SE3352a: Software Requirements and Analysis

Smash Studios

Modern Software Requirements Specification
For WE Admission System

Version 2.0

SMASH

STUDIOS

Revision History

Date (dd/mm/yy)	Version	Description	Author
12/11/15	1.0	Created the shared document for the group work	Everyone
12/11/15	1.1	Completed the introduction, which includes the purpose, scope, definitions, references, and overview	Everyone
13/11/15	1.2	Completed the introduction, survey description and use-case model hierarchy for the use-case model survey	Everyone
15/11/15	1.3	Created the diagram of the use case model	Everyone
		Assumptions and Dependencies?	Everyone
16/11/15	1.5	Created the use case specifications of the requirements	Everyone
18/11/15	1.6	Generated the rest of the requirements of the system	Everyone
19/11/15	1.7	First proof read and UC Diagram revision	Everyone
19/11/15	1.8	Second proof read and requirements revision	Everyone
20/11/15	1.9	Final revision	Everyone
20/11/15	2.0	Formatted the document	Everyone

Table of Contents

Modern Software Requirements Specification

1. Introduction

1.1 Purpose

The driving purpose behind the creation of this Modern SRS software is to facilitate the automation of second year engineering student placement, and thereby reduce the significant workload of Undergraduate Services Office during the placement period. As it currently stands, the Undergraduate Services Office (USO) performs much of this process manually. To ensure that the software under development meets the needs of the clients in the USO, this project also aims to implement a user-friendly interface, allow dynamically created rules, and maintain a centralized database for student records.

1.2 Scope

This Student Admission System software development project will be an automated suite of programs, that interacts directly with a database containing the first year engineering students ITS records, and then allows an employee of the USO to apply rules to the dataset. The efficiency of the USO will be greatly increased and the application process itself will be streamlined. By automating this process and providing a user-friendly interface, this software project will ensure that the needs of the USO are met in a way that guarantees end-user satisfaction.

In particular, the design goals of the system can be modularized by considering several key functionalities, namely, the communication between the system and a database, the interaction between the USO employees and the system and finally, the generation of a list of students to whom the system recommends granting acceptance.

1.3 Definitions, Acronyms and Abbreviations

Term	Definition
SRS	Software Requirements Specification. Document and process defining the function and non-functional requirements for a software project.
ITS	Information Technology Services. Western University's technical support office.
USO	Undergraduate Services Office. Western University's undergraduate Engineering administration.
UC	Use Case. Software term describing a particular functionality of a development project.
IP	Intellectual Property.
API	Application Programming Interface. An imported collection of tools and functions designed to aid in developing software.

GUI	Graphical User Interface.
IP Address	Internet Protocol Address. The address assigned to each internet end system to allow it to be found by, and access other online resources.
SAS	Students Admission System. The process that handles the Intent to Register process.
Database	Software construct that stores information. In this system a database will be storing the student records.
Administrator	A system user with privileges elevated above the norm.

1.4 References

Title: Assignment 1: Developing the Software Requirements Specifications SRS

Reference: Ouda, A. "Assignment 1: Developing the Software Requirements Specifications SRS." SE 3352 Software Requirements and Analysis. Western University, 29 Oct. 2015. Web. 18 Nov. 2015. <https://owl.uwo.ca/access/content/attachment/04dfffe3-4b6b-4215-bbba-9eb24dc12ebb/Assignments/624dd898-2220-47a2-9a44-85377fc6f258/SE3352a_assignment1_2015_.pdf>.

Title: Accessibility Western

Reference: "Accessibility Western" Accessibility UWO. Western University, 2015. Web. 18 Nov. 2015. <http://accessibility.uwo.ca/>.

Title: IBM Common User Access

Reference: "IBM Common User Access" Wikipedia. 21 July 2015, <https://en.wikipedia.org/wiki/IBM_Common_User_Access>

1.5 Overview

Chapter two, entitled Overall Description, provides an informal description of the required functionality that is to be included in the final deliverable. This will serve to give context to the more precise technical specifications that come about in chapter three. Non-technical audiences will find chapter two understandable with any limited technical terms clearly described.

The final chapter is tailored to developers, and those with a technical background. It is within this section that the technical descriptions of use cases are laid out.

2. Overall Description

2.1 Use-Case Model Survey

Introduction

A use case is an essential tool used to describe the functionality of a system. It shows the interaction between the system and external actors, as well as basic functionality flow. The use cases in this project work cohesively to achieve the end goal of an automated distribution of first year engineering students to each engineering faculty. The system is going to be used mainly by the Undergraduate Services Office, which would allow them to efficiently obtain student Intent to register data. Each use case in this model is going to serve one main functionality which is to allow the Undergraduate Services Office to distribute the first year students with no manual data entry.

Survey Description

The Student Admission System is a web-based admission system that connects to the currently in-place Information Technology Services (ITS) system responsible for communicating with the students. The purpose of this system is to automate the admission process carried out by the Faculty of Engineering. The system is maintained by an administrator and allows users to control the admission criteria that are used by the system to distribute students for each program. The system also exchanges data with the ITS system under the admin's control. Additionally, the system authenticates each user at login, and loads an appropriate user-interface based on the user roles associated with the account.

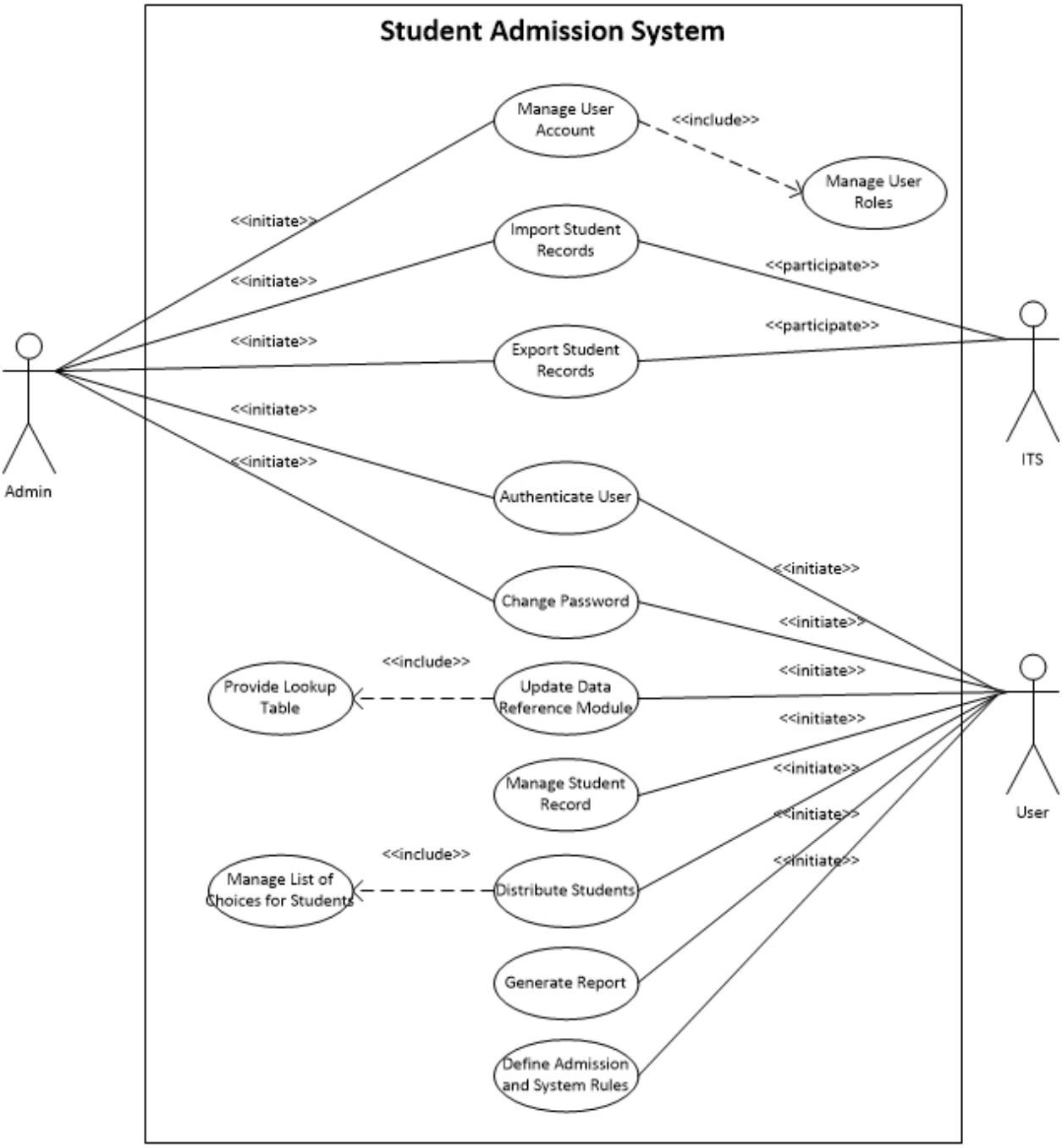
Use-Case Model Hierarchy

Actors	**Descriptions**
Admin	The administrator of the system is responsible for ensuring around-the-clock technical maintenance and support.
User	The user utilizes the system in order to define admission and distributions criterias that is then used by the system.
ITS	The Information Technology Services system, which contains a database that stores information about student records and their list of choices.

Use Cases	**Descriptions**
Manage User Account	System shall allow the admin to add, edit and delete user accounts. It includes the Manage User Roles.
Manage User Roles	System allows the admin to manage the roles and privileges associated with a user accounts.

Import Student Records	System allows the admin to retrieve student records from the ITS system.
Export Student Records	System allows the admin to export student records to the ITS system.
Authenticate User	Authenticate the user by verifying their username and password through a secure login process.
Change Password	System allows the user to change their password.
Update Data Reference Module	System provides the ability to update the reference data module. This includes the Provide Lookup Table use case.
Provide Lookup Table	System provides the ability to search the lookup tables/code tables data that is used in the application, such as faculty codes.
Manage Student Record	System allows addition, modification and deletion of students record through a single data entry form for the user. The record includes basic student information, student's transcript, and list of student's choices. Includes the Manage List of Choices for Students use case.
Manage List of Choices for Students	System allows the user to add, modify or delete the list of program choices to which the students will be distributed.
Distribute Students	System allows the user to distribute the students based on their choices. The rules of distribution are defined by the user in the Define Admission and System Rules use case.
Generate Report	System allows the user to produce a report of all the students currently distributed in a specified system.
Define Admission and System Rules	System allows the user to define the admission and distribution rules for a program or option. Eg. Minimum percentage to enter the program.

Diagram of the Use-Case Model:



SMASH STUDIOS

Assumptions and Dependencies

The development team believes that no assumptions or dependencies regarding a key technical feasibility, subsystem, component availability, or other project-related assumption upon which the viability of the software described is dependent has been made.

3. Requirements

3.1 Use-Case Specifications

Use case name	Manage User Account
Participating actors	Initiated by Admin
Entry condition	Admin is logged into the system
Flow of events	1. Admin clicks on 'Manage User Account' tab/button 2. Admin enters information about a new user, or edits/deletes the profile of an existing user 3. System displays the acknowledgment to the Admin
Exit condition	Admin receives acknowledgment that a user account has been added/modified
Quality requirement	None

Use case name	Manage User Roles
Participating actors	Initiated by Admin
Entry condition	The system is in 'Manage User Account' mode
Flow of events	1. An Admin requests to manage a user's role 2. Admin can create, modify and remove different user roles by which the privileges will be determined 3. System displays the acknowledgement to the admin
Exit condition	Admin receives acknowledgment that a user role has been created or modified
Quality requirement	None

Use case name	Import student records
Participating actors	Initiated by Admin Participated by ITS
Entry condition	Admin is logged into the system
Flow of events	1. Admin clicks on the 'Import Student Records' tab/button 2. Students records gets collected from the ITS system 3. A progress bar will be displayed, showing how long it would take for the process to be completed 4. System displays the acknowledgement to the admin
Exit condition	Admin receives the acknowledgement that student records has been imported
Quality requirement	None

Use case name	Export student records
Participating actors	Initiated by Admin Participated by ITS
Entry condition	Admin is logged into the system
Flow of events	1. Admin clicks on the 'Export Student Records' tab/button 2. Students records get exported to the ITS system 3. A progress bar will be displayed, showing how long it would take for the process to be completed 4. System displays the acknowledgement to the admin
Exit condition	Admin receives the acknowledgement that student records has been exported
Quality requirement	None

Use case name	Authenticate User
Participating actors	Initiated by User Initiated by Admin
Entry condition	Admin or User logs in
Flow of events	1. Admin or User presses the login button 2. Request is acknowledged by the system 3. The actor is prompted to enter his/her credentials 4. The user credentials are submitted 5. The system checks their validity. 6. The system grants the actor access if the credentials are valid. If they are invalid, the system tells the user so and awaits another attempt.
Exit condition	System has checked the validity of the entered credentials, and admin or user receives the acknowledgement that he/she is logged in
Quality requirement	None

Use case name	Change Password
Participating actors	Initiated by Admin Initiated by User
Entry condition	Admin or User is logged into the system
Flow of events	1. User or Admin opens the Change Password form 2. The system requires the actor to be logged in. If they aren't logged in, the system prompts them to log in 3. An already logged in user will be prompted to enter his/her current password 4. Upon successful validation of the entered credential, the actor will be prompted to enter his/her new password twice 5. The form will be submitted, and the system will update the actor entry
Exit condition	Successful changing of the actor's password, and admin or user receives the acknowledgement that password has been changed
Quality requirement	None

Use case name	Update Data Reference Module
Participating actors	Initiated by User
Entry condition	User request to update data reference module
Flow of events	1. User clicks on update module button 2. A form appears with the current configuration of the data reference module 3. User makes alterations to the module as needed 4. The user saves the current data reference model
Exit condition	The system dynamically updates with the most current data reference module
Quality requirement	The system will not accept invalid input in the date reference module

Use case name	Manage Student Records
Participating actors	Initiated by User
Entry condition	User is logged into the system
Flow of events	1. User clicks on “Manage Student Records” button. 2. The user can choose either add, delete or edit for the list of choices for the students. 3. The user has made his alterations to the list of records for the students.
Exit condition	User receives acknowledgment that a student role has been modified
Quality requirement	The system will not accept invalid user inputs (improper address format, improper telephone format, blanks, etc.) If error is found within forms, users should be notified above the sections where the errors occurred

Use case name	Provide Lookup Table
Participating actors	Initiated by User
Entry condition	User requests the reference data model
Flow of events	1. User requests to edit the reference data model

	2. System provides the user with lookup table for the user to enter codes 3. User can search codes currently present in the reference data model for which the system will return the value of from the table
Exit condition	User leaves the page with the lookup table
Quality requirement	The lookup table should filter based on user search input at a reasonable speed

Use case name	Distribute Students
Participating actors	Initiated by User
Entry condition	User request to distribute students
Flow of events	1. Admin clicks on distribute students button 2. A progress bar would appear showing how long it takes until it finish distributing 3. The student records would be distributed in the system
Exit condition	A form would appear showing that the process has finished
Quality requirement	The system will take no longer than 5 minutes to finish exporting student record in the system

Use case name	Manage List of Student Choices
Participating actors	Initiated by User
Entry condition	The user has initiated the Distribute Students use case and requests to manage the list of choices for students.
Flow of events	1. The user can select either add, delete or edit for the list of choices for the students. 2. The user has made his alterations to the list of choices for the student .
Exit condition	The User has finished managing the list of choices for students.
Quality requirement	None

Use case name	Generate Report
Participating actors	Initiated by User
Entry condition	The user requests a report of students in each program to be generated
Flow of events	1. The user presses the generate report button 2. The server does the required the computation needed
Exit condition	The user now has a report of students in each program
Quality requirement	The report format must be acceptable by the departments and by the ITS systems

Use case name	Define Admission and System Rules
Participating actors	Initiated by the User
Entry condition	User requests to define the admission and system rules
Flow of events	1. User selects an option to define the admission and system rules 2. User is presented with a form to define the admission and distribution rules for each program or option. 3. User completes entering the required information into the form and clicks submit. 4. Acknowledge to the user that their request has been successfully processed by the system.
Exit condition	A successful submission of the provided form and the according server response
Quality requirement	Form to define the admission and distribution rules must be easy-to-use and provide details regarding each required input field

3.2 Functionality

1. The system should provide a way for users to log in. Upon login, the system should authenticate the username and the encrypted password. The system should dynamically provide user with a set of controls depending on their type of user account.
2. The system will come pre-packaged with a single admin account. This account will be capable of creating, new accounts of different roles/privileges. Additionally, administrators can edit/delete user accounts if they choose.
3. Each user shall have the ability to change his or her password.
4. The system shall provide a customizable reference data module. Reference data includes the lookup tables/code tables data that is used in the application, such as faculty codes, department codes, program

codes, rejection reason codes, etc.

5. The system should provide a form for users to add, modify, and/or delete student records. Entries that can be modified should include profile number, name, email, student transcript, along with the list of student's choices.
6. The system should provide a functional form to specify admission/distribution rules for the choices with which students will be distributed. Such rules can include the maximum students allowed in programs, minimum percentage for specific courses, and required prerequisites.
7. The system shall be capable to automatically distribute the students to their choices according to the defined rules and to provide the reasons if a student get declined.
8. After the distribution takes place, the system shall provide list of students per each program in a format acceptable by the departments and by the ITS systems.
9. The system shall provide a tool to import and export student records from and to ITS database.
10. The system administrator shall have the ability to create different user roles by which the privileges of the users will be determined.

3.3 Usability

1. The newly developed Student Admission System will have an improved user interface that dramatically improves the user experience for the application. With smart and aesthetic design choices, we can create a very user-friendly, yet functional piece of software. The average training times for normal and power users will be extremely low, thus creating a new standard of efficiency.
2. With a very user-friendly interface, we hope to achieve remarkable task times for the functionality of the application. The expected time will be 3-4 clicks to fully complete a task. With strategic navigation and specific tasks in mind, this is easily achieved.
3. A common standard in modern applications, that users know and like, is to utilize a single dynamic page per functionality and to organize functionalities via tabs. The application will have a hybrid model of these practices. Another usability feature that users enjoy is having a progress bar. The newly developed system will incorporate a loading and status bar.
4. In order to maximize the usability of the application, the IBM Common User Access (CUA) standards will be implemented. Some specific CUA conventions that will be highlighted are: all operations can be performed with mouse or keyboard, navigation within fields can be managed via the Tab key and Shift+Tab keys, and the application will have a Help Menu that can be accessed upon request.

3.4 Reliability

1. Availability: The system should ideally be available at all times. System users should be able to access the system at any hours of the day. If maintenance is required, a three day notice should be displayed to all users prior to the maintenance. Maintenance should take no longer than 12 hours to finish.
2. Mean Time Between Failures (MTBF): The mean time between failures should be no longer than 12 hours. Any system failures should automatically and instantaneously be alerted to the maintenance team.
3. Mean Time To Repair (MTTR): The mean time to repair failures should be no longer than 6 hours. Efforts to determine the source of the failure should also take no longer than 6 hours. Altogether, these make up an MTBF of 12 hours.
4. Accuracy: The system's output will be in the form of student-program pairs. Accuracy is not so much a concern with this aspect of our system. The only mathematical component might be calculating student averages, in which case, grades should be rounded to a precision of 2 decimal places for comparison.
5. Maximum bugs or defect rate: The system should have a max of 3 bugs/KLOC. Additionally, the bugs must only be noticeable in extremely rare circumstances and must be able to be resolved easily and quickly.

If these 2 requirements are not satisfied, the bugs must be fixed immediately as there becomes too much of a risk for error during execution.

6. Bugs or defect rate:
    a. Minor bugs: defined by the 2 requirements in the last point. There can only be 3 of the bugs/KLOC.
    b. Significant bugs: bugs that can affect the value of data or student distribution rules. These can result in incorrect student distribution and ultimately, incorrect system output. If these bugs are caught, they must be fixed (0 bugs/KLOC).
    c. Critical bugs: bugs that can result in complete loss of data or complete inability to use certain parts of the system. These bugs must be immediately fixed if caught because they directly affect the functionality of the system and in some cases, completely halt the use of the system.

3.5 Performance

1. The response time of the distribute operation should take no more than 5 seconds to perform. Given this maximum operation time and an estimate of 250 applicants, the system should be able to process approximately 50 transactions per second. Operation times for adding, modifying, and deleting accounts or profiles should appear instantaneous to all user-roles capable of performing such actions.
2. The system will come prepackaged with a customizable reference module data including faculty codes, department codes, program codes, rejection reason codes, etc. The system will communicate with the ITS through its API to retrieve student information to be processed.
3. In the occurrence of malicious event (i.e, malicious input, disruptive user's, possible runtime errors, etc) the system will enter a degradation mode. While the system is in this mode, a notification will be displayed to indicate that there is some error in the current system state. In order to handle this error, the system could take state snapshots (i.e, a backup of the latest working state) and reuse that snapshot in order to get rid of any malicious event that has happened. By loading up the latest working snapshot, the system will dynamically get rid of the degradation mode and the malicious event by reverting to a stable system state that did not encounter the original interruption.

3.6 Supportability

1. Modular Programming: The system will be designed and built in a very specific manner to accommodate supportability and maintainability. The system will utilize a modular programming architecture in order to pinpoint specific areas of interest in the software that requires a form of support or maintenance. By building the system in this fashion it is possible to alter code in one module and not have it affect other modules. This is important because it dissolves the possibility of error occurring in other parts of the software. Also, modular programming allows multiple code entry points from various teams. This means a developer will not interfere with another developer, thus creating a more efficient way to support and maintain different parts of the software at the same time when needed.
2. Cross-Browser Support: Because the system will be web-based, the libraries and programming languages used must be supported across all browsers available on the user's machine. This means that instead of JavaScript, the system would use JQuery which provides cross-browser support for JavaScript functions and DOM structures.
3. Responsive Programming: Effort should also be put into responsive programming. Since web-based applications can be accessed from a myriad of devices (including desktops, laptops, phones, and tablets), multiple device sizes will present problems for the structure and appearance of the web-application. Tools such as Bootstrap will ensure that the system can be accessed, independent of the device used (i.e, dynamic resizing).

3.7 Design Constraints

1. External design: Efforts should be put into making code modular. For example, external stylesheets and scripts should be referenced inside the HTML document to keep the programming environment more readable and maintainable. Each page should be contained within its own folder and images that are on these pages should be in their associated page folders. External class libraries and developmental tools should be put into the root of the website.
2. Accessibility: Focus should be put on improving accessibility for peoples with disabilities; in particular, the visually-impaired. One such way to achieve this is to ensure alternative text is linked to non-text page elements (flash objects, videos, images, etc.). Another way to achieve this is to use the new HTML5 elements that provide insight into the structure of HTML pages (headers, asides, articles, footers, etc.).
3. Dynamic Behaviour: The system should be designed in such a way that there is no static behaviour. The program must automatically generate new information based on a dynamic configuration that can be altered at runtime. The system should not in any way be required to restart in order to see new results or to handle a new configuration. The application should change in response to modifications of the data-set.

3.8 Online User Documentation and Help System Requirements

The development team will be providing technical documentation to ensure basic troubleshooting capabilities for the USO staff. However, this will not be online. Insofar as online resources, each member of the development team will be providing two email addresses to guarantee that in the case of a serious error members of the USO will be able to contact the team. All team members will be returning to Western University next year and thus the USO will receive one year of in-person support.

3.9 Purchased Components

All components will be created by the development team and thus be proprietary software. The team does not believe that the scope of this project will require any third-party APIs. This will alleviate many of the legal concerns involving purchasing rights and third-party legal constraints.

3.10 Interfaces

User Interfaces

There will be different views depending on the privileges of the current user. Namely, Administrator users will be presented with an interface that is different from USO Employee users and so on. The interface itself will be designed for an optimal user experience. This will allow for users to implement selection rules with no need for programming experience.

Hardware Interfaces

This project does require a hardware interface as there will be a server that is accessed via HTTP requests. The final deliverable will make use of a Secure-Socket Layer (SSL) to ensure the security of the requests. The HTTP requests will make use of port 80 on the client side. For the server, the port number will be specified at a later date. The IP address of the server will also be specified at a later date. The server may or may not be implemented with a static IP address.

Software Interfaces

1. Many newly developed systems utilize an entirely new database for the software being produced. However, that does not mean we should ignore the legacy system and backwards compatibility. One software interface that should be considered is the legacy ITS database. This should still interact with the newly developed system and its components without any errors. This is a component that is considered to be reused from another application (the old ITR system).
2. It is also not uncommon for software systems to need additional support via purchased components. In the case of the student admission system that is currently being developed, we need additional support to host the application and handle the traffic that will be generated. The interface that interacts with the system would be the University of Western Ontario's web servers. This is a component that is purchased and is required to interact with the newly developed system in order to achieve a stable and successful product.

Communications Interfaces

The communications interface was briefly discussed in the Hardware Interfaces Section above. In short, communication will need to occur between the client machine that a USO employee, or other actor, is using and the server that contains the Student Records. This communication will be handled via HTTP with SSL providing encryption for the communication.

3.11 Licensing Requirements

The final deliverable will be the proprietary creation of Smash Studios. The system will be sold to the Western University Engineering Department where it will be licensed for use for the Undergraduate Services Office. This is the limit of the usage of the system. Any usage outside this domain will require written consent from Smash Studios.

3.12 Legal, Copyright and Other Notices

The information of students attained by the system from the ITS including names, IDs, addresses, grades, and registration decisions cannot be distributed to members or parties outside the Engineering Faculty's Undergraduate Services office without prior permissions from said students.

3.13 Applicable Standards

This project will have applicable standards that it must comply with. The standards arise from three main domains: quality and regulatory, industry standards for usability, browser compatibility.

The first set of standards will require the final deliverable to handle all student records with care. As this can be considered sensitive data security measures will need to be in place to protect access to the database. It is also vital to the success of the system that no erroneous records are created. The system must guarantee no corruption of student data even in traumatic server events.

Usability requirements will be met by ensuring that those with visual impairments can still access the system. This will conform to Western University's policy on accessibility and accommodation for its employees and students as stated on the Accessibility Western site.

The final set of standards is the most technical of the three. The required functionality taken from the document provided states that "All users should be able to access the system with any web browser supporting cookies, and JavaScript installed in any desktop computer or laptop". Therefore, the development team will ensure that the final deliverable is compatible with technology across the web browser spectrum.