

CARDIFF UNIVERSITY

FINAL REPORT

---

Developing a video plugin for OBO-Edit  
to support physiotherapy applications of  
this open-source ontology editor

---

*Author:*

Thomas EDWARDS

*Supervisor:*

Dr. I SPASIĆ

*Moderator:*

Professor Paul Rosin

CM3203 - One Semester Individual Project - 40 Credits

May 2014

## 0. Contents

---

<b>Contents</b>	<b>1</b>
<b>List of Figures</b>	<b>4</b>
<b>1 Abstract</b>	<b>6</b>
<b>2 Acknowledgements</b>	<b>7</b>
<b>3 Introduction</b>	<b>8</b>
<b>4 Background</b>	<b>10</b>
4.1 Ontologies . . . . .	10
4.1.1 The importance of ontologies . . . . .	11
4.2 Ambiguity within written descriptions . . . . .	13
4.3 Existing efforts of creating unambiguous descriptions within ontologies . . . . .	14
4.4 The aims of the project . . . . .	17
<b>5 Social experiment</b>	<b>19</b>
5.1 Overview . . . . .	19
5.2 Creating a fair experiment . . . . .	19
5.2.1 Creating the exercise videos . . . . .	19
5.2.2 Obtaining test subjects . . . . .	20
5.2.3 Undertaking the experiment . . . . .	21
5.2.4 Evaluation of results . . . . .	22
5.3 Conclusion of the Social experiment . . . . .	24
<b>6 Video plug-in Approaches</b>	<b>25</b>
6.1 Overview of section . . . . .	25
6.2 Features found within existing applications . . . . .	25
6.3 Investigation of technologies that can support video . . . . .	26
6.4 Approaches to implement the video plug-in . . . . .	27
6.4.1 Creating a Plug-in in Matlab . . . . .	27
6.4.2 Adapting the Term image plug-in . . . . .	27
6.4.3 Creating a new plug-in in Java . . . . .	28
6.4.4 Extending the functionality of OBO Edit . . . . .	28
6.4.5 Creating a separate ontology viewer . . . . .	29
6.4.6 Conclusion of findings . . . . .	29

<b>7</b>	<b>Specification and Design</b>	<b>30</b>
7.1	Application development tools . . . . .	30
7.2	System scope and Boundaries . . . . .	30
7.2.1	System scope . . . . .	30
7.2.2	System Boundaries . . . . .	31
7.3	System Requirements . . . . .	32
7.3.1	Functional requirements . . . . .	32
7.3.2	Non-Functional requirements . . . . .	33
7.4	GUI Design . . . . .	33
7.4.1	Prototype . . . . .	34
7.4.2	Heuristic Evaluation . . . . .	36
7.5	System Design . . . . .	38
7.5.1	Use case diagram . . . . .	39
7.5.1.1	Actors that will interact within the system: . . . . .	39
7.5.2	Activity diagram . . . . .	42
7.5.3	Class diagram . . . . .	43
7.5.4	Sequence diagram . . . . .	44
<b>8</b>	<b>Implementation</b>	<b>45</b>
8.1	Displaying an ontology tree . . . . .	46
8.2	Displaying and updating term descriptions . . . . .	49
8.3	Playing a video based on the current tree item . . . . .	51
<b>9</b>	<b>Evaluation of the system</b>	<b>54</b>
9.1	Functionality testing . . . . .	55
9.1.1	Summary of the results . . . . .	56
9.2	Usability testing . . . . .	56
9.2.1	Number of participants . . . . .	56
9.2.2	Type of participants . . . . .	57
9.2.3	Task given to the participants . . . . .	57
9.2.4	Performance of usability testing . . . . .	57
9.2.5	Summary of findings . . . . .	58
9.2.5.1	Common problems . . . . .	59
9.3	Utility testing . . . . .	59
<b>10</b>	<b>Future work</b>	<b>60</b>
10.1	Moving the user base over to protégé . . . . .	60
10.2	Narration on videos . . . . .	61
10.3	Implementing 3D models . . . . .	61
10.4	On-line application . . . . .	61
10.5	Mobile application . . . . .	61
10.6	Language packs for TRAK . . . . .	62
<b>11</b>	<b>Conclusion</b>	<b>63</b>
<b>12</b>	<b>Reflection on learning</b>	<b>65</b>
12.1	Project management problems . . . . .	65
12.2	Project Development problems . . . . .	68

12.3 Lessons learned . . . . .	69
<b>13 Appendices</b>	<b>71</b>

## 0. List of Figures

---

4.1	The TRAK ontology loaded in OBO-edit . . . . .	11
4.2	Set containing subsets . . . . .	12
4.3	Search space decreasing as subcategories are accessed . . . . .	12
4.4	Flat Structure . . . . .	13
4.5	TRAK image . . . . .	14
4.6	Using the term image plug-in . . . . .	15
4.7	Mediaslot displaying a video . . . . .	16
5.1	CUSTV account showing a fitness induction . . . . .	20
5.2	Feedback questionnaire . . . . .	22
5.3	Average feedback scores . . . . .	23
5.4	Graph of results . . . . .	23
6.1	Matlab video . . . . .	26
6.2	Technology comparison table . . . . .	27
6.3	OBO Edit System scope . . . . .	28
7.1	System boundaries . . . . .	31
7.2	Initial screen . . . . .	34
7.3	Video display screen . . . . .	34
7.4	Editing a Description . . . . .	35
7.5	Description saved screen . . . . .	35
7.6	Neilsen Heuristics . . . . .	36
7.7	Severity rating system . . . . .	36
7.8	Heuristic problem 1 . . . . .	37
7.9	Heuristic problem 2 . . . . .	37
7.10	Heuristic problem 3 . . . . .	38
7.11	System Use Case Diagram . . . . .	40
7.12	System activity Diagram . . . . .	42
7.13	System Class Diagram . . . . .	43
7.14	System activity Diagram . . . . .	44
8.1	Displaying an ontology tree . . . . .	46
8.2	SQL query retrieving term names . . . . .	47
8.3	SQL query retrieving term names . . . . .	47
8.4	Displaying term descriptions . . . . .	49
8.5	Updated term descriptions . . . . .	49
8.6	Updated term descriptions . . . . .	50
8.7	Update term descriptions . . . . .	50

---

8.8	Application Playing a video . . . . .	51
8.9	TRAK code to URI . . . . .	52
8.10	Video Looping Code . . . . .	53
9.1	Information System Properties . . . . .	55
9.2	Test case template . . . . .	56

## 1. Abstract

---

This project is aimed at working with members of both the school of computer science as well as the school of medicine in order to conceptualise a plug-in for OBO Edit that will allow the software to view videos. This will allow for greater understanding of explanations, which in plain text form may be open to interpretation. For this project an experiment took place to demonstrate the descriptive power of video versus a more traditional text description. The system itself will be a representation of how a plug-in would look and feel in the existing OBO edit software. The final project will be distributable and runnable without additional software installations.

## 2. Acknowledgements

---

This section acknowledges and thanks all those who helped make this project possible. I would like to give a special thanks to the following people:

Dr Irena Spasić	For academic and moral support
Dr Kate Button	For academic support
Mrs Gerian Edwards	For moral support
Miss Aleksandra Nacheva	For moral support
Cardiff University Sport	For use of their facilities
CUROP	For project funding
Cardiff University School of Computer Science	For project funding
Mr L. Semmens	For use of equipment



### 3. Introduction

---

OBO-Edit is an open source ontology editor written in the Java programming language that supports the OBO format, a web ontology format. OBO format is a “Biologist friendly” alternative to OWL and has been used in order to represent a wide variety of biomedical ontologies. One successful contribution to the “OBO-Edit” project is the Term Image Plug-in. The Term Image Plug-in allows a user to attach an image to an ontology term and in turn will allow the user to view images related to ontology terms.

One of the projects currently being worked on as joint effort between the School of Computer Science and in Informatics and the School of Healthcare Sciences is a physiotherapy ontology (TRAK) designed to help the treatment of knee conditions. The project relies on detailed descriptions of several physiotherapy exercises and has been adapted to make use of the Term Image Plug-in where images have been used in order for exercises to be understood in a non-interpretive manor. Where images have been a tremendous help in the project it is still felt that ambiguity remains within the explanations of exercises.

OBO-edit and the Term image plug-in in their current forms lack the ability to use more interactive methods of content delivery. One of the proposed mediums for content delivery to further aid the consumption of information in a truly non-interpretative manor is video. Video has been proposed as a content delivery mechanism for exercise as it is felt that if implemented correctly videos will be able to show an end user exactly what is suggested by a description

One of the main reasons that the School of Computer Science and the School of healthcare studies are looking to remove ambiguity of the descriptions within their project is that the project once complete will focus on aiding patients in performing exercise. A worry is that improper exercise could end up harming a patient rather than helping them recover. When discussing lunges NHS Physiotherapist Nick Sinfield had this to say

”Using improper form not only has less benefit for the thighs and buttocks,  
but it can result in injury, especially to the knees and back”

When working with patients who have already undergone injury, the risk of the patient injuring themselves further could act as detrimental to the project and could potentially undermine the work that the project as a whole stands for.

This project investigates the use of video for the consumption of unambiguous interpretation of information and aims to conceptualise how video functionality could be brought to the OBO-Edit software in a manner similar to that of the Term Image plug-in.

In addition to this, this project will also attempt to give a scientific basis to the claim that videos will give a true unambiguous interpretation of information. This will be done through a social experimentation.

## 4. Background

---

This section will discuss the history of ontologies as well as current efforts of removing ambiguity from data within ontologies. It will also discuss how ambiguity occurs within written language and how this ambiguity could directly effect this project as well as how video and other mediums of communication have been used in the past to combat ambiguity within the written word.

### 4.1 Ontologies

As defined by Gresereth & Nilsson:

”An ontology is an explicit specification of a conceptualization. The term is borrowed from philosophy, where an Ontology is a systematic account of Existence. For AI systems, what ”exists” is that which can be represented. When the knowledge of a domain is represented in a declarative formalism, the set of objects that can be represented is called the universe of discourse.”

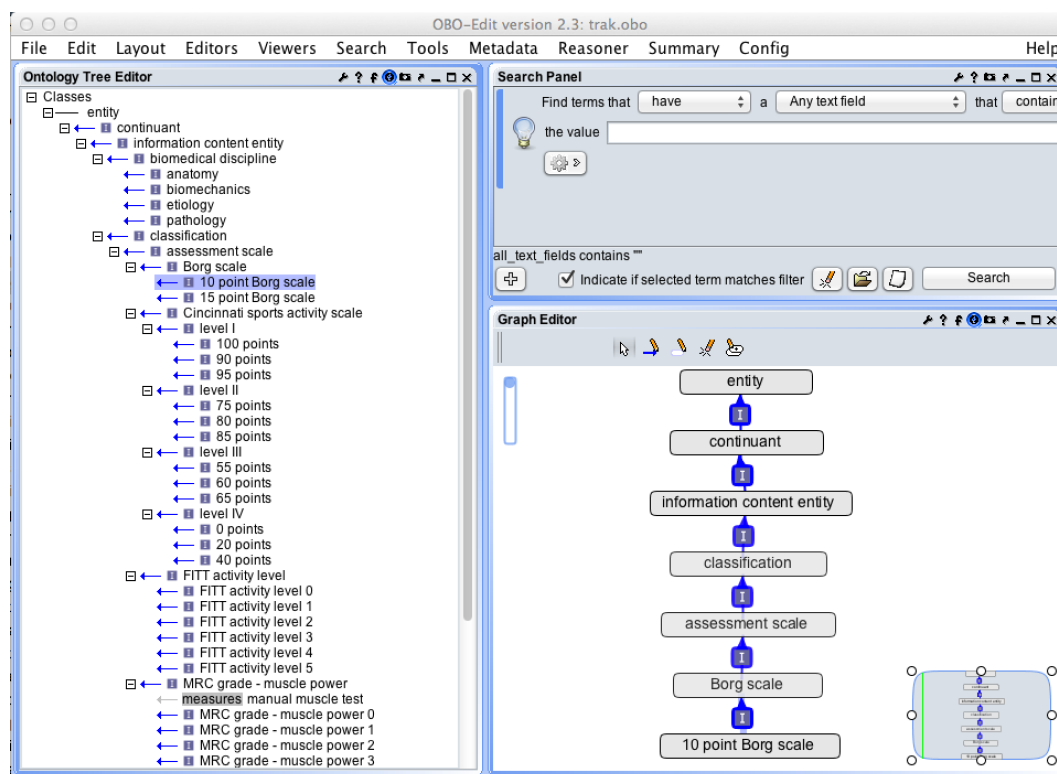
For the purposes of this project we need to think of an ontology as a structure that holds terms. These terms are objects of reality. The ontology allows us to hold these terms in a tree like structure that gives a drill down approach.

In this project we will be dealing with the TRAK ontology. TRAK is defined by the NCBI as:

”An ontology that formally models information relevant for the rehabilitation of knee conditions. TRAK provides the framework that can be used to collect coded data in sufficient detail to support epidemiology studies so that the most effective treatment components can be identified, new interventions developed and the quality of future randomized control trials improved to incorporate a control intervention that is well defined and reflects clinical practice”

An example of the TRAK ontology being viewed in an ontology editor is shown in the figure below:

FIGURE 4.1: The TRAK ontology loaded in OBO-edit

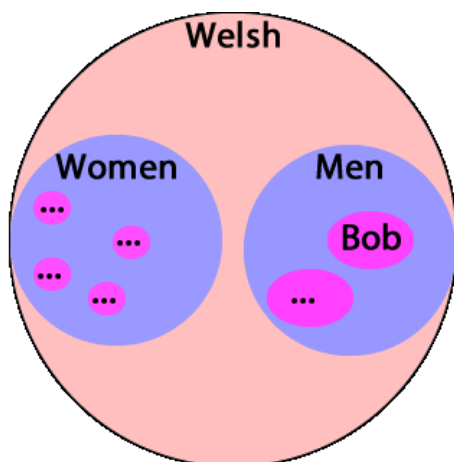


TRAK is being used in a number of projects to help support treatment of knee injuries within the physiotherapy domain. As the terms and definitions already exist within the TRAK ontology this project will focus on creating videos that accurately depict the existing definitions.

#### 4.1.1 The importance of ontologies

When looking at why an ontology is a useful structure of information we need to consider alternative methods of storing, as well as for looking through the information. In definition an ontology can be thought of as a universe of "things" that have "things" within "things". For example, we would be able to have the entity Bob. Bob is welsh and a man. However, from this we would be able to say a lot about BOB but fairly little about men or Welsh people. This can be compared to sets. Wherein, sets can have subsets but that subset does not necessarily contain all of the elements of the greater sets. For illustrative purposes the example of Bob is shown in the figure below:

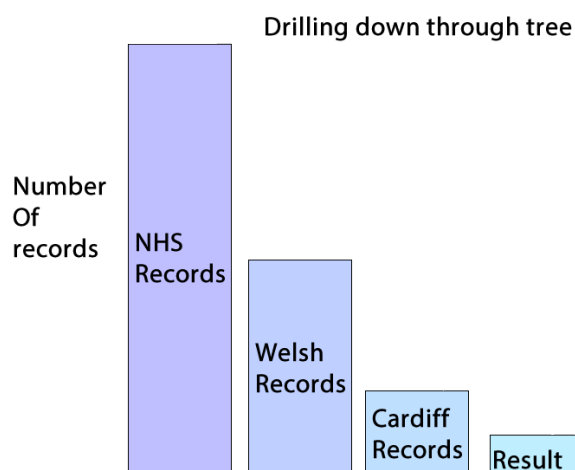
FIGURE 4.2: Set containing subsets



Ontologies are based on the theory that at a core level they are representing the universe as a series of sets. This can be seen as beneficial to us as it is able to make complex information more understandable to the human mind. Further to this, digesting information as an ontology is able to give us the ability to search through a large amount of data without the need for complex search algorithms.

As a pseudo comparison the ontology itself can be thought of as searching algorithm, wherein every step we make looking through the ontology will allow us to gain access to a smaller set. Each one of these steps usually gives the benefit of cutting the search space by a huge amount. The illustration below demonstrates how categories could fit within categories. It also shows how drilling down through categories can give a greatly reduced search space.

FIGURE 4.3: Search space decreasing as subcategories are accessed



The other way of thinking of how we will be able to store and retrieve information without the concept of "things" within "things" is to think of a flat structure of data.

The same information as shown as a concept of "things" within "things" would take a different visualisation if seen as a flat structure as seen in the figure below:

FIGURE 4.4: Flat Structure



When looking at data within this structure we are able to see that even when sorted we would have to scan through potentially all of the items to get to the item that we want. This will give us a run time complexity of  $O(n)$  or linear time where as the set gets bigger it will take longer to search for the desired item.

In conclusion, as this project aims to help users with correct exercise, we will be dealing with a large initial amount of terms that in the future can only grow larger. Because of this it is foreseeable that with a flat structure we will end up having a users wasting a large amount of time looking for the definitions that they need. This makes an ontology an ideal structure for the project as it will allow the users to use identifiable categories to sort through in order to get to the element that they need.

## 4.2 Ambiguity within written descriptions

A written document will always be open to interpretation because of the nature of language. When looking at ambiguity for this project we will focus on two types of ambiguity:

- Lexical ambiguity - Lexical ambiguity occurs when a word has more than one meaning.

"Examples of lexical ambiguity are everywhere. In fact, almost any word has more than one meaning. "Note" = "A musical tone" or "A short written record."

- Structural ambiguity - Structural ambiguity occurs when a sentence can be taken in more than one way an example of this can be seen in a the TRAK description for TRAK:0000306 where in it was stated

"jog forwards weaving in and out of the floor markers."

For this project lexical ambiguity is not as important to take into account as it is reasonable to assume that descriptions within the TRAK ontology will be relating to medical terms. More specifically, the descriptions that this project will be dealing with, will almost all of the time when a second meaning of a word exists be referring to a part of the body or movement.

The issues for the descriptions of this project far more lie in structural ambiguity. For example, if we take the description for TRAK:0000306, when asked some users felt this meant weave tightly between the markers and others felt that the description implied a loose weave. On a description of structural ambiguity it is mentioned that "In normal speech, ambiguity can sometimes be understood". With the above example it is felt that with a user being able to see an actor performing the weave between the markers would give a clear idea of what is meant within the description.

### 4.3 Existing efforts of creating unambiguous descriptions within ontologies

When looking into the history of text alternatives for ontologies there are a few noteworthy examples that directly relate to this project. These are:

- The Term image plug-in

The Term image plug-in is a plug-in for OBO Edit that allows an image to be linked to a term within the ontology. The TRAK ontology has previously been annotated with images for use with the Term image plug-in. The images allowed for a graphical representation of the text descriptions. As many users reported that illustrated images were far clearer than their text description counterparts the project has acted as inspiration for this project. One of the images used for the TRAK ontology is shown below:

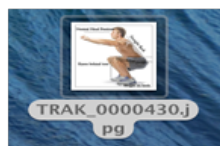
FIGURE 4.5: TRAK image



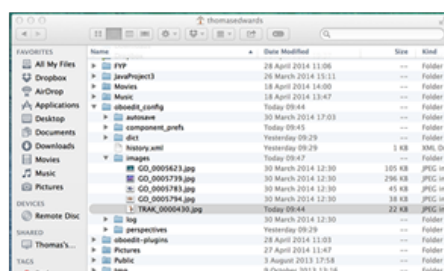
The term image plug-in is simple to use and only needs the user to rename an image with the desired TRAK ID and place it in a pictures folder within the OBO-Edit configuration folder. An example of how to use the Term image plug-in is shown in the figure below:

FIGURE 4.6: Using the term image plug-in

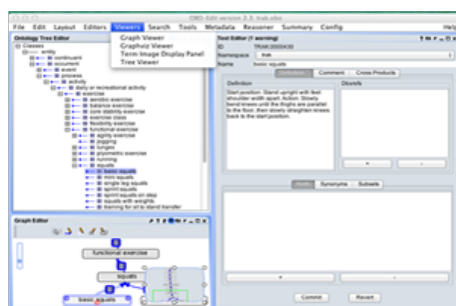
**Rename the image that you want to add with the following Schema "ontology"\_"item number"**



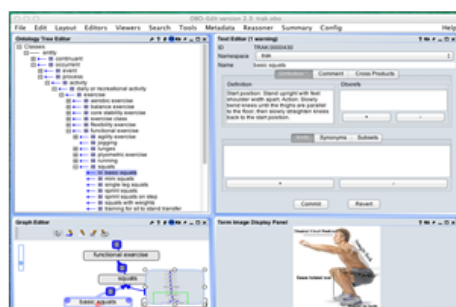
**Move the image to your pictures folder in your OBO-Edit configuration folder**



**In OBO Edit go to the "Viewers" Menu and select "TERM Image Display Panel"**



**If you navigate to the term within the ontology your image will now be displayed**



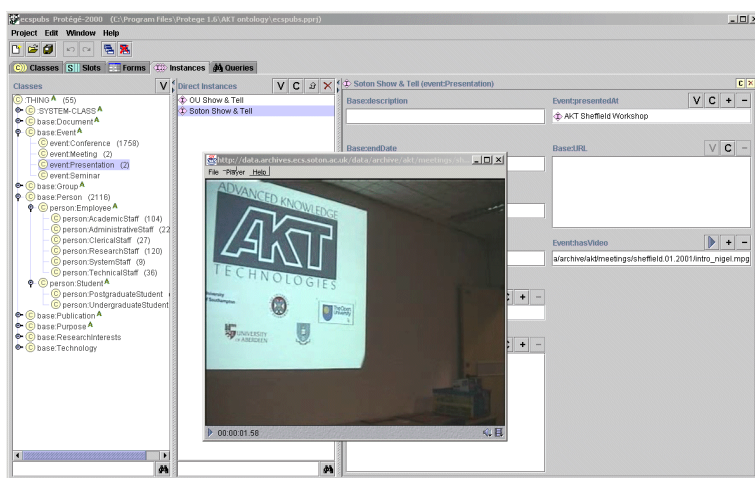


The main advantages that images were able to offer the TRAK ontology are as follows:

- Images allow users to see relatable movements
  - Images allow for text descriptions to be more clearly broken down into their steps
  - Coupled with the description the user is able to get a richer understanding of what is being asked of them
- Mediaslot for protégé

Mediaslot is a video plug-in developed for the protégé ontology viewer. It encapsulates all of the functionality that we want to add to OBO edit. It allows a user to tie a video to a term. A main limitation of the Mediaslot plug-in is that it has exclusively been developed for the protégé ontology viewer and is incomparable with OBO Edit at a code level. An example of the Mediaslot plug-in is shown in the figure below:

FIGURE 4.7: Mediaslot displaying a video



As illustrated in the figure above another disadvantage of the Mediaslot plug-in is that the video content is displayed in a separate interface. In comparison the Term image plug-in allows for content to be viewed as a "panel", this allows for the content to be displayed in the same interface as the ontology viewer. This gives us a more seamless look and feel to the application.

Mediaslot will be able to act as inspiration for designing the video plug-in for OBO edit as it already encapsulates the features we need in OBO Edit.

Whilst it is arguable that the functionality of this project has already been accomplished by the Mediaslot plug-in, it still remains true that the functionality is unavailable to OBO Edit. One solution to this problem would be to migrate users over to the protégé software. However, it is unlikely that we would be able to get the users of TRAK ontology to migrate over to protégé as the project is already well established and the learning curve of protégé has been judged to be too high for the users.

## 4.4 The aims of the project

This project has two main aims:

- The first aim of this project is to undertake an experiment

This experiment will aim to help prove that using video as a medium for delivering our provided content is indeed a more efficient medium than the text descriptions or illustrated images that are in use today.

In order to accomplish this task videos for the terms related to exercises within TRAK ontology will be needed. The videos will become part of the TRAK ontology and not just used for this project. Therefore, it is important that they are of a high quality and feature actors of a semi-professional nature.

The main benefit of this aim will be to offer a clear indication on whether or not the plug-in for OBO edit will give a noticeable advantage to its target audience. Another benefit of creating the exercise videos is that projects like "TRAK for Facebook" are looking to implement videos. Therefore, by making sure that the videos are created to a high standard and making them available under licence, other projects will be able to benefit from the videos created for this project.

- The second aim of this project is to create a video plug-in for OBO Edit ontology viewer.

In order to accomplish this aim this project will need to investigate the possibilities of expanding the functionality of OBO Edit. This will be done by analysing existing projects like the Term image plug-in and the Mediaslot plug-in in order to gather how their functionality is accomplished and how their design can be adapted in order to fulfil the goals of this project.

This aim will also investigate the possibility of creating a standalone application that will allow for an ontology to be accompanied with video.

There are two main target users for this part of the project:

- The first and main set of target users are patients recovering from knee injuries. These users will be able to use OBO Edit to view the TRAK ontology along with the videos in order to exercise correctly without the need for help from a physiotherapist. It is hoped that in this way stress will be able to be relieved from an ever stretched NHS system.
- The second set of target users are researchers and physiotherapists who will use the TRAK ontology for learning purposes.

## 5. Social experiment

---

### 5.1 Overview

This section will discuss the social experiment that was undertaken in order to judge if using video is an acceptable technique of removing ambiguity from data. This section will detail how the video supplements were gathered, how the experiment was undertaken, and the results of the experiment.

### 5.2 Creating a fair experiment

The only variable that could effect the fairness of this experiment is the exercises themselves. In order to eliminate any bias that could occur the experiment only used exercises that video supplements have been obtained and all volunteers were asked to perform multiple random exercises.

#### 5.2.1 Creating the exercise videos

For the purposes of the experiment 23 different videos were recorded each corresponding to an exercise within the TRAK ontology that had both a description and corresponding image.

Cardiff University Sports Training Village (CUSTV) allowed us to use their facilities in order create the videos and carry out the experiment. This was done to remove any variable in the environment that could influence the result.

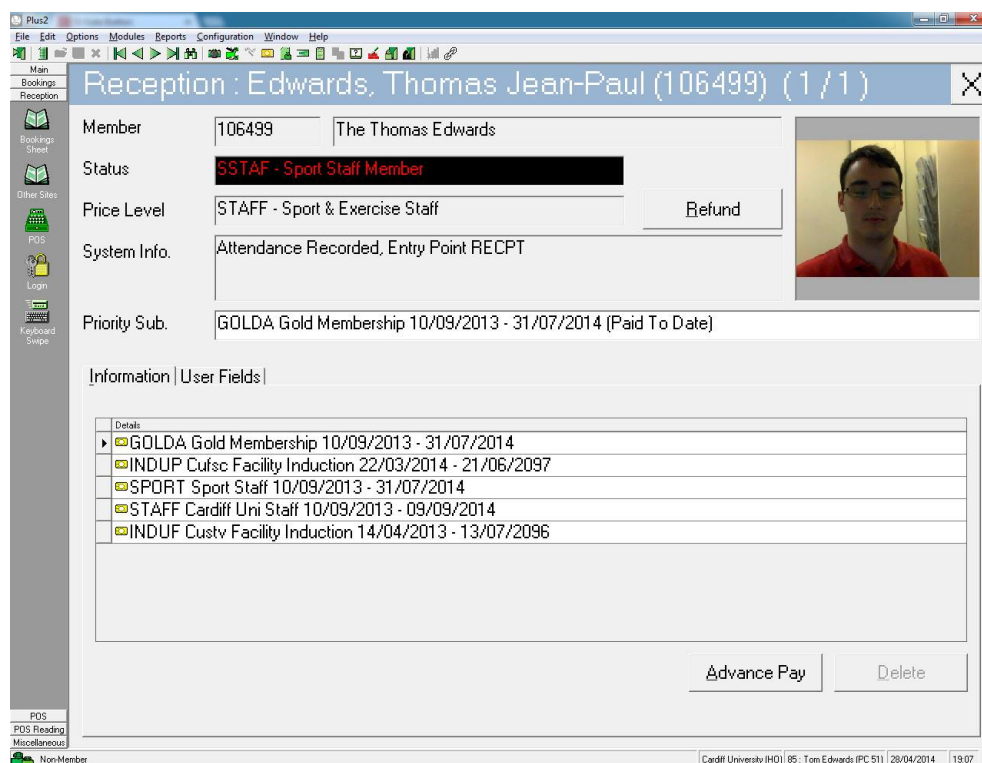
In order to make the videos for the experiment as accurate as possible a volunteer physiotherapist was recruited in order to perform the exercises. Each exercise was recorded with multiple takes each with different angles and other adjustments. All of them were overseen by Dr Kate Button, a research fellow at Cardiff university who was able to give guidance on what takes to use for each exercise. This was done to guarantee that no information was misinterpreted when the videos were created.

## 5.2.2 Obtaining test subjects

To guarantee the sample space for this experiment was large enough to ensure that there was no unforeseen connections influencing the results 30 different volunteers would be used.

The volunteers all needed to have a basic understanding of exercise. To accomplish this we only accepted volunteers that had completed an induction with CUSTV. This was ideal as CUSTV was hosting the experiment and therefore we would be able to check volunteers accounts for a fitness induction as detailed in the figure below:

FIGURE 5.1: CUSTV account showing a fitness induction



Volunteers were gathered on a word of mouth bases as well as on the day by promotional posters hung around CUSTV

As the sample space was quite large and obtaining the required amount of volunteers could prove challenge a cash incentive was offered by the School of Computer Science and Informatics in the form of Amazon gift vouchers. This was able to act as a catalyst for obtaining volunteers for the experiment. However, it did not affect the quality of the volunteers as we still required them to have a CUSTV induction.

### 5.2.3 Undertaking the experiment

To keep the experiment fair the project was overseen by a member of physiotherapy at Cardiff University Sports Training Village. This was done to aid the quality of information provided during feedback.

The experiment got each volunteer to perform the steps detailed below for each exercise:

1. Supervisor decides if the exercise will be accomplished with photo or video supplements
2. The volunteer is given time to study the exercise
3. The volunteer performs the exercise in their own time
4. Once the volunteer feels that they have completed an exercise they are asked to rate the exercise on a scale of 1 - 5 with 1 meaning they had not understood what they were meant to do, and 5 meaning that they had fully understood what was asked of them. At this point the physiotherapist is also asked to give a rating on the same scale. Both of these ratings were then recorded and the volunteer moves on to the next exercise

The Cycle above continues until the volunteer has performed 10 exercises 3 of which will be accompanied with video and 3 with photos. The remainder of the exercises were to be performed from only descriptions.

### 5.2.4 Evaluation of results

During the testing the user was asked to fill in a questionnaire as illustrated in the figure below.

FIGURE 5.2: Feedback questionnaire

Name:.....  
Student Number:.....

For this experiment you will be asked to perform 10 exercises. For each of the exercises performed you will be given a text description. For some exercises you will be given a video or photo supplement to help you understand what is needed from you. Please, fill in the bellow table with the exercise number for the exercises you have performed, whether a supplement was given as well as a rating from 1 -5 with how easy you felt the exercise was with 1 being that you did not feel you understood what was being asked of you, and 5 being that you understood exactly what was being asked of you.

Exercise number	Video Supplement	Photo Supplement	Rating	Physiotherapist Rating

After each user had undertaken the experiment we collected the questionnaire from them, making sure that it was complete. The results were then entered into an excel document so that the user experience scores would be able to be compared against each other. The full document of results can be found in the Appendix 1.

Because we asked each of the 30 participants to perform 3 exercises of each type we were able to obtain a sample set of 90 results for each method of testing. This was seen to be an adequate number for testing, as it should be large enough for the findings to be without bias.

At this point the information was anonymised. This was done because identifying personal information such as student numbers were only ever collected in order to comply with the requirements that Cardiff University's School Of Computer science had laid out in order to offer a prize and have no bearing on the results.

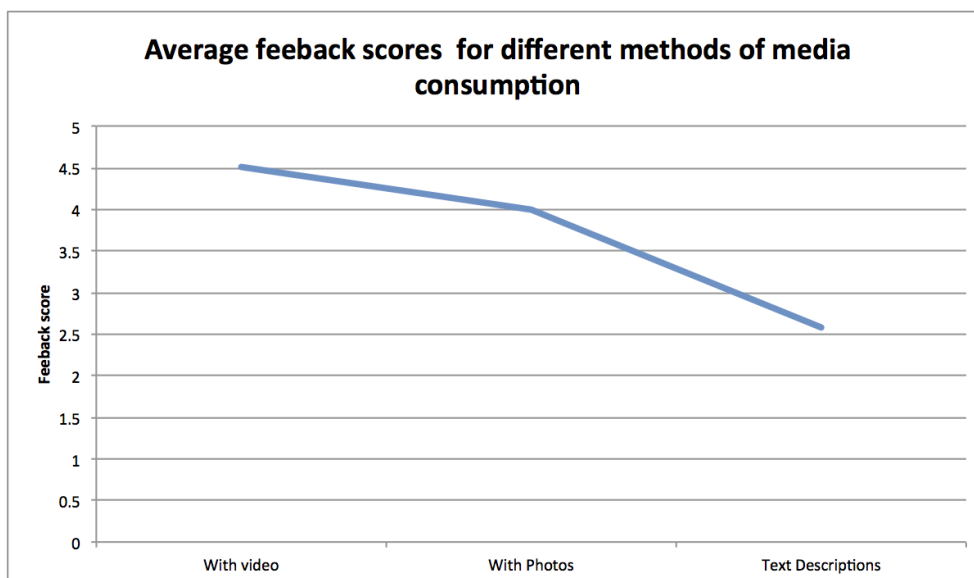
Excel was then used in order to get the average values of the user scores given to each category of test. This is illustrated in the figure below:

FIGURE 5.3: Average feedback scores

	Average score from users	Average score from physiotherapists	Average score
Descriptions with video	4.51	4.5	4.51
Description with photos	3.98	3.99	3.99
Descriptions only	2.59	2.57	2.58

The above table shows that the gap between having the description and form of visual aid is grater than the gap between how users rated exercises with videos and exercises with photos. This can be visualized on a chart as shown in the figure below:

FIGURE 5.4: Graph of results



From the results we can see that videos have been able to provide the users with an advantage over using both the text descriptions as a standalone resource or using the text descriptions with a photo counterpart. It is worth noting however, that the jump between users not having a supplement and using video and users moving from descriptions with a photo supplement are not equivalent. This can be compared to the law of diminishing returns, wherein the initial effort provides the greatest jump in in the project and as the project evolves it will take exponentially greater effort to improve the project as it nears perfection



### 5.3 Conclusion of the Social experiment

In conclusion it can be stated that the experiment was able to demonstrate all of the goals that it aimed to prove and therefore can be called a success.

When looking at the findings of the data gathered in this experiment we now have evidence to state that a user does indeed find video to be a more ideal learning supplement than anything currently available with the TRAK ontology.

As the exercises that were supplemented with videos were able to reach such high feedback scores we can also safely assume that the videos created for this section were a success.

Finally it was shown in the summary of results that a user would be more likely to understand a text description when it was accompanied with a supplement and that the preferred supplement was indeed the videos created for this project.

As this section caused concern about the ambiguity of the Data within the TRAK ontology we can now conclusively state that there was indeed user ambiguity within the description of the terms of TRAK ontology and that the video supplements created for this project have been able to give us the most unambiguous solution to date.

When looking at the user feedback we can still see that on average the scores still show that the video supplements despite giving clarity to the data still leave some ambiguity. This will mean that in future further projects could investigate the implementation of a truly unambiguous system.

We are now able to continue the project by investigating and implementing technologies that will allow for videos to accompany the TRAK ontology in the knowledge that this project will indeed be beneficial to its intended users.

## 6. Video plug-in Approaches

---

### 6.1 Overview of section

This section will be split two main parts. The first part will discuss existing features within applications that can assist in accomplishing the goals of this part of the project. It will also discuss the different approaches that could be taken in order to develop a plug-in for OBO Edit as well as the the possibility of creating a standalone application that is able to be able to link video terms within an ontology and the benefits that this approach could give.

The second part of this section will demonstrate how the most suitable approach found in section one could be designed and implemented in order to fully meet the requirements of this project.

### 6.2 Features found within existing applications

When investigating existing applications there are several features that would be beneficial to retain for this project.

- The Term image plug-in

Some of the functionality that needs to be retained from the Term image plug-in is the ability to match a file to a term based on a naming scheme.

For example, if we have a term in the TRAK ontology named TRAK:000023, then the file name for this term would be "TRAK\_000023.jpg" an underscore is used for compatibility with file structures. To load the file we would need to take the ID of the selected term, replace the semicolon and ".jpg" to the end of the URI. At this point we are then able to string match with the term in the ontology with the file that needs to be loaded.

- Mediaslot

Mediaslot has been able to get video working within the protégé ontology editor. To accomplish this Mediaslot has used the Java Media Framework(JMF). This

framework allows for photo, video and audio be used within a Java application. One limitation of the JMF is that the project is now seen as outdated and therefore, it is only a matter of time before support is unavailable for the tool.

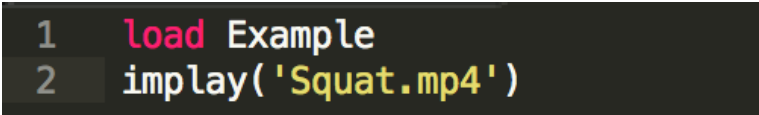
## 6.3 Investigation of technologies that can support video

There are a few common technologies that are used to support video within an application. As OBO Edit is a Java application, this section will only discuss technologies that are seen as relevant to a Java application.

- Matlab

Matlab is a programming language that is designed to handle video. Getting video to play in Matlab can be accomplished easily with the syntax needed to play video can be reduced to 2 lines of code. This is demonstrated in the figure below:

FIGURE 6.1: Matlab video



```
1 load Example
2 implay('Squat.mp4')
```

- JMF

The JMF is the traditional way of handling Video in a Java project. it is supported by the latest release of Java and only requires that the jar file is distributed with the applications created with it.

- JavaFX

JavaFX is the latest way of handling media within Java. There is no need for a separate file to be distributed with the package as it is pre-installed with the latest builds of Java.

- CUDA for Java

CUDA for Java gives the functionality of the CUDA computing platform to Java applications. CUDA is able to take advantage of threading and therefore on a modern computer would enable the project to run quickly. The main limitation with this technology is that without complex registry hack CUDA is hardware specific and requires an nVidia graphics card in order to run. This could potentially restrict a percentage of the projects target audience from running the application.

The figure below compares the positive and negative aspects of the technologies described above:

FIGURE 6.2: Technology comparison table

Technology	Matlab	JMF	JavaFX	CUDA
<b>Positive aspects</b>	<ul style="list-style-type: none"> <li>• Easy to Program</li> <li>• Natively handles video</li> </ul>	<ul style="list-style-type: none"> <li>• Extends Java to support video</li> <li>• Uses the same syntax as Java</li> </ul>	<ul style="list-style-type: none"> <li>• Extends Java to support video</li> <li>• Uses the same syntax as Java</li> </ul>	<ul style="list-style-type: none"> <li>• Fast</li> <li>• Utilizes threads</li> </ul>
<b>Negative aspects</b>	<ul style="list-style-type: none"> <li>• Requires the user to install Matlab</li> </ul>	<ul style="list-style-type: none"> <li>• Relatively outdated</li> <li>• Requires separate files to be bundled with the application</li> </ul>	<ul style="list-style-type: none"> <li>• Requires the developer to use a signing authority for public distribution</li> </ul>	<ul style="list-style-type: none"> <li>• Has it's own Syntax that can be complex</li> <li>• Requires specific hardware</li> </ul>

From the table above we are able to see that most of the main technologies could be seen as viable options for creating an application that is able to support video. The only exception to this is CUDA that has been discounted at this stage as it is reliant on specific hardware.

## 6.4 Approaches to implement the video plug-in

This section will discuss the different approaches that were considered in the creation of this project.

### 6.4.1 Creating a Plug-in in Matlab

When the project was being worked on as part of CUROP during the summer of 2013 it was suggested that it should be moved to the Matlab platform. When investigated the Matlab platform would be able to support the aims of the project. However, as this project aims to be a self contained application, Matlab was ultimately perceived as unviable as it would require the users to have an installation of Matlab on their machines.

### 6.4.2 Adapting the Term image plug-in

As source code was available for the Term image plug-in, the next approach that was looked at was to adapt the functionality of that plug-in. During the early stages of the investigation of this approach a message from the developer (Available in Appendix 2) revealed that the project was no longer being developed and the original team had no further plans to support it. As the project had limited documentation this approach had to be abandoned.

### 6.4.3 Creating a new plug-in in Java

When looking at creating the video player section of the plug-in two main technologies were looked at. These were:

- JMF

The JMF was considered for the development of the Video player. However, because it is now perceived as relatively outdated and requires more distribution files in order to run. This approach was abandoned.

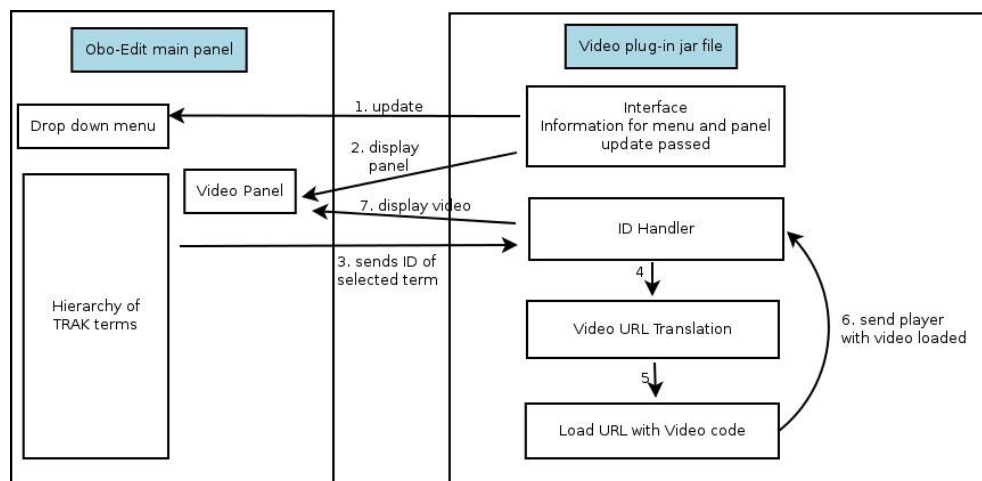
- JavaFX

JavaFX was able to offer a platform to create a video player with relatively few drawbacks. As it has been distributed with builds of Java since 2011 it is acceptable to assume that the majority of the machines that this project will run on will be JavaFX compatible. The only issue with JavaFX is that in order to distribute the application it will need to be signed by a signing authority. As at this time this project will only be used in a development environment the signing issue was not considered to be a problem.

### 6.4.4 Extending the functionality of OBO Edit

Once the core video player was built communication with the OBO Edit ontology viewer was looked at. The method that OBO Edit would use for communication with the video plug-in is explained in the figure below:

FIGURE 6.3: OBO Edit System scope



Attempts to integrate the plug-in into OBO however ultimately failed. This was due to a combination of the documentation for implementing plug-ins being extremely limited, and much like the Term image plug-in the project itself being unsupported by its development team.

#### **6.4.5 Creating a separate ontology viewer**

The development of a plug-in for OBO Edit at this stage is not within the scope of this project, and the core video application fulfils the requirements set out by this project. Therefore, the project was expanded to create an interface that would allow the terms to be viewed along with their TRAK code, description and video

The application would also be developed in Java, and the core code for the video player functionality will be left unchanged. This has been done so that in future, if the documentation becomes available the project can easily be moved to the OBO Edit ontology viewer.

#### **6.4.6 Conclusion of findings**

In conclusion, despite JavaFX being found as a suitable technology for the creation of a plug-in, investigations into expanding the functionality of OBO Edit have ultimately been unsuccessful. This is because of a lack of support and documentation for the ontology viewer.

However, by taking the approach of creating an independent ontology viewer that is able to use a video player created with JavaFX the project is able to achieve its main goal of linking videos to TRAK terms. Therefore, it has been decided that the rest of the project will focus on this approach.

## 7. Specification and Design

---

This section will outline the user requirements for the application and discuss the process of creating the application through design documentation and prototyping.

### 7.1 Application development tools

In order to work with JavaFX a suitable development environment is needed. Oracle recommend "Netbeans". Netbeans will allow development of JavaFX code across multiple operating systems. Netbeans also gives the ability to self sign the JavaFX code, that will allow us to run the code within the development environment without the need to use a signing authority.

### 7.2 System scope and Boundaries

#### 7.2.1 System scope

The system will consist of an ontology viewer implemented in JavaFx. It will be a desktop application that is compatible with the Macintosh, Linux and Windows operating systems.

The ontology viewer must be able to display an ontology tree, allow the user to play video for a user selected exercise term from the ontology, and change the description for the term selected.

Taking into account that the target users for this application will be coming from different backgrounds and have different levels of knowledge, it is important for the system to provide easy to use functions. This will include:

- Well positioned application components for displaying information
- Use of natural language on display components

### 7.2.2 System Boundaries

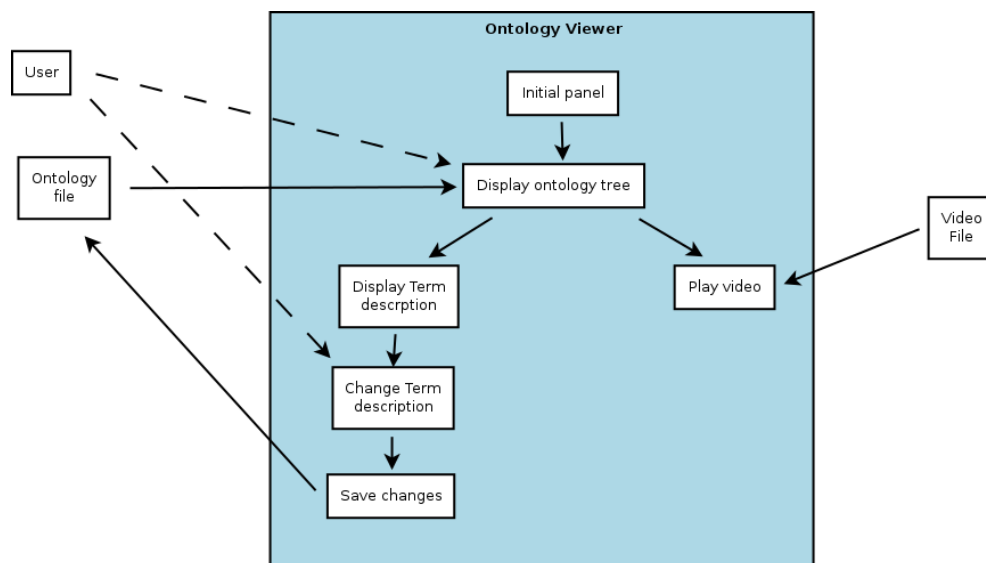
The system needs to interact with the users. Further to that, it will need to be able to save the changes the user has made to the ontology. Therefore, a data structure for saving the modifications made should be created. The main features of the system are:

The system needs to interact with the users. Further to that, it will need to be able to save the changes the user has made to the ontology. Therefore, a data structure for saving the modifications made should be created. The main features of the system are:

- The ability to load an ontology from an external structure
- Have display components to hold an ontology tree, video, and ontology term description
- Be responsive to user interactions: playing video, displaying descriptions of selected terms and providing options for updating a term's description
- Save any changes made by the user

A graphical representation of the system boundaries is given in the figure below:

FIGURE 7.1: System boundaries





## 7.3 System Requirements

In this section the essential functional and non- functional system requirements of the application will be outlined and for each of them give a description and acceptance criteria. This will help us visualise the main system functionalities as well as give use a base for user testing.

### 7.3.1 Functional requirements

Requirement: The system must load an ontology tree in the appropriate panel

Description: When the user loads the application, the system must be able to display the provided ontology as a tree structure within the appropriate panel on the screen.

Acceptance criteria:

- The system on load displays the appropriate ontology as a tree structure in the appropriate panel.

Requirement: The system should display description and name of a term selected

Description: When the user has selected a term from the ontology tree, the relevant description of this term should be displayed in the appropriate panel

Acceptance criteria:

- The system displays the description of the term selected by the user

Requirement: The user must be able to change the description of a particular term

Description: Once the description of the selected term is displayed on the screen, the user must be able to update the displayed description and save the changes.

Acceptance criteria:

- The system allows the user to change the displayed description
- The updated description is saved to the ontology file

Requirement: The system must be able to display a relevant video for the selected term

Description: When the user has selected term for the ontology tree, the system must be able to play a video associated with the selected term in the video panel.

Acceptance criteria:

- The system plays a video associated with the selected term

### 7.3.2 Non-Functional requirements

Requirement: Performance

Description: The system must update its components in sync with the user actions. The videos must be displayed shortly after the user selects an exercise term.

Acceptance criteria:

- A video is displayed within seconds after an exercise term is selected

Requirement: Reliability

Description: Only video, that is appropriate to the selected exercise are displayed by the system.

Acceptance criteria:

- The percentage of the correctly displayed videos for 10 exercises is 100

Requirement: Usability

Description: The functions of the system should be easy to learn and use. The components displayed by the system should be clear and easy to understand.

Acceptance criteria:

- In user feedback usability must obtain an average score of 8/10

## 7.4 GUI Design

One of the most important aspects of this project is the creation of a product that will be easy to use and to learn by its target users. Taking into account that the users will have different technological knowledge levels, it is important for the application to provide simple and easy to use interface. Therefore, to ensure the system design is of a high quality, we need to create a prototype.

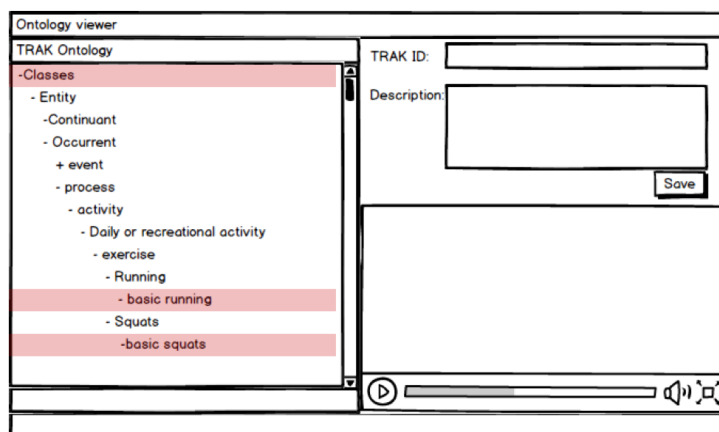
This prototype will also help define a clear action flow for the systems functionality. After the prototype has been created, it will be evaluated using the Nielson's heuristic approach. In this way we will ensure that no significant changes to the design of the final product will be needed, as most of the problems will be solved during the prototype creation.

### 7.4.1 Prototype

The prototype will be created using the “Balsamiq Mockups” software. The images presented below represent the created screens, each representing a step of the basic flow of the system.

#### Step 1: Initial screen

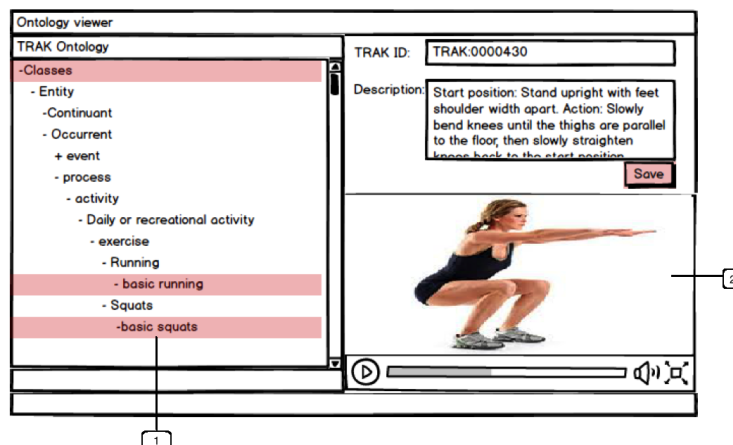
FIGURE 7.2: Initial screen



In the beginning, only the ontology tree will be displayed in the Ontology viewer.

#### Step 2: Play video

FIGURE 7.3: Video display screen



Every time when a user selects an exercise term a video associated with this exercise will be displayed in the video panel.

### Step 3: Change description

When a user has selected an exercise term, along with the video they will also be also given a description for the exercise. The actions that need to be performed in order to change a description for exercise are as follows:

1. Select exercise term
2. Change the given description in the Description panel
3. Click the 'Save' button
4. A message that the description s updated is given

FIGURE 7.4: [Editing a Description

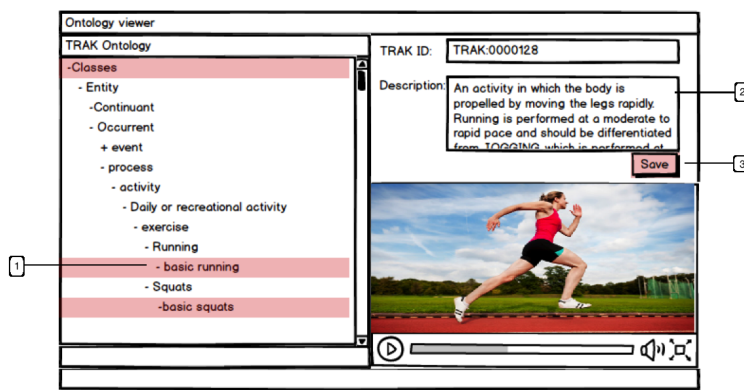
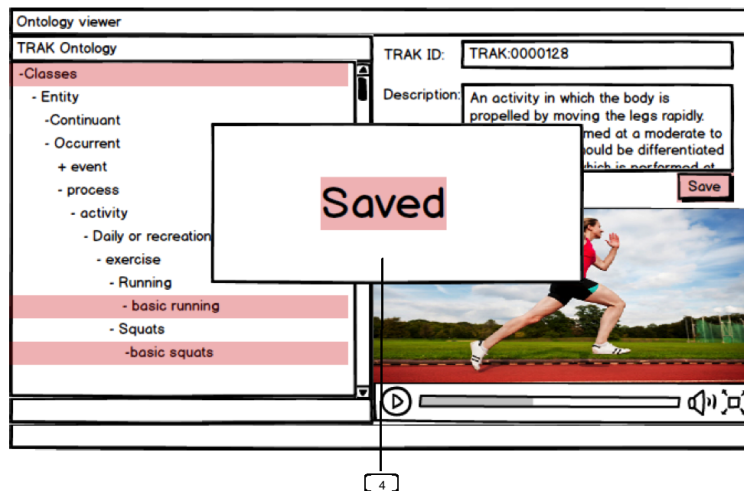


FIGURE 7.5: Description saved screen



### 7.4.2 Heuristic Evaluation

As mentioned above the prototype will be evaluated using Nielsen Heuristics, which are 10 general principles for interaction design. The heuristics that will be used for the evaluation of the prototype are given in the figure below:

FIGURE 7.6: Nielsen Heuristics

No	Heuristic	General usability principles
1	Match the real world	Learnability (L)
2	Consistency and standards	Learnability (L)
3	Help and documentation	Learnability (L)
4	User control and freedom	User Control and freedom (UC)
5	Visibility of system status	Visibility (V)
6	Flexibility and efficiency	Efficiency (EF)
7	Error prevention	Error handling (ER)
8	Recognition, not recall	Error handling (ER)
9	Error reporting, diagnosis and recovery	Error handling (ER)
10	Aesthetic and minimalist design	Graphic design

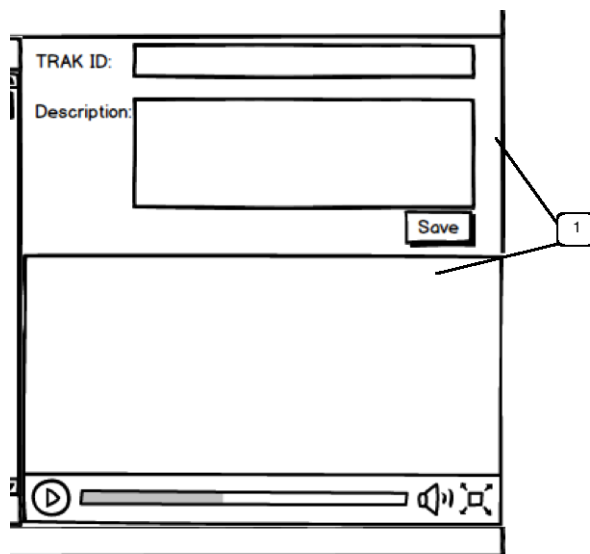
The problems found will be prioritized using the severity rating system shown in the figure below:

FIGURE 7.7: Severity rating system

Rating	Description
1	Superficial usability problem: May be easily overcome by the user or occurs extremely infrequently. This does not need fixing unless extra time is available
2	Minor usability problem: Occurs more frequently than level 1 or is more difficult to overcome. Level 2 is given low priority to fix
3	Major usability problem: Occurs frequently or presents the user with a situation that they are unsure how to proceed. High priority to fix
4	Catastrophic usability problem: Seriously impairs the user. Must be fixed.

The rest of this section will list the problems encountered, by giving the heuristic number, a short description, a rank, and a solution for each one of them.

FIGURE 7.8: Heuristic problem 1



Problem: No flexibility for Video and Description panels

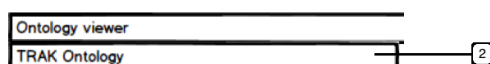
Description: The user is not provided with the option of closing one of the panels: Description panel or Video panel. Further to that, the Ontology viewer is too rigid and does not provide the user with the opportunity to minimize, resize or change the panel's positions.

Heuristic: User control and freedom

Severity rank: 2

Solution: Options for closing, minimizing and swapping the positions of the Video and the Description panel can be provided

FIGURE 7.9: Heuristic problem 2



Problem: No possibility for displaying other ontologies than TRAK

Heuristic: Flexibility and efficiency

Description: There is not provided an option for selecting other ontologies than TRAK. Even though, at this stage this functionality is not important in future it will be better to be able to display more than one ontologies

Severity rank: 1

Solution: A drop down menu can be put in the upper part of the Ontology viewer form where the users can select one of the displayed ontologies.

FIGURE 7.10: Heuristic problem 3

TRAK ID:  3

Description:

Problem: Inappropriate use of component for displaying TRAK ID

Heuristic: Aesthetic and minimalist design

Description: The text field used for displaying the TRAK ID of the selected term might be confusing for the user since it implies that the information can be changed.

Severity rank: 1

Solution: Another design component can be used such as Label for displaying the TRAK ID

## 7.5 System Design

It is important to define the different functionalities and aspects of the system before starting the implementation stage of development. In order to be able to create a reliable and responsive system that implements all requirements, first we need to define answers to the following questions:

- How the system is going to be used?
- By whom the system is going to be used?
- How the system is going to be deployed?

To answer these we can use UML for modelling. This will include the creation of the following diagrams:

- Use case diagram

A use case diagram will be created in order to capture all functionalities that should be implemented. It will also help for identifying the main actors that will interact with the system and need to be taken into account during the implementation.

- Activity diagram

The activity diagram will help to model how the system will accomplish its specified goals. It will specify the steps that need to be performed in order to play a video in our application

- Class diagram

While the use cases describe the behaviour of the system as a set of concerns, the class diagram allows defining the objects that will be needed within the system to meet these concerns. The class diagram will help to identify the classes that need to be implemented and how these classes communicate with each other.

- Sequence diagram

A sequence diagram will be used for modelling particular parts of the system where needs to be captured the order of interactions.

### 7.5.1 Use case diagram

This section will present the Use case diagram and the use cases will be defined. Each use case will be given a description and Basic Flow. Further to this, the actors that will interact with the system will be defined.

#### 7.5.1.1 Actors that will interact within the system:

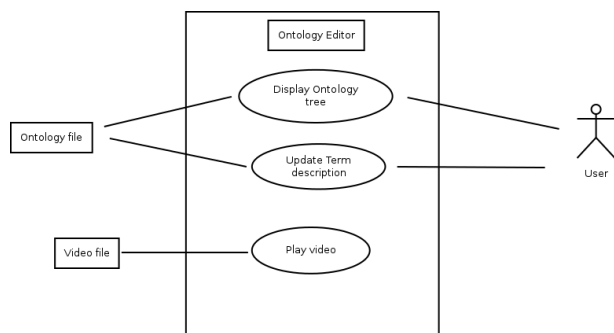
The main entities that the system will be interacting with and need to be taken into account when implementing the system are:

- Users, whose actions should be taken into account such as selecting ontology, changing term description, adding new terms
- Ontology file: The system must be able to read the ontology structure from an external file, display its content and save any changes made to the file
- Video file: The system needs to be able to display videos saved to an external file. It is important to display the relevant video for the selected term by the user. Therefore, a way for matching the term TRAK ID to the video Id should be implemented



The figure below shows the use case diagram for the entire system:

FIGURE 7.11: System Class Diagram



Use case: Display Ontology tree

Description: This use case starts when the user has opened the ontology viewer. When the user selects the option display ontology, the ontology tree must be displayed in the appropriate panel of the viewer.

Preconditions:

- An ontology has been created and saved to the Ontology file

Basic Flow:

1. The user selects 'Open ontology' option
2. The system displays ontology tree

Use case: Update Term description

Description: This use case starts when the user has selected a term from the ontology tree. The user is able to change the description of the selected term and save the changes made.

Preconditions:

- An ontology has been created and saved to the Ontology file

Post-conditions:

- The changes made are saved to the ontology file

Basic Flow:

1. User selects term from the displayed ontology tree
2. The system displays description for the selected term
3. The user makes updates to the displayed description
4. The user clicks 'Save' button

Use case: Play video

Description: This use case starts when the user selects exercise term from the displayed ontology tree. The video is displayed in a video panel and it is relevant to the exercise selected. The video can loop until the user decides to close it.

Preconditions:

- An ontology tree has been displayed
- A video file which stores the videos has been created

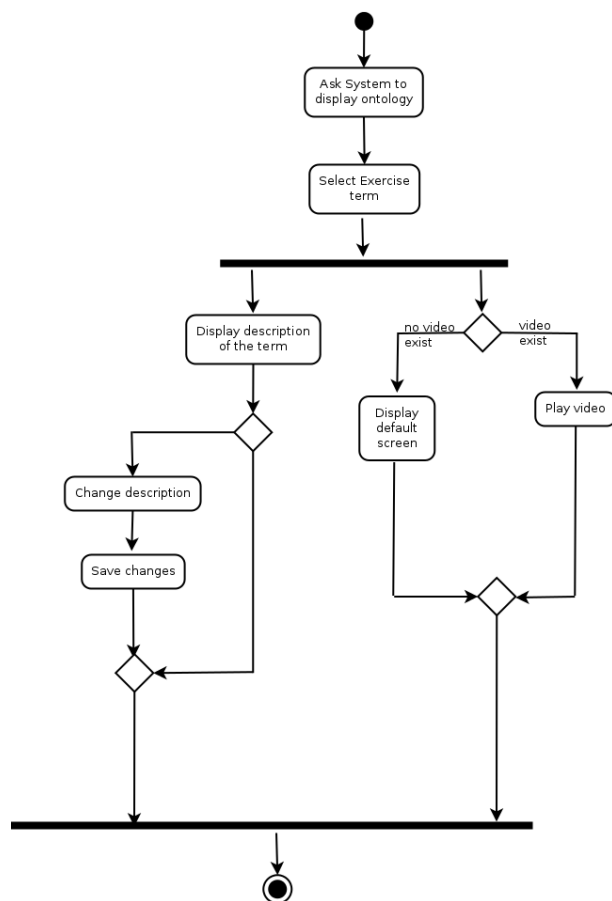
Basic Flow:

1. The user selects exercise term from the ontology tree
2. The system plays the video relevant for the exercise

### 7.5.2 Activity diagram

The activity diagram shown in the figure below captures the entire action flow of the system by specifying all steps that need to be performed in order to play a video in the application

FIGURE 7.12: System activity Diagram



Initially, the system will display only the ontology tree. From here, the user will be able to select an exercise term from the ontology. In this moment two actions will be performed by the system simultaneously:

- The description of the term selected by the user will be displayed in the Ontology viewer
  - The user has the option to change the description given for a term and save the changes made

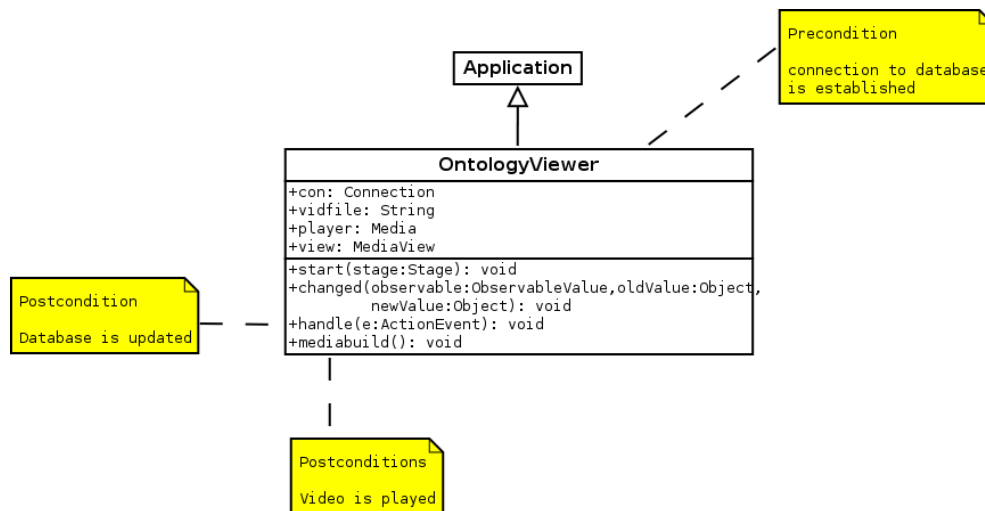
**AND**

- A video associated with the selected term is displayed in the Ontology viewer
  - If a video does not exist for selected term, a default screen will be displayed

### 7.5.3 Class diagram

The class diagram will represent the classes that the system has and how they communicate.

FIGURE 7.13: System Class Diagram



The application will consist of a single class that extends **Application**. This has occurred as the file was initially intended to only act as a video player and separate methods have been added over time in order to keep the application running as a single JavaFX stage.

In this class 4 methods with the following functionalities will be implemented :

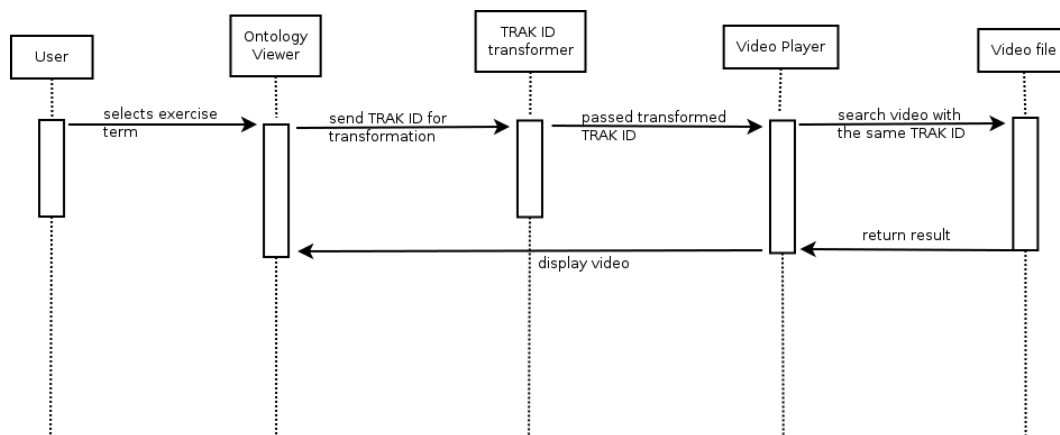
- **start () method:**
  - As this is JavaFx application, the main method is not used. Instead we use `start()` method
- **changed() method:**
  - Check to see if the user has selected a new Item in the ontology and display relevant description and term code on the screen
- **handle() method**
  - Save any changes made by the user to the database
- **mediabuild() method**
  - Receives a TRAK code and converts it into a file name to be used to create a video file

### 7.5.4 Sequence diagram

An important functionality of the system that has not been explained clearly by the other diagrams is the ability to handle the "TRAK IDs" of terms selected by the users. This is an important aspect of the system, as it ensures that the played videos are those that are related to the exercises the users are interested in.

The sequence diagram below will model this functionality: In order to be able to ensure

FIGURE 7.14: System activity Diagram



that the correct video is played for the selected term we need to consider two functionalities during the Implementation stage:

- Transformation of the TRAK ID of the selected term to match the TRAK ID format used for the videos  
Similar to the Term Image plug-in, the matching of term TRAK IDs and video IDs will be based on string matching, where the TRAK ID of the term represented as: TRAK\_000 will be converted to the format used by the videos: TRAK:000
- Sets the right path of the file storing the videos  
It is important to make sure that the right path of the file where the videos are stored is set Both of these functionalities can be implemented in a single method, which will do the following:
- Receive a TRAK code of the selected term and converts it into an appropriate file name
- Search for the file name in the directory where the videos are
- Check if the video exist
- Play the video

## 8. Implementation

---

This section will present The main features of the methods implemented in order to achieve the system requirements.

The GUI consists of a single anchor pane that uses split panes to split the application into 3 main sections. These are:

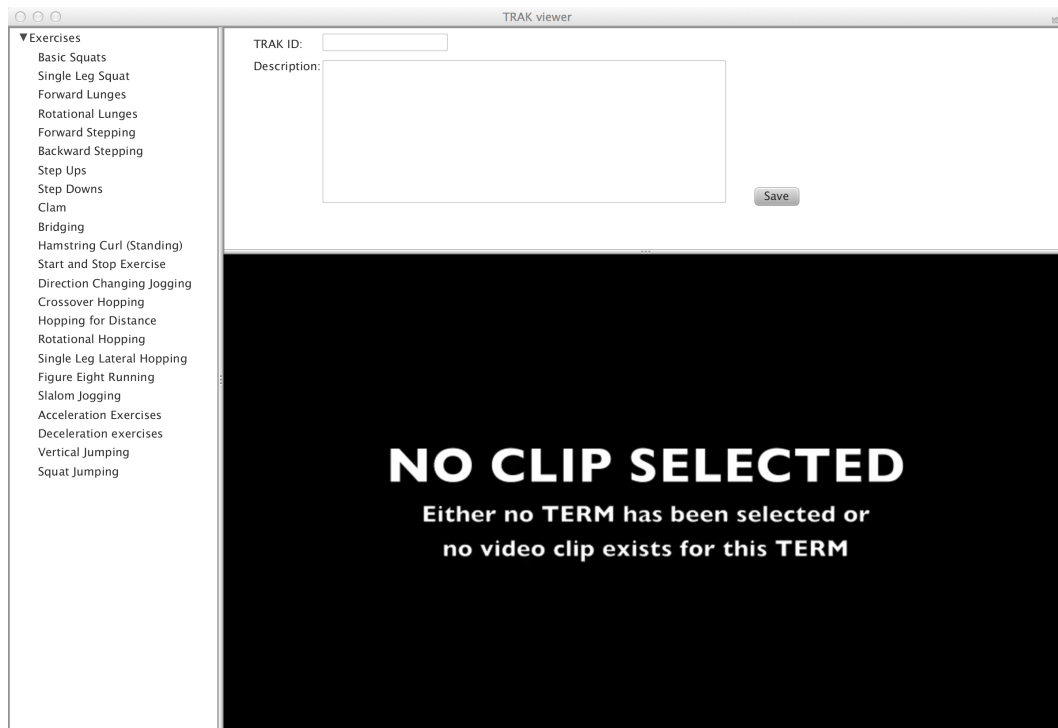
- The ontology tree
- Term descriptions
- The video pane

The rest of this section will be split into three parts. Each part states a requirement of the system and how this requirement has been achieved.

## 8.1 Displaying an ontology tree

The figure below shows the application on load displaying terms from TRAK ontology. The tree populated with TRAK terms can be seen to the left of the image.

FIGURE 8.1: Displaying an ontology tree



The tree shown in the image above loads automatically when the system loads and is displayed for the life of the application. A user is able to interact with the tree by clicking the graphical representation shown in the image above. By doing so the item the user clicks in the tree will be selected and become current.

The graphical representation of the tree and subsequently the tree itself relies on several techniques in order to be displayed. These are:

- The external file

The external file will contain the Terms that the interface needs to display. for this example we have only included the name, TRAK Code, and description of each term. This is because we are dealing with relatively little data. However, another field of parent could be added for further leaf nodes to be parsed in the final tree.

In order to implement this file the current TRAK ontology was used in order to create an SQLite file. SQLite was chosen as it does not use any propriety technologies and gives us a single point where TRAK terms can be stored and updated.

- JDBC

The JDBC is an oracle based project that allows us to connect an SQLite file to our Java project. The JDBC is used to act as a bridge between the stored terms in the SQLite file and the project interface. In order to get the items to populate the tree the code in the figure below is used.

FIGURE 8.2: SQL query retrieving term names

```
// Fill ontology from Database
List terms = new ArrayList();
String sql = "Select Term_Name from TRAK ";
Statement stat;
stat = con.createStatement();
ResultSet res = stat.executeQuery(sql);

// While we havent reached the end of the list keep adding items to
// the root node
while (res.next()){
    String buffer = res.getString(1);
    TreeItem<String> item = new TreeItem<> (buffer);
    rootItem.getChildren().add(item);
}
// Close the connection to the Database
stat.close();
res.close();
```

The above code queries the Database for the Term names to populate the tree with using the query "Select Term\_Name from TRAK" this returns every term name stored in the database one by one the next section of code then set each result it receives as a tree item and adds it to the node. We then close the connection to the Database. There is no need to keep the connection to the database or to retrieve other elements of the records as each term is unique and therefore can be used to query the database at a later stage in the program.

- Array lists

An array list is used in order to hold the objects received from the SQLite file. In JavaFX we are able to specifically define an array list as a TreeItem this gives us access to methods such as .getChildren(). This subsequently allows us to easily define items as leaf nodes of current items to give us our tree representation

- TreeView

TreeView is a feature of JavaFX used for the graphical representation of trees. In order to use tree view we hand it a set of nodes at the root location. this method will then build us a tree representation of the data structure. This is done using the using the line of code shown in the figure bellow:

FIGURE 8.3: SQL query retrieving term names

```
// Add the root node to the tree
TreeView<String> tree = new TreeView<> (rootItem);
```



Once we have got the root node added to the tree view we only need to add it to the stage in order to view it.

- Panes

The Pane allows us to add the previously created `TreeView` to the current window (Stage). This is done by setting the current stage to a pane and then adding the object to the pane. If for example we had `button1` and `button 2` that we wanted to add to the left and right sides of a split pane we would use the following code:

```
SplitPane.getItems().addAll(button1,button2);
```

To make this pane visible to the user all we then have to do is add the pane to the view. This would be done with the following code

```
stage.setScene(SplitPane);
```

This results in the object being viewable by the user as illustrated in figure 8.1.

## 8.2 Displaying and updating term descriptions

The Figures below show the application showing a term description before and after it has been updated.

FIGURE 8.4: Displaying term descriptions

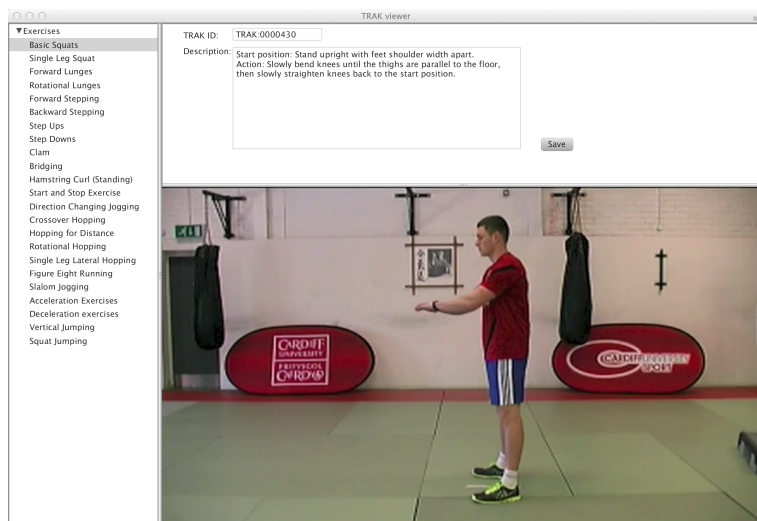
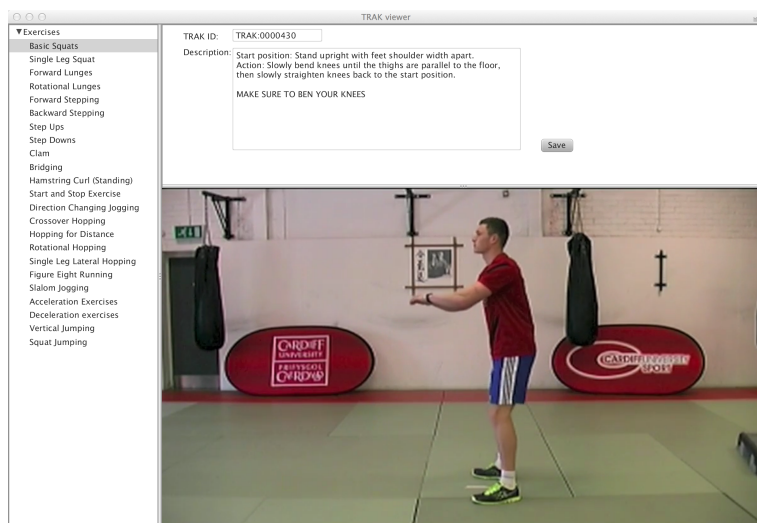


FIGURE 8.5: Updating term descriptions



The description box in the above figures are created using a couple of techniques to select the data for the currently selected term from the database and submit any update data to the database. These techniques are:

- Action listeners

Two action listeners have been implemented in order to achieve this functionality. One on the tree and one on the submit button. The action listener on the tree listens for a change in the list and when one occurs runs a query to the database

using the name of the currently selected item and sets the the description field based on the result. This is illustrated in the pseudo code in the figure below:

FIGURE 8.6: Updating term descriptions

```

1  Send : The selected tree node
2  Recieve : The description corresponding to the selected node
3
4  String current_name = tree.getSelectedNode().getName();
5
6  String query = "Select TRAK_ID, description from TRAK where TRAK_Name = current_name"
7
8  result = query.execute.response;
9
10 track_idbox.setText(result.description);
11 descriptionbox.setText(result.description);

```

The second Action listener invokes an update command to be sent to the database that uses the `getText()` method to submit any updates the user makes to the description.

- SQL queries SQL queries are used for both setting the text of the description box as shown in the example above and for updating the database to reflect the current description. The figure below shows the update text query on the submit buttons action listener:

FIGURE 8.7: Update term descriptions

```

// Action listener for the save button
saveButton.setOnAction(new EventHandler<ActionEvent>() {

    @Override public void handle(ActionEvent e) {

        // Get the user edited description and update the term in the
        // database
        currdesc = tDes.getText();
        input = trAKfeild.getText();
        String sql = "UPDATE TRAK SET Description='"+currdesc+"' WHERE TRAK_Code='"+input+"'";
        try {
            statDescUpdate = con.createStatement();
            statDescUpdate.executeUpdate(sql);
        } catch (SQLException ex) {
            Logger.getLogger(OntologyViewer.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
});

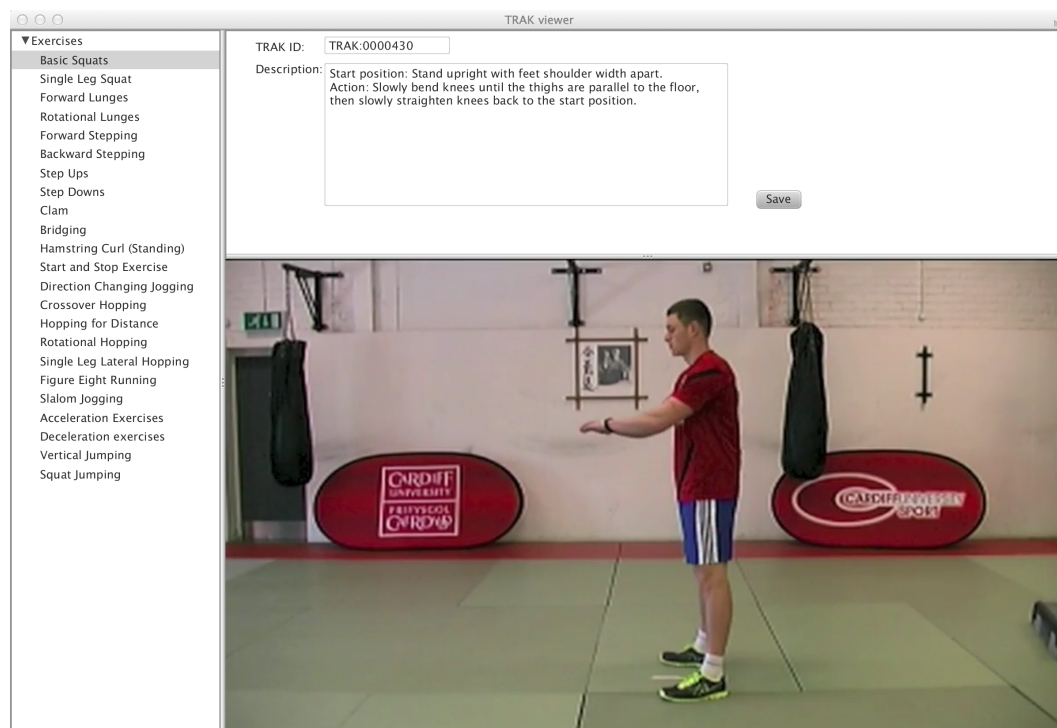
```

The code above takes the text in the description box and uses the TRAK code of the currently selected term in order to perform an update command on the Database. the TRAK code is used as it will always be unique and therefore we can ensure that we are updating the correct value.

### 8.3 Playing a video based on the current tree item

The figure below shows a video being played in the active window.

FIGURE 8.8: Application Playing a video



The video pane in the above image is reliant on several techniques within the application. The video itself is a Java FX MediaView that uses the following techniques in order to run:

- Default

The Java FX media player needs a video to be displayed by default in order to set the stage to the right dimensions. The default file has the same Dimensions as the videos being used to demonstrate the exercises and will also double as a error prevention method by being loaded when there is no file to load.

- TRAK code to URI

As we have to obtain the TRAK code in order to populate the TRAK\_ID box we are able to pass the same code into a method that translates the TRAK code to a URL that will be able to be string matched to any matching video in the projects videos folder. This is illustrated in the code below:

FIGURE 8.9: TRAK code to URI

```
// Replace the semicolon with an underscore to be able to search for the file
input = input.replace(":", "_");

// Sets the file path to represent the correct path
File video = new File("videos/" + input + ".mp4" );

// Check to see if the video exists. if it does not exist set the file back to
// default
if(!video.exists()){
    video = new File("videos/" + "DEFAULT" + ".mp4" );
}

// Convert the file path to one readable by the video player
vidfile = video.toURI().toASCIIString();

// Create a video player based on the passed file
Media media = new Media(vidfile);
player = new MediaPlayer(media);
view = new MediaView(player);
player.play();
player.setAutoPlay(true);
```

The above code takes the TRAK ID of the currently selected input and firstly replaces the semicolon with an underscore. This is done for file compatibility reasons as mentioned in earlier sections. The code then prefixes videos/ to the beginning of the TRAK code and appends .mp4 to the end of the TRAK code so for example if you were to pass the string TRAK:0000111 you would get videos/TRAK\_0000111.mp4 as a resulting string. We then check to see if the file exists and if it does this string then goes through the method .toURI() this has been done because we don't necessarily know the file location we only know that it will be in the videos folder. Finally we pass the resulting string to the media player and start the player.

- JavaFX MediaPlayer and MediaView

Both the Java FX methods MediaPlayer and MediaView are needed in order to load and display the files. MediaView allows the video to be attached to a frame much the same as treeView works for the ontology. The media player allows use to play media files that we know the location of all we need to do is pass it an appropriate URI.

- Looping

As the videos are short the video player implements an action listener in order to figure out when the video reaches the end and loops it back to the beginning. The figure below demonstrates how this is done programmatically:

FIGURE 8.10: Video Looping Code

```
// Have the current video loop
player.currentTimeProperty().addListener(new ChangeListener<Duration>() {
    @Override
    public void changed(ObservableValue<? extends Duration> observableValue, Duration duration, Duration current) {
        if(current.toSeconds() > player.getTotalDuration().toSeconds() - 0.1){
            player.seek(Duration.seconds(0));
        }
    }
});
```

The code Illustrated above sets an action listener on the change property of the video. It then, when the video reaches the end uses the `.seek()` method to reset the video to the beginning (0 seconds)

- Memory Management

Each time a user clicks on a term a new video is loaded. Therefore, we need a method to destroy videos that are not currently playing to prevent memory leaks. As we will only ever have one video playing at a time the we can accomplish this by putting the `.dispose()` method within the video constructor with the condition that a counter incremented each time a video is created is grater than one. This has been done to prevent the application trying to destroy an element that doesn't exist.

## 9. Evaluation of the system

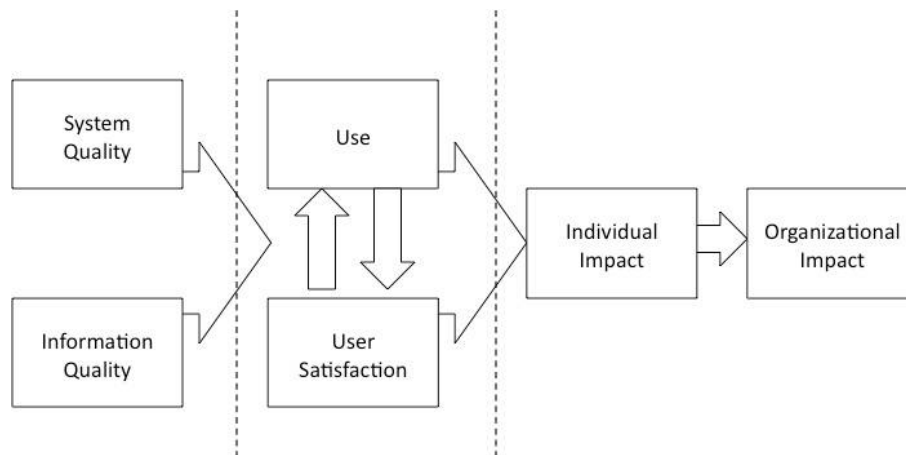
---

In order to state the development of the system was successful and prove it has a positive impact on the target users, we need to evaluate it. For this section we will use the DeLone and McLean model. DeLone and McLean provide a general definition of the Information System (IS) success that covers evaluation of different aspects of the IS. They accomplish that by establishing 6 major categories of IS success, with interdependencies between them these are: system quality, information quality, use, user satisfaction, individual impact and organizational impact. In order to evaluate all these categories, the model suggests considering the IS in terms of three properties: functionality, usability and utility.

- **Functionality**  
Defined as: “What the system does or should do”. Determined by examining organizational requirements.
- **Usability**  
Defined as: “How easy is to use the system for the purpose it was constructed for”. Determined by examining the user satisfaction.
- **Utility**  
Defined as: “How acceptable the system is in terms of doing what is needed. Determined by examining the system’s impact on the individuals and the organization.

The figure below illustrates the way in which the IS categories are interrelated and how they should be evaluated:

FIGURE 9.1: Information System Properties



### Information Systems Success Model (DeLone & McLean 1992)

DeLone and McLean model states in order to test all aspects of the system we need to perform the following types of testing:

- Functionality testing
- Usability testing
- Utility testing

## 9.1 Functionality testing

The functionality of the system will be evaluated by testing whether all stated requirements are accomplished. To accomplish this we will use test cases created from the use cases. This will help make sure that no major features go untested. For the creation of the test cases we will focus on capturing the basic flow of the system. Therefore, the test cases will evaluate the basic flows of the following use cases:

- Display Ontology tree
- Update Term description
- Play video



A template of the structure of the test cases used is illustrated in the figure below:

FIGURE 9.2: Test case template

<b>Test Case ID:</b>		<b>Test Purpose:</b>	
<b>Environment:</b>			
<b>Preconditions:</b>			
<b>Test Path:</b>			
<b>Test Case Steps:</b>			
<b>Step No</b>	<b>Procedure</b>	<b>Expected Response</b>	<b>Pass/Fail</b>
<b>Checker:</b>			

The test cases for the application are available in Appendix 3.

### 9.1.1 Summary of the results

The test cases have been performed on the following three platforms:

- Mac OS X: 10.9.x Mavericks with 1680x1050 display resolution Java version 1.7.0 release 40
- Window 7 service pack 1 with java version 1.7.0 release 25
- Ubuntu 14.0.4 (Linux) with Java version 1.7.0 release 25

All tests performed on each of the platforms have been successful. No functionality problems have been found

## 9.2 Usability testing

An important property of the system is its usability. In order to evaluate the usability of the system we need to measure the user satisfaction of using it. For this purpose the system needs to be tested by sample of users.

### 9.2.1 Number of participants

An important aspect that needs to be considered when giving the system to sample of users is the number of the participants in the usability testing. A study conducted

by Jacob Nielsen is proved that 80% of the usability findings are discovered with 5 participants. Even though, this study is contradictory, it is the only one that gives suggestions on how a usability testing should be performed and has enough research to back up the idea. Therefore, for the purposes of the project we will ask 5 people to test the system.

### **9.2.2 Type of participants**

One of the groups of target users for this project are people with knee problems that need rehabilitation. Therefore, for people were selected for the testing from different age groups(20-60) and knowledge backgrounds (Students, professionals etc.) that could be potential users of the system.

### **9.2.3 Task given to the participants**

Each of the participants needed to perform three main steps:

- Open ontology viewer
- Select a term form the ontology displayed
- Change description given for a term

By not giving guidance to the participants we ensure that each participant will receive a similar experience from the system. Further to that, the testers were not given guidance on how to accomplish each step. In this way we can test the systems ease of use.

### **9.2.4 Performance of usability testing**

The usability testing was conducted in a single room with one supervisor. All participants were able to test the program on a single laptop one after another without the ability to speak to each other. After finishing with the given tasks, the users were asked to fill a questionnaire. The questionnaire consists of 10 questions. For each of them the participant needs to give a rating from 1 to 7 (1 being strongly disagree and 7 being strongly agree). The answers to these questionnaires will be used as feedback in order to find usability problems that need to be solved. The document given to the testers is available in Appendix 4

### 9.2.5 Summary of findings

This section will discuss the feedbacks received from the testers, it will outline problems encountered and give suggestion for future improvements. The questions given to the users were intended to test the following features of the application: Overall reaction, Organisation and Presentation of information, Learning, and Application capabilities. A summary of result will be given for each of these features.

1. Overall reaction

The overall reaction to the application is positive with average score of 6. The participants found the application useful and easy to use.

2. Organisation and Presentation of information

The participants found the organisation of the information on the screen intuitive and easy to follow. However, some of the participants thought that it has not been made clear that the only information they can change relating to a term is its description. The reason for that is a misleading display of term code. Another complaint was that a "help display" option had not been provided.

3. Learning

There were not any complaints about this aspect of the system. The participants found the application simple and easy to learn

4. Application capabilities

All participants stated that the application has the main functionalities needed to perform its main goal. However, it is too rigid and does not provide many advanced options such as: changing the of display of different kinds of information, closing one of the panels (the video one and the term description panel).

### 9.2.5.1 Common problems

Two problems stand out as being the most significant:

1. Lack of help documentation

- Solution

In future a help icon can be implemented that will provide the user with information about the ontology viewer and how it can be used.

2. The application is too rigid

- Solution

The application can be expanded to include functionality that provides more freedom and flexibility to the user. However, at this stage the most important goal was the implementation of the main requirements and secondary functionalities were not considered.

## 9.3 Utility testing

The system will have a beneficial impact for the target users because of the following reasons:

- The system provides an easy to use interface for displaying exercise terms with their descriptions and videos showing how this exercises should be performed  
In this way people who need to make rehabilitation exercises are supported with enough material in order to perform the activities without the help of physiotherapists. As proven in the Social experiment, supplementing the descriptions of the exercises with videos will be beneficial for the users to understand the exercises better.
- Provide a feature of changing the description of exercises  
In this way users and physiotherapists will be able to provide, improve, or extend exercise descriptions where this is necessary. Once the new descriptions are saved, an administrator can go through the suggestions and include the changes if they are appropriate.

## 10. Future work

---

This section will discuss future potential for expanding this project as well as some solutions to having a fully fledged ontology viewer that can support video.

Some of the ideas for the expansion of this project are described in the sections below.

### 10.1 Moving the user base over to protégé

During this project contact with the developers of the Term image plug-in revealed that development of OBO-Edit has more or less come to a complete stop.

This project may in the future be able to breathe a little bit of life into the ageing OBO Edit. However, without continual development to the OBO Edit project it is only a matter of time before the project becomes incompatible with modern operating systems. There are three main methods of addressing the issue:

- Get developers to start contributing to the project.  
This is a viable option but would need a team to investigate the source code of OBO Edit in order to produce adequate developer information for the project.
- Get the user base of OBO Edit to move to protégé.  
With this option we would need to investigate how the current audience could be introduced to protégé and how much of a learning curve would be involved with this. It is feasible that the academic and professional users of the TRAK ontology would be able to handle the switch. However, as this project aims to deal with patients who are using the ontology to aid the recovery process protégé ontology viewer may ultimately be seen as too complex.
- Expand the functionalities of the ontology viewer created for this project to create a rival ontology viewer comparable to OBO Edit.  
This approach a key advantage of this approach is that all the documentation needed to start expanding this project can be found in this project. However, one of the issues seen with this approach is that the development time needed to bring

this project up to scratch with projects like OBO Edit can not really be justified as its functionality already exists within other projects.

## **10.2 Narration on videos**

A small contribution that could be made to the project would be to have the videos narrated. This could be done by editing the videos to match narration with slowdown and looping techniques being used in order clarify. If this addition were to be made volume controls and playback buttons (play / pause) would have to be added to the video player for users who wish to read the description without the video.

## **10.3 Implementing 3D models**

As this project is aimed at making complex information more universally understandable a natural progression would be for this project is for the support of 3D models. This will allow for further detail to be given to the data and can be seen as a natural expansion of this project.

## **10.4 On-line application**

Currently there is an ongoing project on the Facebook social network. This project aims to bring the functionality of the TRAK ontology to the web in a easy to use application. As mentioned in an earlier section this project is looking to implement videos in much the same way as this project. As the videos used in this project are of a high standard there is potential for their use with this project.’

## **10.5 Mobile application**

Feedback from the project testers revealed to me that they would be interested in having the functionality of this project in a mobile application. As the project is written in the Java language the android platform is an obvious choice. Although a new interface would need to be developed in order for the project to be optimised for touch.

## 10.6 Language packs for TRAK

Furthering the feeling that this project aimed at bringing clarity of information to the descriptions within TRAK it is understandable that not all users of the ontology will use English as there main language. Therefore the ability for localised versions of the TRAK ontology could aid users with the digestion of its content.

## 11. Conclusion

---

The project has two main aims. The first aim of the project was to conduct a social experiment that proves the importance of videos in knee rehabilitation conditions. For the purposes of this aim, videos of knee rehabilitation exercises were made. The social experiment involved the participation of physiotherapists as well as a large number of volunteers who needed to perform exercises after being presented with description, image or a video of the exercise. After that each of them needed to give a rating of each of the supporting materials given. This social experiment can be of help because of the following reasons:

- Prove the importance of the creation of software that facilitates a new way for physiotherapy rehabilitation by providing videos associated with rehabilitation exercises.

- Provides high quality videos that can be used in other projects

The main achievements regarding this part of the project are:

- Creation of high quality videos related to knee rehabilitation
- Conduction of survey enclosing a large enough set of people in order to get feasible results
- Summarising and analysing the results form the experiment

The second aim of the project was to create a software product that provides the functionality of displaying videos associated with exercise terms available in TRAK ontology. It is an ontology viewer that helps to view and select exercises, display description as well as videos associated with them. This part of the project can be of a tremendous help to people with knee problems that need to perform exercises as part of their rehabilitation. Further to that, by facilitating the self-rehabilitation of patients, this project promotes a lesser burden on physiotherapy domain. The main achievements regarding this part of the project are:

- Providing an ontology viewer with the ability to display terms from TRAK ontology



- Provide the ability for the user to select exercise terms from TRAK ontology, view and update their description as well as to play video associated with the selected exercise term

The project has been successful as all objectives have been met. Even though the direction of the project has been changed and instead of implementing a plug-in, a separate ontology viewer has been created, the main goal of providing software that display videos with accompanying exercise terms have been accomplished. Further to that, a profound research into the problem of creating plug-ins for OBO -Edit has been done. In this way a lot of existing problems were outlined and propositions for how they should be solved were given. This can help for future projects based on the creation of plug-ins for OBO-Edit.

## 12. Reflection on learning

---

During the development of this project there were two main areas where problems that if left unchecked could have been detrimental to the project. In order for the successful completion of this project these issues needed to be overcome. These areas can be identified as:

- Project management problems
- Project development problems

In the following section I will describe each problem that I have encountered within each group, how these problems arose, how they were overcome and how I will aim to avoid these problems in the future.

### 12.1 Project management problems

The section will deal with issues that arose connected to project management.

- Time management This project had us working to a project plan that was submitted as part of the initial report. As I had not undertaken a project of this level before and we were only given one week to complete this report I found that I misjudged some of the time scales for the project. I also ran into the issue that some of my tasks greatly relied on outside resources that were beyond my control. Therefore, some of the aspects of the development of the system ended up being delayed until resources became available to me. A prime example of this was working with outside testers in order to have corrections to the application tested within a timely manner. What ended up happening was some of the tester took feedback forms with them in order to finish completing them. On the surface this seems like an acceptable predicament. However, as people tend to be unreliable this ended up costing my project time.

In order to overcome this issue I ended up juggling my time schedule to reinvest time that was not planned for the project into the project. This meant that I was

able to quite quickly catch up and overcome further issues created by this problem. However, it did leave me stretched with my commitments outside of this project. When thinking of this as it would have happened within a work environment this issue could have been catastrophic to the overall project and could have ended up reducing the overall project quality or even making the project un-fishable.

In the future I would have multiple plans for gathering data that relied purely on outside sources and will have contingency plans for what may happen if key project Data becomes unavailable. This should ensure that if these issues were to arise again in a future project I would not only be prepared for it but it would not end up costing the project any time or quality. Furthermore, by taking this approach it should ensure that all information gathered for any future project should be of a high standard.

Time management also became an issue at the end of the project where smaller tasks had piled up into larger stacks of tasks that all relied on each other. The concern with tasks piling up was that the project might not be able to be finished in the time scale outlined in the initial documentation.

In order to solve this issue I split tasks back into smaller manageable chunks that could be accomplished in the remaining time scale. This was done by giving myself tasks to do on a day-by-day basis. Work for the project for that day was not finished until the list was complete. With this plan I was able to get back on schedule with the initial plan and even finish elements of the project before their scheduled completion date laid out in the initial plan.

In the future to avoid this, instead of having an overarching plan of how it is going to be completed. I'll use a plan that needs to be updated on a weekly or even a daily scale. By doing this I will be able to catch issues that are going to put the project off track at a far earlier stage and before they have been able to significantly alter the plan of the project.

- **Conduction of experiments** When conducting the experiment for part one of this project. I ran into problems at almost every single stage.

The first problem occurred with trying to find a venue. The initial venue that was planned for this experiment was one of the rooms within the Computer science Buildings. However, the room management meant that we were not able to get a room for the length of time that we required or for the uses that we would need them for.

The problem was overcome when my place of work CUSTV offered me the use of their facilities with the proviso that any work that I carried out in their facilities would give them acknowledgement.

In the future problems like this can only be solved on case-by-case basis, as there is no permanent solution to this problem.

The next problem occurred because the planed date of the experiment was scheduled during the Easter holiday period. This meant that many of these initial volunteers that I has been able to get to say that they would take part in the experiment were unable to do so.

This problem was overcome by recruiting more volunteers on the day of the experiment. This ended up being a superior option as we were able to enforce the precursor that volunteers must have a CUSTV induction and therefore the quality of the volunteers was greatly increased.

In the future it would benefit events like this to check with social calendars to make sure that conflicts like this are unlikely to occur.

The final problem that occurred, was that, on the day, the person who was supplying the recording equipment for the project was unavailable. This problem would have left the project without the video supplements that were needed.

I eventually resolved this problem by finding another source that was able to provide me with the recording equipment called for by the project.

The overall solution to this problem would be not to rely too heavily on outside resources and all required equipment should be sourced and secured prior to the day of the event.

- Interacting with other people

During this project there were several times that I needed to contact a member of a team only to find that the project has been discontinued

The instances concerned both the OBO edit team and the Term image plug-in team. The first issue came about while trying to work out how the Term image plug-in worked. I contacted the developer listed on the website who revealed to me that he had no recollection of the project and that it had been unsupported for years. A similar thing happened with the OBO Edit team when contacting them about documentation for plug-in development.

For this project these issues end up expanding the overall goals of my project to include an application that is able to read in a variety of terms. This allowed me to get on with the project aims and not lose time on a lost cause.

In the future I will try to stick to well-documented projects that are in development so that if help is needed I have somewhere to turn. However, if needed I would still take on the challenge of working with an unsupported project if the time scale allowed it.

## 12.2 Project Development problems

This section will deal with problems that arose that were connected to the development process.

- Choosing the correct approach

One of the sections that turned out to take up a lot more time than originally planned was choosing the technologies that would be used for the project.

During this project I solved this problem by fully investigating a number of approaches that I could have used only using the strongest one. This led to a lot of start/ stopping in the project, as it was fairly late on in the project when the technologies being used were finalized.

In future it would be wise for the project to have a more in depth investigation stage where in the technologies that could be used are fully examined and a suitable technology is able to be found before development of any application begins. This will mean that at a ground level we will have all of the technologies to implement the core functionality of the project laid out so that development of project sections that need to be scrapped due to incompatibility do not occur.

- Lack of documentation regarding important problems related to the project

By far the biggest issue that I faced in this project was trying to get my application working within OBO Edit. This problem occurred because of a lack of developer documentation for their plugin system. Only after contacting the development team of OBO Edit was I made aware that the project is unsupported and there were at current no plans of sorting any aspect of the project out.

In the end this problem was overcome by expanding my project goals to create a mini ontology viewer that would be able to read terms and more or less mock up how the system within OBO Edit would have worked. In doing this I did however leave my video code in a compatible state with what was wanted from the initial plug-in for OBO Edit and therefore if developer documentation does become available in the future the current project could be reintroduce to the OBO edit ontology viewer.

In future I could try to avoid this problem by using software with clear documentation and or development support. By doing this I will remove issues with not knowing how the application fits together. Or, what I am meant to call to be able to hook into it. If given enough time however, I would not be discouraged from using an unsupported platform as long as a backup alternative was clearly outlined.

## 12.3 Lessons learned

In this section I will outline the main lessons I learned during the development of this project. I gained a lot of technical as well as soft skills that can be split in the following sections:

- Project management skills

One of my main problems during most of the project I have participated in the time management. Therefore, for the implementation of this project I tried to establish a plan with a set of tasks for each week. This helped me a lot for the successful completion of the project. From this project I learned to be more organised, to establish a plan of work and follow this plan. Further to that, I learned to prioritise the tasks that need to be accomplished and take into account risks that might occur.

- Communication skills

A significant part of my project involved the communication with and reliance on people from different backgrounds. For instance, the social experiment and usability evaluation of the software created. In addition to this, I also needed to contact other developers whose work relied on as a bases for my project. From all these interactions I learned to present a technological problem on different levels. I learned to present an idea in a clear and concise manner.

- Problem solving skills

This project was plagued with potentially project ending problems, during the encounters with problems I learned to be agile with my approach to solving the problem at hand as well as learning that there would always be an approach that we are able to take in order to move forward with a project. This project also led to me expanding my plan to include the development of features that were not in the initial plan. This meant that my problem solving skills were also used in order to reschedule myself to meet a deadline.

- Software development skills

During the development of this project I have learned a lot about prototyping software and the software development life cycle. For the first time I was able to engage myself with the development documentation of a project, an area, that up until now I have had relatively little interaction with. During this project I encountered software tools such as “Netbeans” that aid in the development of applications and make it possible to package and distribute your projects easily.

I also got a taste of what it was like communicating with developers from other projects and how other projects can be used to enhance my own.

## 13. Appendices

---

Below is listed all of the appendices for this document:

Appendix 1: Survey results from the social experiment

Appendix 2: Email from the OBO edit developers

Appendix 3: Test cases

Appendix 4: Application feedback form

Also attached to this as a zip file is the Netbeans project file for the Ontology Viewer created for this project.



## 13. References

---

- [1] BYU,  
Delone and McLean IS success model - IS Theory. [Webpage].  
Available from: [http://istheory.byu.edu/wiki/  
Delone\\_and\\_McLean\\_IS\\_success\\_model](http://istheory.byu.edu/wiki/Delone_and_McLean_IS_success_model), [accessed Tuesday, 6 May 14]
- [2] Button K,  
TRAK ontology: defining standard care.  
.[J Biomed Inform. 2013] - PubMed - NCBI. [Webpage].  
Available from: <http://www.ncbi.nlm.nih.gov/pubmed/23665300>, [accessed Tuesday, 6 May 14]
- [3] Cecilia Quiroga-Clare, Language Ambiguity. [Webpage].  
Available from: <http://www.seasite.niu.edu/trans/articles/Language>[accessed Tuesday, 6 May 14]
- [4] Jakob Nielsen, 2003.  
The magic Number five: The "magic number 5": is it enough for web testing?  
. chi.
- [5] NHS,  
Top 10 gym exercises done incorrectly - Live Well - NHS Choices.[Webpage].  
Available from: [http://www.nhs.uk/Livewell/fitness/Pages/Top-10-gym-exercises-  
done-incorrectly.aspx](http://www.nhs.uk/Livewell/fitness/Pages/Top-10-gym-exercises-done-incorrectly.aspx), [accessed Tuesday, 6 May 14]
- [6] Oracle,  
Trail: JDBC(TM) Database Access (The Java Tutorials). [Webpage].  
Available from: <http://docs.oracle.com/javase/tutorial/jdbc/>, [accessed Tuesday, 6 May 14]
- [7] Oracle,  
Client Technologies: Java Platform, Standard Edition (Java SE) 8  
Release 8. [Webpage].  
Available from: <http://docs.oracle.com/javase/8/javase-clienttechnologies.htm>, [ac-  
cessed Tuesday, 6 May 14]

- [8] SO OPINIONATED,  
Information quality — So Opinionated ... [Webpage].  
Available from: <https://sopinion8ed.wordpress.com/tag/information-quality/>, [accessed Tuesday, 6 May 14]
- [9] Tom Gruber,  
What is an Ontology? [Webpage].  
Available from: <http://www-ksl.stanford.edu/kst/what-is-an-ontology.html>, [accessed Tuesday, 6 May 14]
- [10] WILLIAM h. DeLone and ephraIM r. McLean,  
The DeLone and McLean Model of Information Systems Success: A Ten-Year Update. [Webpage].  
Available from: <http://www.mesharpe.com/MISVirtual/07Delone.pdf>, [accessed Tuesday, 6 May 14]
- [11] oracle.  
JavaFX Developer Home. [Webpage].  
Available from: <http://www.oracle.com/technetwork/java/javase/overview/javafx-overview-2158620.html>, [accessed Tuesday, 6 May 14]