Lab 2 - CORSICA Product Specification

Red Team

Latimer Gerle

CS411W

Professor Janet Brunelle

October 5, 2014

Version 1

Table of Contents

1 Introduction 2

1.1 Purpose 3

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 7

1.4 References 9

1.5 Overview 9

2 Product Description 10

2.1 Prototype Architecture Description 10

2.2 Prototype Functional Description 13

2.3 External Interfaces 13

List of Figures

*Figure 1*. Current Registration Process. [5](#Figure1)

*Figure 2*. Improved Registration Process Utilizing CORSICA. [6](#Figure2)

*Figure 3*. CORSICA Major Components. [11](#Figure3)

List of Tables

*Table 1*. Comparison of CORSICA Real World Product to Prototype. [12](#Table1)

# Introduction

At the start of each academic term at institutions of higher learning across the country, students attempt to register for courses that are already filled to capacity. This causes headaches for both students and faculty; the students pester the instructors and advisors to be allowed into closed classes, and the faculty is barraged by emails, phone calls, and visitations from worried students. This cycle, which continues until the students are either let into their desired class or give up, leads to wasted time and frustration for both parties. And as the number of people seeking bachelor’s degrees continues to increase, the aforementioned situation will occur more often. Between 2001 and 2011 alone, the number of students enrolled in post-secondary institutions increased by thirty-two percent (“Postsecondary Education”, 2012). Meanwhile, the number of spots available for enrollment is not keeping pace, with the number of new post-secondary institutions increasing by about ten percent during the same time period (National Center for Education Statistics, Table 5).

Waitlists are one solution for decreasing the frustration and excessive communication that results from students wishing to register for closed courses. This is an option that numerous universities offer, and while the details differ depending on the specific institution, the basic implementation is the same; if a class is full, a student can place him or herself on the waitlist, and he or she will be notified when a seat becomes available. However, this is not a perfect solution for those students wishing to register for a closed course. First of all, these waitlists often have certain restrictions; for example, a student is required to have a certain number of credits. Secondly, not all universities have course waitlists, and those that do typically offer them only within certain departments or courses. And finally, the waitlists lack basic features, such as the ability to view your position or be notified via text.

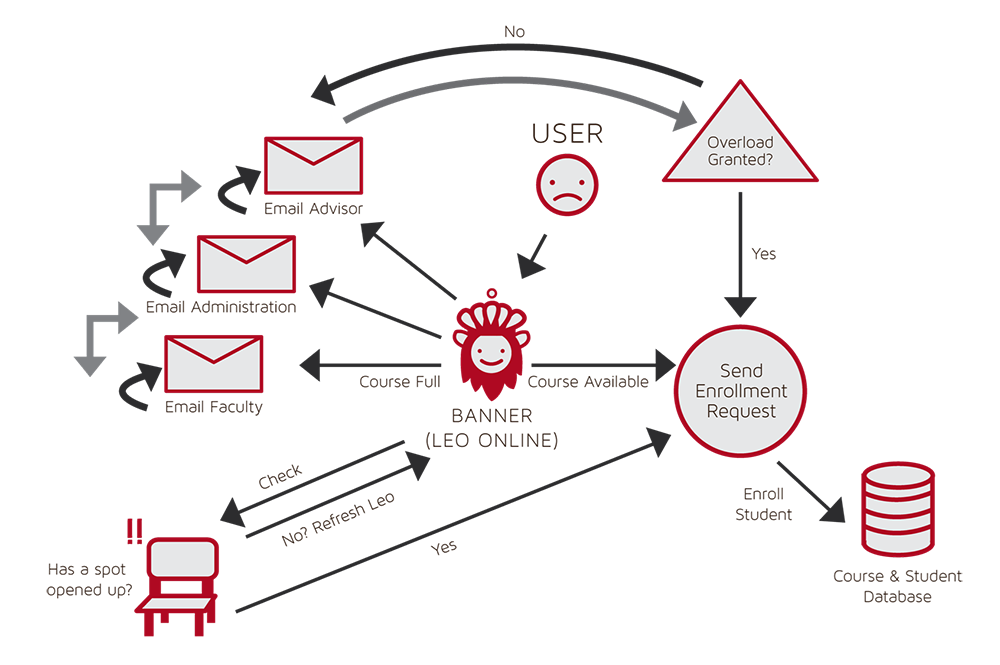
## 1.1 Purpose

CORSICA, the COmputeR ScIence Course wAitlist, was planned to be an enhanced and more efficient waitlist than the current waitlist available via Old Dominion University’s LEO Online; the existing waitlist is inadequate because it does not have the ability to handle Computer Science courses, which differ from most courses in that they include lab and/or recitation sections. The hallmarks of CORSICA, a graphical user interface (GUI), ability to handle labs and recitations, and improved notification system, combine to make a simple and streamlined process for both students and instructors.

## 1.2 Scope

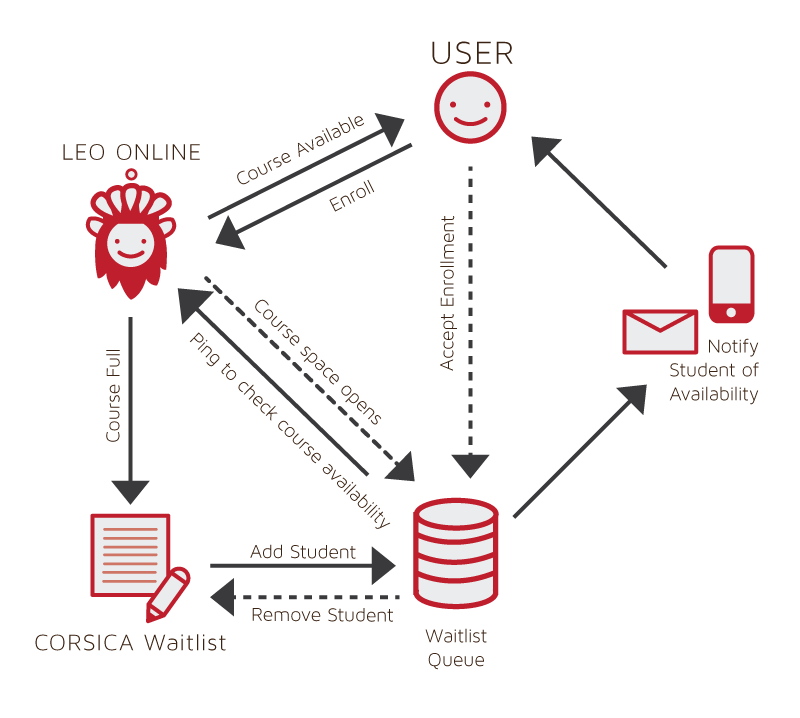
CORSICA was designed for ODU’s Computer Science department and, specifically, Computer Science courses, which differ from the majority of other undergraduate courses. However, if CORSICA is a successful and beneficial product, it could possibly be used by numerous post-secondary institutions.

Professor Janet Brunelle, the Assistant Chair and Chief Advisor of the ODU Computer Science department, initially suggested the idea for CORSICA, and recommended developing it in conjunction with Dr. Irwin Levinstein. Dr. Levinstein is an Associate Professor in Computer Science at ODU, as well as the course scheduler for the department. Professor Brunelle explained that the current registration process is quite flawed. The Computer Science courses typically fill up quickly; therefore, any student who is delayed in registering may find him or herself unable to sign up for all of his or her desired classes. This in turn leads to the students consistently checking the course enrollment figures and emailing advisors and faculty with requests to be allowed into the class or classes; the advisors and faculty must then take time away from other tasks to respond to these requests. This time-consuming cycle, illustrated in Figure 1, repeats itself until the registration period is over.



*Figure 1*. Current Registration Process

In order to resolve this issue, Professor Brunelle recommended developing a waitlist for ODU Computer Science courses. As shown in Figure 2, CORISCA would stop students from continually checking enrollment numbers, and it would also cease the constant emails to advisors and faculty requesting course overrides. Thus, the cycle from Figure 1 would transform into a simplified and less stressful process. In addition, the analysis component of CORSICA will be of assistance to Dr. Levinstein in that it will determine how many seats should be added or removed from future Computer Science courses. Upon successful testing and demonstration, CORSICA will be available for use by any post-secondary institution wishing to utilize it.



*Figure 2*. Improved Registration Process Utilizing CORSICA

The CORSICA prototype is designed to demonstrate the concept that its improved and efficient waitlist system will save both time and frustration for students and faculty. The prototype will accomplish this by showing that the CORSICA waitlist process will render it unnecessary for students to stay in constant communication with faculty and advisors when they wish to register for a closed course. Thus, the time that would be spent on these communications will now be free for other purposes.

One objective of the prototype is to successfully demonstrate that students will be notified when a registration spot opens in a class. However, the prototype will not display the end product notification system; rather, it will be simulated by integrating with that of ODU. The ODU notification system works by alerting students of emergencies and school closures via text and email.

Another objective of the CORSICA prototype is to exhibit the waitlist functionality. This will be achieved by filling the database with data files containing mock student information. Once the database is populated, the prototype will demonstrate a simplified version of CORISCA’s capabilities in a rudimentary version of the real-world GUI. However, functionality will only be available for Student, Scheduler, and Administrator users. This will allow the basic functions of the waitlist to be displayed, such as a student adding or dropping herself from a waitlist, or an administrator changing course information.

## 1.3 Definitions, Acronyms, and Abbreviations

**Algorithm**:A set of instructions that is carried out in order to solve a problem; typically used in the context of Computer Science.

**Banner**: The centralized academic and administrative records system used by Old Dominion University.

**C++**:An object-oriented computer programming language.

**CSS**:Cascading Style Sheets. A style sheet language used for formatting a document; typically used in conjunction with HTML for websites.

**Computer**: An electronic device that can store information and carry out programmed instructions.

**CORSICA**: Computer Science Class Waitlist. A piece of software developed for the Old Dominion University Computer Science department.

**Database**:A collection of (usually related) information.

**Email**: A method of sending messages from one computer to another on a network.

**GUI**: Graphical User Interface. A program that allows a user to interact with a computer by means of graphical icons such as buttons and text boxes.

**HTML**: Hypertext Markup Language. A markup language that is used for creating websites.

**Internet**: A worldwide system of interconnected computer networks.

**JavaScript**:A dynamic computer programming language, typically used in web browsers.

**Lab**:A section of a college class that commonly involves hands-on work.

**Lecture**: An oral presentation given by a college professor within a university setting.

**LEO Online**: Old Dominion University’s online student and employee information portal.

**MySQL**:A relational database management system.

**Notification**:The act of informing someone of something.

**ODU**: Old Dominion University. A public post-secondary institution in Norfolk, Virginia.

**PHP**:A scripting language commonly used for web development.

**Prototype**: A preliminary model of something.

**Recitation**: A college class section where small groups of students from the corresponding lecture meet to review weekly material.

**Server**: A computer that provides information and services to other computers.

**SQL**:Structured Query Language.A programming language designed for managing data held in a relational database management system.

**Text Message**: An electronic message that is sent between cell phones.

**UIN**:University Identification Number. A unique identification number provided to students at Old Dominion University.

**Waitlist**: A list of people who are waiting for something.

## 1.4 References

Lab 1 - CORSICA Product Description. Version 1.2. (9 October 2014). CORSICA. Red Team. CS411W: Latimer Gerle.

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.

## 1.5 Overview

This product specification describes the goals, capabilities, features and interfaces of CORSICA and its prototype. The remainder of this document contains a high-level description of the CORSICA prototype, as well as in depth explanations of the prototype architecture and functionality. Also included are descriptions of the hardware, software, and user interfaces associated with the prototype.

(This space intentionally left blank.)

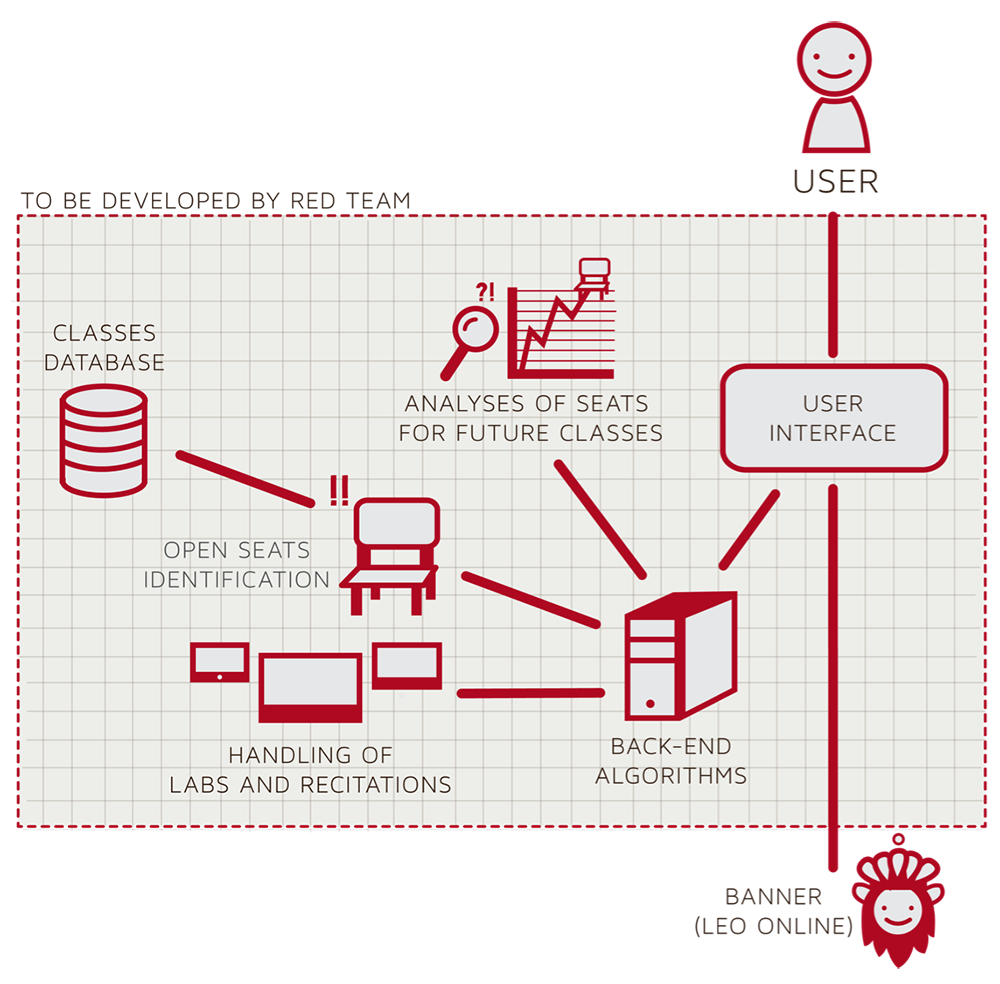
# 2 Product Description

CORSICA is a course waitlist system, designed for ODU Computer Science courses, through which students can wait for an enrollment spot to become available in a full course. Each student on a CORSICA waitlist is emailed his or her current position number every twenty-four hours. When a spot becomes available in the corresponding class, the first student on the waitlist received a notification email, and he or she is allowed to register for the course if they still wish to do so. CORSICA is utilized by way of a GUI, through which students can add themselves to the waitlist for a particular course if it has reached the total enrollment capacity. Via the GUI, students can also remove themselves from a waitlist, as well as view their current position on the list.

## 2.1 Prototype Architecture Description

CORSICA is a software service, with both the prototype and real world product comprised of four main components, as illustrated in Figure 3: a database, various back-end algorithms, a GUI, and a notification system. The database is a vital part of the system; without it, there is no information regarding the waitlist. It will utilize MySQL, and will contain all of the information relevant to CORSICA, such as administrator names, CRNs, and UINs. The GUI, which would be driven by the algorithms, is where users would perform different actions based upon their login credentials. And the notification system, also driven by the algorithms, would alert students of an opening in a class.

(This space intentionally left blank.)



*Figure 3.* CORSICA Major Components

The CORSICA prototype architecture will be nearly the same as the real-world product; the differences are explained in Table 1. The prototype consists of a database, algorithms, GUI, and notification system. The database will be created with MySQL, and queries will be written using SQL. These queries will add information to, delete information from, and search the database. Connection to the database will be established using both PHP and Javascript. Multiple algorithms will be required for the prototype, which are as follows: Increase Course Capacity, Close Course, Notification, Add Student to Waitlist, Drop Student from Waitlist, Check for Open Seats, and Load Enrollment Data Files. These algorithms will be coded using C++, as will the notification system. And finally, the GUI will be coded using HTML, CSS, and Javascript.

|  |  |  |
| --- | --- | --- |
|  | 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 Waitlist | Yes | Yes |
| Drop Student from Waitlist | 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*. Comparison of CORSICA Real World Product to Prototype

## 2.2 Prototype Functional Description

The CORSICA prototype will first and foremost exhibit basic waitlist functionality. Students will be able to add or drop themselves from course waitlists, as well as view their positions on whatever waitlists they are currently on.

Another feature the CORSICA prototype will display is a simplified version of its notification system. Even though the prototype system will be basic, however, it will still achieve the same goal as the real world system; that is, when a registration spot opens up in a previously closed course, all students currently on the waitlist for that course will be notified. This notification will be carried out through email, text, or both.

A final feature the prototype will demonstrate is its automation. The majority of CORSICA will be automated, making for an efficient and easy to operate system. User actions will trigger the back-end algorithms, which will in turn power the automation process.

## 2.3 External Interfaces

Following are the hardware, software, and user interfaces that will be utilized by and within CORSICA.

## 2.3.1 Hardware Interfaces

No hardware interfaces will be implemented by CORSICA. However, in order to function, CORSICA will require a computer and Internet connection, as well as access to the ODU network.

## 2.3.2 Software Interfaces

While CORSICA will utilize pre-existing computer programming languages for its algorithms, it will create a customized database. The database will be created using MySQL, with queries being written in SQL.

## 2.3.3 User Interfaces

CORSICA will provide users with a single interface, that being the GUI, which can be accessed through ODU’s LEO Online system. Via the GUI, students and administrators can log on to CORSICA and utilize its capabilities.