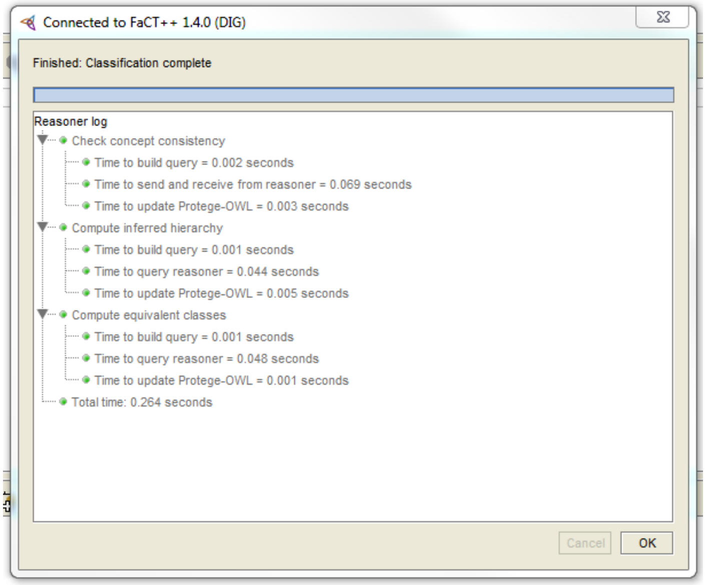
Knowledge Based Systems – Continuous Assignment.

Darren Coutts – 51227666.

Task One – CGS 8.5 – 11.49

1. File downloaded and running in Protégé successfully.

2. By running the FaCT++ server reasoner on the provided ontology. After running this reasoner, it has move the Author and Customer classes to become a sub class of the Person class.



Finished: Classification complete

```
Reasoner log
▼ ● Check concept consistency
    ● Time to build query = 0.002 seconds
    ● Time to send and receive from reasoner = 0.069 seconds
    ● Time to update Protege-OWL = 0.003 seconds
▼ ● Compute inferred hierarchy
    ● Time to build query = 0.001 seconds
    ● Time to query reasoner = 0.044 seconds
    ● Time to update Protege-OWL = 0.005 seconds
▼ ● Compute equivalent classes
    ● Time to build query = 0.001 seconds
    ● Time to query reasoner = 0.048 seconds
    ● Time to update Protege-OWL = 0.001 seconds
  ● Total time: 0.264 seconds
```

Figure 1.1 – FacCT++ Classification wizard.

The classification from FaCT++ is as follows:

```
#Generated by Protege OWL. Export of classification results.
#Sat Nov 08 16:10:25 GMT 2014
Author=Moved from owl\:Thing to Person
Customer=Moved from owl\:Thing to Person
```

Classification can also be found in the file - ***Section 1.2 - Classification Results***

3. I have created some test data to be used in the assignment. In particular I have created:

    - 13 Author instances, each of whom have at last one book stored in the ontology.
    - 21 Book instances, each mapping to one author.
    - 8 Customer instances.

    Despite the assessment paper not calling for me to add in any Purchase instances, I have done so, as this will be required in order to test the ontology correctly, and satisfy the Customer relationship.

    Assignment Submission in files Assignment Part 1.pprj and Assignment Part 1.owl

Task Two – CGS 11.5 – 14.49

1. Using a pen and paper method, I have calculated that there would be 6 recommendations in this ontology based on purchases by other people. I have manually inputted these all into the recommendations class in the ontology.

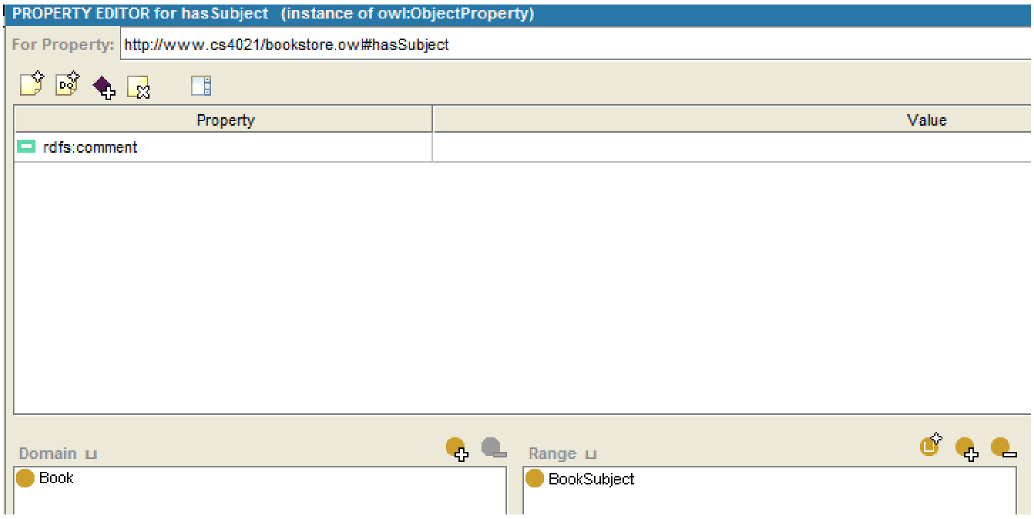
   Therese are as follows:

   - Jeff – Fly Fishing
   - Anna – My Sisters Keeper
   - Ben – Artificial Intelligence: The Basics
   - Ben – Harry Potter and the Chamber of Secrets
   - Buddy – Chess for Kids
   - Jacqui – Hunger Games
   - Anna – Twilight
   - Buddy – Twilight

2. I created a small hierarchy structure underneath the BookSubjects class to order all of the books into their respective categories.

3. From reading the required link, and following method two of the suggestions, I have created a subject instance under each class. For example, in the Football class, I have created an individual of the Football class with name of FootballSubject. I have done this for all levels of the hierarchy, however in reality, I am only going to use the leaf nodes of the tree in this ontology.

   I have then created a property called **hasSubject** which resides on the domain of Book and the range of BookSubject. This can be seen in the following screenshot.



PROPERTY EDITOR for hasSubject (instance of owl:ObjectProperty)

For Property: http://www.cs4021/bookstore.owl#hasSubject

Property	Value
rdfs:comment	

Domain: Book

Range: BookSubject

Figure 2.1 – hasSubject property settings.

I have then gone through the list of books I have in the ontology and assigned them to the most appropriate subject class available. Please note, some of the Fiction books may be in slightly the wrong grouping, however it was to prove how the rules will work in the next step.

Assignment Submission in files Assignment Part 2.pprj and Assignment Part 2.owl

Task Three – CGS 14.5 – 17.49

1. All manual recommendations have been removed.

2. The Jess file that has been provided works by iterating through all of the books in the ontology, looking at who has purchased them, and comparing the other purchases that they have made. This will provide a list of recommendations based around the idea like on the amazon website of "People who bought this, also bought."

3. After changing the vocabulary in the bookstore-rules.txt to match the requirements from the protégé ontology the rules were run in the Jesstab console. The changed rules file can be found in the file: **bookstore-rules-1-protégé.txt**. The output from the Jess console is:

```
Recommendation for Jeff: other people that bought "Moby Dick" also
bought "Fly Fishing"
Recommendation for Anna: other people that bought "Chess for Kids: How
to Play and Win" also bought "My Sisters Keeper"
Recommendation for Buddy: other people that bought "My Sisters Keeper"
also bought "Chess for Kids: How to Play and Win"
Recommendation for Jacqui: other people that bought "My Sisters Keeper"
also bought "The Hunger Games"
Recommendation for Anna: other people that bought "Chess for Kids: How
to Play and Win" also bought "Twilight"
Recommendation for Buddy: other people that bought "My Sisters Keeper"
also bought "Twilight"
Recommendation for Ben: other people that bought "Discrete Mathematics
For Computing" also bought "Artificial Intelligence: The Basics"
Recommendation for Ben: other people that bought "Discrete Mathematics
For Computing" also bought "Harry Potter and the chamber of secrets"
```

4. Having inspected the instances of the Recommendations, and they did successfully match my predictions from the previous exercise.

5. A Jess rules file has been created in order to make recommendations based on books that have also been written by the same author that they have already purchased. This jess rule file is available in the file: **bookstore-rules-2-protege.txt –** this file has fully qualified vocabulary, however for ease of reading, I have also included the file: **bookstore-rules-2.txt** which does not contain the fully qualified vocabulary.

   By running the rules file, it has created 12 more recommendations for the customers. Having checked these manually before and after running the Jess rules, I am happy that the file will correctly recommend books. The output from the jess console is:

```
Recommend book by Stephenie Meyer to Jacqui such as Twilight New Moon
Recommend book by J. K. Rowling to Darren such as Harry Potter and the
Philisopers Stone
Recommend book by Stephenie Meyer to Jacqui such as Twilight Breaking
Dawn
Recommend book by J. K. Rowling to Darren such as Harry Potter and the
Prizoner of Askaban
Recommend book by J. K. Rowling to Darren such as Harry Potter and the
goblet of fire
Recommend book by J. K. Rowling to Darren such as Harry Potter and the
Orderof the Pheonix
Recommend book by J. K. Rowling to Darren such as Harry Potter and the
half bloody prince
Recommend book by J. K. Rowling to Darren such as Harry Potter and the
deathly hallows
Recommend book by Stephenie Meyer to John such as Twilight New Moon
Recommend book by Stephenie Meyer to John such as Twilight Breaking
Dawn
Recommend book by Suzanne Collins to Buddy such as The Hunger Games
Catching Fire
Recommend book by Suzanne Collins to Buddy such as The Hunger Games
Mockingjay
```

6. A second Jess rule file has been created to recommend books to customers based on the subject of their previous book purchase. For example, someone who purchased a book on chess before, has now been recommended all other books in the chess category.

   This jess rule file is available in the file: **bookstore-rules-3-protege.txt –** this file has fully qualified vocabulary, however for ease of reading, I have also included the file: **bookstore-rules-3.txt** which does not contain the fully qualified vocabulary.

   This Jess rule file has created 26 recommendations (based on a blank recommendation class) all of which have been checked and the rule appears to be functioning in the correct manor. The output from the jess console is as follows:

```
Anna suggest The Right Way to Play Chess
Ben suggest Artificial Intelligence: The Basics
Jacqui suggest The Right Way to Play Chess
Jacqui suggest Twilight New Moon
John suggest Match of the Day: 50 Years of Football
Jacqui suggest Twilight Breaking Dawn
Darren suggest Harry Potter and the Philisopers Stone
Darren suggest Harry Potter and the Prizoner of Askaban
Darren suggest Harry Potter and the goblet of fire
Darren suggest Harry Potter and the Orderof the Pheonix
Jacqui suggest The Hunger Games Catching Fire
Darren suggest Harry Potter and the half bloody prince
Darren suggest Harry Potter and the deathly hallows
Jacqui suggest The Hunger Games Mockingjay
Jacqui suggest The Hunger Games
John suggest Twilight New Moon
John suggest Twilight Breaking Dawn
John suggest My Sisters Keeper
Buddy suggest Twilight New Moon
Buddy suggest Twilight Breaking Dawn
John suggest The Hunger Games Catching Fire
Buddy suggest Twilight
John suggest The Hunger Games Mockingjay
John suggest The Hunger Games
Buddy suggest The Hunger Games Catching Fire
Buddy suggest The Hunger Games Mockingjay
```

Task Four – CGS 17.5 – 22.0

1. Assessment Project has been downloaded, and the OWLAPI jar file downloaded from the http://owlapi.sourceforge.net/ website. The version I am using in this project is 4.0.0-distrabution.

2. Complete.

3. By running the protégé ontology through the FaCT++ reasoner, the hasWritten property was asserted on all individuals who had a hasAuthor property linked with it. The reason that this happens is that the hasWritten property is the inverse of the hasAuthor property.

   In protégé 4.3, there does not appear to be a way to export the response of the reasoner, I will however list this bellow.

   The individuals that now have a hasWritten property are as follows:

   - Frank Herbert (http://www.cs4021/bookstoreTest1.owl#Frank_Herbert)
   - Hawking (http://www.cs4021/bookstoreTest1.owl#Hawking)
   - Herman Melville (http://www.cs4021/bookstoreTest1.owl#Herman_Melville)
   - J. R. Hartley (http://www.cs4021/bookstoreTest1.owl#J_R_Hartley)
   - Roger Penrose (http://www.cs4021/bookstoreTest1.owl#Roger_Penrose)
   - Tolkien (http://www.cs4021/bookstoreTest1.owl#Tolkien)

   I am not 100% that this section has been completed correctly. However this does appear to be the most obvious solution.

4. The way that my implementation of the inverse rule functions is as follows.

   - Locate the edges from the HashMap which begin from the given node.
   - Iterate though given edges
   - Use the property.getInvese function to check if there are any inverse properties to the given one.
   - Add a new edge to the edges HashMap to represent the inverse property.
   - Flag *changed* to true.

   This will add a new edge to the HashMap.

   As an intermediate point, I created a script that would provide an output like the following.

```
We have a link from subject
<http://www.cs4021/bookstore.owl#Return_of_the_King> of
<http://www.cs4021/bookstore.owl#hasAuthor> to
[<http://www.cs4021/bookstore.owl#Tolkien>] - So we should add a link
from [<http://www.cs4021/bookstore.owl#Tolkien>] of
[<http://www.cs4021/bookstoreTest1.owl#hasWritten>] to
<http://www.cs4021/bookstore.owl#Return_of_the_King>
```

5. By including the above functionality to the tableau, I can confirm that the same individuals indeed came out as having a hasWritten property, albeit, not in the same order, as a whole this should prove the functionality.

End of Report.pdf

Appendix One – File Directory.

This table has been included to detail which files in the ZIP archive as for which part of the assignment.

Filename	File Type	Notes
Assignment Part 1.owl	OWL	Ontology at the end of part one.
Assignment Part 1.pprj	Protégé Project	Part one protégé project.
Assignment Part 2.owl	OWL	Ontology at the end of part two.
Assignment Part 2.pprj	Protégé Project	Part two protégé file.
Assignment Part 3.owl	OWL	Ontology at the end of part one.
Assignment Part 3.pprj	Protégé Project	Part three protégé file.
Assignment Part 4.owl	OWL	Ontology from part four. Protégé 4.3
bookstore-rules-1-protege.txt	Jess Rules	Fully Qualified Jess Rule 1.
bookstore-rules-2.txt	Jess Rules	Human Readable Jess Rule 2.
bookstore-rules-2-protege.txt	Jess Rules	Fully Qualified Jess Rule 2.
bookstore-rules-3.txt	Jess Rules	Human Readable Jess Rule 3.
bookstore-rules-3-protege.txt	Jess Rules	Fully Qualified Jess Rule 3.
bookstoreTest3.owl	OWL	OWL file for java project.
Java Project.zip	ZIP Archive	Java project in a zip archive.
Report.pdf	PDF	This report
Section 1.2 - Classification Results	TXT	Classification result from FaCT++ in part 1.
Start Point.owl	OWL	Initial Ontology.
Start Point.pprj	Protégé Project	Initial protégé project.