Chapter 3

High Level Design of classification of unstructured data from online course web pages

Software sometimes can be a complex entity. Its development usually follows what is known as Software Development Life Cycle (SDLC). The second stage in the SDLC is the Design Stage. The objective of the design stage is to produce the overall design of the software. The design stage involves two sub-stages namely:

1. High-Level Design
2. Detailed Design

In the High-Level Design, the Technical Architect of the project will study the proposed applications functional and non-functional (qualitative) requirements and design overall solution architecture of the application, which can handle those needs. High Level Design precisely discusses an overall view of how the system should work and the top-level components that will form the system that work to achieve the proposed solution. It should have very little detail on implementation, i.e. no explicit class definitions, and in some cases not even details such as database type (relational or object) and programming language and platform [44].

The high level design consists of the detail structuring of the preprocessing phase. It consists of a crawler that finds and retrieves web pages, a trainer that analyses a list of relevant (training pages) and irrelevant links and compute probabilities bout the feature-category pairs found and an indexer which is responsible for identifying and extracting all suitable features from each web page.

3.1 Design Considerations

This section addresses the issues that need to be discussed or resolved before attempting to devise a complete design solution.

3.1.1 General Constraints

The initial stage of the project i.e. is the pre-processing stage involves crawling which can be time consuming in order to crawl for relevant online course web pages.

The application can only strive for highest levels of accuracy however cannot guarantee the absence of variations. This software has been designed in accordance to the basic factors of user convenience and improved functionality together. The constraints are:

- Slow internet connections or outdated versions of browsers due to which the software may not deliver effectively.
- The functioning of the server, upon on which the functionality of the software depends.
- User understanding and familiarity upon which the usefulness of the software depends, which varies across systems.

3.1.2 Development Methods

The development method used in this software design is modular/functional development method [44].The approach is categorized into various phases. The input-output data that flows from one module to another shows the dependency. The phases are: crawling, extraction (feature set), application of classification algorithms (including the modified algorithm), comparison of results, design of front end. Data flow diagrams have been used in the modular design of the system.

3.2 Architectural Strategies

This section describes the design decisions and strategies that affect the overall organization of the system and its higher-level structures. These strategies will provide insight into the key abstractions and mechanisms used in the system architecture.

3.2.1 Programming Language

The programming language plays a major role in the efficiency as well as the future development of the project. This project is developed in Java. Since Java is system independent and portable, it is the most suitable language to be used for this project. The front end is built using java swing. Swing is the primary Java GUI widget toolkit, an API for providing a Graphical User Interface (GUI) for Java programs. Swing is a platform-independent, Model-View-Controller GUI framework for Java, which follows a single-threaded programming model. Additionally, this framework provides a layer of

abstraction between the code structure and graphic presentation of a Swing-based GUI. It also provides good database connectivity.

3.2.2 Future Plans

Upon the completion of the project, the following are to be implemented

- To build a crawler for AJAX based web pages
- To build a mobile application
- Improve data set as per increasing number of online course web pages.
- Research into more efficient classification algorithms.

3.2.3 Error Detection and Recovery

The software should be built in such a way that it should avoid errors and prevent failures from occurring. The correct set of rules for classification are used to ensure appropriate set of features(existent in all web pages) are extracted for classification in the classifier stage. This ensures and improves the quality of the data and renders good results for the machine learning algorithms used. The training of the model and testing needs have been performed appropriately using bootstrapping and cross validation methods. Necessary precautions are taken to ensure faulty data is avoided in the database such that errors can be prevented. Care should be taken to ensure that the database is valid and up to date.

3.2.4 Data Storage Management

The data is stored in CSV files. The data retrieval is done with the help of CsvJdbc driver. CsvJdbc is a read-only JDBC driver that uses Comma Separated Value (CSV) files or DBF files as database tables. It is ideal for writing data import programs or analyzing log files. The driver enables the access of a directory or a ZIP file containing CSV or DBF files as if it were a database containing tables. CsvJdbc accepts only SQL SELECT queries from a single table

3.2.5 Communication Mechanism

The project involves communication between the user and the system through an interface. The interface is responsible for querying the repository for required online

courses from the user end. There is also a communication established between the repository and the crawler and repository and the classifier.

3.3 System Architecture

The software has the following major phases which guide the working of the whole system:

- Crawler(fast multi-threaded java crawler)
- Extraction of features.
- Classification of web pages using various machine learning algorithms.
- Comparison of the machine learning algorithm results.
- Designing the front end.

3.4 Data Flow Diagrams

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system. Data Flow models are used to show how data flows through a sequence of processing steps. The data is transformed at each step before moving on to the next stage. During the live migration the data flow occurs between the two physical machines. These processing steps or transformations are program functions when data flow diagrams are used to document a software design. The DFD for the performance analysis of live migration of the virtual machines can be decomposed into two levels such as level-0 and level-1 [44].

3.4.1 Data Flow Diagram – Level 0

Level 0 is the initial level Data Flow Diagram and it is generally called as the Context Level Diagram (CLD). It is a common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. This context level DFD is then "exploded" to show more detail of the system being modelled. Figure 3.1 below shows the Level 0 Data Flow Diagram. It shows the interaction between the user the online course repository. The client passes query requests in the front end. The results for the requests are searched in the repository and results are displayed back on the front end.

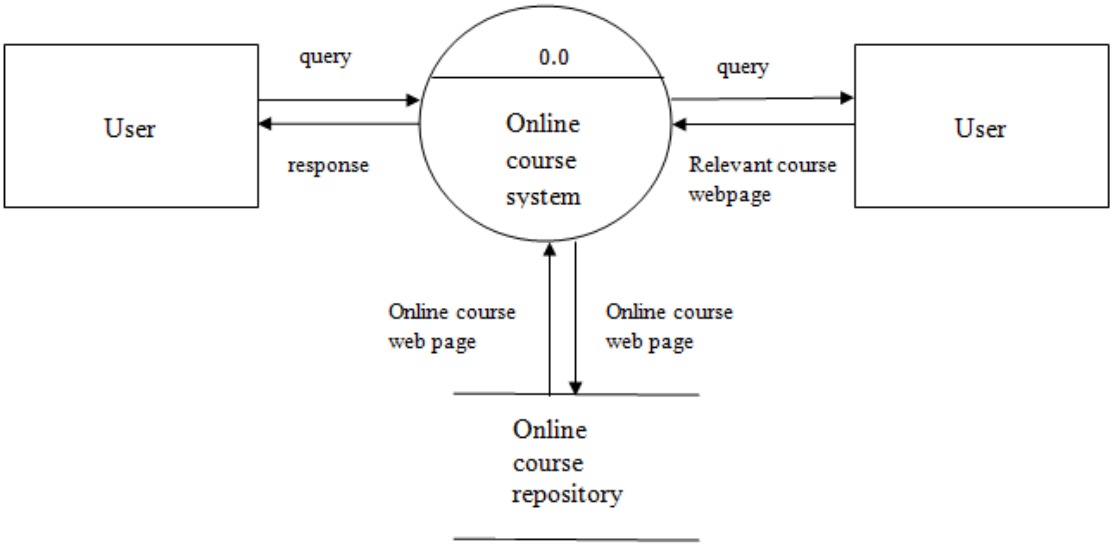

Figure 3.1: Data Flow Diagram level 0 for online course web page classification system

3.4.2 Data Flow Diagram – Level 1

The Level 1 Data Flow Diagram gives more information than the level 0 Data Flow Diagram. The Figure 3.2 below shows the Level 1 Data Flow Diagram. It shows the back end mechanism. This mechanism does the job of retrieving the data and having the data to be classified. It involves a crawler which essentially crawls the urls of online course web pages. The fetched data is stored in a repository in a structured format upon which the classifier does the job of classifying the web pages as online course web pages.

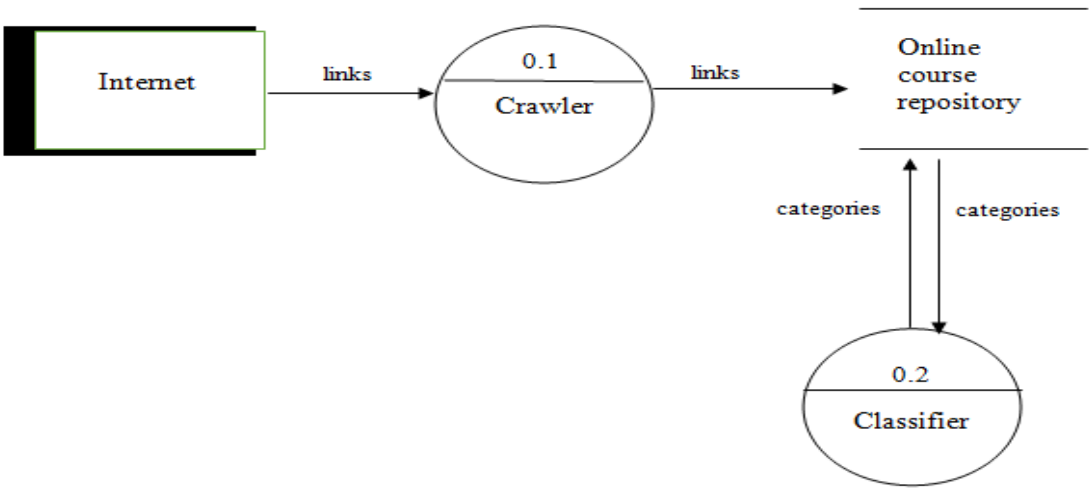

Figure 3.2: Data Flow Diagram level 1 for online course webpage classification system

3.4.3 Data Flow Diagram – Level 2

The Figure 3.3 below shows Level 2 data flow diagram. It represents more detailed flow of data between the processes. It elaborates the procedure of fetching the data which is the in the unstructured format. After the crawler crawls the urls, the featurizer extracts the feature set from the web pages and consolidates the data in the structured format. This structured data is then put in the repository for the classifier to work upon. The classifier is trained with a trained data set and then used to classify the test data set.

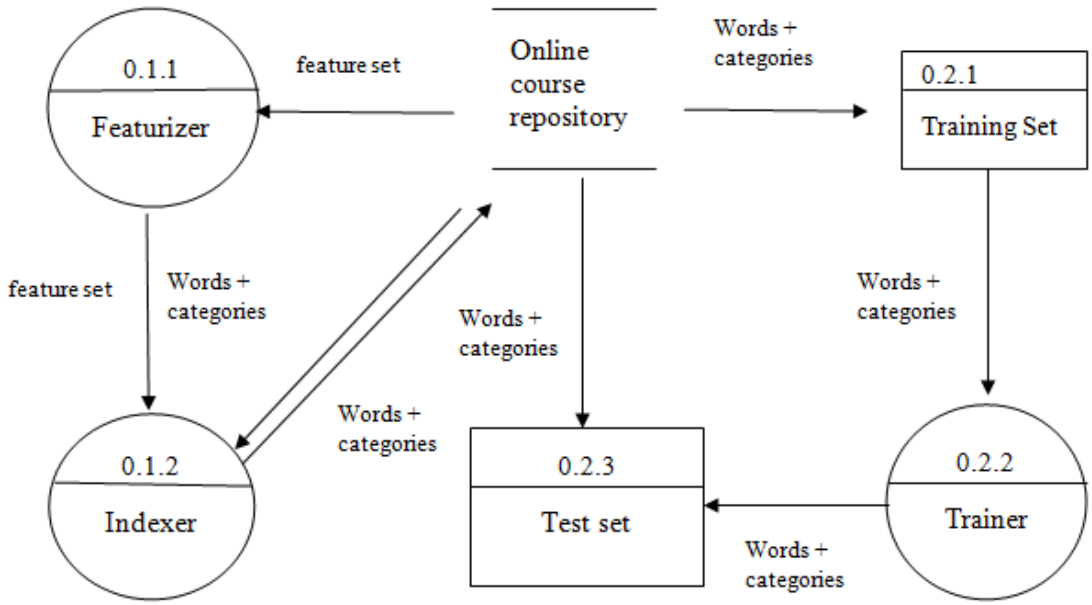

Figure 3.3: Data Flow Diagram level 2 for online course webpage classification system

3.6 Summary

This chapter details the phases required by architects to understand the overall flow and functioning of the system as a whole, as well as a collection of individual modules which have been elaborated with various features and descriptions of data flow within the system. The chapter has also detailed various phases of the development and

working of the project right from preprocessing and crawling to classification and the user interaction. This flow of data, architecture and other design related constraints and specifications constitute the high level design of the project in completion.