

VIMDB : Constructing a Visualisation for the Internet Movie Database

Project Proposal

Jareth Disley

Abstract

The Internet Movie Database (IMDB) is one of the best known internet databases containing information about thousands of films and TV shows from around the world. The IMDB among other methods provides this data in textual form. This report presents a proposal for a project in this area with particular emphasis on data visualisation, and component re-usability.

Project Aims

- To reverse engineer / re-design an ER diagram of the schema
- Design an XML document format to represent the textual data
- Create a component to convert textual data into the above format
- Create a component which reads the above format and loads into a DBMS
- Research, design and create an aesthetically pleasing data visualisation

Project Phases

- Analysis of the underlying data and design of components to convert to XML
- Design / Implementation of database as well as database loading
- Research / design / implementation of data visualisation

The expected outcomes of the project are an application that allows the complex data of the IMDB to be visualised in a useful and elegant way, as well as re-usable components should the underlying data change.

1. Introduction

This project will be focussing on visualising the complex textual representation of the IMDB so that the relationships within it can be explored more efficiently, and with more functionality than the standard web interface of the IMDB.

The proposed system will include a data visualisation component which will allow the data from the text files to be represented visually, so that the relationships between data can be easily shown. Possible developments should the project progress well could include additional functionality such as the ability for films to be suggested to a user based on preferences and other heuristics. Other developments could include the ability to automatically source images when representing films and include this as part of the data visualisation.

The next section in this report will talk about the background of the IMDB and data visualisation techniques to set the scene for the rest of the report. The proposed project is split into *Aims and Objectives* and *Methodology*. The methodology section will describe the type of software engineering approach that will be used.

The evaluation section will discuss the ways that the system can be evaluated in order to find out if the original aims and objectives have been met. In the programme of work section each task will be described in terms of how long it will take so that a plan in the form of a Gantt chart can be produced as an overall schedule of work.

The penultimate section – Resources Required will discuss what resources are needed in order to implement the project. The report ends with a set of references.

2. Background

The IMDB is a database which contains information about movies and TV shows, as well as Computer games. The main interface for viewing information allows users to search for a specific title or select from a set of common list such as “*Top Movies*”. A query is generated and sent to the Database Management System which then returns a full or empty set of information back to the user. The returned list will typically only have a small amount of information in relation to each of the results therefore it is necessary to choose a specific result in order to generate another query and retrieve more detailed information.

The diagram below in Figure 1 diagrammatically shows how a user typically interacts with the system.

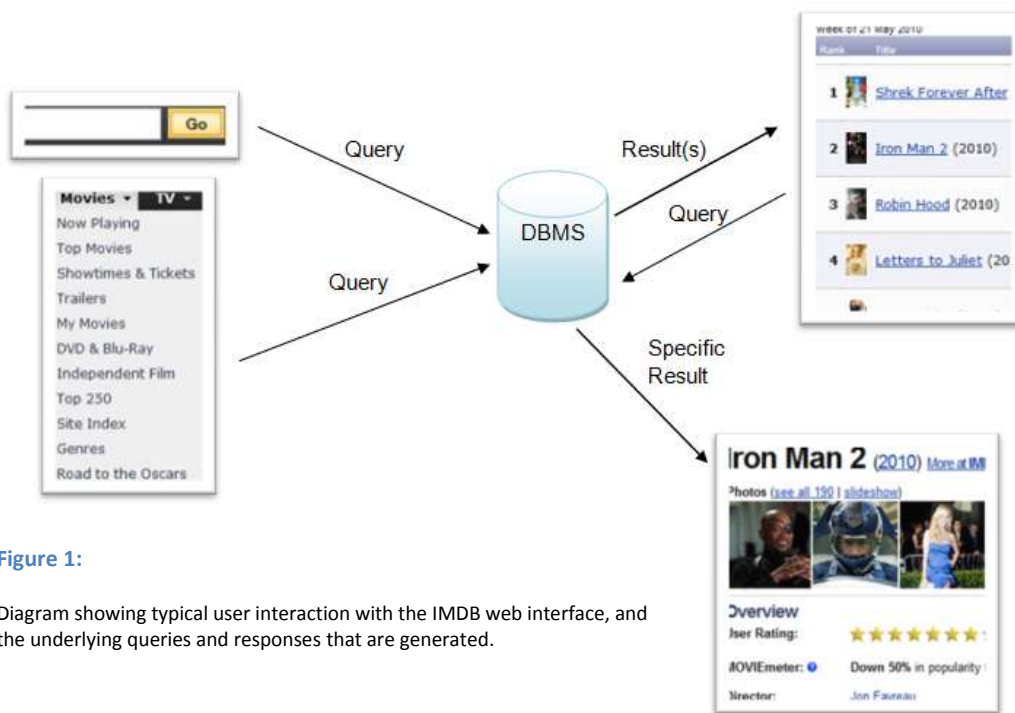


Figure 1:

Diagram showing typical user interaction with the IMDB web interface, and the underlying queries and responses that are generated.

Currently a web interface is used to interact with the IMDB and server side code deals with communicating with the DBMS to display information. There are often large amounts of information associated with different titles and relationships can be explored by clicking on links within the displayed information as can be seen in Figures 2 and 3.

Relationships are explored through the use of hyperlinks which jump to a new page where a query may be generated to retrieve further information. This approach of using hyperlinks to explore information on the IMDB means that the relationships are not represented visually, and can allow the user to lose track of where they are in relation to the original data.

Currently the web interface is the only way to explore the information and relationships of the IMDB in some form of visual environment. The data within the IMDB is kept up-to date by allowing people to submit information about titles which should then be checked by operatives and submitted to the database. Text files that are provided by the IMDB are released regularly, however they may not be as up-to date as their live database.

An example of an early (1995) data visualisation for the IMDB is Ben Schneiderman's “*FilmFinder*” which allowed large amounts of data from a DBMS to be represented using a Starfield display and manipulated with AlphaSliders and other controls. The controls within the application allow queries to be dynamically generated and sent to a DBMS with the result being displayed in a Starfield display. The “*FilmFinder*” could easily represent many thousand records but not display very much information about each of them unless they were explicitly selected to be displayed in the information box. An improvement to this could be to display more information in relation to a smaller number of records to allow the user to make more informed decisions and an example of the “*FilmFinder*” can be found in Figure 4 below.

There are currently several legacy applications which make use of the provided IMDB text files however they are command line based and do not provide any data visualisation. When using the text files it is necessary to regularly parse these files in order to ensure that the application that relies upon it can be kept up-to date.

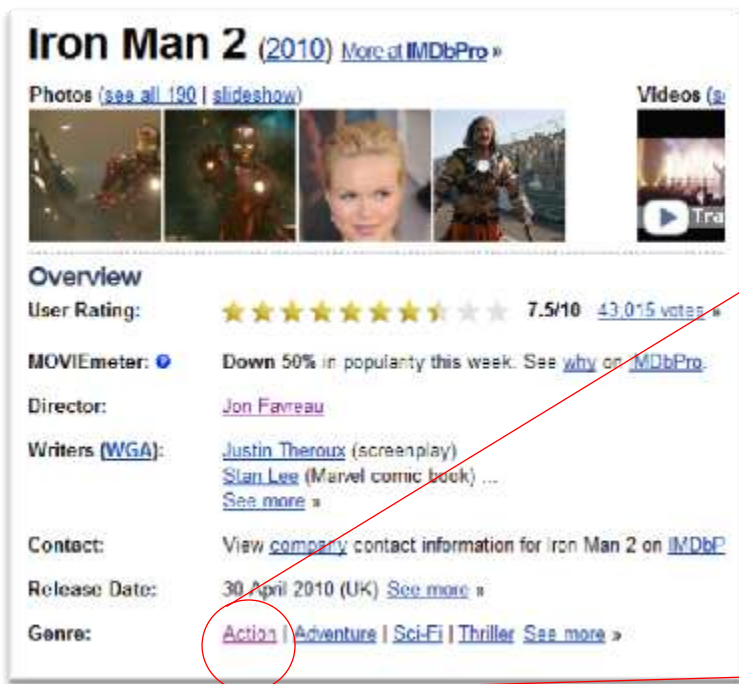


Figure 2 (Above): Screenshot of a typical movie information page on the IMDB web interface, with the Genre relationship hyperlink highlighted.

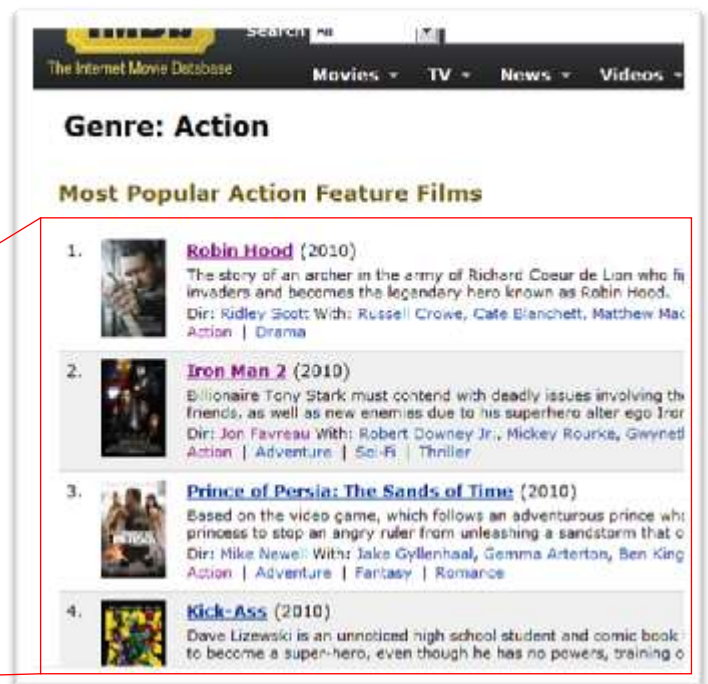


Figure 3: Screenshot of the Genre category on the IMDB web interface after using the hyperlink from the movie interface Page (Figure 2: Left)

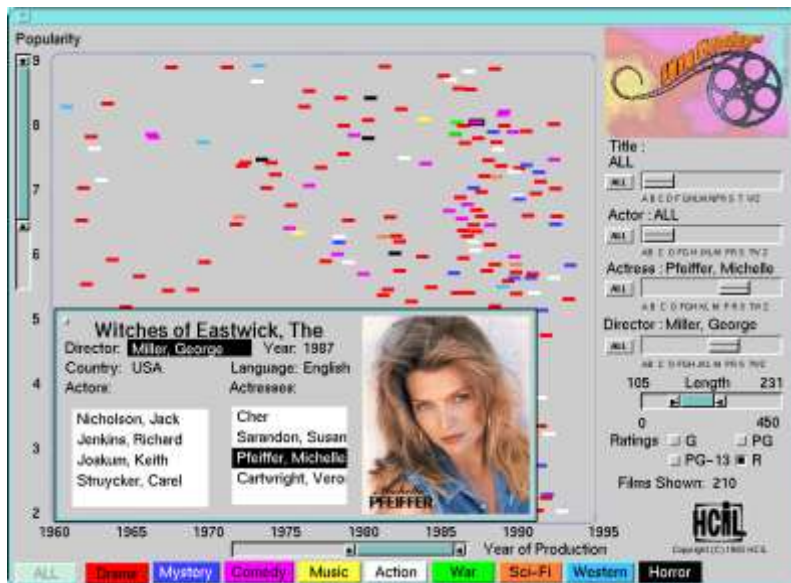


Figure 4 (Left):

Screenshot of Ben Schneiderman's "FilmFinder" showing a Starfield display in the centre, an information box at the bottom and AlphaSliders and controls to the right.

3. The Proposed Project

The overall aim of the project is to produce a data visualisation for the IMDB that makes it easy to explore relationships between data. Another of the aims is to have re-usable components within the system should the underlying data change.

3.1 Aims and Objectives

3.1.1 Text analysis and conversion components

The first part of the project will involve reverse engineering the IMDB data so that an entity relationship diagram of the schema can be produced; to allow for a full understanding of the relationships between the data. It may be necessary to re-design the schema to make it more efficient or so that relationships can be better represented for

additional processing into. Should the schema be re-designed then an additional component can be created which will modify the existing text files to meet the new schema. One of the aims is to have re-usable components so should the underlying data change (i.e. data other than the IMDb) the visualisation will still function. This can be achieved by adding an extra layer of abstraction between reading the data source and the loading of data into a DBMS using XML. This abstraction means that the data source will first need to be converted to XML and then before being loaded into an appropriate DBMS. The process of using XML as an intermediary means that should the underlying data change all that will be needed to change is the process of converting the source data into XML as well as a small configuration file for specific data filtering.

Once a better understanding of the underlying data has been achieved then an XML format can be devised which will allow full representation of all of the data from the text files. This format will be represented in the form of an XML Schema(s) which can then be used to validate the XML produced from the component which converts textual data into XML. The reason XML is to be used is so that components within the system can be re-used should the underlying data change with a requirement to only change the way that the data is converted into XML.

The next component is necessary to allow the conversion of textual IMDb data into the XML specified format to allow for entry into the DBMS. This process will involve creating a component that will parse each of the text files into an equivalent XML files that meet the schema. This would be achieved by creating a Java application which would parse each line of the files and produce an XML document.

3.1.2 Database Setup / Loading / Health Checking

In order to load these XML documents into a DBMS another component must be created that can parse and load them into the DBMS.

There are many different types of DBMS that are available today most supporting the Structured Query Language (SQL) for querying the database. SQL allows complex queries to be performed on a database with a single command, allowing the application to concern itself with the results. MySQL is such an example of a DBMS that supports SQL and is easily linked in with Java as below.

This component could take advantage of built in methods within many DBMS such as MySQL which allows XML files to be directly loaded so long as tables and columns are correctly setup. There are also other XML parsers which can be used for similar purposes to allow XML files to be parsed and interpreted. The MySQL connector allows Java applications to access and manipulate MySQL databases which can be used to automate XML file loading into the database. This project requires the use an SQL based database which could be located on a separate server if networked, or run locally should this not be available. It may be necessary to only use a representative subset of the IMDb should storage space become limited and or time becomes an issue. The diagram in Figure 5 below shows the flow of control from the original text files to data visualisation. An additional component is also necessary to allow the relational integrity of the database to be checked. Relational integrity ensures that the every value that is in the column of one table is also in related tables.

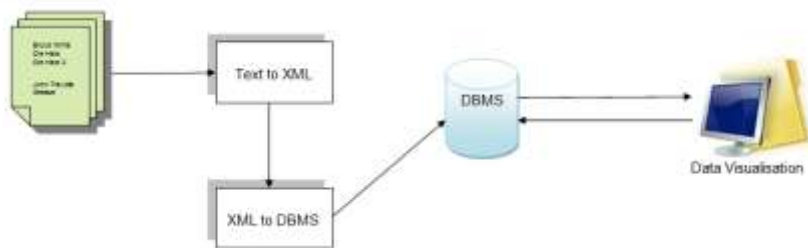


Figure 5: Diagram showing the flow of data from textual data to DBMS, and between the DBMS and the Visualisation component.

3.1.3 Data Visualisation

Data visualisation is one of the key aims of this project and this component will require significant work. Initially it will be necessary to research existing data visualisation techniques and assess their suitability for the IMDB. This research may be carried out through the use of the Internet as well as written material. Some preliminary research that has been carried out has indicated that the Processing programming language and the Prefuse toolkit may be suitable types of data visualisation techniques. The Processing programming language is designed to support visual programming and relies on a very simple syntax that is similar to Java. Large amounts of data can be quickly and easily represented through a simple syntax.

The Prefuse toolkit is a plug-in for Java that allows Java code to be used to represent complex data visualisations, but uses pure Java and is significantly more complex than Processing.

An approach to data visualisation that has been suggested is to use Nodes and Arc to represent entities and their relationships in a visual way as can be seen in Figure 6 below. The example below also shows how additional information could be displayed in relation to a record by allowing a user to hover or click on a node.

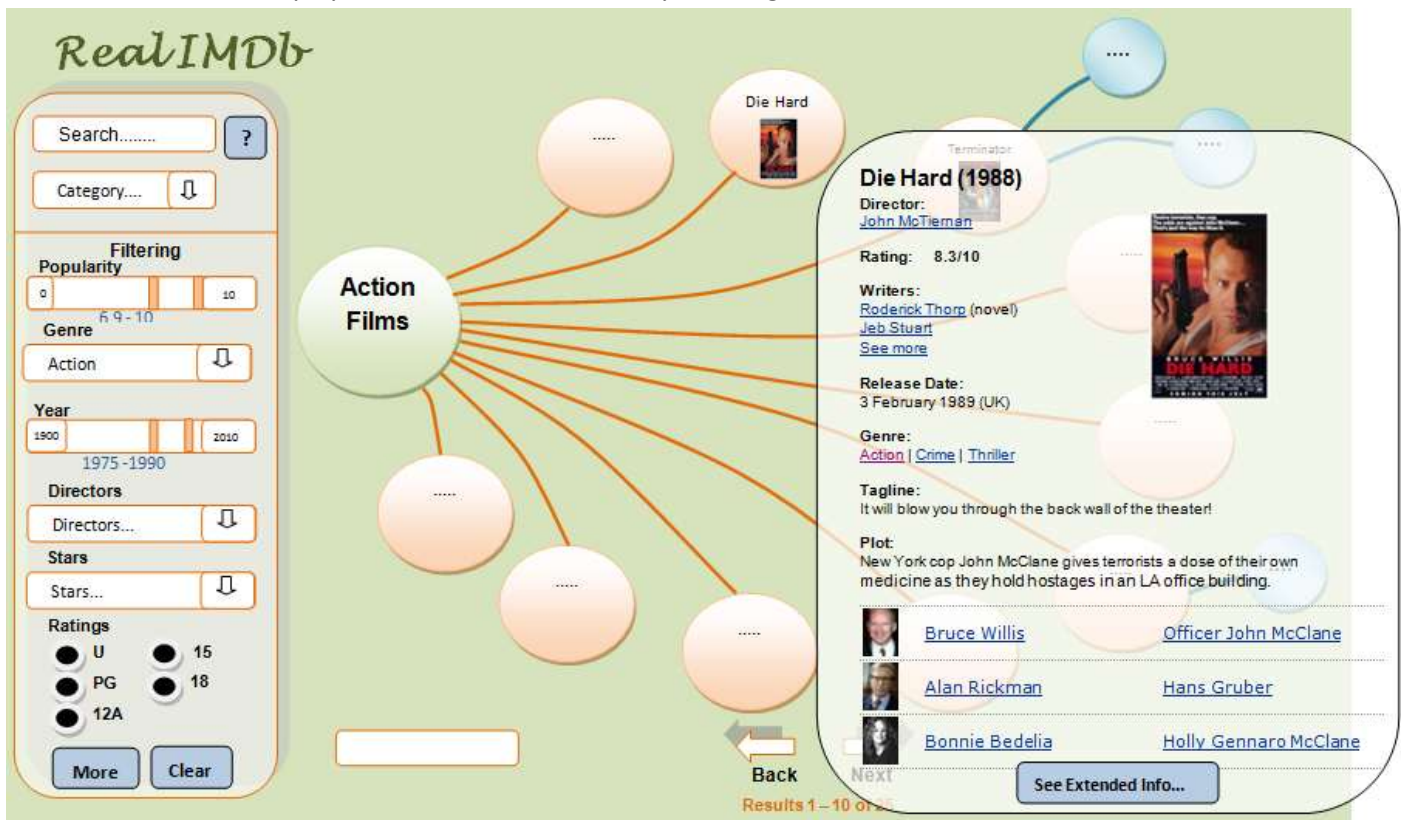


Figure 6: Diagram showing what a data visualisation for the IMDB could look like along with methods of querying the data.

Large volumes of data will need to be represented within the data visualisation and as a result of this it will be necessary to implement further query mechanisms to allow results that a user is not interested in to be removed. This mechanism could be implemented by allowing a user to filter results with additional queries such as the popularity or the year the film was made, or by specifying genres of movie stars that they are interested in. The example in Figure 6 above shows such filtering through the use of sliders, combo boxes and radio buttons.

An additional aim should the project progress well could be to add additional functionality to the data visualisation so that films can be suggested to users based upon heuristics such as favourite genre or director. Some form of user study will be required in order to formulate an algorithm that can be used to suggest films. This component will be created alongside the data visualisation.

3.2 Methodologies

3.2.1 Approach

The methodology that will best suit the style of this project will be a combination of the Waterfall Method and Rapid Application Design. The waterfall method suits the type of development early on in the project where there are a lot of dependencies on the previous task and a single failure at the start can cause significant problems further down the line. RAD suits the development when it comes to the data visualisation and possible extra functionality as the data visualisation may be subjective and require prototyping in order to ensure that it is fully functional.

3.2.2 Evaluation Techniques

The evaluation and testing of the project will be accomplished in several different ways through the use of unit testing for individual components as well as acceptance tests and a small user study.

The use of unit testing for all components will ensure that low level components are correctly functioning and producing the correct outputs this could be accomplished through the use of a Java plug-in such as Junit which allows for tests to be setup on each individual component.

Acceptance tests can be used to ensure that "Happy Day" or typical scenarios work well to ensure a minimum of functionality that is acceptable. These acceptance tests can also be used as part of a user study to assess how well the system functions and whether or not the data visualisation makes it easy to see relationships and explore data.

Several user studies could be performed as part of the RAD methodology to feed back information in relation to the perception of prototypes during this process.

4 Programme of Work

What follows is an estimated programme of work with associated timescales:

Phase 1: Weeks 1 – 7 Textual Analysis and Design

Task 1: Weeks 1 - 2 Reverse engineer an ER diagram of the Schema

This task will involve analyzing the textual data and representing all of the relationships between entities of the schema in an Entity Relationship ER diagram. This task is crucial to ensuring an accurate understanding for the rest of the project.

Task 2: Week 3 Re-Design / alter Schema to be more efficient

This task will involve analysing the produced ER diagram of the schema so that changes can be made to make it easier to represent the relationships in the database. This stage may or may not find significant changes that can be made.

Task 3: Weeks 3 - 4 Re-Factor Textual Files

Task 4: Week 4 Design XML document format suitable for the IMDB

Task 5: Week 5 Design component to convert IMDB textual data into XML

Task 6: Week 5 Design component to load XML into an SQL database

Task 7: Week 6 Implement textual to XML component

Task 8: Week 7 Unit test textual to XML

Phase 2: Weeks 7 – 11

Task 9: Weeks 7 - 8 Setup database and Implement XML to SQL database

Task 10: Week 8 Unit test XML to SQL

Task 11: Week 9 Design Health checking for database

Task 12: Week 10 Implement health checking for database

Task 13: Week 11 Unit test health checking for database

Phase 3: Weeks 1 – 20

- Task 14: Weeks 1 - 11 Research data visualisation systems and perform exploratory programming
- Task 15: Weeks 12 - 13 Design architecture and mechanisms
- Task 16: Weeks 13 - 15 Implement data visualisation
- Task 17: Weeks 15 - 16 Unit test / Perception test / User test visualisation
- Task 18: Weeks 17 - 20 Write report

Gantt chart

The chart below in Figure 7 shows the programme of work:

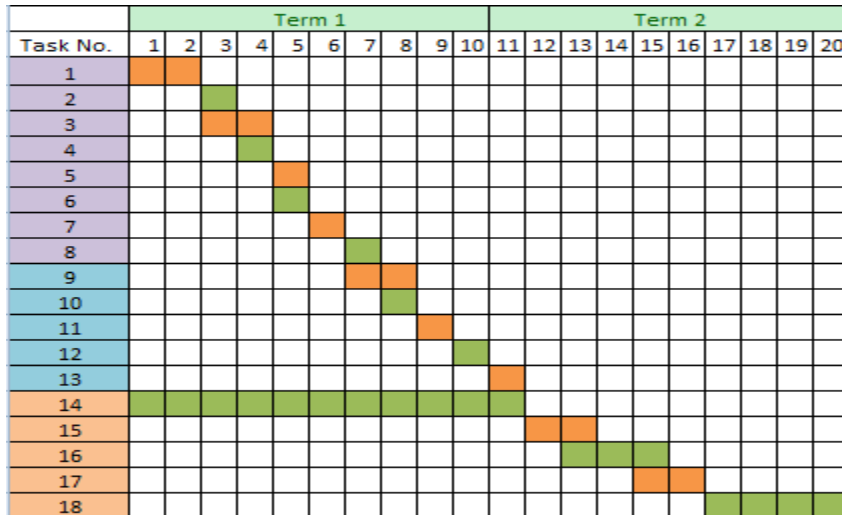


Figure 7 (Left):

Gantt chart to show the distribution of tasks and their associated timescales.

5 Resources Required

- Networked workstation for development :
 - To allow development and for use to test data visualisation components.
- Networked SQL Database Server:
 - Used to host SQL database for the IMDB data, also for use in the processing of text files and database loading.
- IMDB text files:
 - These files are needed so they can be processed and loaded into the SQL database.
- Java Integrated Development Environment (IDE):
 - Software to allow Java application development.
- Data visualisation API:
 - A package to support the development of data visualisation.

6 References

- The Internet Movie Database (IMDb):

Screenshots of the IMDB web interface as well as research into the working of the IMDB and for supply of the textual data.

<http://www.IMDb.com>

23/06/2010

- Lancaster University Computing Intranet CSc 202:

Intranet resources used to assist in the production of this proposal.

<http://cs-intranet.lancs.ac.uk/Csc202/FYPPProposal>

23/06/2010

- W3Schools Web Standards Information and tutorials:

Research into XML based data representation and XML schema's.

<http://www.w3schools.com/dom/default.asp>

23/06/2010

- MySQL documentation – XML Loading / MySQL Connector:

Research into the loading of XML files into a DBMS such as MySQL, as well as the manipulation of MySQL databases within a Java application.

<http://dev.mysql.com/tech-resources/articles/xml-in-mysql5.1-6.0.html>

23/05/2010

<http://dev.mysql.com/usingmysql/java/>

23/05/2010

- Processing Programming Language and Environment:

Used as an example of a suitable data visualisation language that can be used alongside Java.

<http://www.processing.org/>

23/05/2010

- Prefuse Visualisation Toolkit:

Used as an example of a suitable data visualisation toolkit that can be plugged into Java.

<http://prefuse.org/>

23/05/2010

- Images used in diagram of data visualisation:

Image of the original Die Hard Film

<http://snarkerati.com/movie-news/files/2009/12/die-hard-poster.jpg>

23/06/2010

Image of the original Terminator Film

<http://www.reutil.com.ar/z/images/terminator1.jpg>

23/05/2010

- Junit testing framework for Java:

Information used to describe the use of unit testing as well as an example of a suitable example.

<http://www.junit.org/>

23/05/2010

- Ben Schneiderman's "FilmFinder"

Information and a screenshot of Ben Schneider's research including "FilmFinder".

<http://www.cs.umd.edu/hcil/spotfire/>

02/06/2010