Software Design Specification for Analysis Tool

Version 1.0 approved

Document ID: WCMS-SDS-1.0

Prepared by:

1.	*Parinaz Barakhshan*	*27675518*
2.	*NagaPrathibha Gummaraju*	*27854153*
3.	*Saravanan Velusamy*	*27618786*

Group #: 3

Team: B

Logfile:		
Student: Name	**ID**	**Main contribution(s)**
Parinaz Barakhshan	**27675518**	***Code: readXML class*** ***Domain Model, Analysis Model,*** ***All Use cases-*** ***Prioritization of Goals*** ***first, second ,and third SSD and SD*** ***system Architecture, system Design*** ***class Diagram.***
NagaPrathibha Gummaraju	**27854153**	***Export Use Case Sequence and System Sequence Diagram and Analysis Documentation ,Review.***
Saravanan Velusamy	***27618786***	***Save as RDF sequence diagram and system sequence diagram, Analysis documentation.***

Table of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**
Team 3	*21,Feb,2016*	Architectural Baseline	1.0
Team 3	*22,Feb,2016*	Sequence Diagram Change	1.0

1. Introduction

The primary goal of this project is to develop the data Analysis component of the 'Google Play Application Data Collection and Analysis Tool'. Data analysis component is only one of two parts of the overall 'Google Play Application Data Collection and Analysis Tool'. The scrapping component is also needed to collect the data prior to analysis.

1.1 Purpose

This Software Design Specification serves the purpose of presenting the design of the Analysis part of the 'Google Play Application Data Collection and Analysis Tool'. The purpose of this design document is to provide all details of the Architectural Design (AD), software Interface Design , and Internal Module Design (IMD) for the data Analysis component of the 'Google Play Application Data Collection and Analysis Tool'. The Architectural Design part focuses on the high-level project decomposition, the MID focuses on the software interfaces between the high level modules, and the IMD focuses on the low level description of the implementation classes and all their attributes and methods.

This document is mainly intended for the internal uses of FactsRUS and Team3 project managers and software developers. Support staff and end users may also read and understand this document. It will serve as a basis for the final phase of the project.

We will describe the overall functions of the component, the possible interactions users can have with the component as well as the various requirements the component will need to function within normal parameters.

1.2 Project Scope and Product Features

The purpose of this tool is to extract data from an XML file in order to create graphs and tables to easily compare different data points and help in the analysis of trends and patterns in google play applications and their usage. This Software Design Specification will describe the overall functions of the component, the possible interactions users can have with the component as well as the various requirements the component will need to function within normal parameters. The component itself will be implemented in Java.

The analysis component of the GPADCAT is a data manipulation application. Data collected from the scraping component of the GPADCAT is provided to the analysis component via XML. The XML file is then parsed for the relevant data and this data is then ordered and displayed to the user. The user may then select the data he or she may wish to display in graph form or table form. Once the display selection is made the relevant data points are taken from the XML file to create the visual representation of the data. Finally, the graphical representation may be saved in a RDF format.

The objective of this component is to provide an extremely user friendly experience for data comparison an analysis of relatively large data sets. The main benefit of this tool is the time it saves. It reduces time spent by companies organizing and classifying data by completing this task automatically. For any firm wishing to analyze and understand trends and patterns in Google Play apps, this is an ideal tool.

In this project we will provide a GUI for statistical analysis and visualization of the stored data in graph and table format and export of the data to a standardized exchange format will be supported.

1.3 Definitions, Acronyms and Abbreviations

SRS: Software Requirement Specification

Tool: The Google Play Application Data Collection and Analysis Tool being built by GROUP3.

GPADCAT: Google Play Application Data Collection and Analysis Tool. Pronounced gee-pad-cat

Component: A subsection of the Google Play Application Data Collection and Analysis Tool, of which there are two: the scraping component that collects data, and the analysis component described in this SRS, which that manipulates data.

IEEE standard: International electrical and electronics engineers organization (std 830-1993).

SRS: Software Requirements Specification

SDD: Software Design Document

AD: Architectural Design

MID: Module Interface Design

IMD: Internal Module Design

RDF: Resource Description Framework. Files that contain the .rdf file extension are documents that have been written in the Resource Description Framework language. This format is used to store "metadata" information.

1.4 References

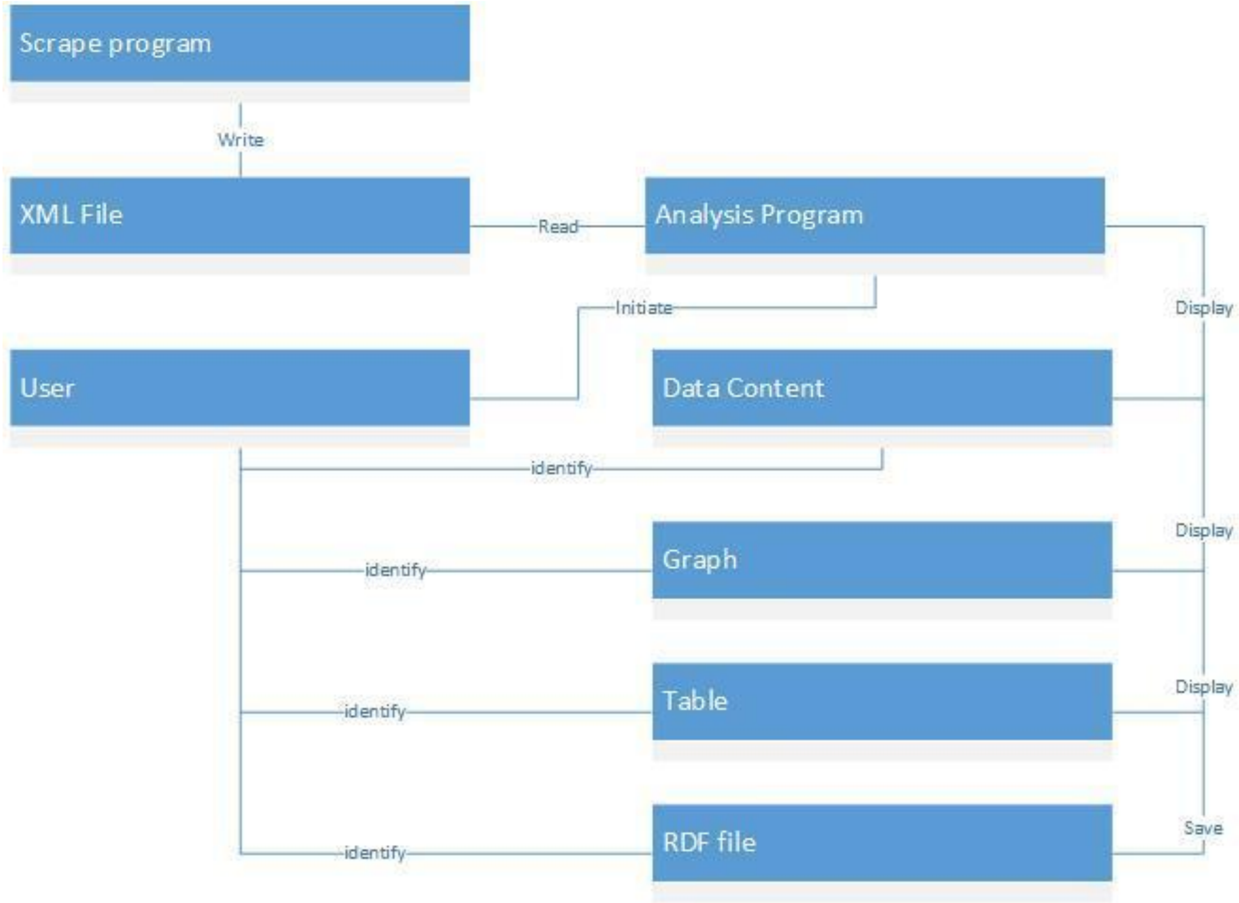
[1] MVC pattern- https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller
[2] System Architecture- https://en.wikipedia.org/wiki/Systems_architecture
[3] Singleton pattern- https://en.wikipedia.org/wiki/Singleton_pattern
[4] Façade pattern- https://en.wikipedia.org/wiki/Facade_pattern
[5] Factory pattern- https://en.wikipedia.org/wiki/Factory_method_pattern

1.5 Assumptions

A-01. This software will require Java software installed on any computer using our software.
A-02. There will be only one user: the end-user.
A-03. We will be using data from Google Play so the software depends on data scraped from the designated application.
A-04. If data from Google Play is modified, information displayed by GPADCAT will be modified as well.

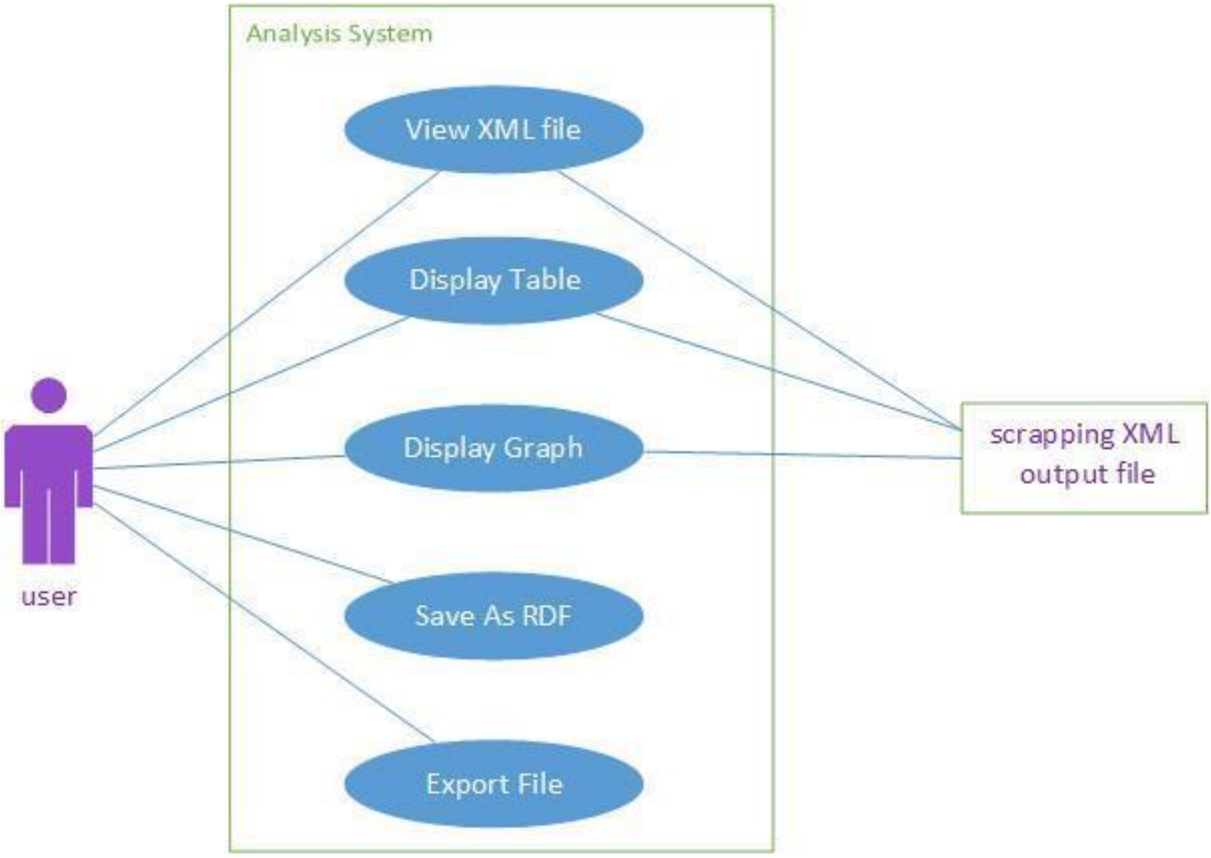
The analysis component of the GPADCAT is a data manipulation application. Data collected from the scraping component of the GPADCAT is provided to the analysis component via XML. The XML file is then parsed for the relevant data and displayed to the user. The user may then select the data he or she may wish to display in graph form or table form. Once the display selection is made the relevant data points are taken from the XML file to create the visual representation of the data. Finally, the graphical representation may be saved in a RDF format.

1.6 Domain Model



1.7 Analysis Model

1.7.1 Use case diagram for Analysis Program



UC-#	*UC Name*
UC-1	*View XML File*
UC-2	*Display Data as table*
UC-3	*Display Data as Graph*
UC-4	*Save As RDF*
UC-5	*Export File*

Table 1 -List of Use Cases

RQ-#	*Requirement*
RQ-1	*View XML file Elements*
RQ-2	*Choosing proper Elements to be compared*
RQ-3	*Viewing selected Elements as graph*
RQ-4	*Viewing selected Elements as Table*
RQ-5	*Saving the current view as RDF file*
RQ-6	*Exporting files*

Table 2- List Of requirements:

1.7.2 Use Case #1: View XML File

Use Case UC-1	View XML File	
Related Requirements:	*RQ-1*	
Pre-conditions	XML file has been created by scraper Program and is provided to the analysis program	
Post conditions	XML elements ca be Selected	
Initiating Actors	End User	
Actor's Goal	To view XML File Elements	
Participating Actor	Scraping XML output File	
Trigger	User clicks on the "View XML file" button.	
Main Scenario	**Step**	**Action**
	1.	User selects to 'View XML file'
	2.	System reads, verifies, and parses the XML file to show the elements of the XML file.
Extensions	**Step**	**Branching Action**
	2a.	Data is corrupted and cannot be parsed correctly. An error is thrown.

1.7.3 Use Case #2: Display Data as Table

Use Case UC-2	**Display Data As Table**	
Related Requirements	RQ-4	
Priority	2	
Initiating Actors	End User	
Actor's Goal	User can view the scraped data in the format of tables	
Participating Actor	Scraping program provides its output as input to this program	
Pre-conditions	XML File is open and the file is verified and parsed. The elements of the XML file are recognized. Its contents are already in memory.	
Post conditions	None	
Trigger	User clicks on Display As Table button	
Main Scenario	**Step**	**Action**
	1.	User chooses the elements ,and selects Display As Table
	2.	System provides user with data analysis result of selected data

1.7.4 Use Case #3: Display Data as Graph

Use Case UC-3	**Display Data As Graph**	
Related Requirements	RQ-3	
Priority	2	
Initiating Actors	End User	
Actor's Goal	User can view the scraped data in the format of Graphs	
Participating Actor	Scraping program provides its output as input to this program	
Pre-conditions	XML File is open and the file is parsed. The elements of the XML file are recognized.	
Post conditions	None	
Trigger	User clicks on Display As Graph button	
Main Scenario	**Step**	**Action**
	1.	User chooses the elements ,and selects Display As Graph
	2.	System provides user with data analysis result of selected data

1.7.5 Use Case #4: Save as RDF

Use Case UC-4	Save as RDF	
Related Requirements	RQ-5	
Priority	2	
Initiating Actors	End User	
Actor's Goal	To save metadata in the RDF file	
Participating Actor	File System	
Pre-conditions	XML elements are recognized, and selected by user.	
Post conditions	RDF file is created	
Trigger	Choosing 'save as RDF' option by user	
Main Scenario	**Step**	**Action**
	1.	User selects 'save as RDF' option
	2.	Metadata info is parsed and structured in RDF format by system. system will provide user with let user choose where to save the file and name the file
	3.	User selects the file name and where it should be saved.

Extensions	**Step**	**Branching Action**
	2a.	Creating RDF file may face problem.an Error message is thrown and saving process is canceled.
	3a.	There is a problem with naming or the place the file should be saved. An error message is thrown and saving is canceled.

1.7.6 Use Case #5: Export File

Use Case UC-2	Export File	
Related Requirements	RQ-6	
Priority	1	
Initiating Actors	End User	
Actor's Goal	To Export data in other formats, in addition to RDF	
Participating Actor	File system	
Pre-conditions	XML elements are recognized by system, and selected by user.	
Post conditions	Requested file is saved in requested place	
Trigger	User selects the 'Export File' option	
Main Scenario	**Step**	**Action**
	1.	User selects 'export file' option
	2.	System asks about the type of file to be created
	3.	User selects one of the predefined types of files by system
	4.	Based on the user's request the related file will be created. System will provide user with the option to choose where to save the file and name the file.
	5.	User selects the name and the place for the file to be saved.
Extensions	**Step**	**Branching Action**
	5a.	Creating the requested file may face problem.an Error message is thrown and saving process is canceled.
	5b.	There is a problem with naming or the place the file should be saved. An error message is thrown and saving is canceled.

1.8 Priority of Requirements

		Use cases				
Requirement	PW	UC1	UC2	UC3	UC4	UC5
RQ-1	5	X				
RQ-2	4	X				
RQ-3	2			X		
RQ-4	2		X			
RQ-5	2				X	
RQ-6	1					X
Max PW		5	2	2	2	1
Total PW		9	2	2	2	1

Table 3- Mapping: System requirements to Use cases

The middle column shows the priority weight (PW) of each requirement, with a greater number indicating a higher priority. The priority weight may be assigned by the customer or derived from the urgency-to-deliver the requested capabilities to the customer. The range of priority weights is decided arbitrarily, here it is 1–5. It is preferable to have a small range (10 or less), because the priorities are assigned subjectively and it is difficult to discern finely-grained priorities. Larger projects with numerous requirements may need larger range of priorities. PW is used to schedule the work for implementing the use cases.

Priority Table 4- presents the prioritization of the functional and Non Functional requirements of the System for the first iteration. These priorities are considered and reflected in the architectural and system design decisions relating to this product.

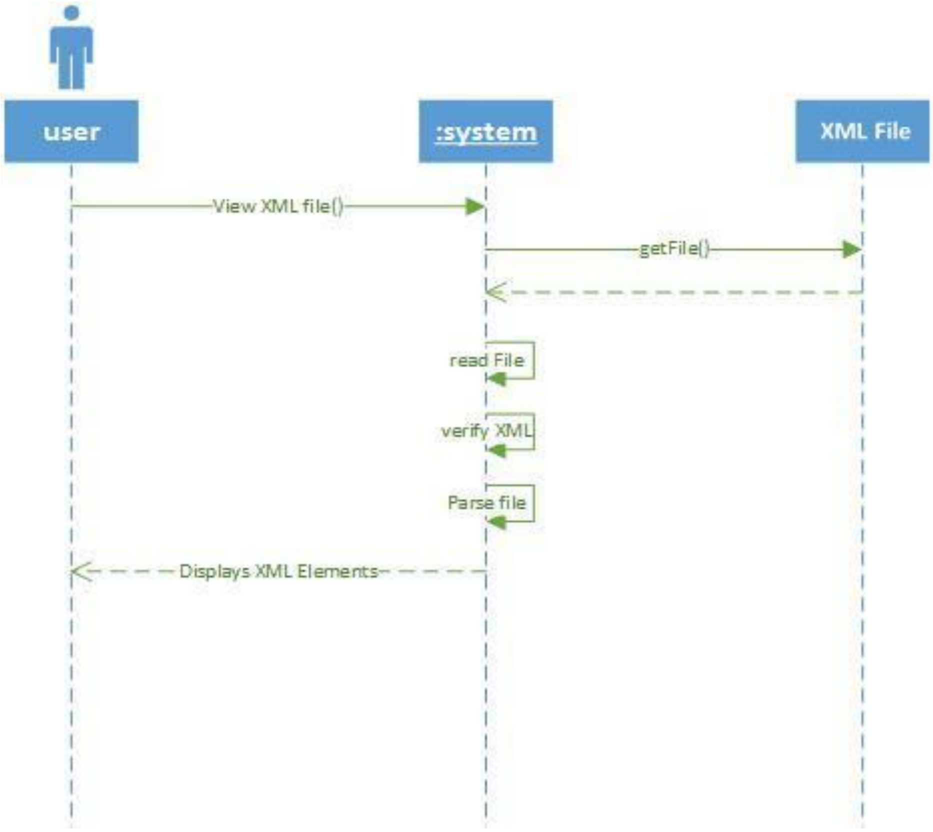
Priority	**Requirement**	**Reason**	**Implemented/considered in Iteration**
High	UC#2,UC#3	Core feature of the Analyzing program	#1
Medium	Extensibility	- Design should support additional types of statistical analysis - Design should support export to different file types	#2
High	UC#1	XML file should be provided by scraping program and XML Elements should be recognized in order to make other system functionalities possible.	#1
Low	Efficiency		#2
Low	Reliability		#2
High	Portability	Program should be executable on different platforms	#1
High	Usability	Design should be Easy to use	#1
Low	Security		#2

Table 4- Prioritization of Goals

2. Sequence Diagrams

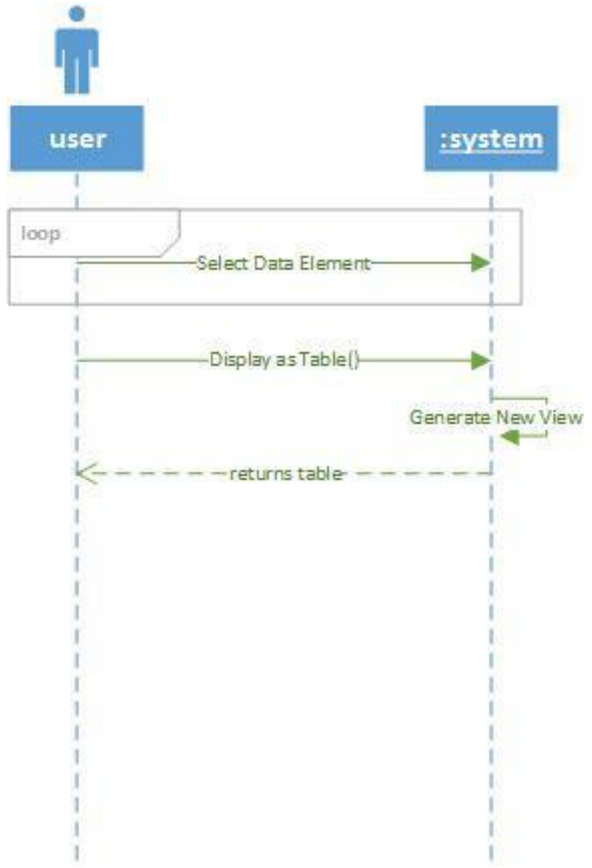
2.1 System Sequence and Sequence Diagram for UC-1: *View XML File*

<u>*SD for* UC-1: *View XML File*</u>

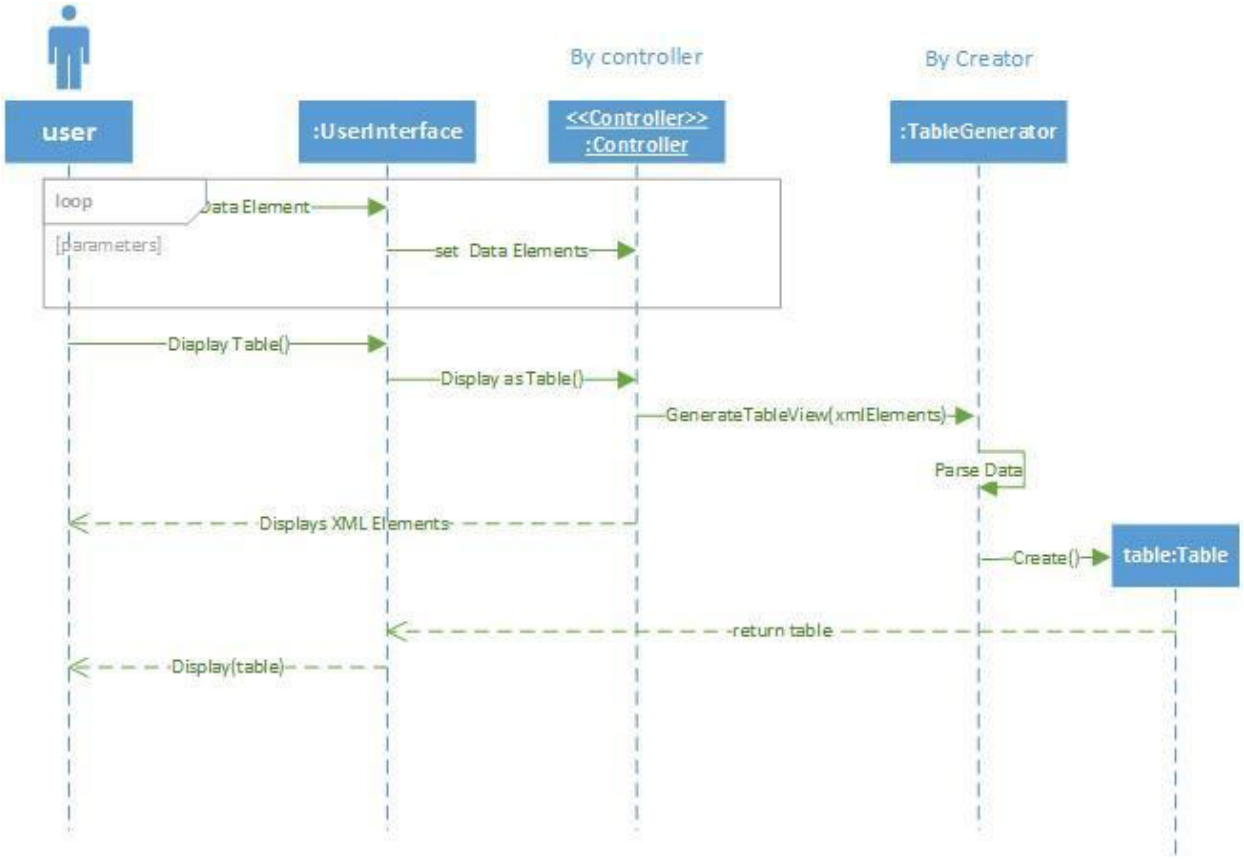


2.2 System Sequence and Sequence Diagram for UC-2: *Display as table*

SSD for UC-2: Display as table

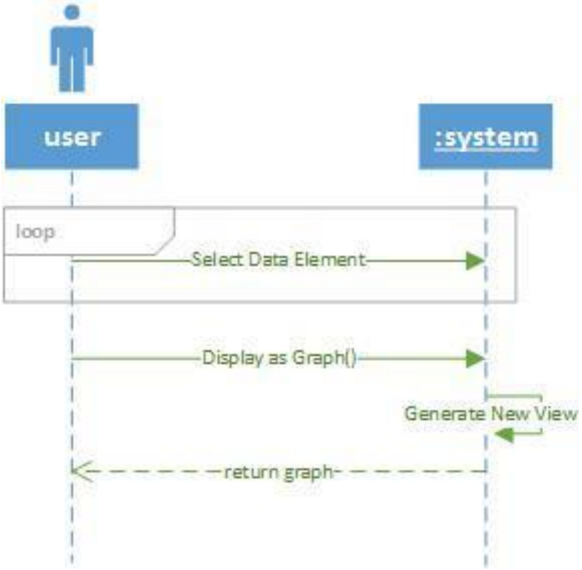


SD for display as a table

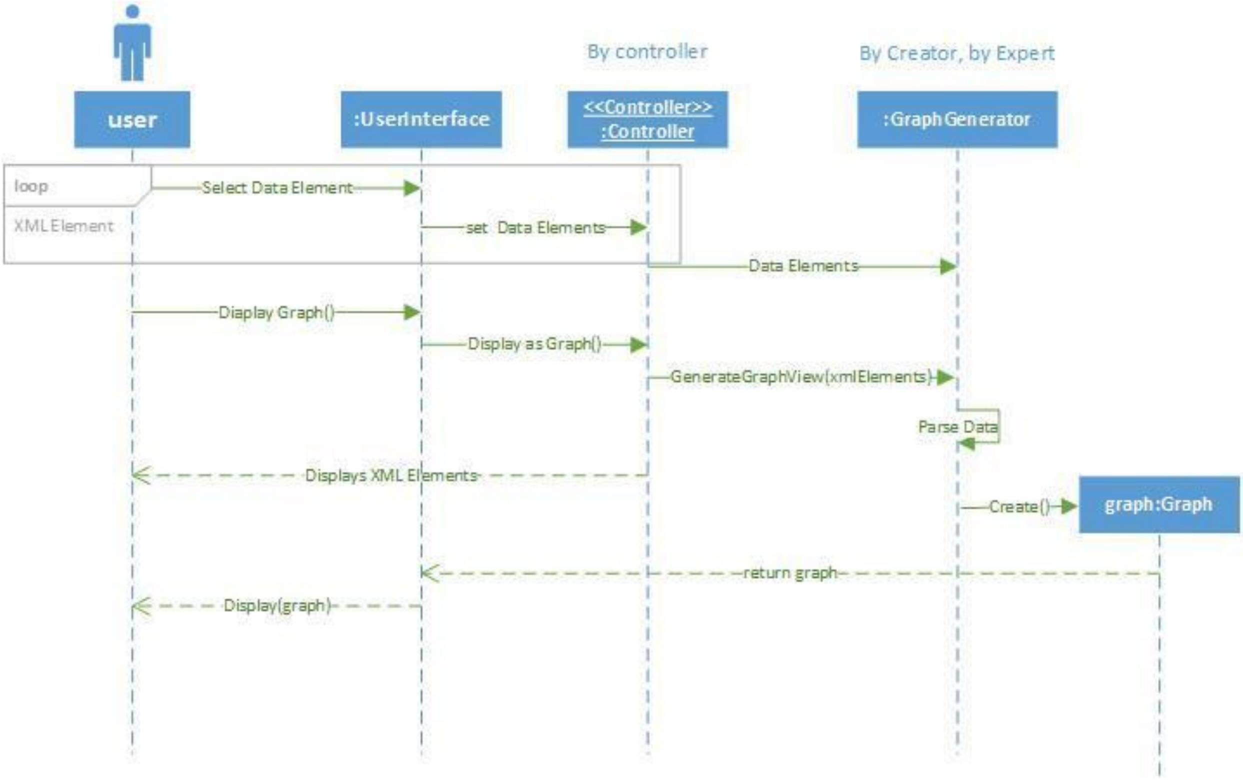


2.3 System Sequence and Sequence Diagram for UC-3: *Display as Graph*

SSD for UC-3: Display as Graph

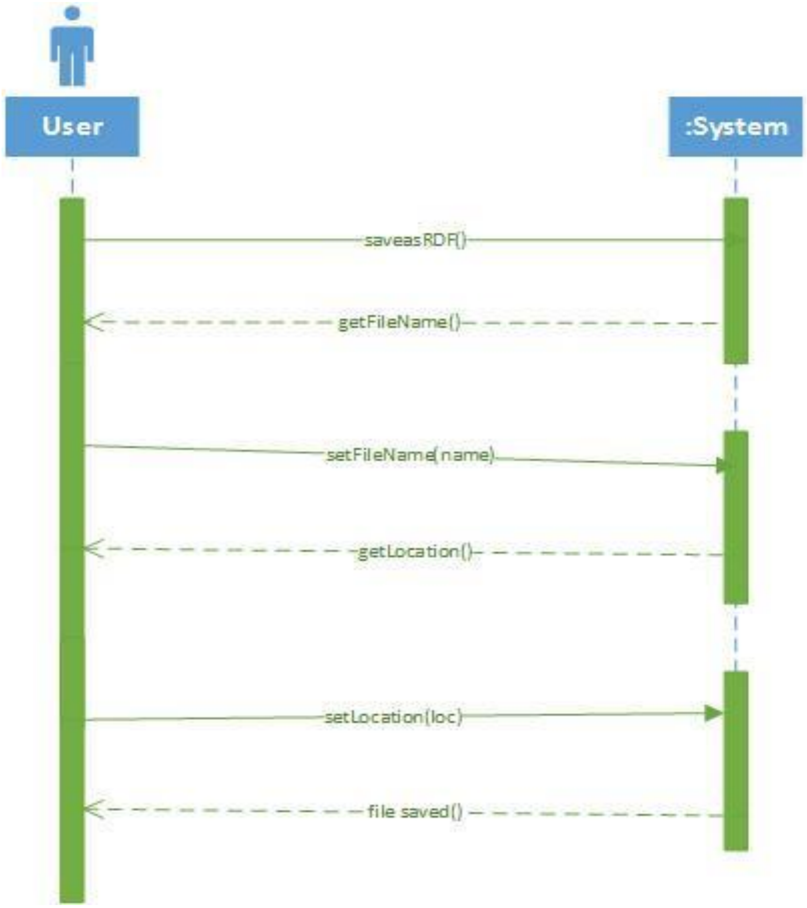


SD for UC-3: Display as Graph

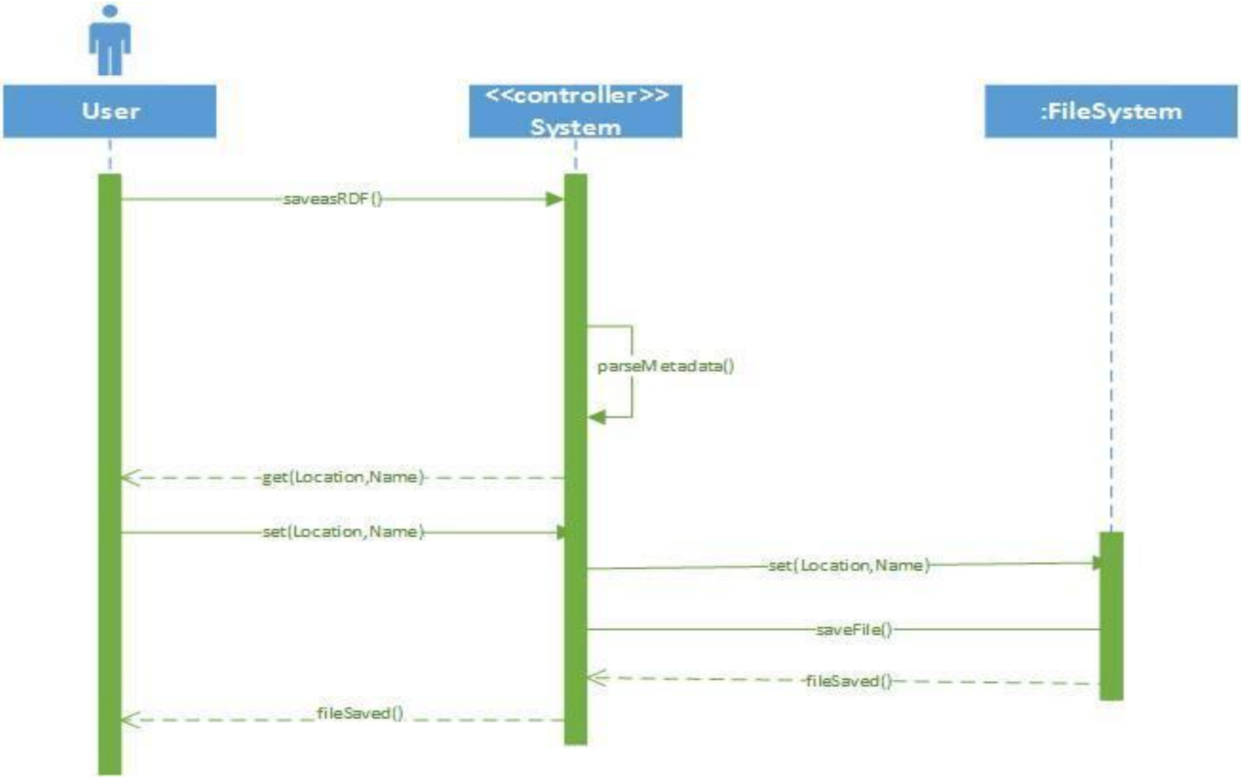


2.4 System Sequence and Sequence Diagram for UC-4: Save as RDF

SSD for UC-4: Save as RDF

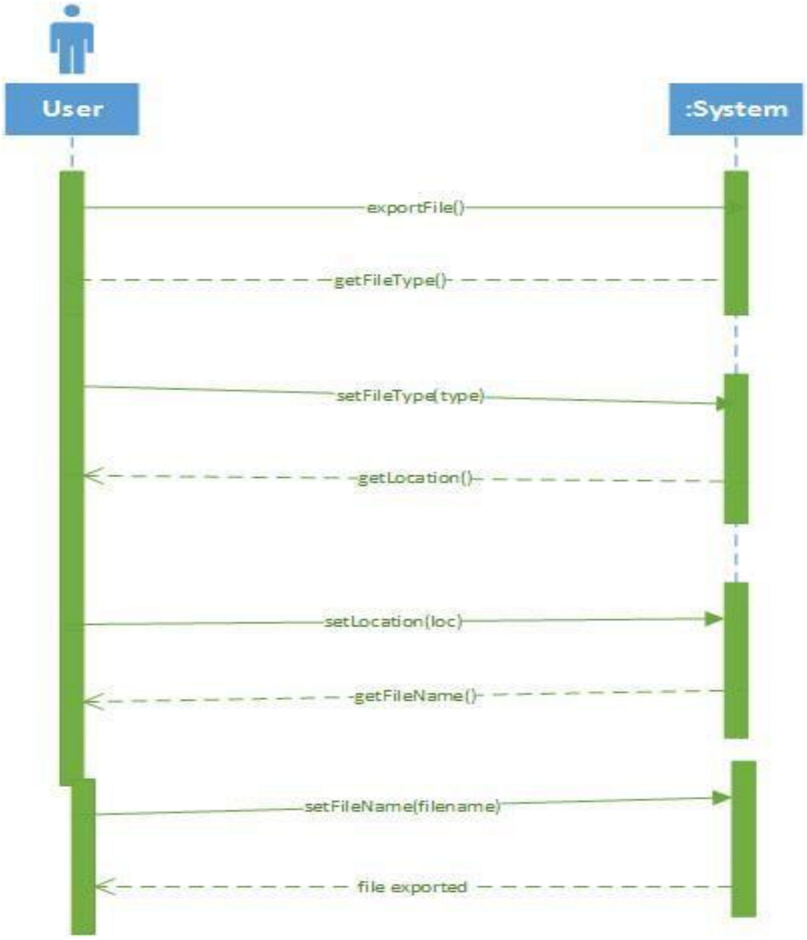


SD for UC-4: Save as RDF

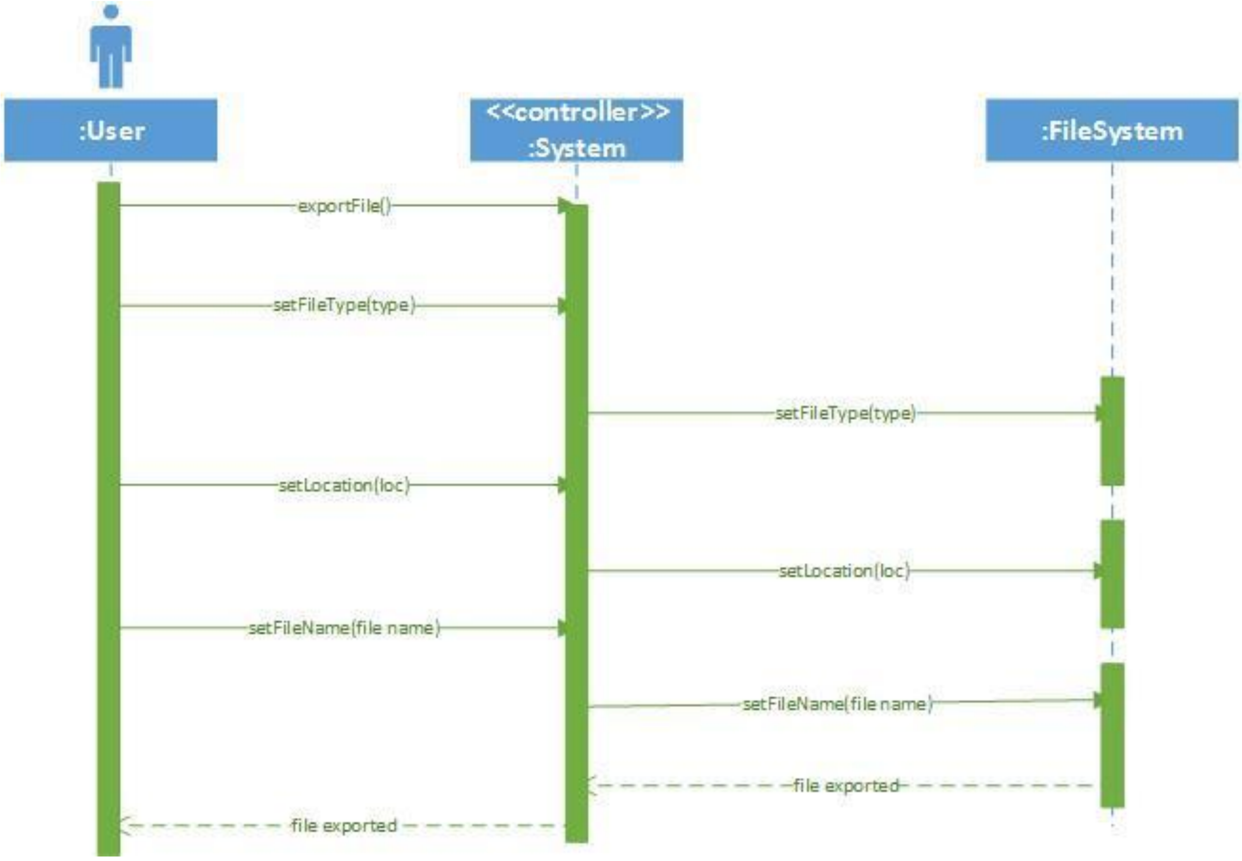


2.5 System Sequence and Sequence Diagram for UC-5: Export file

SSD for UC-5: Export file



SD for UC-5: Export file



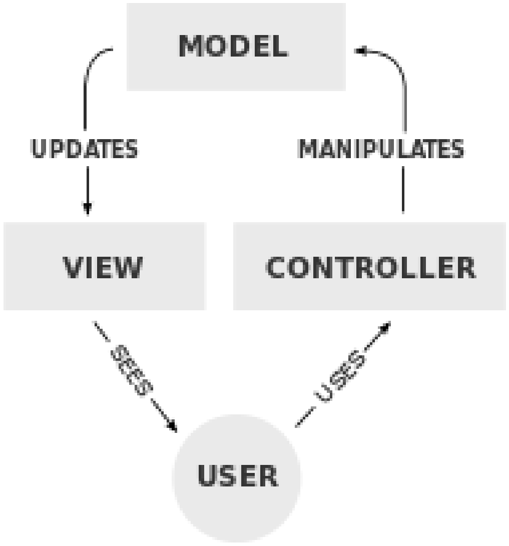
3. System Design

3.1 System Architecture

*The selected Architectural pattern for this system is **Model–view–controller (MVC).** It divides the application into three interconnected parts, so as to separate internal representations of information from the ways that information is presented to the user. MVC expresses the "core of the solution" to a problem while allowing it to be adapted for each system. In addition to dividing the application into three kinds of components, the model–view–controller design defines the interactions between them.*

The central component of MVC, the model, captures the behavior of the application in terms of its problem domain, independent of the user interface.

- *The model directly manages the data, logic and rules of the application. A model stores data that is retrieved according to commands from the controller and displayed in the view.*
- *A view can be any output representation of information, such as a chart or a diagram. Multiple views of the same information are possible, such as a graph or a table. A view generates an output presentation to the user based on changes in the model. A view controller generates an output view and an embedded controller.*
- *The third part, the controller, accepts input and converts it to commands for the model or view. A controller can send commands to the model to update the model's state. It can also send commands to its associated view to change the view's presentation of the model. [1]*



Collaboration of the MVC components.

3.2 Subsystem Architecture

3.2.1 Purpose

systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

A system architecture can comprise system components, the externally visible properties of those components, the relationships between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. [2]

3.2.2 Subsystem Class Structure

4.2.2.1 Singleton pattern

The singleton pattern is a design pattern that restricts the instantiation of a class to one object. This is useful when exactly one object is needed to coordinate actions across the system. [3] In our project we use it in the readXML class which extracts all the elements and their values from the XML file, and gives global access just to one instance of the XML file.

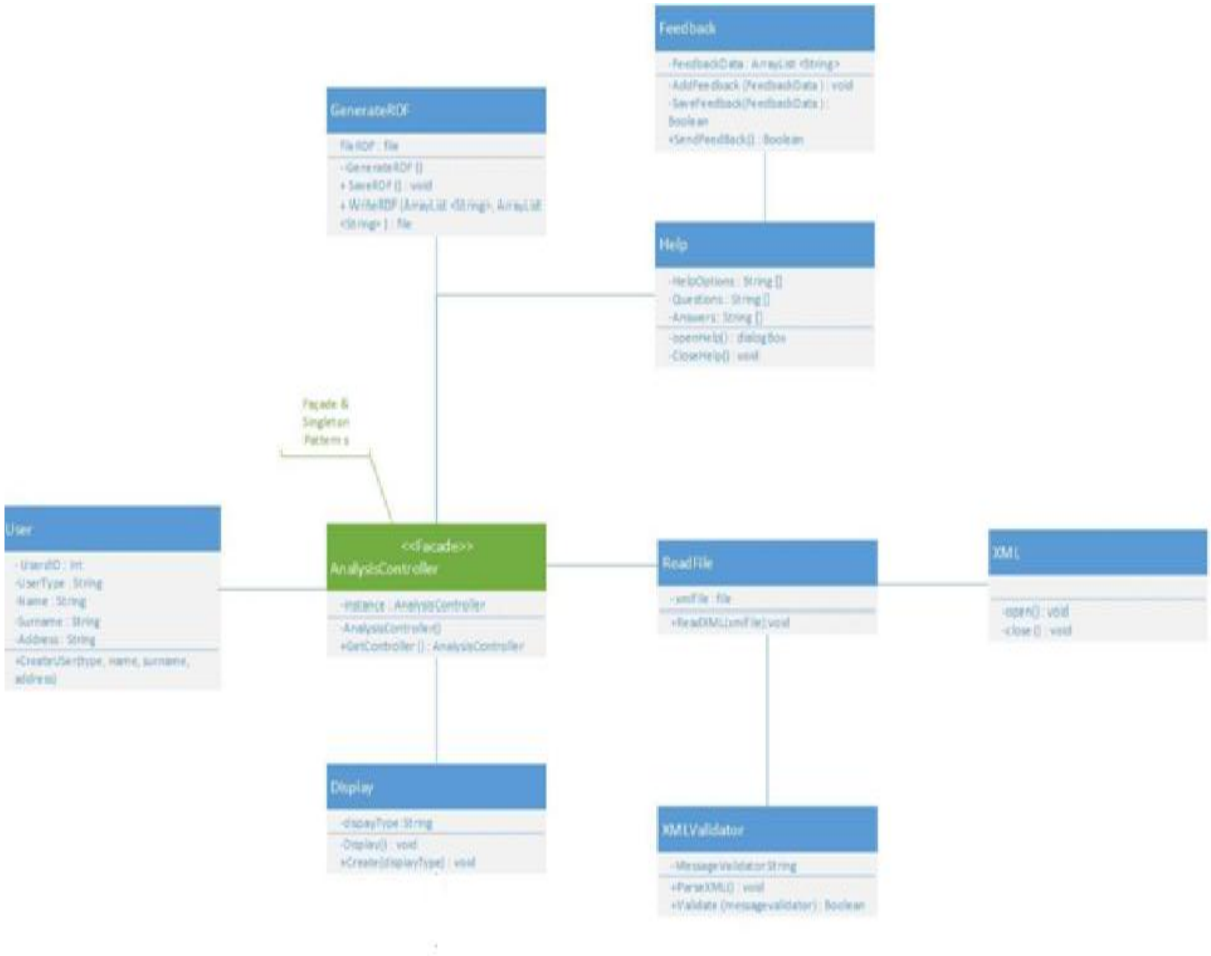
4.2.2.2 Facade pattern

A facade pattern is just an interface which separates client from the beneath subclasses, it provides a simplified interface to a larger body of code. The Facade design pattern is often used when a system is very complex or difficult to understand because the system has a large number of interdependent classes or its source code is unavailable. This pattern hides the complexities of the larger system and provides a simpler interface to the client. It typically involves a single wrapper class which contains a set of members required by client. These members access the system on behalf of the facade client and hide the implementation details. [4] In our project we use it in the main class UI.

4.2.2.3 Factory pattern

Factory pattern uses factory methods to deal with the problem of creating objects without having to specify the exact class of the object that will be created. This is done by creating objects by calling a factory method rather than by calling a constructor.
We have used the factory design pattern to generate the different views. When we select the view we want, the class will check the type of it and will execute the method they want. [5] In our project we use it in the class view.

<u>Subsystem Class Structure</u>



4. Detailed Design

Method Name	ParseXML() : List<samplexmlbean>
Class Name	It depends on the ParseXML
Functionality	It will parse xml data into array list and this list will be used for generating the table and graph.

Pseudo Code	Begin
	DocumentBuilde.parse(“xml file location”);
	Setbean.getElementByTag(“each tag name”);
	ArrayList.add(bean);
	End

Return Type	Here the Array List is being Returned

Method Name	Chart3DDemo(String, String, String, String, String, String): ChartPanel
Class Name	PieChart
Functionality	The basic functionality of this method is to generate the pie chart for the selected data.

Pseudo Code	Begin
	DataSet.add(selected data);
	ChartFactory.createPieChart3D();
	chartPanel.add(chart);
	end

Return Type	Pie Data set is displayed

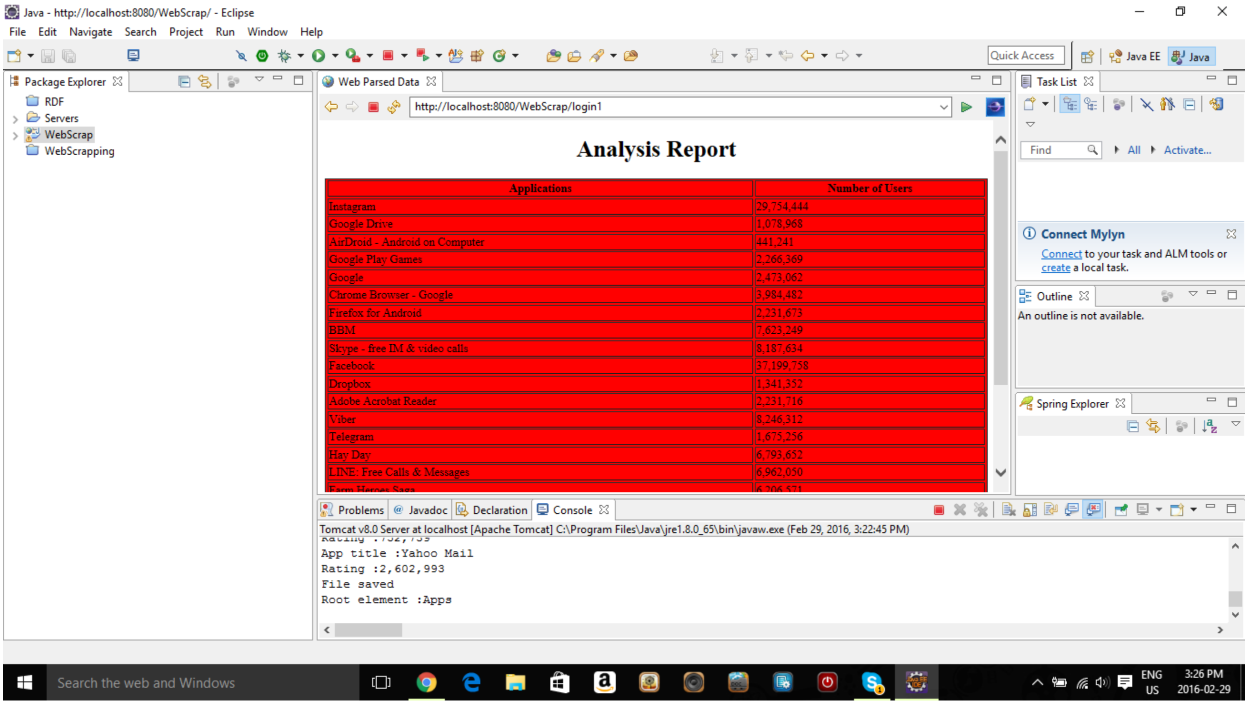
Method Name	DisplayGUI():void
Class Name	AnalysisMain
Functionality	The basic functionality of this method is to create a user interface which provides the panels of the graphs.

Pseudo Code	Begin
	Frame = new JFrame(“title”);
	rightPanel = getPanel();
	bottomPanel = getPanel();
	Frame.add(JPanel);
	End

Return Typeq	Frame is displayed

5. Prototype

5.1 Viewing XML Data As Table

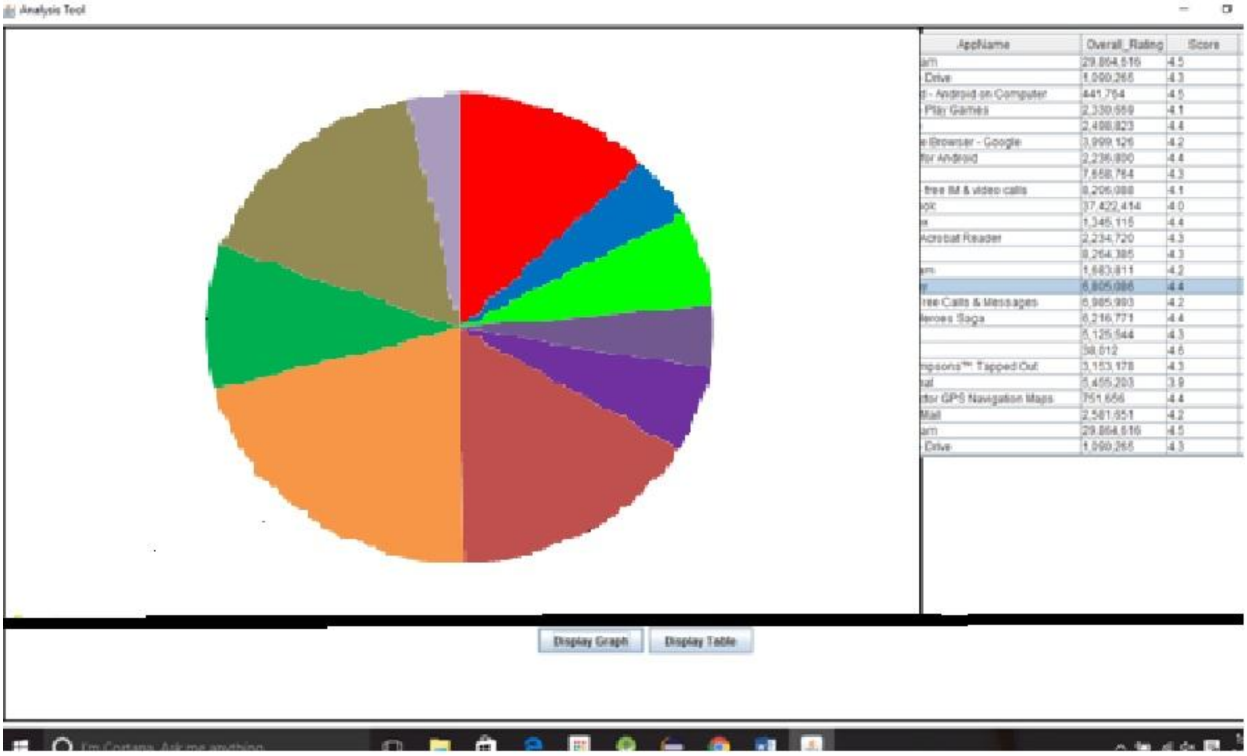

Java - http://localhost:8080/WebScrap/ - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Java
Package Explorer
RDF
Servers
WebScrap
WebScrapping
Web Parsed Data
http://localhost:8080/WebScrap/login1
Task List
Find
All
Activate...
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
An outline is not available.
Spring Explorer
Problems
Javadoc
Declaration
Console
Search the web and Windows
ENG
US
3:26 PM
2016-02-29

Analysis Report

Applications	Number of Users
Instagram	29,754,444
Google Drive	1,078,968
AirDroid - Android on Computer	441,241
Google Play Games	2,266,369
Google	2,473,062
Chrome Browser - Google	3,984,482
Firefox for Android	2,231,673
BBM	7,623,249
Skype - free IM & video calls	8,187,634
Facebook	37,199,758
Dropbox	1,341,352
Adobe Acrobat Reader	2,231,716
Viber	8,246,312
Telegram	1,675,256
Hay Day	6,793,652
LINE: Free Calls & Messages	6,962,050
Farm Heroes Saga	6,206,571

```
Tomcat v8.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jre1.8.0_65\bin\javaw.exe (Feb 29, 2016, 3:22:45 PM)
Rating :732,735
App title :Yahoo Mail
Rating :2,602,993
File saved
Root element :Apps
```

5.2 Displaying Data as Pie Chart



AppName	Overall_Rating	Score
am	29,864,616	4.5
Drive	1,090,268	4.3
d - Android on Computer	441,764	4.5
Play Games	2,330,569	4.1
	2,498,823	4.4
e Browser - Google	3,999,126	4.2
for Android	2,236,890	4.4
	7,658,784	4.3
free IM & video calls	8,206,088	4.1
ook	37,422,414	4.0
x	1,346,115	4.4
Acrobat Reader	2,234,720	4.3
	8,264,385	4.3
am	1,683,811	4.2
y	6,805,086	4.4
ree Calls & Messages	6,985,993	4.2
Heroes Saga	6,216,771	4.4
	5,125,544	4.3
	98,612	4.6
mpsons™: Tapped Out	3,153,178	4.3
at	5,455,203	3.9
tor GPS Navigation Maps	751,656	4.4
Mail	2,581,651	4.2
am	29,864,616	4.5
Drive	1,090,268	4.3