Ministry of Education and Science of the Russian Federation
Peter the Great St. Petersburg State Polytechnic University
Institute of Computer Sciences and Technologies
**Graduate School of Cyber-Physical Systems and Control**

# Course Project Report

Object Identification Knowledge Base
Discipline: Knowledge Engineering
16 May 2017

Student Group: 13541/8

Christopher W. Blake

Professor

Onufriev V.

St. Petersburg
2017

# Contents

# Introduction (and Goal)

Knowledge Engineering is the task of identifying, storing, and providing access to bodies of knowledge. As such, a series of tasks are performed to gain a fundamental understanding of the concepts involved. These tasks involve selection of a topic area, performing interviews about the topic, definition of triples and microdata, production of concept and UML maps, creation of rdf/turtle files, and finally C# programs for easy consumption.

After a fundamental understand of all concepts is obtained, a web interface is developed for serving the ontology of the Object-Identification Knowledge Base. This web interface contains two components, an API for other computers to and applications to consume data and a graphical web portal for exploring the ontology structure and data. Below is the URL address to this web application.

http://objectidentification.azurewebsites.net/

# Tasks

## P1: Working Plan

Task: Develop a plan for the project answering questions such as “What is the topic?”, “What kind of system is best?”, “What kind of information needs to be present?” Below is a list of answers to each question.

1.) **What is the topic of my thesis? (or what is the field/direction of your thesis?)**
Below is the first paragraph from my thesis concept document.

In all systems of life, there is a common theme: the need to collect, learn, and control. From this process, additional capabilities are discovered which lead to higher-level collecting which leads to higher level control. This thesis plans to describe a system that simulates this system of learning to produce knowledge. As such it has the following general objectives.

1. Automatic control system development via input and output correlation.
2. Automatic layered knowledge generation.
3. Automatic distribution of a centralized knowledge database.

2.) **What exact problem am I going to solve?**
To generate knowledge of functionality of a black box device, and then enable it to learn regularly occurring tasks. This adaptive system is intended to replace complicated tasks, normally performed by a human operator.
A task is manually controlled a few times by a human operator, and the system recognizes it as recurring. This regular task can then be performed as a single operation. As such, these auto-identified regular tasks can then be combined by a human operator, and identification can further repeat at the higher level

3.) **Choose a knowledge base system that may be able to be a sub system of my masters thesis project.**

It appears an expert system (or maybe searching system) is the most appropriate, as this system is heavily dependent on analysis of previous usage of the devices.

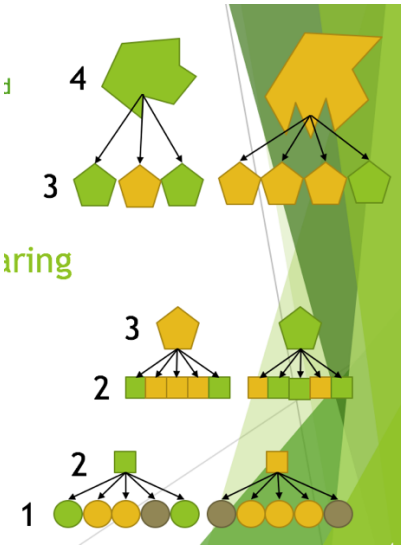
Such a system would use the history to identify regular tasks and possible predict future required tasks. It would allow an operator to write instructions, maybe similar to SQL, to search for available inputs/outputs that were previously performed/recorded. These returned inputs are then sent to the device for operation. Additionally, like SQL systems, additional entries could be manually added to the system, for quicker device learning.

Thoughts about other systems:

a.) Pattern Identification System – to recognize regular inputs/outputs and convert them into formulized tasks
b.) Prediction System – to predict relevance of one task to another. Estimate the likelihood that one task will be requested after the current one is finished. This may be used to suggest new regular functionality to the operator.
c.) Searching System – to allow an operator to find input/output combinations and turn them into a regular task.

4. **What kind of information will need to be stored.**

Both control data for the output device, and response data from the input device. This will be broken into layers. The lowest layer will contain time vs voltage. Additional layers will be referential, according to a pointer type system. Each complex object will point to other less complex objects, or to a time voltage entry.



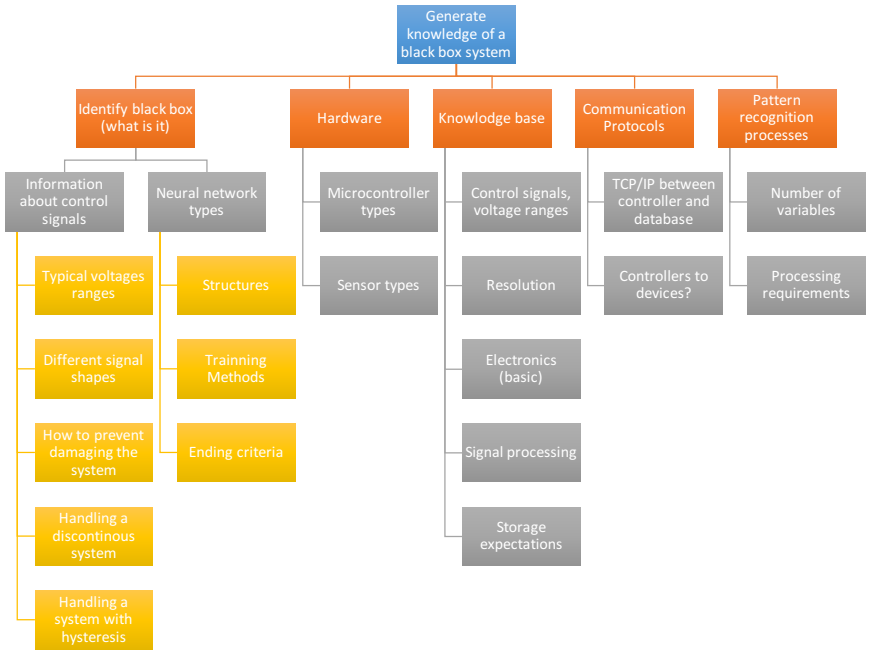

Layer 1 is voltage vs time entries in the database.
Layers 2 is an object referencing a defined pattern of voltage vs time entries.
Layer 3+ is an object referencing a defined pattern o lower level objects.

## P2: Knowledge Extraction (by hand)

Task: An interview-style knowledge assessment is performed with a peer. This peer knows nothing about the project and asks questions to gain enough understanding of how one would perform this project on one's own.

Below is a chart showing the requirements discovered through the interview.

**Blue = Aim**
**Orange = Main Tasks**
**Grey = Need to know to solve the task**
**Yellow = Additional detail**



## P3: Knowledge Extraction (100 Triples)

Task: Focusing on the selected subject, create a list of triples to organize the knowledge.

Below is a table of 47 task relevant triples. An additional 59 triples were performed for practice about separate unrelated topic. Hence, they have been left out.

**Available Relations**

| Common Wording | Property | Description / Example | Property traits |
|---|---|---|---|
| | Transitive | Both words relate to the same meaning. | |
| | Symmetric | a is related to b if and only if b is related to a | |
| | Reflexive | Having an object equal to the subject. | |
| A **is the same as** B (what) | Synonym | Each listed synonym denotes the same as this entry. Pretty/Attractive, Sick/Ill | Symmetric, Reflexive |
| A **is opposite of** B (what) | Antonym | Each listed antonym denotes the opposite of this entry. Up/Down, Dead/Alive | Symmetric |
| A **has a subtype** B (what) | Hypernym | Animal is a hypernym of mammal (mammals are animals), mammal/dog (dogs are mammals) | Transitive |
| A **is a type of** B (what) | Hyponym | Dog is a hyponym of mammal, mammal/animal (dog is a kind of mammal and mammal is a kind of dog) | Transitive |
| A **is a component of** B (what) | Meronym | Bark is a meronym of tree (bark is a part of what makes up a tree), Elbow is a meronym of arm which is a meronym of body | Transitive |
| A **has the component** B (what) | Holonym | Forest is a holonym of tree, a tree is a holonym of bark | Transitive |
| A **is a way to do** B (how) | Troponym | “to trim” and “to slice” are troponyms of “to cut”. “to drown” and “to poison” are troponyms of “to kill” | Transitive |
| A **is a group for both** A and B (what) | Coordinate Term | Shares a hypernym. man and woman share the hypernym human. | Symmetric, Reflexive |
| **Need** A **in order for** B (what for) | | | ? |
| A **is because of** B A **is caused by** B (why) | | | ? |

| | | | |
|---|---|---|---|
| A **has conditions of** B<br>A **if** B<br>(when) | | | ? |
| A **needs** B<br>(how) | | | ? |
| A **occurs on** B (how) | | actions | ? |

**Triples (47)**

| Type | Subject | Property | Object |
|---|---|---|---|
| **Basic Elements** | | | |
| What | ID | is a type of | Integer |
| What | Value | is a type of | Double |
| What | Time | is a type of | Date |
| What | Voltage | has the component | ID |
| What | Voltage | has the component | Value |
| What | ControlVoltage | is a type of | Voltage |
| What | ResponseVoltage | is a type of | Voltage |
| What | EventObject | is a type of | Object |
| What | EventObject | has the component | ID |
| What | EventObject | has the component | DeviceID |
| What | EventObject | has the component | List<EventObject> |
| What | Command | is a type of | EventObject |
| What | Situation | is a type of | Object |
| What | Device | is a type of | Object |
| | | | |
| **Voltage to EventObject Conversion** | | | |
| How | VoltageToEventObject | creates | List<EventObject> |
| What | VoltageToEventObject | is a type of | Process |
| How | VoltageToEventObject | requires | List<ReponseVoltage> |
| How | ResponseVoltage.ID | is linked to | EventObject.ID |
| When | EventObject | does not exist | new EventObject |
| | | | |
| **EventObject to Voltage Conversion** | | | |
| How | EventObjectToVoltage | creates | List<Voltage> |
| What | EventObjectToVoltage | is a type of | Process |
| How | EventObjectToVoltage | requires | List<EventObject> |
| How | EventObject.ID | is linked to | ResponseVoltage.ID |
| | | | |
| **Pattern Detection Process (new EventObject)** | | | |
| How | PatternDetection | creates | EventObject |
| What | PatternDetection | is a type of | Process |
| How | PatternDetection | requires | NN Recognition |
| How | NN Recognition | requires | List<EventObject> |
| How | NN Recognition | requires | Pattern Schemes |
| | | | |
| **Common Command Identification** | | | |
| How | CommandIdentification | creates | Command |

| | | | |
|---|---|---|---|
| What | CommandIdentification | is a type of | Process |
| How | CommandIdentification | requires | List<EventObject> |
| How | CommandIdentification | counts | EventObject instances |
| When | Count(EventObject) | is greater than | Threshold |
| | | | |
| **Control Signal Creation** | | | |
| How | ControlSignalCreation | creates | List<ControlVoltage> |
| What | ControlSignalCreation | is a type of | Process |
| How | ControlSignalCreation | requires | Command |
| How | Command | is convert by | EventObjectToVoltage |
| | | | |
| **Labeling System** | | | |
| What | Label | is a type of | Object |
| What | Label | contains | string |
| What | Label | contains | EventObject |
| What | CommandLabel | is a type of | Label |
| What | EventObjectLabel | is a type of | LabeL |
| | | | |
| **Device Identification** | | | |
| What | DeviceID | is a type of | ID |
| What | Device | contains | DeviceID |
| What | Device | contains | Label |
| | | | |

## P4: DBPedia Mappings

Task: Locate a non-existing mapping in DBPedia and create it

Below is the mapping that was created. It is for the German language and is about an info box for a glacier.

mapping de:Infobox_Gletscher

```
{{TemplateMapping | mapToClass = Glacier
| mappings =
        <!-- {{ PropertyMapping | templateProperty = BESONDERHEITEN | ontologyProperty = }} -->
        <!-- {{ PropertyMapping | templateProperty = BILD | ontologyProperty = }} -->
        <!-- {{ PropertyMapping | templateProperty = BILDBESCHREIBUNG | ontologyProperty = description
}} -->
        {{ PropertyMapping | templateProperty = BREITE | ontologyProperty = width }}
        {{ PropertyMapping | templateProperty = DICKE | ontologyProperty = thickness }}
        <!-- {{ PropertyMapping | templateProperty = ENTWÄSSERUNG | ontologyProperty = drainage }} -->
        {{ PropertyMapping | templateProperty = FLÄCHE | ontologyProperty = Area}}
        {{ PropertyMapping | templateProperty = GEBIRGE | ontologyProperty = MountainRange}}
        {{ PropertyMapping | templateProperty = LAGE | ontologyProperty = location }}
        {{ PropertyMapping | templateProperty = LÄNGE | ontologyProperty = length }}
        {{ PropertyMapping | templateProperty = NAME | ontologyProperty = foaf:name }}
        {{ PropertyMapping | templateProperty = NAME-ZUSATZ | ontologyProperty = foaf:name }}
        {{ PropertyMapping | templateProperty = REGION-ISO | ontologyProperty = region }}
        {{ PropertyMapping | templateProperty = TYP | ontologyProperty = type }}

        {{GeocoordinatesMapping | BREITENGRAD = lat_d | LÄNGENGRAD = long_d }}
}}
```

Links to real entries on DBPedia:
http://mappings.dbpedia.org/server/templatestatistics/de/?template=Infobox_Gletscher
https://de.wikipedia.org/wiki/Vorlage:Infobox_Gletscher

## P5: Page Microdata

Task: Create a manual HTML page that includes micro tags for 5 instances based on Schema.org.

### Page Description

1. Book: Pattern Recognition and Machine Learning (Information Science and Statistics)
2. Person: Albert Einstein Biography
3. Event: Product Innovators Panel: Knowing Your Custmer

### Structure Data Testing Tool Results

Book 0 ERRORS 0 WARNINGS

| | |
|---|---|
| **@type** | Book |
| **name** | Pattern Recognition and Machine Learning (Information Science and Statistics) |
| **thumbnailUrl** | https://images-na.ssl-images-amazon.com/images/I/61FKyOeM7KL._SX368_BO1,204,203,200_.jpg |
| **description** | This book describes the important ideas in a variety of fields such as medicine, biology, finance, and marketing in a common conceptual framework. While the approach is statistical, the emphasis is on concepts rather than mathematics. Many examples are given, with a liberal use mathematics. Many examples are given, with a liberal use of colour graphics. It is a valuable resource for statisticians and anyone interested in data mining in science or industry. The book's coverage is broad, from supervised learning (prediction) to unsupervised learning. The many topics include neural networks, support vector machines, classification trees and boosting---the first comprehensive treatment of this topic in any book. This major new edition features many topics not covered in the original, including graphical models, random forests, ensemble methods, least angle regression & path algorithms for the lasso, non-negative matrix factorisation, and spectral clustering. There is also a chapter on methods for "wide" data (p bigger than n), including multiple testing and false discovery rates. |
| **numberOfPages** | 745 |
| **bookEdition** | 2nd |
| **copyrightYear** | 2009 |
| **inLanguage** | English |
| **isbn** | 0387848576 |
| **isbn** | 978-0387848570 |

EducationEvent — 0 ERRORS — 3 WARNINGS

| | |
|---|---|
| **@type** | EducationEvent |
| **url** | https://www.meetup.com/Team-Austin-Networking/events/237247020/ |
| **name** | Product Innovators Panel: Knowing Your Customer |
| **startDate** | 2017-02-22T06:00:00 |
| **endDate** | 2017-02-22T06:00:00 |
| **description** | Hear from some of the city's best product innovators about their journeys into product management and entrepreneurship. You'll have a chance to ask these experts your burning questions on creating, maintaining, and growing a product based on your customers' needs. Takeaways - About Product Management and how to get into it. - Identifying ideas worth pursuing and dedicating resources to. - Detecting customer pain points and running customer interviews without bias. - Evaluating which product metrics to track and which to ignore. |
| **location** | |
| **@type** | PostalAddress |
| **name** | GA Austin, WeWork Congress |
| **streetAddress** | 600 Congress Avenue, 14th Floor |
| **addressLocality** | Austin |
| **addressRegion** | TX |
| **contributor** | |
| **@type** | Person |
| **givenName** | Dan |
| **familyName** | Corbin |
| **jobTitle** | Product Manager, Return Path |
| **contributor** | |
| **@type** | Person |
| **givenName** | Rob |
| **familyName** | Feinstein |
| **jobTitle** | Product Management Executive |
| **contributor** | |
| **@type** | Person |
| **givenName** | Rick |
| **familyName** | Orr |
| **jobTitle** | Founder RealSavvy, TabbedOut |
| **contributor** | |
| **@type** | Person |
| **givenName** | Josh |
| **familyName** | Lipton |
| **jobTitle** | Sr. Director, Product & Growth, Favor |
| ⚠ **eventStatus** | The *eventStatus* field is recommended. Please provide a value if available. |
| ⚠ **image** | The *image* field is recommended. Please provide a value if available. |
| ⚠ **offers** | The *offers* field is recommended. Please provide a value if available. |

Person 0 ERRORS 0 WARNINGS

| | |
|---|---|
| **@type** | Person |
| **image** | http://www.nobelprize.org/nobel_prizes/physics/laureates/1921/einstein.jpg |
| **givenName** | Albert |
| **familyName** | Einstein |
| **birthDate** | 1879-03-14 |
| **jobTitle** | teacher |
| **jobTitle** | Director |
| **jobTitle** | Professor |
| **deathDate** | 1955-04-18 |
| **birthPlace** | |
| **@type** | Place |
| **name** | WÃ¼rttemberg, Germany |
| **nationality** | |
| **@type** | Country |
| **name** | Swiss |
| **worksFor** | |
| **@type** | Organization |
| **name** | Swiss Patent Office |
| **worksFor** | |
| **@type** | Organization |
| **name** | Kaiser Wilhelm Physical Institute |
| **worksFor** | |
| **@type** | Organization |
| **name** | University of Berlin |
| **nationality** | |
| **@type** | Country |
| **name** | German |
| **nationality** | |
| **@type** | Country |
| **name** | United States |
| **deathPlace** | |
| **@type** | Place |
| **name** | Princeton, New Jersey |

## HTML Code

```
<html>
<body>

<div itemscope itemtype="http://schema.org/Book">

  <h2><b><span itemprop="name">Pattern Recognition and Machine Learning (Information
Science and Statistics)</span></b></h2>
  <table>
      <tr>
        <td style="vertical-align: top; text-align: left;">
          <img height=400 style="margin-right: 15px;" itemprop="thumbnailUrl"
src="https://images-na.ssl-images-
amazon.com/images/I/61FKyOeM7KL._SX368_BO1,204,203,200_.jpg"/>
        </td>
        <td>
          <h3><b>Description</b></h3>
          <span itemprop="description">
          <p>This book describes the important ideas in a variety of fields such as
medicine, biology, finance, and marketing in a common conceptual framework. While the
approach is statistical, the emphasis is on concepts rather than mathematics. Many examples
are given, with a liberal use of colour graphics. It is a valuable resource for
statisticians and anyone interested in data mining in science or industry. The book's
coverage is broad, from supervised learning (prediction) to unsupervised learning. The many
topics include neural networks, support vector machines, classification trees and boosting-
--the first comprehensive treatment of this topic in any book.</p>
          <p>This major new edition features many topics not covered in the original,
including graphical models, random forests, ensemble methods, least angle regression &amp;
path algorithms for the lasso, non-negative matrix factorisation, and spectral clustering.
There is also a chapter on methods for "wide'' data (p bigger than n), including multiple
testing and false discovery rates.</p>
          </span>
          <h3><b>Product Details</b></h3>
          <ul>
            <li><b>Series:</b> Springer Series in Statistics<br></li>
            <li><b>Hardcover:</b> <span itemprop="numberOfPages">745</span> pages</li>
            <li><b>Publisher:</b> Springer; <span itemprop="bookEdition">2nd</span> ed.
<span itemprop="copyrightYear">2009</span>, Corr. 9th printing 2017 edition (April 21,
2017)</li>
            <li><b>Language:</b> <span itemprop="inLanguage">English</span></li>
            <li><b>ISBN-10:</b> <span itemprop="isbn">0387848576</span></li>
            <li><b>ISBN-13:</b> <span itemprop="isbn">978-0387848570</span></li>
            <li><b>Product Dimensions:</b> 6.2 x 1.5 x 9.2 inches</li>
            <li><b>Shipping Weight:</b> 3.3 poundsÂ </li>
          </ul>
        </td>
      </tr>
  </table>
</div>


<hr/>


<div itemscope itemtype="http://schema.org/Person">
  (<a href="http://www.nobelprize.org/nobel_prizes/physics/laureates/1921/einstein-
bio.html">original page</a>)

  <h1><b>Albert Einstein - Biographical</b></h1>
  <p>
    <img style="float: left; margin: 0 10px 10px 0;" width=150 itemprop="image"
src="http://www.nobelprize.org/nobel_prizes/physics/laureates/1921/einstein.jpg"/>
    <b><span itemprop="givenName">Albert</span> <span
itemprop="familyName">Einstein</span></b> was born at Ulm, in <span
itemprop="birthPlace">WÃ¼rttemberg, Germany</span>, on <span itemprop="birthDate">March 14,
1879</span>.
```

```
    Six weeks later the family moved to Munich, where he later on began his schooling at
the Luitpold Gymnasium.
    Later, they moved to Italy and Albert continued his education at Aarau, Switzerland and
in 1896 he entered the Swiss Federal Polytechnic School in Zurich to be trained as a <span
itemprop="jobTitle">teacher</span> in physics and mathematics.
    In 1901, the year he gained his diploma, he acquired <span
itemprop="nationality">Swiss</span> citizenship and, as he was unable to find a teaching
post, he accepted a position as technical assistant in the <span itemprop="worksFor">Swiss
Patent Office</span>.
    In 1905 he obtained his doctor's degree.
    <br>
    <br>
    During his stay at the Patent Office, and in his spare time, he produced much of his
remarkable work and in 1908 he was appointed Privatdozent in Berne.
    In 1909 he became Professor Extraordinary at Zurich, in 1911 Professor of Theoretical
Physics at Prague, returning to Zurich in the following year to fill a similar post.
    In 1914 he was appointed <span itemprop="jobTitle">Director</span> of the <span
itemprop="worksFor">Kaiser Wilhelm Physical Institute</span> and <span
itemprop="jobTitle">Professor</span> in the <span itemprop="worksFor">University of
Berlin</span>.
    He became a <span itemprop="nationality">German</span> citizen in 1914 and remained in
Berlin until 1933 when he renounced his citizenship for political reasons and emigrated to
America to take the position of Professor of Theoretical Physics at Princeton*.
    He became a <span itemprop="nationality">United States</span> citizen in 1940 and
retired from his post in 1945.
    <br>
    <br>
    After World War II, Einstein was a leading figure in the World Government Movement, he
was offered the Presidency of the State of Israel, which he declined, and he collaborated
with Dr.
    Chaim Weizmann in establishing the Hebrew University of Jerusalem.
    <br>
    <br>
    Einstein always appeared to have a clear view of the problems of physics and the
determination to solve them.
    He had a strategy of his own and was able to visualize the main stages on the way to
his goal. He regarded his major achievements as mere stepping-stones for the next advance.
    <br>
    <br>
    At the start of his scientific work, Einstein realized the inadequacies of Newtonian
mechanics and his special theory of relativity stemmed from an attempt to reconcile the
laws of mechanics with the laws of the electromagnetic field.
    He dealt with classical problems of statistical mechanics and problems in which they
were merged with quantum theory: this led to an explanation of the Brownian movement of
molecules.
    He investigated the thermal properties of light with a low radiation density and his
observations laid the foundation of the photon theory of light.
    <br>
    <br>
    In his early days in Berlin, Einstein postulated that the correct interpretation of the
special theory of relativity must also furnish a theory of gravitation and in 1916 he
published his paper on the general theory of relativity.
    During this time he also contributed to the problems of the theory of radiation and
statistical mechanics.
    <br>
    <br>
    In the 1920s, Einstein embarked on the construction of unified field theories, although
he continued to work on the probabilistic interpretation of quantum theory, and he
persevered with this work in America.
    He contributed to statistical mechanics by his development of the quantum theory of a
monatomic gas and he has also accomplished valuable work in connection with atomic
transition probabilities and relativistic cosmology.
    <br>
    <br>
    After his retirement he continued to work towards the unification of the basic concepts
of physics, taking the opposite approach, geometrisation, to the majority of physicists.
    <br>
```

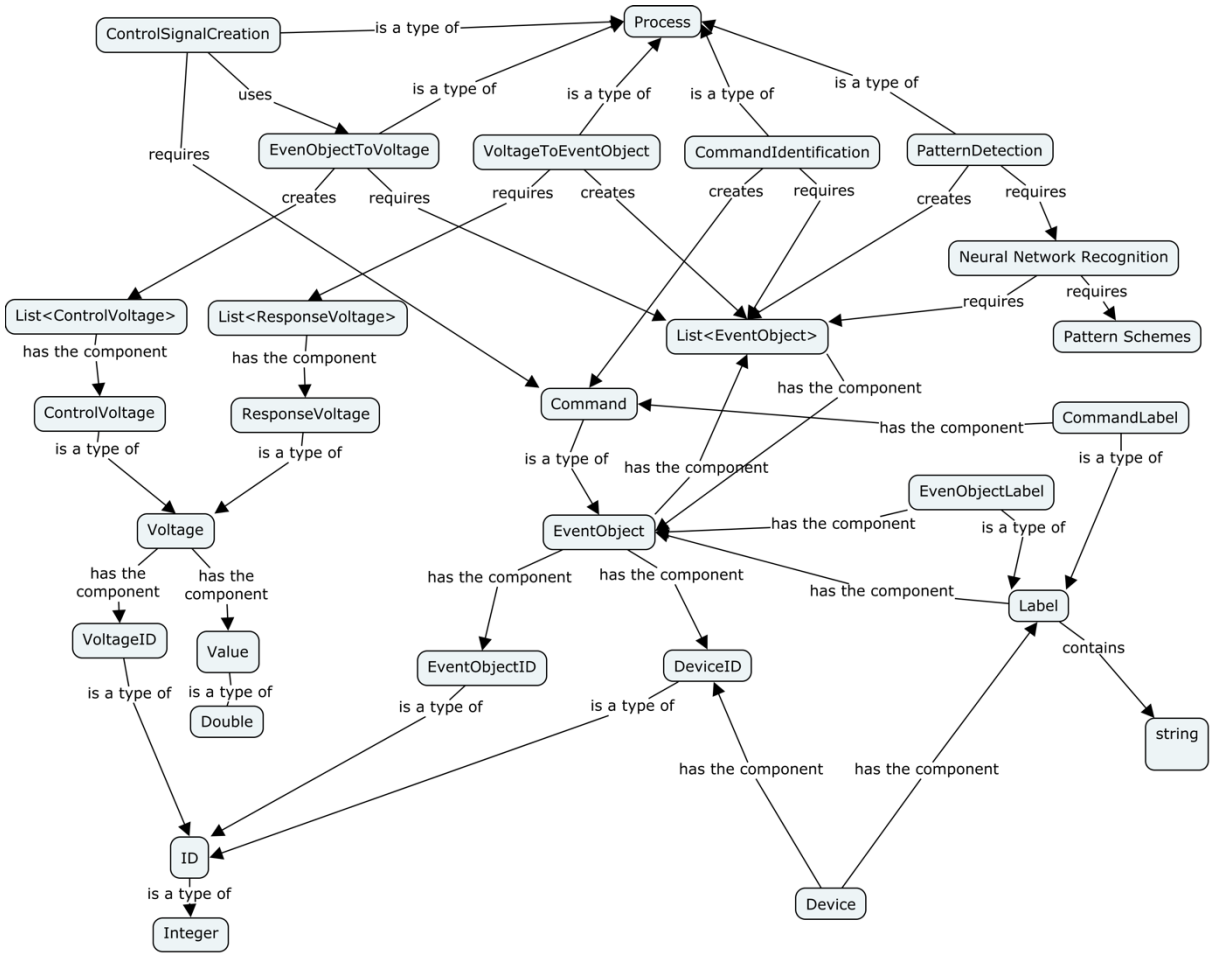
```
    <br>
    Einstein's researches are, of course, well chronicled and his more important works
include <i>Special Theory of Relativity</i> (1905), <i>Relativity</i> (English
translations, 1920 and 1950), <i>General Theory of Relativity</i> (1916), <i>Investigations
on Theory of Brownian Movement</i> (1926), and <i>The Evolution of Physics</i> (1938).
    Among his non-scientific works, <i>About Zionism</i> (1930), <i>Why War?</i> (1933),
<i>My Philosophy</i> (1934), and <i>Out of My Later Years</i> (1950) are perhaps the most
important.
    <br>
    <br>
    Albert Einstein received honorary doctorate degrees in science, medicine and philosophy
from many European and American universities.
    During the 1920's he lectured in Europe, America and the Far East, and he was awarded
Fellowships or Memberships of all the leading scientific academies throughout the world.
    He gained numerous awards in recognition of his work, including the Copley Medal of the
Royal Society of London in 1925, and the Franklin Medal of the Franklin Institute in 1935.
    <br>
    <br>
    Einstein's gifts inevitably resulted in his dwelling much in intellectual solitude and,
for relaxation, music played an important part in his life.
    He married Mileva Maric in 1903 and they had a daughter and two sons;
    their marriage was dissolved in 1919 and in the same year he married his cousin, Elsa
LÃ¶wenthal, who died in 1936.
    He died on <span itemprop="deathDate">April 18, 1955</span> at <span
itemprop="deathPlace">Princeton, New Jersey</span>.
  </p>
</div>


<hr/>


<div itemscope itemtype="http://schema.org/EducationEvent">
  (<a itemprop="url" href="https://www.meetup.com/Team-Austin-
Networking/events/237247020/">original page</a>)
  <h1><b><span itemprop="name">Product Innovators Panel: Knowing Your
Customer</span></b></h1>
  <h2><b><span itemprop="startDate"><span itemprop="endDate">February 22, 2017 @ 06:00
PM</span></span></b></h2>
  <h2><b>RSVP's: 160</b></h2>
  <div itemscope itemtype="http://schema.org/PostalAddress" itemprop="location">
  <h3><b><span itemprop="name">GA Austin, WeWork Congress</span>, <span
itemprop="streetAddress">600 Congress Avenue, 14th Floor</span> <span
itemprop="addressLocality">Austin</span>, <span itemprop="addressRegion">TX</span></b></h3>
  </div>

  <p>Ever wonder how organizations design and produce the innovative products we use daily?
How do they know what customers want?
  How do they communicate the requirements to develop the product?</p>
  <p>
  <b>Find Out From The Experts.</b><br/><br/>
  <b>Speaker Series Featuring:</b><br/>
  <span itemscope itemtype="http://schema.org/Person" itemprop="contributor">
    <b><span itemprop="givenName">Dan</span> <span itemprop="familyName">Corbin</span>:</b>
<span itemprop="jobTitle">Product Manager, Return Path</span>
  </span>
  <br/>
  <span itemscope itemtype="http://schema.org/Person" itemprop="contributor">
    <b><span itemprop="givenName">Rob</span> <span
itemprop="familyName">Feinstein</span>:</b> <span itemprop="jobTitle">Product Management
Executive</span>
  </span>
  <br/>

  <span itemscope itemtype="http://schema.org/Person" itemprop="contributor">
    <b><span itemprop="givenName">Rick</span> <span itemprop="familyName">Orr</span>:</b>
<span itemprop="jobTitle">Founder RealSavvy, TabbedOut</span>
```

```
  </span>
  <br/>

  <span itemscope itemtype="http://schema.org/Person" itemprop="contributor">
    <b><span itemprop="givenName">Josh</span> <span
itemprop="familyName">Lipton</span>:</b> <span itemprop="jobTitle">Sr. Director, Product
&amp;amp; Growth, Favor</span>
  </span>
  <br/>
  </p>
  <p>
  ----------------------------------------------------------------</p>
  <span itemprop="description">
  <p>
  Hear from some of the city's best product innovators about their journeys into product
management and entrepreneurship.
  You'll have a chance to ask these experts your burning questions on creating,
maintaining, and growing a product based on your customers' needs.
  </p>

  <p>
  <b>Takeaways</b><br/>
  - About Product Management and how to get into it.<br/>
  - Identifying ideas worth pursuing and dedicating resources to.<br/>
  - Detecting customer pain points and running customer interviews without bias.<br/>
  - Evaluating which product metrics to track and which to ignore.<br/>
  </p>
  </span>
</div>

</body>
</html>
```

## P6: Concept Mapping

Task: Create a node-based concept map of the project concepts.

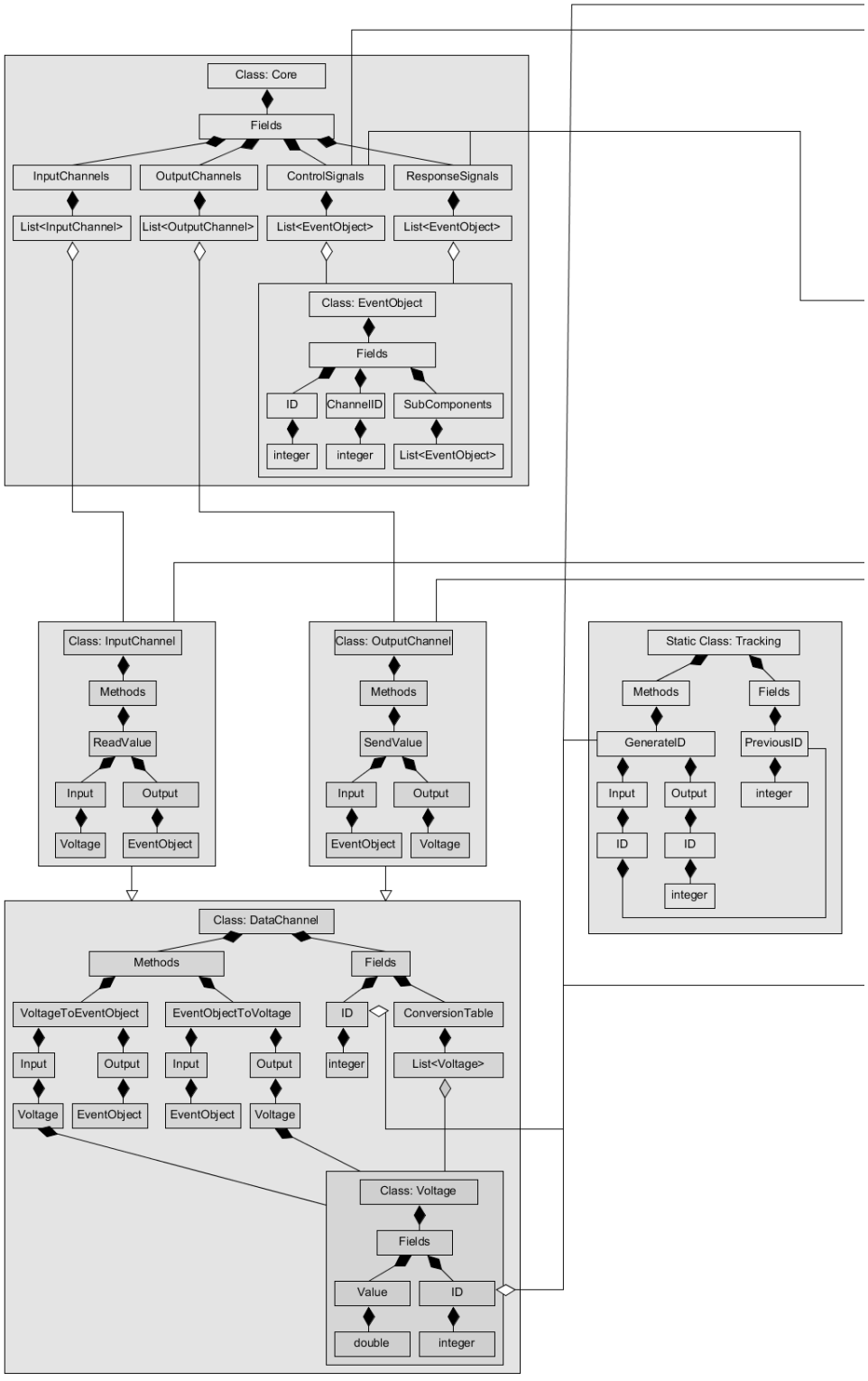
Below is a concept map showing the basic elements of the system such as core storage component “EventObject”.

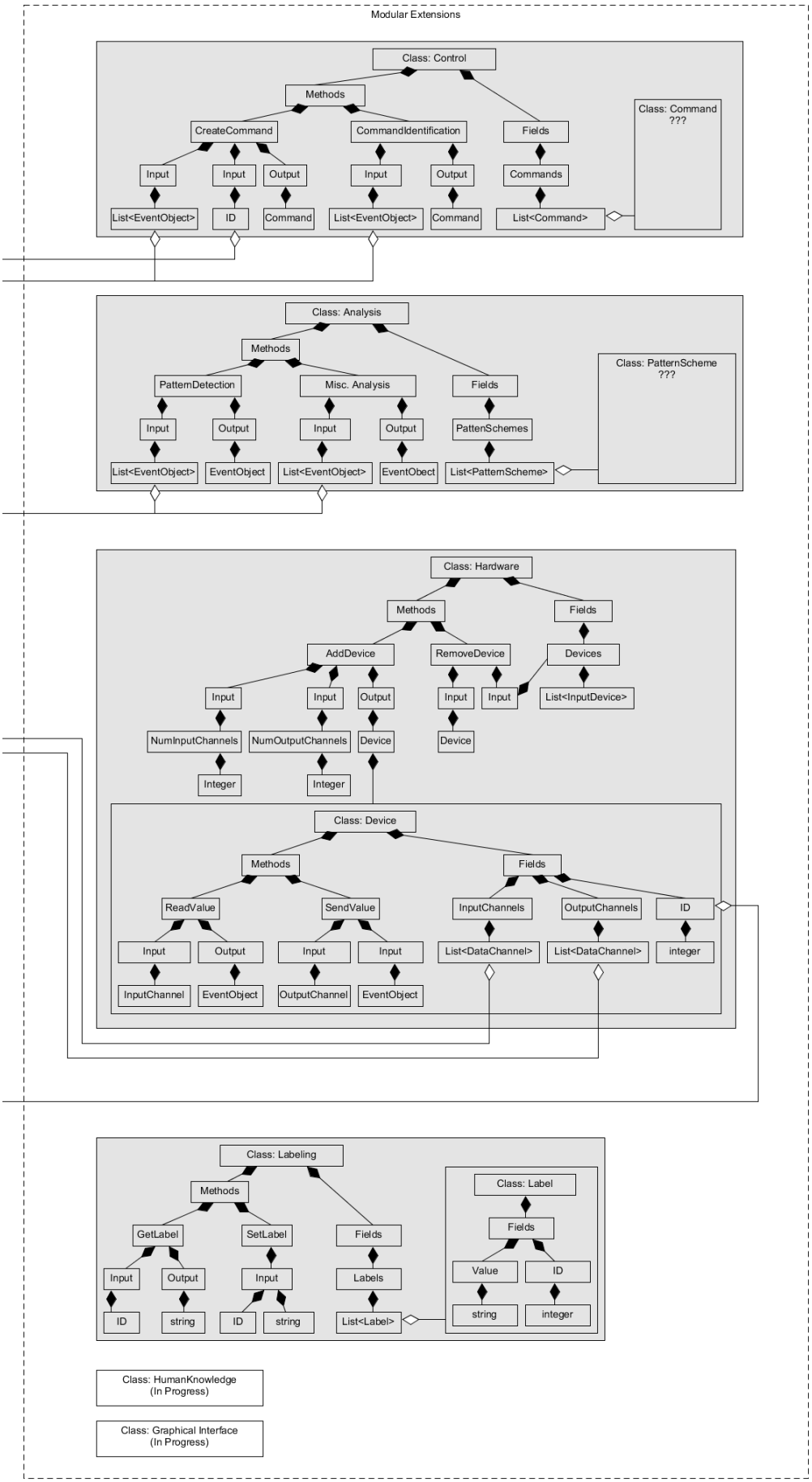


## P7: UML Mapping

Task: Create a detailed UML map of the workings of the project's system.

Below is a broken view of the UML. The required components are on the first page. The modular, optional, components are listed on the second page.




Modular Extensions
Class: Control
Methods
CreateCommand
CommandIdentification
Fields
Input
Input
Output
Input
Output
Commands
List<EventObject>
ID
Command
List<EventObject>
Command
List<Command>
Class: Command
???
Class: Analysis
Methods
PatternDetection
Misc. Analysis
Fields
Input
Output
Input
Output
PattenSchemes
List<EventObject>
EventObject
List<EventObject>
EventObect
List<PatternScheme>
Class: PatternScheme
???
Class: Hardware
Methods
Fields
AddDevice
RemoveDevice
Devices
Input
Input
Output
Input
Input
List<InputDevice>
NumInputChannels
NumOutputChannels
Device
Device
Integer
Integer
Class: Device
Methods
Fields
ReadValue
SendValue
InputChannels
OutputChannels
ID
Input
Output
Input
Input
List<DataChannel>
List<DataChannel>
integer
InputChannel
EventObject
OutputChannel
EventObject
Class: Labeling
Methods
GetLabel
SetLabel
Fields
Input
Output
Input
Labels
ID
string
ID
string
List<Label>
Class: Label
Fields
Value
ID
string
integer
Class: HumanKnowledge
(In Progress)
Class: Graphical Interface
(In Progress)

## P8: Turtle Code

Task: Learn the basics of Turtle (ttl) syntax. Deomonstrate this by creating a turtle file about the project.

Below is the text of the turtle file.

```
@prefix ois: <http://www.ChristopherWBlake.com/ObjectIdentification#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

<http://www.w3.org/1999/02/22-rdf-syntax-ns#> a owl:Ontology ;
       dc:title “Object Identification Concepts Vocabulary (OI)" ;
       dc:description "This is the RDF for the object identification system." .

#Properties#
ois:id a rdf:Property ;
       rdfs:label "id" ;
       rdfs:range rdfs:Literal ;
       rdfs:comment "A tracking identifier for all objects in the system." .
ois:value a rdf:Property ;
       rdfs:label "value" ;
       rdfs:range rdfs:Literal ;
       rdfs:comment "A numeric amount." .
ois:EventObjectList a rdf:List ;
       rdfs:range ois:EventObject ;
       rdfs:label "EventObjectList" ;
       rdfs:comment "A list of event objects." .
ois:VoltageList a rdf:List ;
       rdfs:range ois:Voltage ;
       rdfs:label "VoltageList" ;
       rdfs:comment "A list of voltages." .
ois:CommandList a rdf:List ;
       rdfs:range ois:Command ;
       rdfs:label "CommandListList" ;
       rdfs:comment "A list of commands." .
ois:Process a rdf:Class;
       rdfs:range rdfs:Resource ;
       rdfs:label "EventObjectList" ;
       rdfs:comment "A list of event objects." .


#Class#
ois:Core a rdfs:Class ;
       rdfs:subClassOf rdfs:Resource ;
       rdfs:label "Core" ;
       rdfs:comment "The central structure of the system, where all generated information
is stored."
##Fields##
ois:inputChannels ois:InputChannelList ;
ois:outputChannels ois:OutputChannelList ;
ois:controlSignals ois:EventObjectList ;
ois:responseSignals ois:EventObjectList .


#Class#
ois:EventObject a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "EventObject" ;
       rdfs:comment "The central information unit of the system." ;
##Fields##
ois:id rdfs:Literal ;
ois:channelID rdfs:Literal ;
ois:EventObjectList "subComponents"
```

```
#Class#
ois:InputChannel a rdfs:DataChannel ;
       rdfs:subClassOf rdfs:Resource ;
       rdfs:label "InputChannel" ;
       rdfs:comment "A data channel for receiving data." ;
##Methods##
ois:readValue ois:Process ;
ois:Input ois:Voltage ;
ois:Output oils:EventObject .


#Class#
ois:OutputChannel a rdfs:DataChannel ;
       rdfs:subClassOf rdfs:Resource ;
       rdfs:label "OutputChannel" ;
       rdfs:comment "A data channel for sending data." ;
##Methods##
ois:sendValue ois:Process ;
ois:input ois:EventObject ;
ois:output oist:Voltage .


#Class#
ois:DataChannel a rdfs:Class ;
       rdfs:subClassOf rdfs:Resource ;
       rdfs:label "DataChannel" ;
       rdfs:comment "A pathway for transfering data." ;
##Fields##
ois:id rdf:Literal
ois:converTable ois:VoltageTable
##Methods##
ois:voltageToEventObject ois:Process ;
ois:input ois:Voltage ;
ois:output ois:EventObject ;
ois:eventObjecToVoltage ois.Process ;
ois:input ois:EventObject ;
ois:output ois:Voltage .


#Class#
ois:Voltage a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Voltage" ;
       rdfs:comment "A numeric value representing the voltage of a device." ;
##Field##
ois:value rdf:Literal ;
ois:id rdf:Literal .


#Class#
ois:Tracking a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Tracking" ;
       rdfs:comment "A class for adding tracking objects in the system." ;
##Fields##
ois:previousID ois:id ;
##Methods##
ois:generateID ois:Process ;
ois:input ios:id
ois:output ios:id

#Class#
ois:Control a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Control" ;
```

```
       rdfs:comment "A modular class for creating human-readable commans." ;
##Fields##
ois:commands ois:CommandList
##Methods##
ois:createCommand
ois:input ois:EventObjectList ;
ois:input ois:id ;
ois:output ois:command ;
ois:commandIdentification ;
ois:input ois:EventObjectList ;
ois:output ois:command .


#Class#
ois:Analysis a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Analysis" ;
       rdfs:comment "A list of patterns schemes for the system." ;
##Fields##
ois:patternSchemes ois:PatternSchemesList ;
##Methods##
ois:patternDetection
       ois:input ois:EventObjectList ;
       ois:output ois:EventObject ;
ois:miscAnalysis ois:Process;
       ois:input ois:EventObjectList ;
       ois:output ois:EventObject .

#Class#
ois:Hardward a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Hardware" ;
       rdfs:comment "A modular class to represent attached hardward." ;
##Fields##
ois:devices ois:DeviceList
##Methods##
ois:addDevice ois:Process ;
ois:numInputChannels rdfs:Literal ;
ois:numOutputChannels rdfs:Literal ;
ois:output ois:Device ;
ois:removeDevice ois:Process ;
ois:input ois:Device .


#Class#
ois:Device a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Device" ;
       rdfs:comment "A piece of hardware." ;
##Fields##
ois:inputChannels ois:InputChannelsList ;
ois:outputChannels ois:OutputChannelsList ;
ois:id rdfs:Literal ;
##Methods##
ois:readValue ois:Process ;
ois:input ois:inputChannel ;
ois:output ois:EventObject ;
ois:sendValue ois:Process ;
ois:input ois:outputChannel ;
ois:input ois:EventObject .


#Class#
ois:Labeling a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Labeling" ;
```

```
       rdfs:comment "A modular class for applying human-readable labels to system objects."
;
##Fields##
ois:labels rdfs:List ;
##Methods##
ois:getLabel ois:Process
       ois:input ois:id;
       ois:output rdfs:label;
ois:setLabel ois:Process
       ois:input ois:id ;
       ois:input rdfs:label .


#Class#
ois:Label a rdfs:Class ;
       rdfs:subClassOf rdfs:Literal ;
       rdfs:label "Label" ;
       rdfs:comment "A human-readibly label of a system object." ;
##Fields##
ois:value rdfs:label ;
ois:id rdfs:Literal .
```

## P9: OWL

Task: Create a sample turtle file extending it to include the OWL namespace. Ensure to include examples of the following elements. The document does not need to be project related.

1.) owl:Class
2.) owl:DatatypeProperty
3.) owl:ObjectProperty
4.) owl:sameAs (for indiv)
5.) owl:differentFrom (for indiv)
6.) owl: AllDifferent

Below is the turtle file text.

```
@prefix :<http://mysite.org>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.

### Class, subClassOf, equivalentClass, disjointWith, UnionOf ###
#################################################################
:female a owl:Class.
:male a owl:Class.

:Animal a owl:Class.
:Plant a owl:Class.

:Carnivore rdfs:subClassOf :Animal;
       owl:disjointWith :Herbivore.
:Herbivore rdfs:subClassOf :Animal;
       owl:disjointWith :Carnivore.
:Omnivore rdfs:subClassOf :Animal;
       owl:UnionOf (:Carnivore :Herbivore).
[] rdf:type :AllDisjointClasses;
       owl:members (:Carnivore :Herbivore :Omnivore).

:Canine rdfs:subClassOf :Carnivore.
:Feline rdfs:subClassOf :Carnivore.
:Mouse rdfs:subClassOf :Herbivore.
:HousePet rdfs:subClassOf :Animal.

:HouseCat rdfs:subClassOf :Feline, :HousePet.
:Cat owl:equivalentClass :HouseCat.

### DatatypeProperty, oneOf, intersectionOf ###
###############################################
:hasName a owl:DatatypeProperty.
:hasGender a owl:DatatypeProperty.
:hasAge a owl:DatatypeProperty.

:GenderRestriction a owl:Class;
       owl:oneOf(:male :female).
:hasGender owl:intersectionOf :GenderRestriction.
:NonNegativeAge a owl:Class; #not sure how to actually implement this.
       owl:onProperty :age.
:Animal owl:intersectionOf :NonNegativeAge. #Prevent age of animals from being negative.

### Domain, Range ###
#####################
:eats a owl:ObjectProperty;
       rdfs:Domain :Carnivore;
       rdfs:Range :Herbivore.
```

```
:eats a owl:ObjectProperty;
       rdfs:Domain :Herbivore;
       rdfs:Range :Plant.

### sameAs, differentFrom, AllDifferent ###
##########################################
:Catsy a :HouseCat;
       :hasName "Catsy";
       :hasage 10;
       :hasgender  "female".
:Kitty a :HouseCat;
       :hasName "Kitty".
:Sheena a :HouseCat;
       :hasName "Sheena".

:Bailey a :HouseCat;
       :hasName "Bailey".
:Penny a :HouseCat;
       :hasName "Penny".

:Catsy owl:sameAs :Kitty. # Catsy and Kitty are nicknames.
:Catsy owl:differentFrom :Sheena.
[] rdf:type owl:AllDifferent;
       owl:distinctMembers (:Catsy :Sheena :Bailey :Penny).
```

## P10: Protége

Task: Using the program Protége, create a fully defined ontology file for the project.

Below is created ontology file for the "Object Identification" project. Note: Some data items (individuals) have been left out to shorten the text for this report. The original file has 3000+ axioms.

```
@prefix : <http://myurl.ObjectIdentification#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://myurl.ObjectIdentification#> .

<http://myurl.ObjectIdentification#> rdf:type owl:Ontology ;
                                     rdfs:comment "Homework practice for task 10."^^rdfs:Literal .

#################################################################
#    Object Properties
#################################################################
###  http://myurl.ObjectIdentification#controlSignal
:controlSignal rdf:type owl:ObjectProperty ;
               rdfs:domain :EventObject ;
               rdfs:range :ControlSignal ;
               rdfs:comment "The subject of this property has a \"ControlSignal\"."^^xsd:string .


###  http://myurl.ObjectIdentification#dataChannel
:dataChannel rdf:type owl:ObjectProperty ;
             rdfs:range :DataChannel ;
             rdfs:comment "The subject of this property has a \"DataChannel\"."^^xsd:string .


###  http://myurl.ObjectIdentification#device
:device rdf:type owl:ObjectProperty ;
        rdfs:range :Device ;
        rdfs:comment "The subject of this property has a \"Device\"."^^xsd:string .


###  http://myurl.ObjectIdentification#eventObject
:eventObject rdf:type owl:ObjectProperty ;
             rdfs:domain :Field ,
                         :List ;
             rdfs:range :EventObject ;
             rdfs:comment "The subject of this property contains one or more a
\"EventObject\"."^^xsd:string .


###  http://myurl.ObjectIdentification#label
:label rdf:type owl:ObjectProperty ;
       rdfs:range :Label ;
       rdfs:comment "The subject of this property has a \"Label\". This is a human-readable string,
used only for display purposes."^^xsd:string .


###  http://myurl.ObjectIdentification#list
:list rdf:type owl:ObjectProperty ;
      rdfs:range :List ;
      rdfs:comment "The subject of this property has a \"List\" of some other \"Thing\"."^^xsd:string .


###  http://myurl.ObjectIdentification#responseSignal
:responseSignal rdf:type owl:ObjectProperty ;
                rdfs:subPropertyOf owl:topObjectProperty ;
                rdfs:domain :EventObject ;
                rdfs:range :ResponseSignal ;
                rdfs:comment "The subject of this property has a \"ResponseSignal\"."^^xsd:string .


#################################################################
#    Data properties
```

```
#################################################################

###  http://myurl.ObjectIdentification#ID
:ID rdf:type owl:DatatypeProperty ;
    rdfs:domain :DataChannel ,
                :Device ,
                :EventObject ,
                :Field ;
    rdfs:range xsd:integer ;
    rdfs:comment "A numeric integer used for tracking all objects in a \"Core\"."^^xsd:string .


###  http://myurl.ObjectIdentification#Value
:Value rdf:type owl:DatatypeProperty ;
       rdfs:domain :Field ;
       rdfs:range xsd:double ;
       rdfs:comment "A numeric value. Usually associated with EventObject types."^^xsd:string .


###  http://myurl.ObjectIdentification#channelID
:channelID rdf:type owl:DatatypeProperty ;
           rdfs:subPropertyOf :ID .


###  http://myurl.ObjectIdentification#description
:description rdf:type owl:DatatypeProperty ;
             rdfs:comment "Provides additional information about something to clarify
usage."^^xsd:string .


###  http://myurl.ObjectIdentification#name
:name rdf:type owl:DatatypeProperty ;
      rdfs:range xsd:string ;
      rdfs:comment "A nickname type value associated to something to allow easier
readability."^^xsd:string .


###  http://myurl.ObjectIdentification#previousID
:previousID rdf:type owl:DatatypeProperty ;
            rdfs:subPropertyOf :ID .


#################################################################
#    Classes
#################################################################

###  http://myurl.ObjectIdentification#Analysis
:Analysis rdf:type owl:Class ;
          rdfs:subClassOf :Module ;
          rdfs:comment "A an extra module to the core. This allows analysis of the current EventObject
Items within the database. Such analysis usually results in additional EventObjects being added to the
core."^^xsd:string .


###  http://myurl.ObjectIdentification#Command
:Command rdf:type owl:Class ;
         rdfs:subClassOf :EventObject ;
         rdfs:comment "The element used by \"Command\" module. This marks an event object as a
refularly occuring command, which can be called by a user."^^xsd:string .


###  http://myurl.ObjectIdentification#Control
:Control rdf:type owl:Class ;
         rdfs:subClassOf :Module ,
                         [ rdf:type owl:Restriction ;
                           owl:onProperty :list ;
                           owl:someValuesFrom :Command
                         ] ;
         rdfs:comment "A an extra module to the core. This allows marking specific EventObjects as
regularly occuring commands."^^xsd:string .


###  http://myurl.ObjectIdentification#ControlSignal
:ControlSignal rdf:type owl:Class ;
               rdfs:subClassOf :Signal ;
               rdfs:comment "An output signal used for controlling an object."^^xsd:string .
```

```
###  http://myurl.ObjectIdentification#Core
:Core rdf:type owl:Class ;
      rdfs:subClassOf owl:Thing ,
                      [ rdf:type owl:Restriction ;
                        owl:onProperty :list ;
                        owl:someValuesFrom :ControlSignal
                      ] ,
                      [ rdf:type owl:Restriction ;
                        owl:onProperty :list ;
                        owl:someValuesFrom :InputChannel
                      ] ,
                      [ rdf:type owl:Restriction ;
                        owl:onProperty :list ;
                        owl:someValuesFrom :OutputChannel
                      ] ,
                      [ rdf:type owl:Restriction ;
                        owl:onProperty :list ;
                        owl:someValuesFrom :ResponseSignal
                      ] ;
      rdfs:comment "The container of all items in the system. It requires a list of control signals,
response signals, output channels, and input channels."^^xsd:string .


###  http://myurl.ObjectIdentification#DataChannel
:DataChannel rdf:type owl:Class ;
             rdfs:subClassOf owl:Thing ,
                             [ rdf:type owl:Restriction ;
                               owl:onProperty :ID ;
                               owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                               owl:onDataRange xsd:integer
                             ] ;
             rdfs:comment "The transportation mechanism of information into and out of the
sytem."^^xsd:string .


###  http://myurl.ObjectIdentification#Device
:Device rdf:type owl:Class ;
        rdfs:subClassOf :Thing ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :list ;
                          owl:someValuesFrom :InputChannel
                        ] ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :list ;
                          owl:someValuesFrom :OutputChannel
                        ] ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :ID ;
                          owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                          owl:onDataRange xsd:int
                        ] ;
        rdfs:comment "Represents a grouping of input and output channels."^^xsd:string .


###  http://myurl.ObjectIdentification#EventObject
:EventObject rdf:type owl:Class ;
             rdfs:subClassOf :Thing ,
                             [ rdf:type owl:Restriction ;
                               owl:onProperty :list ;
                               owl:someValuesFrom :EventObject
                             ] ,
                             [ rdf:type owl:Restriction ;
                               owl:onProperty :ID ;
                               owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                               owl:onDataRange xsd:integer
                             ] ,
                             [ rdf:type owl:Restriction ;
                               owl:onProperty :channelID ;
                               owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                               owl:onDataRange xsd:integer
                             ] ;
             rdfs:comment "The basic building block of all data withing the system. An event object may
be the lowest member of a group, or it may contain references to other EventObjects, defining a complex
event object."^^xsd:string .
```

```
###  http://myurl.ObjectIdentification#Field
:Field rdf:type owl:Class ;
       rdfs:subClassOf :Thing .


###  http://myurl.ObjectIdentification#Hardware
:Hardware rdf:type owl:Class ;
          rdfs:subClassOf :Module ,
                          [ rdf:type owl:Restriction ;
                            owl:onProperty :list ;
                            owl:someValuesFrom :Device
                          ] ;
          rdfs:comment "A an extra module to the core. This allows organizing input and output channels
as \"devices\"."^^xsd:string .


###  http://myurl.ObjectIdentification#InputChannel
:InputChannel rdf:type owl:Class ;
              rdfs:subClassOf :DataChannel ;
              rdfs:comment "The input mechanism for information into the system."^^xsd:string .


###  http://myurl.ObjectIdentification#Label
:Label rdf:type owl:Class ;
       rdfs:subClassOf :Thing ,
                       [ rdf:type owl:Restriction ;
                         owl:onProperty :ID ;
                         owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                         owl:onDataRange xsd:integer
                       ] ,
                       [ rdf:type owl:Restriction ;
                         owl:onProperty :name ;
                         owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                         owl:onDataRange xsd:string
                       ] ;
       rdfs:comment "The element used by \"Labeling\" module. This assigns a human-readable name to an
EventObject."^^xsd:string .


###  http://myurl.ObjectIdentification#Labeling
:Labeling rdf:type owl:Class ;
          rdfs:subClassOf :Module ;
          rdfs:comment "A an extra module to the core. This allows assigning a human-readable name to
an EventObject."^^xsd:string .


###  http://myurl.ObjectIdentification#List
:List rdf:type owl:Class ;
      rdfs:subClassOf :Thing ,
                      [ rdf:type owl:Restriction ;
                        owl:onProperty :list ;
                        owl:someValuesFrom :Thing
                      ] ,
                      [ rdf:type owl:Restriction ;
                        owl:onProperty :name ;
                        owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                        owl:onDataRange xsd:string
                      ] ;
      rdfs:comment "Creates a list of any \"thing\"." .


###  http://myurl.ObjectIdentification#Module
:Module rdf:type owl:Class ;
        rdfs:subClassOf owl:Thing ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :name ;
                          owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
                          owl:onDataRange xsd:string
                        ] ;
        rdfs:comment "A modular extension of the system. This is used for adding additional
functionality to the system such as human readable labels, regularly occuring commands, and analysis
engines."^^xsd:string .
```

```
###  http://myurl.ObjectIdentification#OutputChannel
:OutputChannel rdf:type owl:Class ;
               rdfs:subClassOf :DataChannel ;
               rdfs:comment "The output mechanism for information out of the system."^^xsd:string .


###  http://myurl.ObjectIdentification#ResponseSignal
:ResponseSignal rdf:type owl:Class ;
                rdfs:subClassOf :Signal ;
                rdfs:comment "An input signal received from an object. Usually a device with
sensors."^^xsd:string .


###  http://myurl.ObjectIdentification#Signal
:Signal rdf:type owl:Class ;
        rdfs:subClassOf :EventObject ;
        rdfs:comment "A value either received or sent via a DataChannel."^^xsd:string .


###  http://myurl.ObjectIdentification#Thing
:Thing rdf:type owl:Class ;
       rdfs:subClassOf owl:Thing ;
       rdfs:comment "The generic contain of all elements withing the system."^^xsd:string .


#################################################################
#    Individuals
#################################################################

###  http://myurl.ObjectIdentification#c1_Core
:c1_Core rdf:type owl:NamedIndividual ,
                  :Core ;
         :list :c1_Core_ControlSignals ,
               :c1_Core_InputChannels ,
               :c1_Core_OutputChannels ,
               :c1_Core_ResponseSignals .


###  http://myurl.ObjectIdentification#c1_Core_ControlSignals
:c1_Core_ControlSignals rdf:type owl:NamedIndividual ,
                                 :List .


###  http://myurl.ObjectIdentification#c1_Core_InputChannels
:c1_Core_InputChannels rdf:type owl:NamedIndividual ,
                               :List ;
                       :dataChannel :id_002_InputChannel .


###  http://myurl.ObjectIdentification#c1_Core_OutputChannels
:c1_Core_OutputChannels rdf:type owl:NamedIndividual ,
                                :List ;
                        :dataChannel :id_003_OutputChannel .


###  http://myurl.ObjectIdentification#c1_Core_ResponseSignals
:c1_Core_ResponseSignals rdf:type owl:NamedIndividual ,
                                 :List .


###  http://myurl.ObjectIdentification#c1_Module_Control
:c1_Module_Control rdf:type owl:NamedIndividual ,
                            :Control ;
                   :description "A modular extension to the core which allows marking of some control
signals as \"commands\". These commands are normally often requested by a user, and hence given special
notice."^^xsd:string .


###  http://myurl.ObjectIdentification#c1_Module_Hardware
:c1_Module_Hardware rdf:type owl:NamedIndividual ,
                             :Hardware ;
                    :list :c1_Module_Hardware_Devices ;
                    :description "Modular extension to the core, which allows handling of input and
output channels as devices."^^xsd:string .
```

```
###  http://myurl.ObjectIdentification#c1_Module_Hardware_Devices
:c1_Module_Hardware_Devices rdf:type owl:NamedIndividual ,
                                     :List ;
                            :device :id_001_Device ;
                            :description "A list of all hardware-based devices used in the
system."^^xsd:string .


###  http://myurl.ObjectIdentification#c1_Module_Labeling
:c1_Module_Labeling rdf:type owl:NamedIndividual ,
                             :Labeling ;
                    :list :c1_Module_Labeling_Labels ;
                    :description "Modular extension of core which allows labeling of all elements in
teh system that have an ID."^^xsd:string .


###  http://myurl.ObjectIdentification#c1_Module_Labeling_Labels
:c1_Module_Labeling_Labels rdf:type owl:NamedIndividual ,
                                    :List ;
                           :label :id_001_Label ;
                           :description "A list of human-readable labels for specified objects in the
system. Not all objects with IDs will have labels."^^xsd:string .


###  http://myurl.ObjectIdentification#id_001_Device
:id_001_Device rdf:type owl:NamedIndividual ,
                        :Device ;
               :dataChannel :id_002_InputChannel ,
                            :id_003_OutputChannel ;
               :ID 1 .


###  http://myurl.ObjectIdentification#id_001_Label
:id_001_Label rdf:type owl:NamedIndividual ,
                       :Label ;
              :ID 1 ;
              :description "Converts a 5v signal to ASCII Code"^^xsd:string ;
              :name "CodeConverter"^^xsd:string .


###  http://myurl.ObjectIdentification#id_002_InputChannel
:id_002_InputChannel rdf:type owl:NamedIndividual ,
                              :InputChannel ;
                     :ID 2 .


###  http://myurl.ObjectIdentification#id_003_OutputChannel
:id_003_OutputChannel rdf:type owl:NamedIndividual ,
                               :OutputChannel ;
                      :ID 3 .


###  http://myurl.ObjectIdentification#id_007_ControlSignal
:id_007_ControlSignal rdf:type owl:NamedIndividual ,
                               :ControlSignal ;
                      :dataChannel :id_003_OutputChannel ;
                      :ID 7 ;
                      :Value "0.025"^^xsd:double .


###  http://myurl.ObjectIdentification#id_008_ResponseSignal
:id_008_ResponseSignal rdf:type owl:NamedIndividual ,
                                :ResponseSignal ;
                       :dataChannel :id_002_InputChannel ;
                       :ID 8 ;
                       :Value 1 .


###  http://myurl.ObjectIdentification#id_009_EventObject
:id_009_EventObject rdf:type owl:NamedIndividual ,
                             :EventObject ;
                    :controlSignal :id_007_ControlSignal ;
                    :responseSignal :id_008_ResponseSignal ;
                    :ID 9 .
```

## P11: SPARQL Queries

Task: Create five SPARQL queries with the following requirements.

1. 3 triple patterns, no filters, no ordering
2. 2+ triple patterns, filter with numbers
3. 2+ triple patterns, filter with regex
4. 3+ triples, use “optional”, filter label with “lang()”
5. 2+ triple patterns, use “count”

**Task 1**

Description

3 triple patterns
- no filters
- no ordering

SPARQL Query

```
select distinct
?cityName
where
{
  ?city a dbo:City.
  ?city dbo:isPartOf dbr:Indiana.
  ?city dbp:officialName ?cityName.
}
LIMIT 10
```

Results

| cityName |
|---|
| "City of Alexandria"@en |
| "Columbus, Indiana"@en |
| "City of Fort Wayne"@en |
| "City of Gary"@en |
| "City of  Connersville, Indiana"@en |
| "Lawrence, Indiana"@en |
| "Mitchell, Indiana"@en |
| "Muncie, Indiana"@en |
| "City of Beech Grove, Indiana"@en |
| "Berne, Indiana"@en |

**Task 2**

Description

2.) >= 2 Triple Patterns
+ filter with numbers

SPARQL Query

```
select distinct
?cityName
?population
where
{
  ?city a dbo:City.
  ?city dbo:isPartOf dbr:Indiana.

  ?city dbp:officialName ?cityName.
  ?city dbo:populationTotal ?population filter(?population < 5000).
}
LIMIT 10
```

Results

| cityName | population |
|---|---|
| "Mitchell, Indiana"@en | "4350"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Berne, Indiana"@en | "3999"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Bicknell, Indiana"@en | "2915"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "City of Cannelton, Indiana"@en | "1563"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Delphi, Indiana"@en | "2893"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Greendale, Indiana"@en | "4520"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Jasonville, Indiana"@en | "2222"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "City of Jonesboro, Indiana"@en | "1756"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "City of Knox, Indiana"@en | "3704"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "City of Ligonier"@en | "4405"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |

**Task 3**

Description

>= 2 Triple Pattern

+ filter with regex

SPARQL Query

```
select distinct
?cityName
?population
where
{
  ?city a dbo:City.
  ?city dbo:isPartOf dbr:Indiana.

  ?city dbp:officialName ?cityName.
                         filter regex(?cityName, "D").
  ?city dbo:populationTotal ?population
                            filter(?population < 10000).
}
LIMIT 10
```

Results

| cityName | population |
|---|---|
| "Delphi, Indiana"@en | "2893"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Decatur, Indiana"@en | "2"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Dunkirk, Indiana"@en | "2362"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |

**Task 4**

Description

3 triples

+ optional

+ filter label lang()

SPARQL Query

```
select distinct
?cityName
?population
where
{
  ?city a dbo:City.
  ?city dbo:isPartOf dbr:Indiana.
  optional { ?city rdfs:label ?cityName
                              filter (lang(?cityName) = "fr"). }
  ?city dbo:populationTotal ?population.
}
LIMIT 10
```

Results

| cityName | population |
|---|---|
| "Columbus (Indiana)"@fr | "44061"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Fort Wayne (Indiana)"@fr | "253691"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Gary (Indiana)"@fr | "80294"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Goshen (Indiana)"@fr | "31719"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| | "5145"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Connersville"@fr | "13481"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Lawrence (Indiana)"@fr | "46001"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Mitchell (Indiana)"@fr | "4350"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Muncie"@fr | "70085"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |
| "Angola (Indiana)"@fr | "8612"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> |

**Task 5**

Description

>= 2 triple patterns
+ count()

SPARQL Query

```
select distinct
?stateName
count(?city) as ?numCities
where
{
  ?state dbo:capital ?capital.
  ?state a dbo:Region.

  ?state dbo:country dbr:United_States.
  ?state rdfs:label ?stateName
                    filter (lang(?stateName) = "en").

  ?city a dbo:City.
  ?city dbo:isPartOf ?state.
}
LIMIT 10
```

Results

| stateName | numCities |
|---|---|
| "Alabama"@en | 171 |
| "Iowa"@en | 939 |
| "Kentucky"@en | 426 |
| "Michigan"@en | 280 |
| "Ohio"@en | 249 |
| "West Virginia"@en | 77 |
| "Kansas"@en | 616 |
| "Alaska"@en | 147 |
| "Florida"@en | 261 |
| "New Jersey"@en | 50 |

## P12: Concept Test - C# Program

Task: Create a windows form program using "dotnetRDF" and C#. Create three test SPARQL queries on the ontology file from task P10. Map these to buttons, so the user does not need to know SPARQL.

Below are screenshots of the created program. The concept program opens with the window shown in figure 1. This window confirms that the ontology file was successfully loaded and then lists all triples within the file.

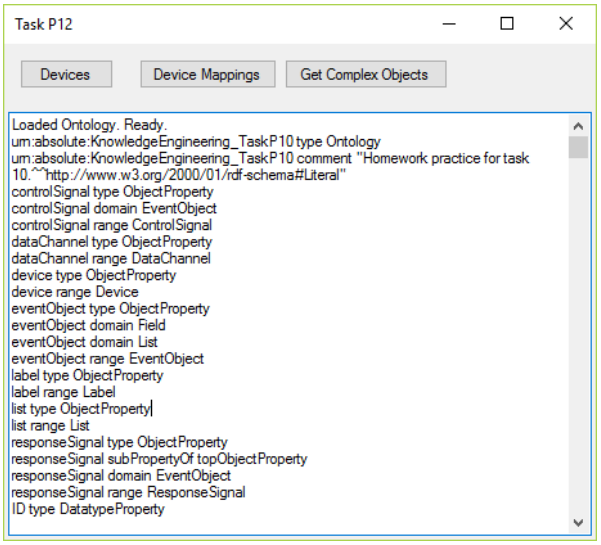



*Figure 1: C# Concept Program: Opening Screen*

Three buttons are shown. "Devices" which shows all devices present in the ontology file. (in this situation, only 1 device). "Device Mappings" which shows the relationship of input voltages to output signals. (in this situation, ASCII codes). "Get Complex Objects" which shows the event objects that have been combined to create more complex objects. (In this case, combinations of event objects to form words and sentences.)

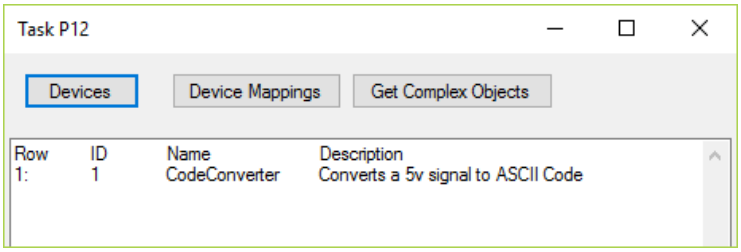



*Figure 2: SPARQL Query - List Devices*

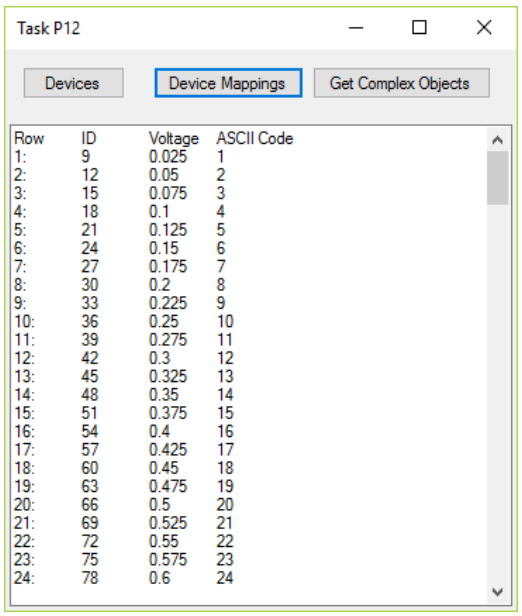



*Figure 3: SPARQL Query - Device Mappings*

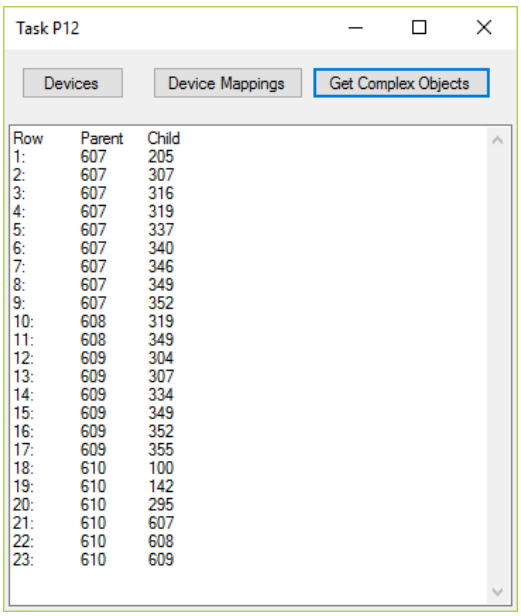



*Figure 4: SPARQL Query - Get Complex Objects*

## Final Program (P13-P16)

Task: Create a final application which allow easy user access to consuming the project's ontology file. It should solve two main problems: 1. The user does not know SPARQL. 2. The user does not know what the project elements represent. Finally, it must provide easy user access to consuming results stored within the ontology file.

To meet these requirement, a web application and web API have been developed. These applications act as wrappers serving both computer-consumable and human-consumable information from the source ontology file.

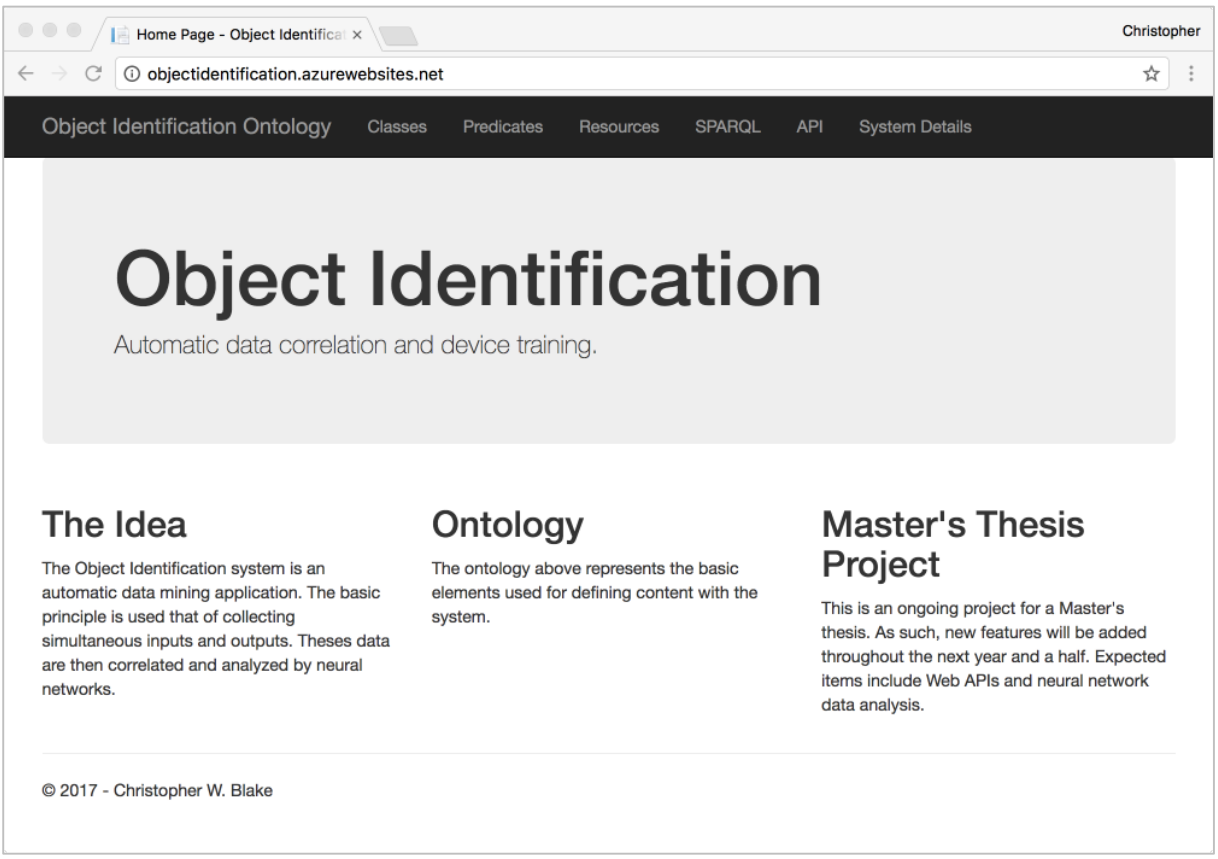



*Figure 5: Home Page*

## Classes, Predicates, and Resources

The first three links in the header, shown in figure 6, provide the basic information about the object identification system. Clicking any of these links will provide lists of the relevant items, as shown in figures 7, 8, and 9. Clicking on any class, predicate, or resource on these pages will provide a description along with the object's properties in tabular format, as shown in figure 10.

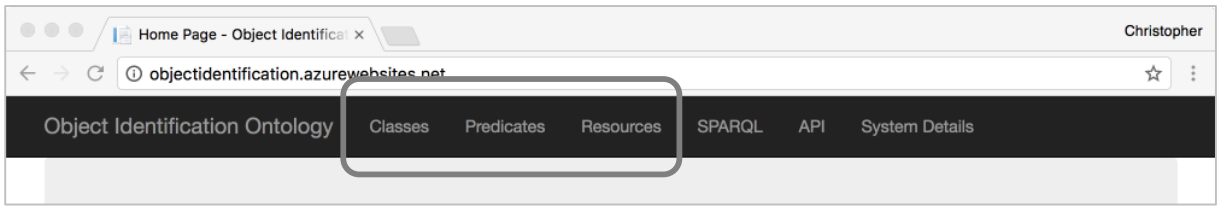



*Figure 6: Links to Class, Predicates, and Resources*

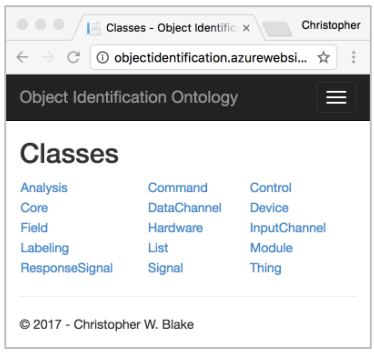



*Figure 7: Definitions of elements*

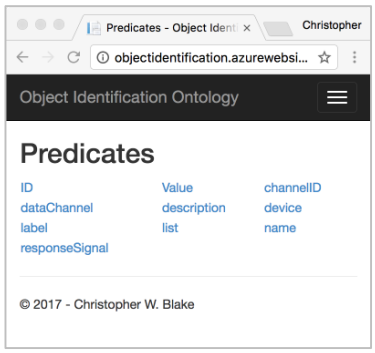



*Figure 8: Definitions of properties*

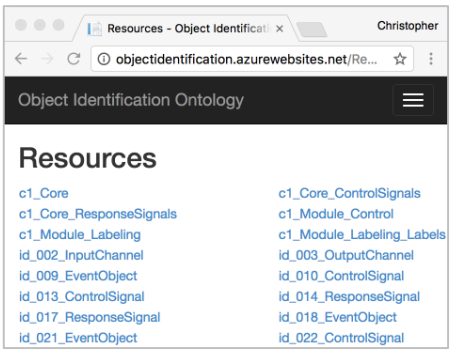



*Figure 9: List of individuals*

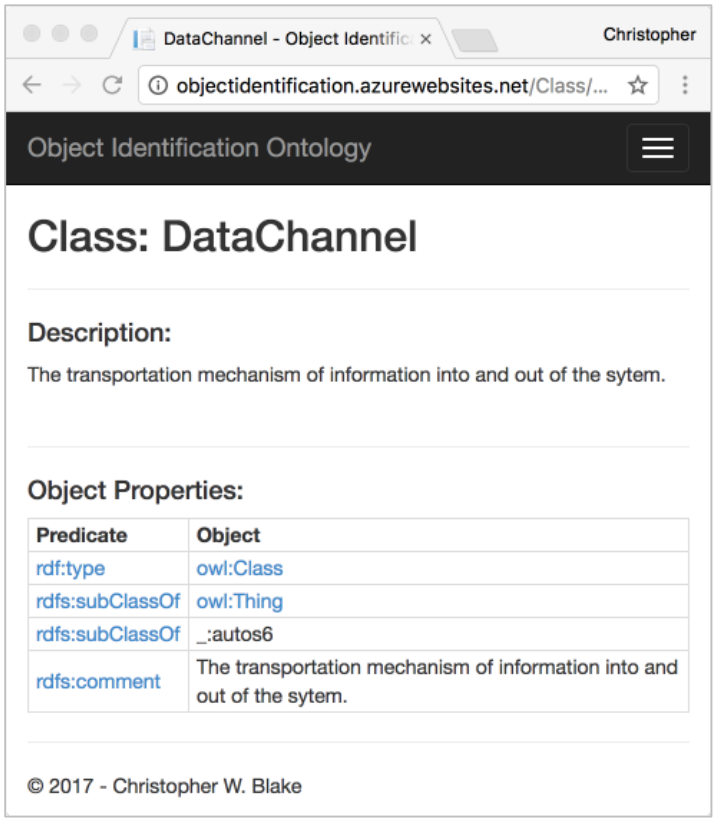



*Figure 10: Item details*

## Custom SPARQL Queries

To support future development and experimenting, a special page has been created for testing custom SPARQL queries. As shown in figure 11, one must only enter the SPARQL text and press the "Submit Query" button. The results will be shown directly below.

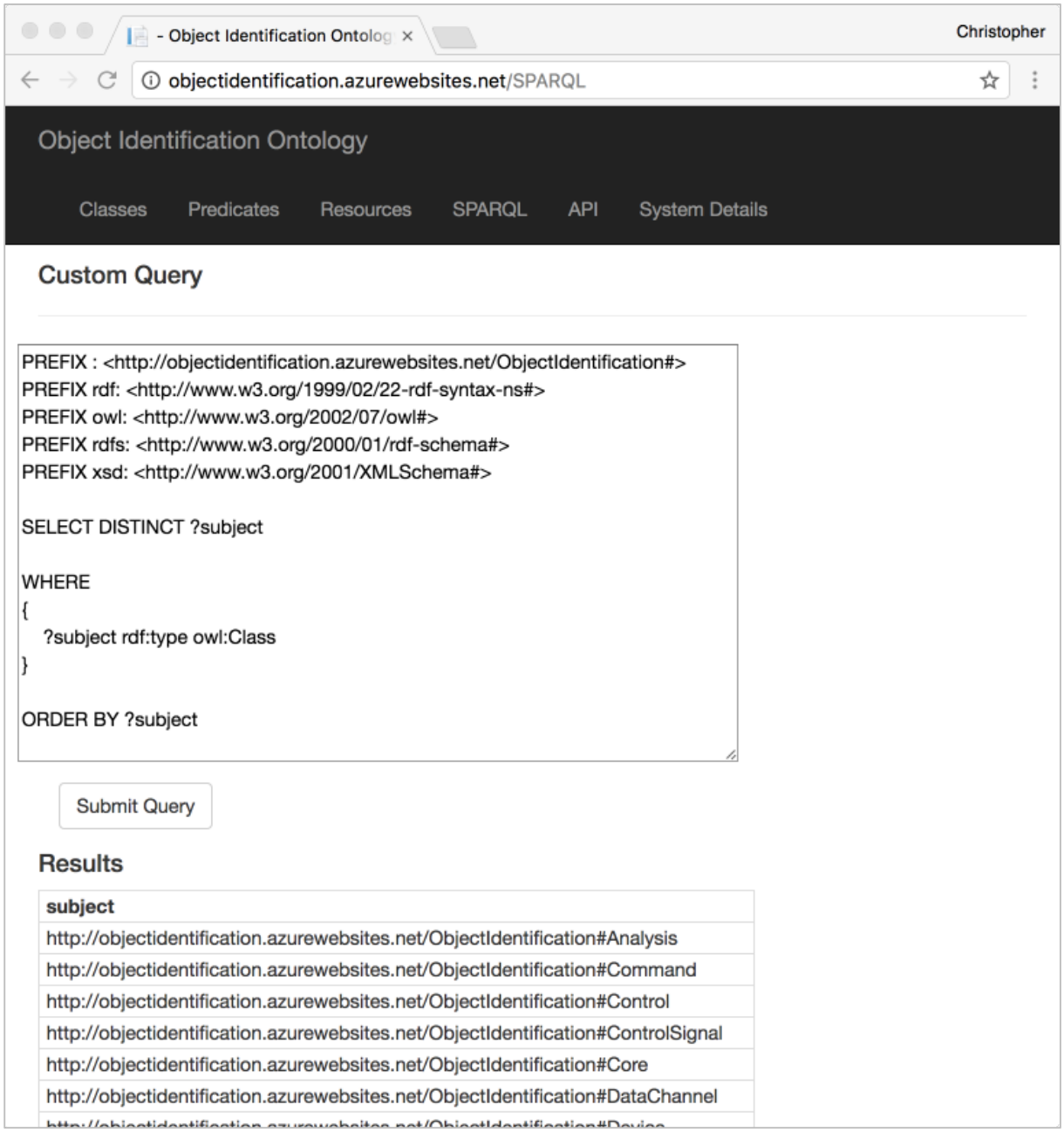



*Figure 11: Custom SPARQL Query + Results*

## Web API

A special link, API, provides usage information about the JSON/XML web API. This API allows other non-platform-specific programs the ability to consume information from this project. It uses the standard "swagger" interface, shown in figure 13. Expanding an item will provide an example as well as allow testing, as shown in figure 14.

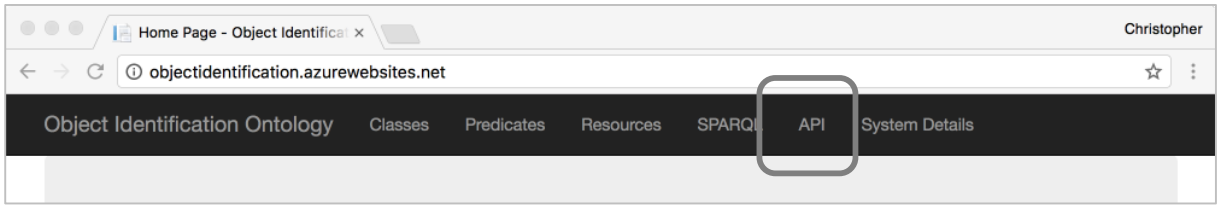



*Figure 12: Link to API information*

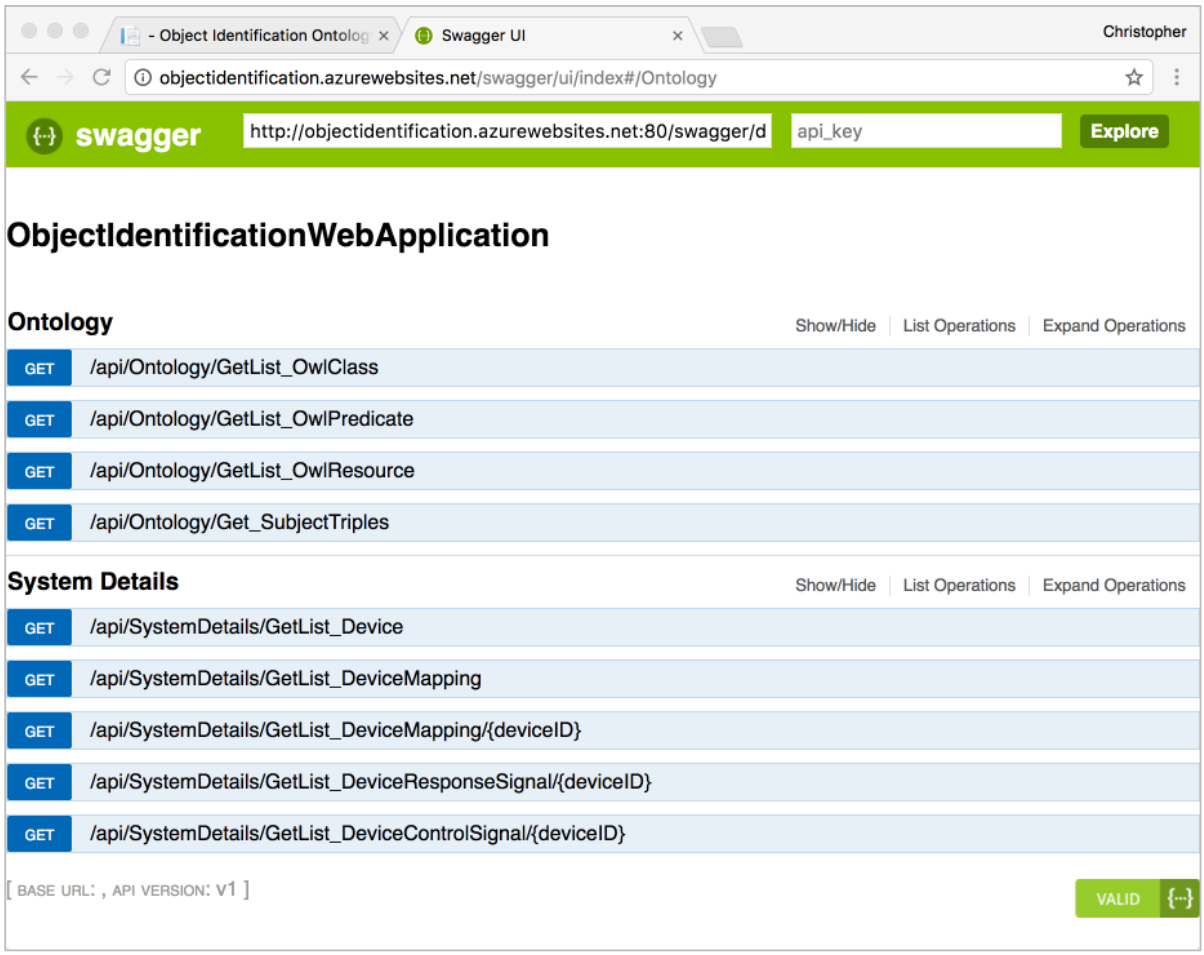



*Figure 13: List of available API commands*

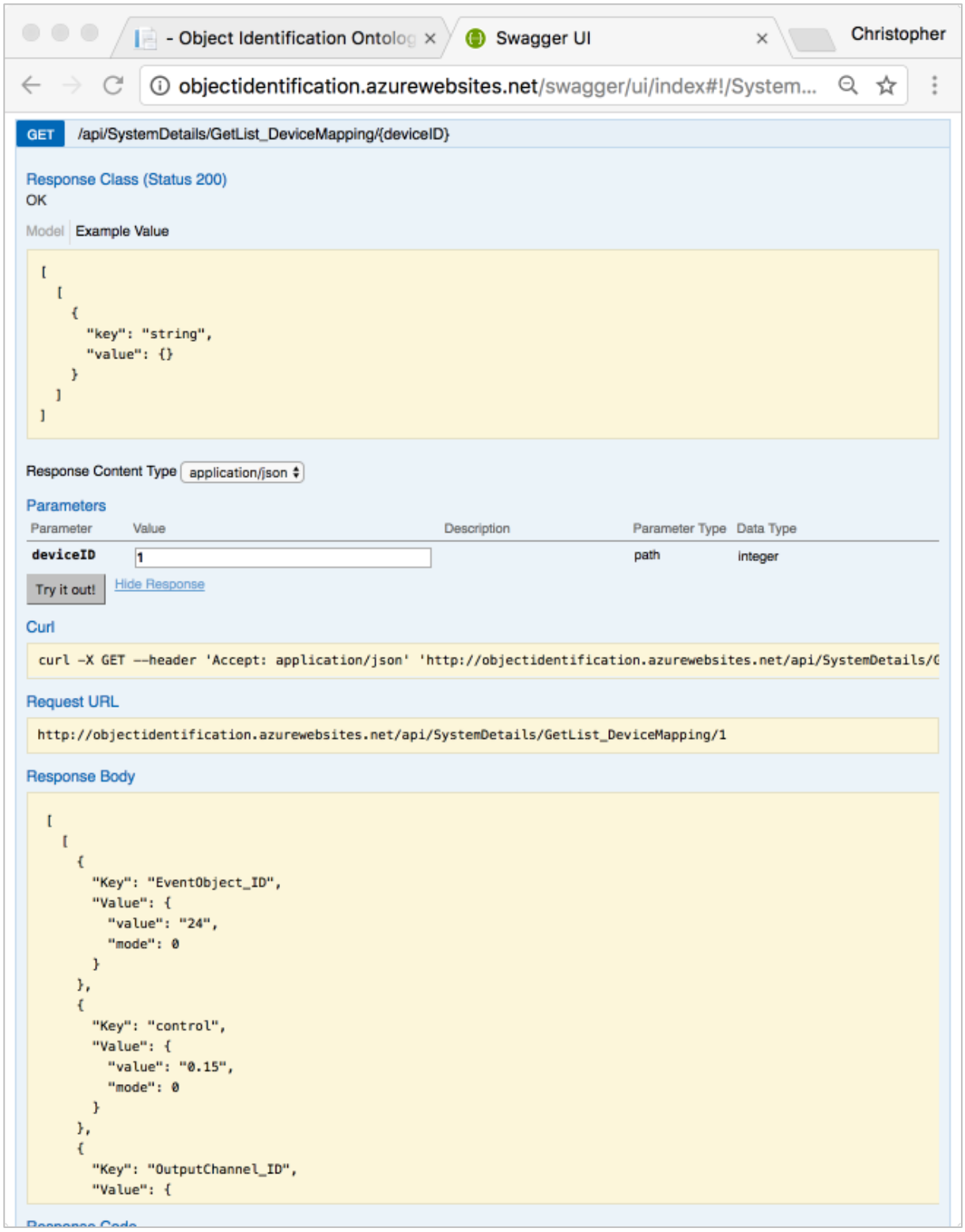



*Figure 14: Example usage of "GetList_DeviceMapping" for the DeviceID 1*

## System Details

The last tab, “System Details” provides easy access information to all users (see figure 15). Four predefined queries are created. An example result for “Device Mappings” is shown in figure 16.

1. Devices – lists all devices with ID, tag, name, and description.
2. Device Mappings – given a device ID, the mappings of inputs that cause outputs are displayed.
3. Device Response Signals – lists all signals that were recorded for a particular device.
4. Device Control Signals - lists all signals that were sent to a device.

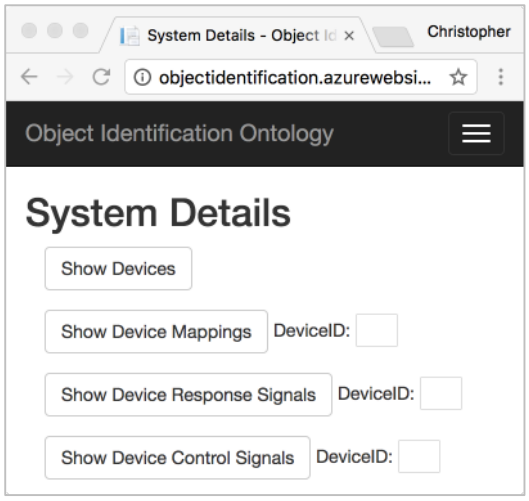
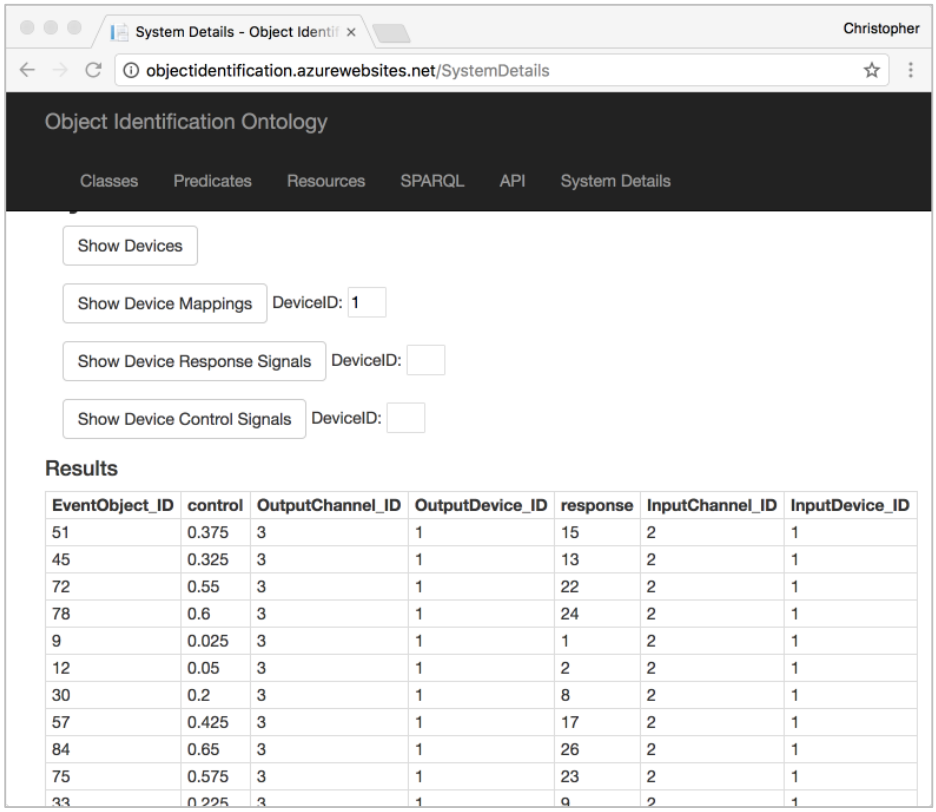



*Figure 15: Predefined tasks*




| EventObject_ID | control | OutputChannel_ID | OutputDevice_ID | response | InputChannel_ID | InputDevice_ID |
|---|---|---|---|---|---|---|
| 51 | 0.375 | 3 | 1 | 15 | 2 | 1 |
| 45 | 0.325 | 3 | 1 | 13 | 2 | 1 |
| 72 | 0.55 | 3 | 1 | 22 | 2 | 1 |
| 78 | 0.6 | 3 | 1 | 24 | 2 | 1 |
| 9 | 0.025 | 3 | 1 | 1 | 2 | 1 |
| 12 | 0.05 | 3 | 1 | 2 | 2 | 1 |
| 30 | 0.2 | 3 | 1 | 8 | 2 | 1 |
| 57 | 0.425 | 3 | 1 | 17 | 2 | 1 |
| 84 | 0.65 | 3 | 1 | 26 | 2 | 1 |
| 75 | 0.575 | 3 | 1 | 23 | 2 | 1 |
| 33 | 0.225 | 3 | 1 | 9 | 2 | 1 |

*Figure 16: Results of "Device Mappings" button*

## Source Code

The entire web application and API is developed in ASP.Net using the MVC code structure and C#. Below is a listing of the critical components and their description. The code for each component is shown in appendix 1.

1. **Ontology** – Methods for accessing the ontology file.
   1.1. Ontology.cs – Methods Only – Data access an querying of the ontology file.
   1.2. Ontology.cs – Query Text Only – Pre-built queries and template queries.
2. **Controllers Web GUI** – Handles web GUI requests to show web pages.
   2.1. HomeController.cs – The introduction page.
   2.2. OntologyUIController.cs – Ontology details such as classes, predicates, resources.
   2.3. SystemDetailsController.cs – Devices, device mappings, signals, …
3. **Controllers API** – Computer and application consumable access to ontology data.
   3.1. OntologyAPIController.cs – Ontology details such as classes, predicates, resources.
   3.2. SystemDetailsAPIController.cs – Devices, device mappings, signals, …
4. **Views** – Standard views for displaying ontology details
   4.1. List.cshtml – Displays a list of objects, such as class, predicates, or resources.
   4.2. Triple,cshtml – Displays all triples for a selected class, predicate, or resource.
   4.3. Details.cshtml – Displays forms for finding devices, device mappings, and signals.

## Conclusion

A fundamental understand of knowledge engineering has been obtained. This was obtained by practicing basic principles such as defining the topic area, performing interviews, creating concept maps, and developing RDF based file structures.

Using these combined skills, a web application was set up on the internet which provides both human-consumable and program-consumable access to the ontology and data related to the task of “Object Identification”. The link to this web application is shown below.

http://objectidentification.azurewebsites.net/

# Appendix 1 - ASP.net MVC C# Source Code

## 1.1 – Ontology.cs – Methods

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using VDS.RDF;
using VDS.RDF.Parsing;
using VDS.RDF.Query;
using VDS.RDF.Writing;
using System.IO;

namespace ObjectIdentificationWebApplication.Ontology
{
    public static class KnowledgeBase
    {
        //Fields
        public static string hostURI =
"http://"+HttpContext.Current.Request.Url.Authority+"/ObjectIdentification#";
        private static string ontologyFile =
HttpContext.Current.Server.MapPath("~/Ontology/ObjectIdentification.owl");
        private static Graph graph = new Graph();

        //Constructor
        static KnowledgeBase()
        {
            //Read text file
            string ontology = File.ReadAllText(ontologyFile);

            //Find original namespace
            int locLessThan = ontology.IndexOf("<")+1;
            int locGreaterThan = ontology.IndexOf(">");
            string origNamespace = ontology.Substring(locLessThan, locGreaterThan -
locLessThan);

            //Replace original namespace with host namespace
            ontology = ontology.Replace(origNamespace, hostURI);

            //Load into graph
            graph.LoadFromString(ontology);
        }

        #region Ontology
        //Lists
        public static List<SparqlResult> getList_OwlClass()
        {
            //Query graph for owl classes
            SparqlResultSet resultSet = graph.ExecuteQuery(Queries.listOwlClass) as
SparqlResultSet;

            //Return results
            return resultSet.ToList();
        }
        public static List<SparqlResult> getList_OwlPredicate()
        {
            //Query graph for owl classes
            SparqlResultSet resultSet = graph.ExecuteQuery(Queries.listOwlPredicate) as
SparqlResultSet;

            //Return results
            return resultSet.ToList();
        }
        public static List<SparqlResult> getList_OwlResource()
        {
            //Query graph for owl classes
            SparqlResultSet resultSet = graph.ExecuteQuery(Queries.listOwlResource) as
SparqlResultSet;

            //Return results
```

```
            return resultSet.ToList();
        }
        public static SparqlResultSet getList_CustomQuery(string qry)
        {
            //Query graph for owl classes
            SparqlResultSet resultSet = graph.ExecuteQuery(qry) as SparqlResultSet;

            //Return results
            return resultSet;
        }

        //Details
        public static SparqlResultSet subjectTriplesByName(string name)
        {
            //Search triples
            SparqlResultSet resultSet = graph.ExecuteQuery(Queries.subjectTriplesByName(name))
as SparqlResultSet;

            //Return results
            return resultSet;
        }
        #endregion

        #region System Details
        //Details
        public static SparqlResultSet getList_Device()
        {
            //Query graph for owl classes
            SparqlResultSet resultSet = graph.ExecuteQuery(Queries.listDevice) as
SparqlResultSet;

            //Return results
            return resultSet;
        }
        public static SparqlResultSet getList_DeviceMapping()
        {
            //Query graph for owl classes
            SparqlResultSet resultSet = graph.ExecuteQuery(Queries.listDeviceMapping) as
SparqlResultSet;

            //Return results
            return resultSet;
        }
        public static SparqlResultSet getList_DeviceMapping(int deviceID)
        {
            //Query graph for owl classes
            SparqlResultSet resultSet =
graph.ExecuteQuery(Queries.listDeviceMappingByID(deviceID)) as SparqlResultSet;

            //Return results
            return resultSet;
        }

        //Signals
        public static SparqlResultSet getList_DeviceResponseSignal(int deviceID)
        {
            //Query graph for owl classes
            SparqlResultSet resultSet =
graph.ExecuteQuery(Queries.listDeviceResponseSignal(deviceID)) as SparqlResultSet;

            //Return results
            return resultSet;
        }
        public static SparqlResultSet getList_DeviceControlSignal(int deviceID)
        {
            //Query graph for owl classes
            SparqlResultSet resultSet =
graph.ExecuteQuery(Queries.listDeviceControlSignal(deviceID)) as SparqlResultSet;

            //Return results
            return resultSet;
```

```
        }
        #endregion

        //Methods
        public static string GetNodeString(INode node)
        {
            string s = node.ToString();
            switch (node.NodeType)
            {
                case NodeType.Uri:
                    int lio = s.LastIndexOf('#');
                    if (lio == -1)
                        return s;
                    else
                        return s.Substring(lio + 1);
                case NodeType.Literal:
                    int delIndex = s.IndexOf("^^");
                    if (delIndex > -1)
                        return s.Substring(0, delIndex);
                    else
                        return string.Format("{0}", s);
                default:
                    return s;
            }
        }
        public static string GetNodeStringWithPrefix(INode node)
        {
            string s = node.ToString();
            switch (node.NodeType)
            {
                case NodeType.Uri:
                    int lioDash = s.LastIndexOf('/')+1;
                    int lioHash = s.LastIndexOf('#');
                    if (lioHash == -1)
                        return s;
                    else
                    {
                        string namespaceText = s.Substring(lioDash, lioHash - lioDash);
                        string name = s.Substring(lioHash + 1);
                        string prefix = namespaceText;

                        //Check for RDF and RDF schema
                        if(namespaceText.Contains("rdf"))
                        {
                            prefix = "rdf";
                            if (namespaceText.Contains("schema"))
                                prefix = "rdfs";
                        }

                        //Combine and return
                        return prefix + ":" + name; ;
                    }


                case NodeType.Literal:
                    int delIndex = s.IndexOf("^^");
                    if (delIndex > -1)
                        return s.Substring(0, delIndex);
                    else
                        return string.Format("{0}", s);
                default:
                    return s;
            }
        }
        public static string exampleQuery
        {
            get {return Queries.listOwlClass;}
        }
    }
}
```

## 1.2 – Ontology.cs – Query Text

```
private static class Queries
    {
      //URL of current host
      private static string kbNamespace = hostURI;

      //Ontology
      public static string subjectTriplesByName(string name)
      {
        string qry = "PREFIX : " +"<"+kbNamespace+">"+
          @"
          PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
          PREFIX owl: <http://www.w3.org/2002/07/owl#>
          PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
          PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

          SELECT DISTINCT ?predicate ?object
          WHERE
          {
            :{name} ?predicate ?object
          }
          ";

        qry = qry.Replace("{name}", name);

        return qry;
      }
      public static string listOwlClass
      {
        get
        {
          string qry = "PREFIX : " + "<" + kbNamespace + ">" +
            @"
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT DISTINCT ?subject

WHERE
{
  ?subject rdf:type owl:Class
}

ORDER BY ?subject
            ";
          return qry;
        }
      }
      public static string listOwlPredicate
      {
        get
        {
          string qry = "PREFIX : " + "<" + kbNamespace + ">" +
            @"
            PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
            PREFIX owl: <http://www.w3.org/2002/07/owl#>
            PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
            PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

            SELECT DISTINCT ?subject ?o

            WHERE
            {
              ?subject rdf:type ?o.
              FILTER (?o IN (owl:ObjectProperty, owl:DatatypeProperty) )
            }
```

```
            ORDER BY ?subject
            ";
        return qry;
    }
}
public static string listOwlResource
{
    get
    {
        string qry = "PREFIX : " + "<" + kbNamespace + ">" +
            @"
            PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
            PREFIX owl: <http://www.w3.org/2002/07/owl#>
            PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
            PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

            SELECT DISTINCT ?subject

            WHERE
            {
                ?subject rdf:type owl:NamedIndividual
            }

            ORDER BY ?subject
            ";
        return qry;
    }
}

//System Details
public static string listDevice
{
    get
    {
        string qry = "PREFIX : " + "<" + kbNamespace + ">" +
            @"
            PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
            PREFIX owl: <http://www.w3.org/2002/07/owl#>
            PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
            PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

            SELECT
            (str(?deviceID) as ?id)
            (?device as ?filetag)
            (str(?deviceName) as ?name)
            (str(?deviceDescription) as ?description)

            WHERE {
            #Get a list of devices
            ?device a :Device.
            ?device :ID ?deviceID.

            #Get the device's name
            ?deviceLabel :ID ?deviceID.
            ?deviceLabel :name ?deviceName.
            ?deviceLabel :description ?deviceDescription.
            }
            ORDER BY ?id
            ";
        return qry;
    }
}
public static string listDeviceMapping
{
    get
    {
        string qry = "PREFIX : " + "<" + kbNamespace + ">" +
```

```
            @"
            PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
            PREFIX owl: <http://www.w3.org/2002/07/owl#>
            PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
            PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

            SELECT
            (str(?linkID) as ?EventObject_ID)
            (str(?controlValue) as ?control)
            (str(?outputChannelID) as ?OutputChannel_ID)
            (str(?outputDeviceID) as ?OutputDevice_ID)

            (str(?responseValue) as ?response)
            (str(?inputChannelID) as ?InputChannel_ID)
            (str(?inputDeviceID) as ?InputDevice_ID)

            WHERE
            {
            #Get the linking event objects
            ?link a :EventObject .
            ?link :ID ?linkID .

            # Get the control signal
            ?link :controlSignal ?controlSignal .
            ?controlSignal :Value ?controlValue .

            # Get the output channel
            ?controlSignal :dataChannel ?outputChannel.
            ?outputChannel :ID ?outputChannelID.

            #Get output device
            ?outputDevice a :Device.
            ?outputDevice :dataChannel ?outputChannel.
            ?outputDevice :ID ?outputDeviceID.

            # Get the response signal
            ?link :responseSignal ?responseSignal .
            ?responseSignal :Value ?responseValue .

            # Get the input channel
            ?responseSignal :dataChannel ?inputChannel.
            ?inputChannel :ID ?inputChannelID.

            #Get input device
            ?inputDevice a :Device.
            ?inputDevice :dataChannel ?inputChannel.
            ?inputDevice :ID ?inputDeviceID.
            }
            ";
        return qry;
    }
}
public static string listDeviceMappingByID(int deviceID)
{
        string qry = "PREFIX : " + "<" + kbNamespace + ">" +
            @"
            PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
            PREFIX owl: <http://www.w3.org/2002/07/owl#>
            PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
            PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

            SELECT
            (str(?linkID) as ?EventObject_ID)
            (str(?controlValue) as ?control)
            (str(?outputChannelID) as ?OutputChannel_ID)
            (str(?outputDeviceID) as ?OutputDevice_ID)

            (str(?responseValue) as ?response)
```

```
            (str(?inputChannelID) as ?InputChannel_ID)
            (str(?inputDeviceID) as ?InputDevice_ID)

            WHERE
            {
            #Get the linking event objects
            ?link a :EventObject .
            ?link :ID ?linkID .

            # Get the control signal
            ?link :controlSignal ?controlSignal .
            ?controlSignal :Value ?controlValue .

            # Get the output channel
            ?controlSignal :dataChannel ?outputChannel.
            ?outputChannel :ID ?outputChannelID.

            #Get output device
            ?outputDevice a :Device.
            ?outputDevice :dataChannel ?outputChannel.
            ?outputDevice :ID ?outputDeviceID.

            # Get the response signal
            ?link :responseSignal ?responseSignal .
            ?responseSignal :Value ?responseValue .

            # Get the input channel
            ?responseSignal :dataChannel ?inputChannel.
            ?inputChannel :ID ?inputChannelID.

            #Get input device
            ?inputDevice a :Device.
            ?inputDevice :dataChannel ?inputChannel.
            ?inputDevice :ID ?inputDeviceID.

            #Filter
            ?inputDevice :ID {deviceID} .
            ?outputDevice :ID {deviceID} .
            }
            ";

      //replace deviceID
      qry = qry.Replace("{deviceID}", deviceID.ToString());

         return qry;
   }

   //Signals
   public static string listDeviceResponseSignal(int deviceID)
   {
      string qry = "PREFIX : " + "<" + kbNamespace + ">" +
         @"
         PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
         PREFIX owl: <http://www.w3.org/2002/07/owl#>
         PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
         PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

         SELECT
         (str(?inputChannelID) as ?InputChannelID)
         (str(?responseSignalID) as ?ResponseSignalID)
         (str(?responseSignalValue) as ?ResponseSignalValue)

         WHERE
         {
         #Get device
         ?device a :Device.
         ?device :ID {deviceID} .
```

```
            #Get input channel of device
            ?device :dataChannel ?inputChannel.
            ?inputChannel a :InputChannel.
            ?inputChannel :ID ?inputChannelID.

            #Get signals collected by this input channel
            ?responseSignal a :ResponseSignal.
            ?responseSignal :dataChannel ?inputChannel.

            #Get response signal ID and value
            ?responseSignal :ID ?responseSignalID.
            ?responseSignal :Value ?responseSignalValue.
            }
               ";

        //replace deviceID
        qry = qry.Replace("{deviceID}", deviceID.ToString());

        return qry;
    }
    public static string listDeviceControlSignal(int deviceID)
    {
        string qry = "PREFIX : " + "<" + kbNamespace + ">" +
            @"
            PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
            PREFIX owl: <http://www.w3.org/2002/07/owl#>
            PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
            PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

            SELECT
            (str(?outputChannelID) as ?OutputChannelID)
            (str(?controlSignalID) as ?ControlSignalID)
            (str(?controlSignalValue) as ?ControlSignalValue)

            WHERE
            {
            #Get device
            ?device a :Device.
            ?device :ID {deviceID} .

            #Get input channel of device
            ?device :dataChannel ?outputChannel.
            ?outputChannel a :OutputChannel.
            ?outputChannel :ID ?outputChannelID.

            #Get signals collected by this input channel
            ?controlSignal a :ControlSignal.
            ?controlSignal :dataChannel ?outputChannel.

            #Get response signal ID and value
            ?controlSignal :ID ?controlSignalID.
            ?controlSignal :Value ?controlSignalValue.
            }
               ";

        //replace deviceID
        qry = qry.Replace("{deviceID}", deviceID.ToString());

        return qry;
    }
}
```

## 2.1 – HomeController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using ObjectIdentificationWebApplication.Ontology;
using VDS.RDF.Query;
using ObjectIdentificationWebApplication.Models;

namespace ObjectIdentificationWebApplication.Controllers
{
    public class HomeController : Controller
    {
        //Home
        [Route("")]
        public ViewResult Index()
        {
            ViewBag.Title = "Home";
            return View();
        }

        [Route("Classes")]
        public ViewResult Classes()
        {
            ViewBag.Title = "Classes";
            ViewBag.Prefix = "Class";
            List<SparqlResult> kb = KnowledgeBase.getList_OwlClass();
            return View("List", kb);
        }

        [Route("Predicates")]
        public ViewResult Predicates()
        {
            ViewBag.Title = "Predicates";
            ViewBag.Prefix = "Predicate";
            List<SparqlResult> kb = KnowledgeBase.getList_OwlPredicate();
            return View("List", kb);
        }

        [Route("Resources")]
        public ViewResult Resources()
        {
            ViewBag.Title = "Resources";
            ViewBag.Prefix = "Resource";
            List<SparqlResult> kb = KnowledgeBase.getList_OwlResource();
            return View("List", kb);
        }

        [HttpPost]
        [ValidateInput(false)]
        [Route("SPARQL")]
        public ViewResult SparqlPost(CustomQry cQry)
        {
            return View("Sparql", cQry);
        }

        [Route("SPARQL")]
        public ViewResult Sparql()
        {
            return View("Sparql");
        }
    }
}
```

## 2.2 – OntologyUIController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using VDS.RDF;
using VDS.RDF.Parsing;
using VDS.RDF.Query;
using System.IO;
using ObjectIdentificationWebApplication.Ontology;

namespace ObjectIdentificationWebApplication.Controllers
{
    public class OntologyUIController : Controller
    {
        [Route("Class/{className}")]
        public ViewResult Class(string className)
        {
            ViewBag.Title = className;
            ViewBag.Prefix = "Class";
            SparqlResultSet triples = KnowledgeBase.subjectTriplesByName(className);
            return View("Triple", triples);
        }

        [Route("Predicate/{predicate}")]
        public ViewResult Predicate(string predicate)
        {
            ViewBag.Title = predicate;
            ViewBag.Prefix = "Predicate";
            SparqlResultSet triples = KnowledgeBase.subjectTriplesByName(predicate);
            return View("Triple", triples);
        }

        [Route("Resource/{resource}")]
        [Route("Res/{resource}")]
        public ViewResult Resource(string resource)
        {
            ViewBag.Title = resource;
            ViewBag.Prefix = "Resource";
            SparqlResultSet triples = KnowledgeBase.subjectTriplesByName(resource);
            return View("Triple", triples);
        }
    }
}
```

2.3 – SystemDetailsController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using Swashbuckle.Swagger.Annotations;
using VDS.RDF.Query;
using ObjectIdentificationWebApplication.Ontology;

namespace ObjectIdentificationWebApplication.Controllers
{
    public class SystemDetailsUIController : Controller
    {
        [Route("SystemDetails")]
        public ViewResult Index()
        {
            return View("Details");
        }

        [HttpPost]
        [Route("SystemDetails")]
        public ViewResult ActionSubmitted(object p)
        {
            SparqlResultSet kb = null;
            try
            {
                //Get requested command
                string queryName = Request["queryName"];

                //Select action to perform
                switch (queryName)
                {
                    case "Devices":
                        {
                            kb = KnowledgeBase.getList_Device();
                            break;
                        }
                    case "DeviceMappings":
                        {
                            int deviceID = Convert.ToInt32(Request["DeviceID"]);
                            kb = KnowledgeBase.getList_DeviceMapping(deviceID);
                            break;
                        }
                    case "DeviceResponseSignals":
                        {
                            int deviceID = Convert.ToInt32(Request["DeviceID"]);
                            kb = KnowledgeBase.getList_DeviceResponseSignal(deviceID);
                            break;
                        }
                    case "DeviceControlSignals":
                        {
                            int deviceID = Convert.ToInt32(Request["DeviceID"]);
                            kb = KnowledgeBase.getList_DeviceControlSignal(deviceID);
                            break;
                        }
                }
            }
            catch { }


            return View("Details", kb);
        }
    }
}
```

## 3.1 – OntologyAPIController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using Swashbuckle.Swagger.Annotations;
using VDS.RDF.Query;
using ObjectIdentificationWebApplication.Ontology;

namespace ObjectIdentificationWebApplication.Controllers
{
    public class OntologyApiController : ApiController
    {
        [SwaggerOperation("GetList_OwlClass", Tags = new[] { "Ontology" })]
        [Route("api/Ontology/GetList_OwlClass")]
        public List<SparqlResult> GetList_OwlClass()
        {
            List<SparqlResult> kb = KnowledgeBase.getList_OwlClass();
            return kb;
        }

        [SwaggerOperation("GetList_OwlPredicate", Tags = new[] { "Ontology" })]
        [Route("api/Ontology/GetList_OwlPredicate")]
        public List<SparqlResult> GetList_OwlPredicate()
        {
            List<SparqlResult> kb = KnowledgeBase.getList_OwlPredicate();
            return kb;
        }

        [SwaggerOperation("GetList_OwlResource", Tags = new[]{"Ontology"})]
        [Route("api/Ontology/GetList_OwlResource")]
        public List<SparqlResult> GetList_OwlResource()
        {
            List<SparqlResult> kb = KnowledgeBase.getList_OwlResource();
            return kb;
        }

        [SwaggerOperation("Get_SubjectTriples", Tags = new[] { "Ontology" })]
        [Route("api/Ontology/Get_SubjectTriples")]
        public SparqlResultSet Get_SubjectTriples(string subject)
        {
            SparqlResultSet kb = KnowledgeBase.subjectTriplesByName(subject);
            return kb;
        }
    }
}
```

## 2.3 – SystemDetailsAPIController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using Swashbuckle.Swagger.Annotations;
using VDS.RDF.Query;
using ObjectIdentificationWebApplication.Ontology;

namespace ObjectIdentificationWebApplication.ControllersAPI
{
    public class SystemDetailsApiController : ApiController
    {
        [SwaggerOperation("GetList_Device", Tags = new[] { "System Details" })]
        [Route("api/SystemDetails/GetList_Device")]
        public SparqlResultSet GetList_Device()
        {
            SparqlResultSet kb = KnowledgeBase.getList_Device();
            return kb;
        }

        [SwaggerOperation("GetList_DeviceMapping", Tags = new[] { "System Details" })]
        [Route("api/SystemDetails/GetList_DeviceMapping")]
        public SparqlResultSet GetList_DeviceMapping()
        {
            SparqlResultSet kb = KnowledgeBase.getList_DeviceMapping();
            return kb;
        }

        [SwaggerOperation("GetList_DeviceMapping", Tags = new[] { "System Details" })]
        [Route("api/SystemDetails/GetList_DeviceMapping/{deviceID}")]
        public SparqlResultSet GetList_DeviceMapping(int deviceID)
        {
            SparqlResultSet kb = KnowledgeBase.getList_DeviceMapping(deviceID);
            return kb;
        }

        [SwaggerOperation("GetList_DeviceResponseSignal", Tags = new[] { "System Details"
})]
        [Route("api/SystemDetails/GetList_DeviceResponseSignal/{deviceID}")]
        public SparqlResultSet GetList_DeviceResponseSignal(int deviceID)
        {
            SparqlResultSet kb = KnowledgeBase.getList_DeviceResponseSignal(deviceID);
            return kb;
        }

        [SwaggerOperation("GetList_DeviceControlSignal", Tags = new[] { "System Details"
})]
        [Route("api/SystemDetails/GetList_DeviceControlSignal/{deviceID}")]
        public SparqlResultSet GetList_DeviceControlSignal(int deviceID)
        {
            SparqlResultSet kb = KnowledgeBase.getList_DeviceControlSignal(deviceID);
            return kb;
        }
    }
}
```

## 4.1 – List.cs

```
@using VDS.RDF.Query;
@model List<SparqlResult>

<h2>@ViewBag.Title</h2>

<table style="width:100%;">
    @for (int i = 0; i < Model.Count; i++)
    {
    <tr>
        @for (int d = 0; d < 3; d++)
        {
            //Get details
            string itemBaseURI = Model[i][0].GraphUri.AbsoluteUri;
            string itemName = Model[i][0].ToString().Replace(itemBaseURI, "").Replace("#",
"");
            string itemURI = itemBaseURI + "/" + itemName;
            string prefix = ViewBag.Prefix;

            //Display in cell
            <td><a href="@prefix/@itemName">@itemName</a></td>

            //Increment counter
            {
                //Increment
                i++;
                //Check for early end condition
                if (i >= Model.Count) { break; }
            }
        }
    </tr>
    }
</table>
```

## 4.2 – Triple.cs

```
@using VDS.RDF;
@using VDS.RDF.Query;
@using ObjectIdentificationWebApplication.Ontology;
@model SparqlResultSet

<style>
    table.properties {
        border: 1px solid #DDDDDD;
    }

        table.properties td, th {
            border: 1px solid #DDDDDD;
            padding-left: 5px;
            padding-right: 5px;
        }
</style>

<h2>@ViewBag.Prefix: @ViewBag.Title</h2>
<hr />
<h4>Description:</h4>
<div class="">
@foreach (SparqlResult sr in Model)
{
    if (sr["predicate"].ToString().ToLower().EndsWith("#comment"))
    {
        <div>@KnowledgeBase.GetNodeString(sr["object"])</div>
    }
}
</div>
<br />

<hr />
<h4>Object Properties:</h4>
<table class="properties">
    <tr>
        <th>Predicate</th>
        <th>Object</th>
    </tr>
    @foreach (SparqlResult sr in Model)
    {
        //Get details
        string predicateName = KnowledgeBase.GetNodeString(@sr["predicate"]);
        string predicateNameWithPrefix =
KnowledgeBase.GetNodeStringWithPrefix(@sr["predicate"]);
        string objectName = KnowledgeBase.GetNodeString(@sr["object"]);
        string objectNameWithPrefix = KnowledgeBase.GetNodeStringWithPrefix(@sr["object"]);

        //Determine predicate URI
        string predicateURI = sr["predicate"].ToString();
        if (sr["predicate"].ToString().StartsWith(KnowledgeBase.hostURI))
        { predicateURI = "/Predicate/" + predicateName; }

        //Determine object URI
        string objectURI = sr["object"].ToString();
        if (sr["object"].ToString().StartsWith(KnowledgeBase.hostURI))
        { objectURI = "/Resource/" + objectName; }

        <tr>
            <td><a href="@predicateURI">@predicateNameWithPrefix</a></td>
            @if (@sr["object"].NodeType == NodeType.Uri)
                {<td><a href="@objectURI">@objectNameWithPrefix</a></td> }
            else
                { <td>@objectName</td> }
        </tr>
    }
</table>
```

## 4.2 – Details.cs

```
@using System.Collections;
@using ObjectIdentificationWebApplication.Models
@using ObjectIdentificationWebApplication.Ontology;
@using VDS.RDF.Query;
@model SparqlResultSet

@{
    ViewBag.Title = "System Details";
}

<style>
    textarea {
        width: 70%;
        height: 100%;
        max-width: 10000px;
    }
    table.withBorders td,th{
        border: 1px solid #DDDDDD;
        padding-left: 5px;
        padding-right: 5px;
    }
</style>

<h2>@ViewBag.Title</h2>


@using (Html.BeginForm())
{
    <div class="form-group col-md-12">
        <div>
            @Html.Hidden("queryName", "Devices")
            <input type="submit" value="Show Devices" class="btn btn-default" />
        </div>
    </div>
}

@using (Html.BeginForm())
{
    <div class="form-group col-md-12">
        <div>
            @Html.Hidden("queryName", "DeviceMappings")
            <input type="submit" value="Show Device Mappings" class="btn btn-default" />
            DeviceID: <input type="text" id="DeviceID" name="DeviceID" size="3"
value="@Request["DeviceID"]"/>
        </div>
    </div>
}

@using (Html.BeginForm())
{
    <div class="form-group col-md-12">
        <div>
            @Html.Hidden("queryName", "DeviceResponseSignals")
            <input type="submit" value="Show Device Response Signals" class="btn btn-
default" />
            DeviceID: <input type="text" id="DeviceID" name="DeviceID" size="3"
value="@Request["DeviceID"]" />
        </div>
    </div>
}

@using (Html.BeginForm())
{
    <div class="form-group col-md-12">
        <div>
            @Html.Hidden("queryName", "DeviceControlSignals")
```

```
            <input type="submit" value="Show Device Control Signals" class="btn btn-
default" />
            DeviceID: <input type="text" id="DeviceID" name="DeviceID" size="3"
value="@Request["DeviceID"]" />
        </div>
    </div>
}



@if (Model != null && Model.Count>0)
{
    <h4>Results</h4>
    <div>
        <table class="withBorders">
            <thead>
                @foreach (string header in Model.Variables)
                {
                <th>@header.ToString()</th>
                }
            </thead>
            <tbody>
                @foreach (SparqlResult row in Model.ToList())
                {
                    <tr>
                        @foreach (string header in Model.Variables)
                        {
                            <td>@row[header].ToString()</td>
                        }
                    </tr>
                }
            </tbody>
            </table>
        </div>
}

@section Scripts
{
    <script type="text/javascript">

    $("textarea").height($("textarea")[0].scrollHeight);
    </script>

}
```