**CORSICA Product Description**

**Anthony Baron**

**Janet Brunelle**

**Old Dominion University**

**May 8, 2014**

**CS 411 Lab 1 Version 2**

#

Table of Contents

[**CS 411W LAB I – PRODUCT DOCUMENT** 3](#_Toc400006515)

[**1.** **INTRODUCTION** 3](#_Toc400006516)

[**2** **CORSICA PRODUCT DESCRIPTION** 4](#_Toc400006517)

[2.1 Key Product Features and Capabilities 5](#_Toc400006518)

[2.2 Major Components (Hardware/Software) 6](#_Toc400006519)

[**3** **IDENTIFICATION OF CASE STUDY** 8](#_Toc400006520)

[**4** **CORSICA PRODUCT PROTOTYPE DESCRIPTION** 8](#_Toc400006521)

[4.1 Prototype Architecture (Hardware/Software) 10](#_Toc400006522)

[4.2 Prototype Features and Capabilities 14](#_Toc400006523)

[4.3 Prototype Development Challenges 15](#_Toc400006524)

[Glossary 17](#_Toc400006525)

[References 18](#_Toc400006526)

List of Tables

Table 1 – User Abilities 5

Table 2 - Real World Product and Prototype Comparison 10

Table 3 - Description of Algorithms 13

Table 4 - Prototype Challenges 16

List of Figures

Figure 1 – Hardware Components 7

Figure 2 - Prototype Major Functional Components Diagram 11

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

# **INTRODUCTION**

Earning a college degree has become a necessity to gain entry into the professional workforce. Individuals who have forms of higher education earn larger incomes and have a lower unemployment rate compared to those who have no higher education (Bureau of Labor Statistics 2014). It is expected that between 2011 and 2021, college enrollment will increase by 13 percent (U.S. Department of Education 2013). With this rise in attendance, universities are struggling to establish appropriate course capacities.

Old Dominion University (ODU) is currently one of those universities having issues with deciding course capacity limits. Today, students who wish to enroll in a course will log into LEO online, ODU’s course enrollment system, and enroll in a course through that interface. Often, enough spaces in a course are not allocated to meet the high demand of student enrollment needs. While the enrollment process is seamless when course capacities are not met, ODU’s current enrollment system lacks an efficient wait-list process. When a course’s capacity is reached, there is no method in place to facilitate notifications of newly available seats. Due to this inefficiency, problems arise for many individuals at the university: the student, faculty, advisors, and schedulers. When a course reaches max capacity, students communicate with faculty and their advisors in attempts have it be overloaded so they may enroll. In addition, students will spend time monitoring LEO to find an opportunity to enroll in a course that has reached capacity.

Due to limited resources and faculty, a wait-list system must be implemented in order to streamline the enrollment process. The COmputeR ScIenCe wAit-list (CORSICA) system will be prototyped for the Computer Science (CS) department to demonstrate its benefit to ODU.

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

 CORSICA is the solution to ODU’s Computer Science Department’s problem of not having a simple, fair, and efficient wait-list process. The program will be available to a set of registered users: Advisors, Schedulers, and Students. It will be managed by Administrative users and allow for Visitor users to interact with the Graphical User Interface (GUI) without sign-in credentials. Users will access the program’s features through LEO, which is the current enrollment system provided by ODU via the Internet. CORSICA will provide notifications to students, informing them of an opportunity to enroll in a course as well as act as a monitoring system should the student wish to view the wait-list status.

 Once a course reaches capacity, it will be managed by the wait-list system. CORSICA will act as a notification system for students until the current semesters add/drop period has ended. When CORSICA is engaged, a student will locate the course they wish to be enrolled in and opt to be added to the course’s wait-list. Upon initial sign up, the student will have the opportunity to provide their cell phone number and opt for text alerts in addition to email notifications. After being placed on the wait-list, the amount of attention the student needs to place on monitoring LEO for an enrollment opportunity in a full course is lessened.

## 2.1 Key Product Features and Capabilities

 CORSICA was designed to provide the CS department with a fair, simple, and efficient wait-list process. Once a course reaches capacity, three procedures will be followed. One, the system blocks LEO online from allowing students to enroll for a course that is now maintained by the wait-list. This ensures that students on the wait-list are the only individuals who have the opportunity to enroll. Two, once students begin to enroll on the wait-list the system will maintain the order in which sign ups have occurred. This helps keep order and ensure a first come first serve method of operation. Three, the system will ensure that a student may sign up for no more than a max cap (user specified) of wait-lists (including different sections of the same course). This helps inhibit student abuse of the system. These procedures ensure a fair process. CORSICA is a simple to use wait-list process for students and its other levels of users. Table 1 details the different user abilities that ensure that CORSICA is a simple and user friendly application.

Table – User Abilities

When the daily enrollment file is loaded into CORSICA, an email will be sent to all students on the wait-list stating that they are position ‘X’ on the wait-list. When an opportunity to enroll becomes available, the first student on the wait-list will receive a notification that states that they are able to enroll. It will also prompt for a yes or no response to the question, “Do you wish to enroll in the course?” If they respond no, they are removed from the wait-list and the notification algorithm is called again. If they respond yes, the student follows through with the enrollment process.

 Courses that have lab or recitation sections are managed by the wait-list system as well. If a lecture section becomes full, all relative labs and/or recitations are treated as being full. The same is true if either the lab or recitation became full first. This helps to ensure that students are able to enroll in all needed sections.

## 2.2 Major Components (Hardware/Software)

 Figure 1 shows how CORSICA will be comprised of three major hardware components: a computer, an Internet connection, and link to a web / database server. The computer and the Internet connection work together in allowing the user to interact with the program. The web server allows the computer to display the web pages, which displays the program’s GUI. The database server allows the CORSICA database (CDB) to provide wait-list data to the rest of the program.

The wait-list system will be comprised of software components: the CDB, GUIs, and back-end algorithms. The CDB will be comprised of user, student, course, wait-list, and course statistic data tables. All tables will hold important data that enable CORSICA to be an efficient wait-list system.

Figure – Hardware Components

 User tables hold all information pertaining to the user categories: administrator, advisor, faculty, student, and visitor. Each category holds privileges and a log in must be required in order to distinguish users. Student tables will contain all relevant information about a student. The program will need to know the student’s name, University Identification Number (UIN), email address, and if provided, their cell phone number. Course tables contain all relevant course information: course prefix, name, course registration number (CRN), meeting times, capacity, amount of enrolled students, and if it has a lab / recitation section. The wait-list tables will only store data about a course’s wait-list and the students on them. Course statistic data tables keep track of how many courses had wait-lists, the number of students on them, and which courses did not. This data is valuable to schedulers as it will assist in decisions regarding course structure (size, number of sections, etc.) in the future.

 CORSICA will have multiple GUIs. They will be created using various scripting languages: HTML, CSS, and JavaScript. Each user group will have their own unique version of the GUI. The purpose of this is to separate user capabilities. Every user has different commands they are able to execute (see Table 1). Back-end algorithms will be a major software component in CORSICA. The algorithms will include load enrollment data files, add student to wait-list, drop student from wait-list, check for open seats, notification, increase course capacity, open course, close course, and linked CRNs.

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

 A development team is implementing CORSICA for ODU’s CS Department. The team is under the mentorship Dr. Irwin Levinstein, who is an Associate Professor at ODU in the CS department. He is also the course scheduler for the CS department. The wait-list program will be tested by the CS department, and statistical data of its operation will be gathered. This data will assist course schedulers in preparing upcoming semester course lists. Pending a successful prototype, CORSICA could be expanded to ODU’s other departments and to other universities.

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

 Capabilities will be reduced in the CORSICA prototype in order to provide proof of concept. The Banner database of students and courses will be simulated with false data files in the prototype as access will have not yet been granted to the development team. The Seat Analysis System will be omitted in the prototype as it is not essential functionality. The prototype will show how CORSICA operates for users: Administrator, Advisor, Scheduler, Student, and Visitor.

 The CORSICA prototype will demonstrate the following functional goals and objectives: operate as a mainly automated system, provide an efficient notification systems, ensure a fair enrollment process, and trigger alerts. CORSICA will operate as an automated system while still accepting user requests / actions. Having a mainly automated system ensures ease of use for all users. In an effort to ensure fairness during course enrollment periods, CORSICA will reserve seats for student users on a course wait-list and prohibit sign-up attempts from students who are not. Finally, alerts will be triggered, by means of a pop up textbox, when an Administrator user of Scheduler changes course information to ensure all changes are intentional. Table 2 shows a comparison between the real world functional product and the prototype.

|  |  |  |
| --- | --- | --- |
|   | 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 2 - Real World Product and Prototype Comparison

## 4.1 Prototype Architecture (Hardware/Software)

 Figure 2 shows that the CORSICA prototype will be comprised of five major

Components: a course database, back-end algorithms, a notification system, a user interface, and a test harness. A user will interact with the prototype through the command prompt using basic I/O. The course database will be simulated with a text file. The file will contain all the information the prototype will require: Course name, Course sections, Course capacity, Number of students enrolled in the course, and the amount of available seats. The data will be arbitrarily generated as CORSICA will not be granted access to real data during the prototype demonstration.



Figure - Prototype Major Functional Components Diagram

 Back-end algorithms will be developed for the prototype. They include the following: Load Enrollment Data File, Open Course, Add Student to Wait-list, Check for Open Seats, Notification, Increase Course Capacity, Drop Student from Wait-list, and Close Course. Table 2 provides explanations as to how the algorithms will function.

|  |  |
| --- | --- |
| 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 queueCourse Y’s wait-list is updated  |
|   Notification | The check for open seats algorithm completes and returns true for an available seatAll 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  |
|  Increase 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 |
| Linked CRN | This algorithm will check to see if a course has any linked CRNs. This will inform CORSICA if a lecture section has a lab or recitation to go along with it. If it does, then if one or the other becomes full, both will be treated as a wait-list course. |

Table 3 - Description of Algorithms

 The prototype will not be created with a fully operational notification system. Instead, it will simulate the notification process with a prompt, alerting the user that students have been notified.

## 4.2 Prototype Features and Capabilities

 The purpose of the CORSICA prototype is to demonstrate functionality and proof of concept. The prototype will show how three of the five user roles may interact with the system. It will be functional for the Administrator, the Scheduler, and the Student users. A visitor user will not prove functionality of the system, and an Advisor user has many of the same abilities as a Scheduler, so they are being left out. Each of the three included user roles will be described in Table 4.

|  |  |  |  |
| --- | --- | --- | --- |
| Prototype Capabilities | Administrator | Scheduler | Student |
| Home page will be a list of courses and their wait-lists. | Yes | Yes | No |
| Home page will be a list of wait-lists they are enrolled in. | No | No | Yes |
| Home page will be a list of wait-lists they are enrolled in. | No | No | Yes |
| Will see a course name, CRN, instructor, and wait-list status and an “Add Course” button on the bottom of their screen. | Yes | Yes | No |
| May query a student and view what course wait-lists they are enrolled in.  | Yes | Yes | No |
| May query a course to show: course name, CRN, instructor, wait-list status, and what students are currently enrolled on the wait-list. | Yes | Yes | No |
| May query for a particular instructor and view what courses they are teaching are currently wait-list options. | Yes | Yes | Yes |
| May choose a course section and enroll in the wait-list (alert will be triggered stating what wait-list user is enrolling in as well as their position). | No | No | Yes |

Table 4 - User Role Descriptions

## 4.3 Prototype Development Challenges

 In software development, various developmental challenges occur. A prototype will often not obtain the full functionality of a real-world product. The goal of a prototype is to demonstrate proof of concept. This is why the CORSICA prototype focuses on “needs” of the program rather than the “wants”. It was developed to be a functional program but has had some features excluded. Table 5 details what challenges the prototype has faced and how they are handled.

|  |  |  |
| --- | --- | --- |
| 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’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 5 - 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.

**\*CORSICA Database (CDB)** – A collection of information required by CORSICA to be functional.

**\*Course Registration Number (CRN)**- A unique number given to each course section for identification purposes.

**\*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.

# References

1. (December, 2013). Digest of Education Statistics: 2012. Retrieved September 17, 2014, from <http://nces.ed.gov/programs/digest/d12/ch_3.asp>
2. (March 24, 2014). Employment Projections.  *Bureau of Labor Statistics.* Retrieved May 1, 2014, from <http://www.bls.gov/emp/ep_chart_001.htm>
3. Yu, R. (2012, September 3). Voice mail in decline with rise of text, loss of patience. *USATODAY.COM*. Retrieved April 27, 2014, from <http://usatoday30.usatoday.com/tech/news/story/2012-09-03/voicemail-decline/57556358/1>

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

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

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