**Lab 1- CORSICA Product Description**

**Red Team**

**Patrick DeBerry**

**Old Dominion University**

**October 9, 2014**

**Version 1.2**

# **Table of Contents**

**1 INTRODUCTION ...........................................................................................................3**

**2 CORSICA PRODUCT DESCRIPTION ………........................................................... 4**

**2.1 Key Product Features and Capabilities ………................................................. 5**

**2.2 Major Components (Hardware/Software) …………….................................... 8**

**3 IDENTIFICATION OF CASE STUDY …………….................................................... 9**

**4 CORSICA PRODUCT PROTOTYPE DESCRIPTION ………................................10**

**4.1 Prototype Architecture (Hardware/Software) ……….................................... 12**

**4.2 Prototype Features and Capabilities ………………........................................15**

**4.3 Prototype Development Challenges ………………..........................................18**

**Glossary……………………………………………………………………………….……….. 19**

**References……………………………………………………………………………….…….. 21**

 **List of Figures and Tables**

**Figure 1 - Hardware Components…………………………………………...…………...….8**

**Table 1 - Real World Product and Prototype Comparison…………………...…………. 11**

**Figure 2 - Major Functional Components Diagram……………………………...….....… 12**

**Table 2 - Description of Algorithms…………………………..………………………...…. 13**

**Table 3 – User Access Table………………………………………………………………....17**

**Table 4 - Prototype Challenges………………………………………...………….……...... 19**

# **CS 411W LAB I - PRODUCT DOCUMENT**

# **1** **INTRODUCTION**

Attending a university has become an important step in obtaining a successful job. The

Bureau of Labor Statistics has shown that citizens with a higher education not only earned a higher salary, but it also increases their chances of getting a job, lowering the unemployment rate (2014). Universities have become more sought after, increasing enrollment over the past years. Universities, such as Old Dominion University (ODU), have the issue of courses reaching capacity due to their student population. An increase in the students being admitted has led to the problem of the current course enrollment system becoming insufficient for handling course overflow.

Old Dominion University uses Banner as its current enrollment system and Leo Online as its user interface for Banner. While most of the enrollment system works well, it lacks an efficient wait-list process. Some departments have been using a wait-list process, such as the Computer Science department. The current wait-list is triggered when a course reaches capacity, closing the course. A student must email the course’s professor and request to either have the capacity increased and be added to the course or to be put on the wait-list. Once a seat opens they could have the opportunity to add the course. The issue with the current wait-list process is when a seat opens in the course the course is no longer closed, allowing any student to add the course. This makes the current wait-list unfair to students who are on the wait-list.

The wait-list causes problems for ODU’s Administration, students, faculty, advisors, and schedulers. The current course wait-list implementation causes excess student communication with faculty and Administration in order to facilitate the student's desire to be enrolled in a specific course when it has reached capacity. While the student and faculty are attempting to resolve this issue, they are pulled away from their other tasks. Implementing an improved wait-list feature in the current course add/drop system; will greatly reduce the number avoidable communications.

A new wait-list system needs to be implemented. CORSICA, or COmputeR ScIenCe wAit-list, will be the solution to these issues. A student wishing to take a Computer Science (CS) course at ODU will be put on a new wait-list which will allow the student to monitor their status and get notifications on their position on the wait-list. This will assist students in joining a class if a space becomes available. CORSICA is only for the CS department; however, it can be adapted to any department.

# **2** **CORSICA PRODUCT DESCRIPTION**

CORSICA is a wait-list addition to the current CS enrollment system. The wait-list process will be a more effective way for students to enroll in courses that have reached capacity, providing a fair process. CORSICA will be made available online for Administrators, Advisors, Schedulers, Students, and Visitors. When a course reaches capacity, CORSICA will start by locking the class, so only students on the wait-list may join the course. If a student wants to join the course, they must first join the wait-list for a course that has no more available seats. A seat may become available through two ways, either a student drops the course, or the capacity of the course is raised. Once the course has an opening, the student will receive a notification through an e-mail and/or text message. E-mails and text messages are the fastest and most efficient methods for delivering messages. The notification system will keep students from constantly checking to see if an available seat has opened. CORSICA allows the student at the top of the list to have priority, allowing them to register first. The list will move the rest of the students up on the list, opening more sections on the wait-list. CORSICA’s notification process will allow students to be notified of an open seat in a quick and efficient manner.

CORSICA is designed to maintain a course’s wait-list after it has reached capacity. CORSICA notifies students of available seat openings. Students are required to take certain classes necessary to graduate with a CS degree. When a student gets to their senior year, they could face the issue of not being able to take a necessary class to graduate due to the class reaching capacity. This would force the student to postpone their graduation. CORSICA would allow the student a more effective way of joining the course they needed.

## 2.1 Key Product Features and Capabilities

 CORSICA’s notification system will be one of the key features. The notification system will notify students in a timely manner when a seat becomes available. The student will have the option to receive the notification through e-mail, text message or both. CORSICA will only notify the students during the add/drop period for course enrollment. After receiving the message, the student is given a 24 hour window, after receiving the message, to register for the course. The 24 hour window gives any student ample time to register for the course if they still need the course. If the student does not register for the course within 24 hours, they will be removed from the wait-list and the next student will be notified. The process continues until the seat is filled. This will allow other students on the list to move up in the list. A student may also respond to the notification asking to be removed from the wait-list, which would also notify the next student. CORSICA would also notify students if the wait-list was closed.

 CORSICA will be provided over the Internet, using a Graphical User Interface (GUI) that is compatible with a variety of browsers, giving users the same abilities. Students will be limited to six course waitlists, preventing students from abusing this privilege. CORSICA’s website will allow students to view their position on the wait-list. They will be able to add and remove themselves from any waitlist, as long as they meet the course requirements. The easy to use GUI will allow students to easily be able to join and monitor their wait-lists.

 A key feature in CORSICA will be the ability to handle courses that have labs and/or recitations with a lecture. When a lecture with a lab and/or recitation reaches capacity, CORSICA will close all three registrations. Each lecture will be linked to the lab and/or recitation that it is associated with. The courses will be linked by their CRNs. If a student joins the wait-list for the lecture, they must also be on the wait-list for the lab and/or recitation that are linked to the course. The student will be given an error message and will not be added to the wait-list, unless they join all courses linked to the lecture. The student will then be placed on all wait-lists associated with the course.

 Another feature of CORSICA will be the ability to track statistics of courses. The statistics can be used to see if a course needs to have the capacity initially raised in the future due to an excess amount of students on the wait-list. CORSICA will monitor how many students join a wait-list. The reports will display how many students add and drop the course. CORSICA will record how long it takes to establish a wait-list, this will verify how wanted the course is for students.

 CORSICA also allows different levels of access for users roles. When a user logs in, they are recognized as an administrator, an advisor, a scheduler or a student. Administrators will be given full access over the system. The administrator’s main task is to oversee all the other users. They will be able to delete or add any user. CORSICA’s statistics can be used by schedulers to decide if a course’s capacity should be raised or decreased. Schedulers will have the ability to manually open and close courses. Advisors will be able to view a student’s position on a wait-list. The advisor will be able to alter the wait-list by changing a student’s position, adding students and removing them. Students will be able to view all the courses that have wait-lists. Courses will display how many students are on the wait-list. Students will be able to any view wait-lists they are on and their position on the list. CORSICA will also have a section for visitors. Visitors will not be able to log-in. They will able to view static pages, the CORSICA home page, an about section page, a contact page and the FAQS page.

## 2.2 Major Components (Hardware/Software)

CORSICA will require hardware and software components in order to complete the tasks it is designed to do. Figure 1 provides the three main hardware components needed in the implementation of CORSICA. 

Figure 1, Hardware Components

A computer with a web browser is required to access the Internet. Through the Internet, the user will access CORSICA’s servers. The server will access CORSICA’s web server, as well as its database server. Multiple servers can be used in order to better maintain the databases and also perform backups on all of CORSICA’s data. The web server will display web pages for the user to access from any computer with internet access. The database server will store all information that CORSICA gathers.

Software components will consist of database software that will maintain all information the CORSICA needs, such as course CRN, course capacity, student information, etc. The database will use the student’s information for its notification system. The database will be created using MySQL. PHP, SQL, javascript and HTML will be used to manipulate the database. The different users require development of different GUIs. HTML, CSS, and JavaScript will be used to maximize the efficiency of CORSICA’s GUI. Algorithms will be developed using C++ that will maintain wait-lists that have labs and/or recitations. Algorithms will also be implemented to identify open seats, loading enrollment data, increase course capacity, close/open course and add/drop students from a waitlist.

# **3** **IDENTIFICATION OF CASE STUDY**

 CORSICA will be developed for Old Dominion University’s Computer Science department. The development process in CS411W will be overseen by faculty mentor Dr. Irwin Levinstein. Dr. Levinstein, an Associate Professor, is the course scheduler for the CS department. Since Dr. Levinstein is the course scheduler, the reports and analysis feature of CORSICA would assist him in setting up the capacity size for CS courses for future semesters. Advisors and faculty will also benefit from the development of CORSICA. During registration periods when classes reach capacity, students will email the professor of the course and their advisor asking to be added to the course. Students would instead be able to just be able to join the wait-list and be able to monitor their position. CORSICA would save advisors and faculty time. They would no longer have to constantly e-mail students who are trying to enroll in a course that has reached capacity.

 The Computer Science department is not the only department that faces these issues. The successful development of CORSICA could be adapted to maintain any department at Old Dominion University. CORSICA also has the potential to assist other universities.

# **4** **CORSICA PRODUCT PROTOTYPE DESCRIPTION**

CORSICA will start out as a prototype. The prototype will not have all the capabilities the real world product would. CORSICA will work with Banner’s database. Since access to Banner is restricted, the student and course information will be simulated. The prototype will show the capabilities and features that the users will have access to. The users that will be implemented in the prototype are the administrator, advisor, scheduler, student and visitor.

A demonstration for CORSICA’s prototype will show its abilities to operate as an automated system and the effectiveness of the notification system. The prototype will also exhibit the fair enrollment process and the alert process if an error occurs. CORSICA will be setup as an automated system that can be overridden by certain users. Developing CORSICA to be an automated system will prevent administration from having to constantly maintain it. CORSICA will lock a course when the class reaches capacity, preventing students to enroll unless they are on the wait-list. When the course is locked, students can click a link to direct them to CORSICA’s page. This will ensure the fairness of CORSICA’s process. Alert triggers will use pop up text boxes in order to notify the user if the changes they are making are correct. Table 1 shows features that the prototype and real world product will have.

|  |  |  |
| --- | --- | --- |
|  | **Real World Product** | **Prototype** |
| **Environments for all Users:** | **Yes** | **No****· Will demonstrate student, admin, and scheduler users** |
| **Notification System** | **Yes** | **No****· Will be simulated with text box** |
| **Check for available seats** | **Yes** | **Yes** |
| **Add Student to Wait-list** | **Yes** | **Yes** |
| **Drop Student from Wait-list** | **Yes** | **Yes** |
| **Fair process** | **Yes** | **Yes** |
| **Alert System** | **Yes** | **No****· Will be simulated with text box** |
| **Mostly automated** | **Yes** | **No****· Will rely heavily on user interaction** |
| **Link to Banner** | **Yes** | **No****· Will be loaded with data.txt files instead** |
| **Link to Leo-Online** | **Yes** | **No****· Will be simulated with command box menu** |
| **GUI** | **Yes** | **Very Basic (Text System)** |
| **Seat Analysis System** | **Yes** | **No** |

Table 1

Real World Product and Prototype Comparison

## 4.1 Prototype Architecture (Hardware/Software)

The CORSICA prototype will consist of four major components: course database, notification system, user interface, and back-end algorithms. Figure 2 shows the process of these four components.

 

Test Harness

Figure 2

Prototype Major Functional Components Diagram

Users will use basic commands in the user interface to interact with the prototype. All data for the database will be text files in the prototype. Simulated data will consist of course sections, course name, course capacity, available seats and number of students enrolled in course. These simulated text files are enroll files provided by Dr. Levinstein. CORSICA’s prototype will simulate the notification process instead of using ODU’s current notification system. The notification process will access the student’s phone number and/or e-mail to send a message to the student. Back-end algorithms for the prototype include loading enrollment data, checking for available seats, adding/dropping students to a wait-list, test harness, changing course capacity, and opening/ closing courses. Table 2 provides the functionality of the algorithms.

|  |  |
| --- | --- |
| **Algorithm** | **How it functions** |
| **Load Enrollment Data Files** | **Course data files are loaded into CORSICA.****Files contain course: Capacity, Number of Enrollments, and Available seats.** |
| **Open Course** | **An Administrator or Scheduler user logs into Banner and opens a course for students to enroll in.****Banner database is updated****CORSICA database is notified of change and is updated** |
| **Check for Open Seats** | **Once a course becomes full, a wait-list is activated for it by CORSICA****CORSICA will continually reference the current course capacity and amount of students enrolled.****If the amount of students enrolled is less than course capacity, a seat has become available.****CORSICA database updates****Calls notification algorithm** |
| **Add Student to Wait-list** | **Student X wishes to enroll in Course Y’s wait-list****CORSICA receives this request and adds Students X to wait-list queue****Course Y’s wait-list is updated** |
| **Notification** | **The check for open seats algorithm completes and returns true for an available seat****All students on the wait-list queue are notified of opening****Students respond** |
| **Drop Student from Wait-list** | **Student X wishes to be dropped from Course Y’s wait-list or the time window for that student has expired****CORSICA receives this request and removes Student X from the wait-list queue****Course Y’s wait-list is updated** |
| **Change Course Capacity** | **Administrator logs into Banner and increases course capacity for Course Y****Banner database is updated****CORSICA database is notified of change and is updated** |
| **Close Course** | **An Administrator or Scheduler user logs into Banner and closes a course as an available option for students to enroll in****Banner database is updated****CORSICA database is notified of change is updated** |

Table 2

Description of Algorithms

## 4.2 Prototype Features and Capabilities

CORSICA’s prototype will demonstrate the products capabilities. The users included in the prototype are the administrator, advisor, scheduler, student and visitor. The prototype will provide example uses of the different types of users.

 The administrator has full control over CORSICA and its users. When an administrator logs into the system, all the courses for the term and their wait-lists are displayed. The administrator has the ability to search for any student using their name or University Identification Number (UIN). Searching for a student will pull up courses they are enrolled in and their wait-list history. The administrator can also search any course by its name or section number. The search would display the course information, number of students enrolled in the course, seats available, and number of students on the wait-list. The administrator could then look through the wait-list and make alterations if needed. The administrator could change a student’s position, add/remove them for the list, or close the wait-list. The administrator’s main purpose is overseeing the other users. Administrators are able to add, delete and change the capabilities of other users.

 An advisor’s task is to assist students enrolling in the courses. They will be able filter courses and wait-list data as well as student information in CORSICA. They have the ability to search for any course or any student. While they cannot change the capabilities of other users, they would be able to change a course’s wait-list. The advisor would also be able to alter the wait-list like the administrator. The advisor is limited to only being to change courses, students, and a course’s wait-list.

 The scheduler would only need access to the courses and their wait-lists. A scheduler could sort courses that have wait-lists and courses without wait-lists. Previous course wait-lists would also be provided for the scheduler. A scheduler could search a course and view a wait-list from a previous semester. The scheduler could also look up the statistics and analytical data for any course. If a course needs to be altered, the scheduler could search the course then easily add, delete or alter the capacity size.

 A student logging in would be able to view all the courses they enrolled in and any wait-list they are currently on. The student would be able to search a course by the course name, course section or the instructor’s last name. If the course is full, they would be able to join the wait-list. CORSICA would use an alert box to inform the student of the course name, semester and the position the student would be on the wait-list. If the user confirms that they want to be on the wait-list, they will be added to the list.

 The visitor will not be permitted access to the prototype, because they are an outside user. They will only have access to examples of the site as well as contact information and FAQs section. CORSICA’s prototype will give potential users the chance to view the different user capabilities. The prototype could be the deciding factor for potential customers as to whether or not the product would be able to solve their issues. Table 3 provides a list of capabilities for each user role.



Table 3

User Access Table

## 4.3 Prototype Development Challenges

 CORSICA will also face challenges in the development of the prototype. The prototype will be limited to its capabilities and features. The challenges CORSICA faces will be dealt with in a variety of ways. Table 3 provides a list of challenges and solutions.

|  |  |  |
| --- | --- | --- |
| **Objectives** | **Prototype** | **Challenges** |
| **Environments for all Users:** | **No****· Will demonstrate student, admin, and scheduler users** | **Working out all the bugs in CORSICA to allow all users to use CORSICA as intended** |
| **Notification System** | **No****· Will be simulated with text box** | **Allowing CORSICA to sync EXACTLY with the University’s Clock** |
| **Alert System** | **No****· Will be simulated with text box** | **Making the Alert System actually recognize each change to help ensure intentional changes** |
| **Mostly automated** | **No****· Will rely heavily on user interaction** | **Users need to be knowledgeable of CORSICA** |
| **Link to Banner** | **No****· Will be loaded with data.txt files instead** | **Using Black Box Testing to certify the text file compatibility** |
| **Link to LEO Online** | **No****· Will be simulated with command box menu** | **Maintaining LEO Online’s layout while appending the CORSICA option on the course registration screen** |
| **GUI** | **Very Basic (Text System)** | **Coding a GUI that looks professional and is simple to navigate** |

Table 4
Prototype Challenges

Glossary

**\*Algorithm -** A set of steps that are followed in order to solve a mathematical problem or to complete a computer process.

**\*\*\*Banner -** Old Dominion University's centralized academic and administrative records system.

**\*Browser -** A computer program that is used to find and look at information on the Internet.

**\*\*C++ -** A general purpose programming language that is free-form and compiled.

**\*\*Cascading Style Sheets (CSS) -** A style sheet language used for describing the look and formatting of a document written in a markup language.

**\*Computer -** An electronic machine that can store and work with large amounts of information.

**\*Database -** A collection of pieces of information that is organized and used on a computer.

**\*E-mail -** A system for sending messages from one computer to another computer.

**\*Graphical User Interface (GUI) -** A program that allows a person to work easily with a computer by using a mouse to point to small pictures and other elements on the screen.

**HyperText Markup Language (HTML) -** A computer language that is used to create documents or Web sites on the Internet.

**\*Internet -** An electronic communications network that connects computer networks and organizational computer facilities around the world.

**\*\*Javascript -** A dynamic computer programming language, used as part of web browsers, whose implementations allow client-side scripts to interact with the user.

**\*Laboratory -** A room or building with special equipment for doing scientific experiments and tests.

**\*Lecture -** A talk or speech given to a group of people to teach them about a particular subject.

**\*\*MySQL -** A database management system.

**\*Notification -** The act of notifying someone.

**\*ODU** - Old Dominion University, a public 4-year university in Norfolk, Virginia.

**\*\*PHP -** A server-side scripting language designed for web development.

**\*Prototype -**  An original or first model of something from which other forms are copied or developed.

**\*Recitation -** A class period especially in association with and for review of a lecture.

**\*Server -** The main computer in a network which provides files and services that are used by the other computers.

**\*\*SQL -** A programming language designed for managing data held in a relational database management system.

**\*Text Message -** A short message that is sent electronically to a cell phone or other device.

**\*\*\*University Identification Number (UIN) -** A unique identification number given out to students at Old Dominion University.

**\*Wait-list -** To be put on a waiting list.

**\***Found at http://www.merriam-webster.com/

**\*\*** Found at <http://en.wikipedia.org/wiki/>

**\*\*\*** Found at https://www.odu.edu

**References**

National Center for Education Statistics, U.S. Department of Education. (2001). Postsecondary Education. In *Digest of* education statistics 2001 (chap. 3). Retrieved September 6, 2014, from http://nces.ed.gov/programs/digest/d12/ch\_3.asp.

National Center for Education Statistics, U.S. Department of Education. (2001). Table 5: Number of education Institutions, by level and control of institution: Selected years, 1980-81 through 2010-11. In *Digest of* education statistics 2001 (chap. 3). Retrieved September 6, 2014, from http://nces.ed.gov/programs/digest/d12/tables/dt12\_005.asp.

National Center for Education Statistics, U.S. Department of Education. (2013). *Digest of      Education Statistics: 2012*. Retrieved September 18, 2014.