|  |
| --- |
| MSC SOFTWARE DEVELOPMMENT |
| Advanced Object Technology |
| Assignment 2 |
|  |
| **Jimmy Collins – R00145569 – Jimmy.Collins@mycit.ie** |
| **5/2/2017** |

|  |
| --- |
|  |

Contents

[Section 1 – Introduction 2](#_Toc480483569)

[Section 2 – Demonstrated Usage of Defined Object Technologies 3](#_Toc480483570)

[Template Pattern 3](#_Toc480483571)

[Proxy Pattern - Remote Proxy 4](#_Toc480483572)

[Usage of Database for Data Persistence 5](#_Toc480483573)

[Façade Pattern 7](#_Toc480483574)

[Command Pattern 8](#_Toc480483575)

[MVC Compound Pattern 9](#_Toc480483576)

[Section 3 – System UML Diagram 10](#_Toc480483577)

[Section 4 – Screen Captures of Most Impressive Code Snippets 10](#_Toc480483578)

[Section 5 – Evaluation of Work 10](#_Toc480483579)

[Appendix 1 – Database Setup Instructions 11](#_Toc480483580)

[Appendix 2 – Possible Development Environment Certificate Issue 12](#_Toc480483581)

[Appendix 3 – Code Commit Analysis 14](#_Toc480483582)

[Contributions 14](#_Toc480483583)

[Code Frequency 14](#_Toc480483584)

## Section 1 – Introduction

TODO

TODO – Improvements on Assignment 1 code

* UI changes to split out general / advanced UI
* Changes to strategy pattern implementation so it just returns a chart - decoupled it from any particular UI element (should have done this in the first assignment – it was tightly coupled to a ‘tab’ element).
* Refactored some code using the Template pattern – see section 2.

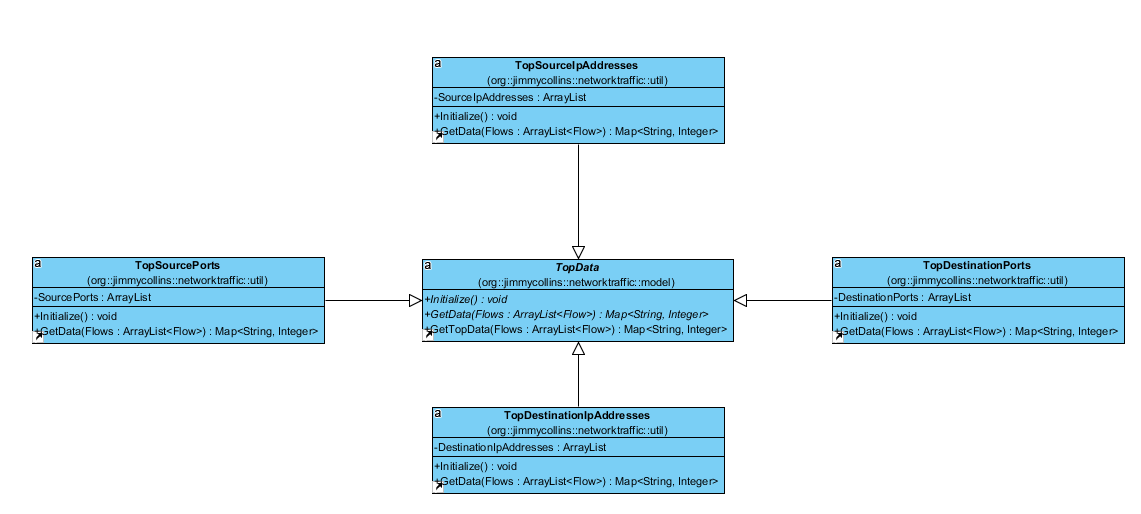
TODO

*Figure 1: The main application user interface.*

## Section 2 – Demonstrated Usage of Defined Object Technologies

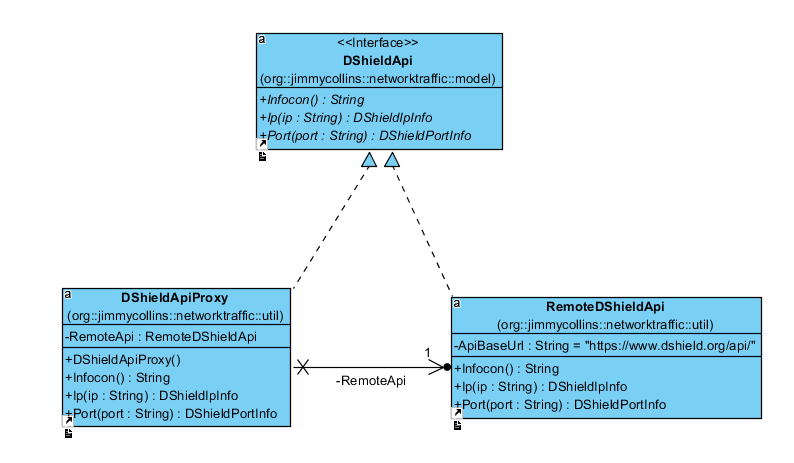
### Template Pattern

I used the template pattern to refactor some code from Assignment 1, which is the logic that is used to generate the top source/destination ports and top source/destination IP address data from the data parsed. Now the TopData class exists, which is an abstract class with the final method GetTopData, and the helper methods Initialize and GetData which are overridden in each of the concrete classes.



### Proxy Pattern - Remote Proxy

The usage of a remote proxy is demonstrated in UML below. In my application I integrate with the DShield[[1]](#footnote-1) API maintained by the Internet Storm Center (ISC). I decided to approach some of the advanced analysis requirements from a security perspective, so I use this API to return some security information on the top IP addresses and ports that are being parsed from the Netflow file. Please see **Appendix 2** for the solution to a possible certificate issue that may occur the first time you run the application and attempt to connect to the API.

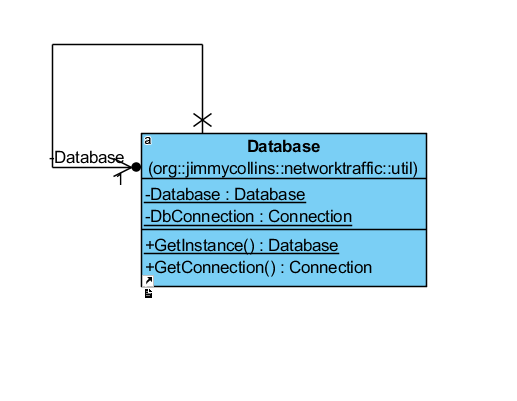


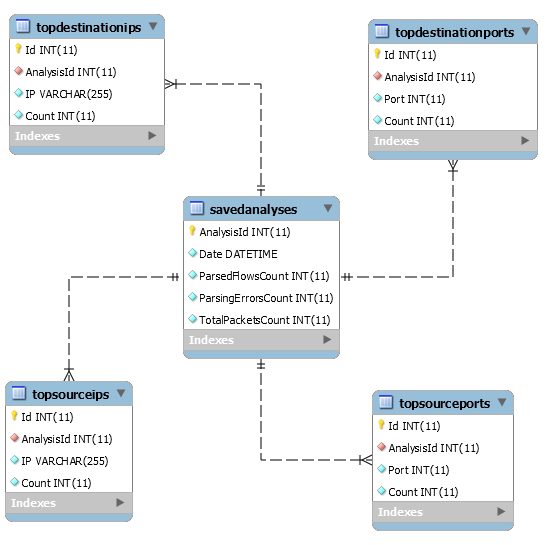
### Usage of Database for Data Persistence

Within the application the user has the ability to save an analysis session to the database. This saves the Top Source/Destination Port and Top Source/Destination IP data to a MySQL database (see **Appendix 1** for the setup details). I consciously chose not to save the data in the ‘Security Analysis’ section, as this data is something that a user would want to check each time an analysis is run, as potential threats are constantly evolving.

The database schema is shown on the next page.

Below is shown the class I use to handle the connection to the database, which implements the Singleton Design Pattern.



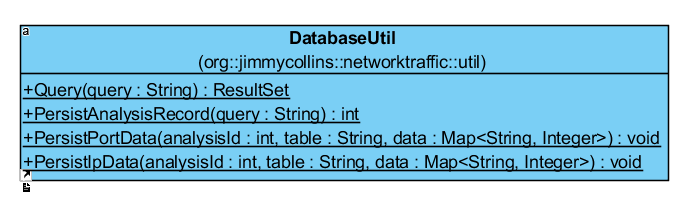


### Façade Pattern

I used the Façade Pattern to refactor my database querying functionality after it had been written, the code was quite complex before this.

I ended up creating a new class, DatabaseUtil, which is a wrapper that contains a set of members that are used for database operations that are easily understood and simple to use.

These members access the database on behalf of the facade user, hiding the implementation details.

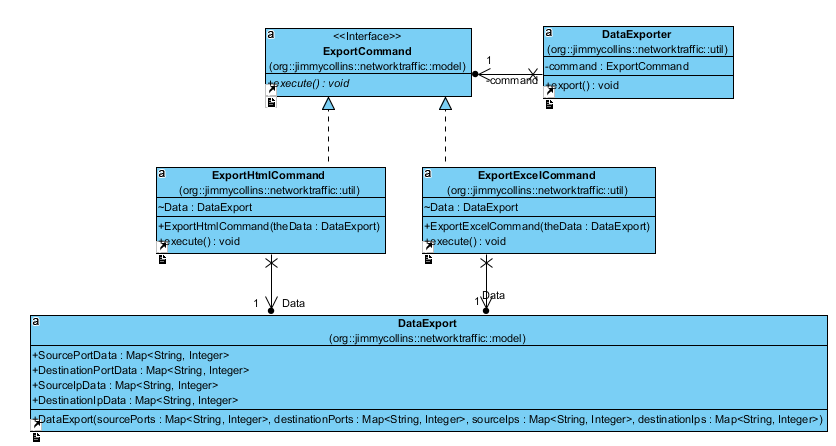


### Command Pattern

I used the command pattern for the functionality that allows a user to export an analysis to different formats (e.g. to email to someone). Currently Microsoft Excel and HTML formats are supported as export formats.

ExportCommand is the interface which both concrete commands ExportExcelCommand and ExportHtmlCommand implement. DataExporter is the receiver and the main UI controller class is the invoker. The DataExport class is just a class used to pass the relevant data to be exported to the constructors of the concrete commands.

**Notes** – adding a new export type could be facilitated easily in this design. I’m also aware that a Strategy Pattern could have been used here to achieve the same functionality.



### MVC Compound Pattern

TODO

## Section 3 – System UML Diagram

TODO

## Section 4 – Screen Captures of Most Impressive Code Snippets

TODO

## Section 5 – Evaluation of Work

TODO

## Appendix 1 – Database Setup Instructions

The below steps should be taken to create the necessary schema in MySQL for the application. This is used to save the analysis data.

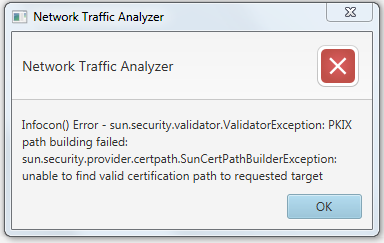
1. Open MySQL Workbench and create a new connection.
2. Open 'schema-no-data.sql' from the 'db' folder in the source tree.
3. Execute this SQL to setup the necessary tables and relationships.

Note I have used MySQL Workbench to create the schema etc. but this could be run from the command line via (ensure to create the networktraffic database first):

mysql -u username -p networktraffic < schema-no-data.sql

## Appendix 2 – Possible Development Environment Certificate Issue

The following issue may occur when you try to run the ‘Security Analysis’ feature under the ‘Advanced Analysis’ menu.



This appears to be because the server has a self-signed certificate.

To work around this issue and get the application running on your local environment, you can add this certificate to the list of trusted certificates of your JVM.

If you see this issue on your environment, please follow the below steps to fix it.

* Go to [www.dshield.org](http://www.dshield.org) and export the certificate. This can be accomplished (using Chrome) as follows:
  + Navigate to [www.dshield.org](http://www.dshield.org) at least once.
  + Next go to chrome://settings/, show advanced settings and click on ‘Manage Certificates…’.
  + In the dialog find the below certificate, and use the export option to export it locally on your machine as a .CER file.



* Next, you need to add this to the list of trusted certificates of your JVM. Go to your JRE\_HOME/bin or JDK/JRE/bin
* Run the below command (substituting the path to your JDK directory and the path to the certification downloaded above):



* If promoted for a password, use the default key-store password ‘changeit’.
* When prompted to trusted the certificate, type ‘yes’.

The application should now work as expected. I’ve successfully tested this procedure on two machines.

## Appendix 3 – Code Commit Analysis

### Contributions

### Code Frequency

1. <https://www.dshield.org> [↑](#footnote-ref-1)