CORSICA Product Description
Red Team
Old Dominion University
May 6, 2014
CS 410 Lab 1

Table of Contents

Figures

CS 411W LAB I - PRODUCT DESCRIPTION DOCUMENT

1   INTRODUCTION (PATRICK)

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 made more,

but had a lower unemployment rate (2014). This has caused higher education to be more desirable.

With the popularity of Universities increasing, so has the number of students that attend them.

Universities, such as Old Dominion University (ODU), have the issue of courses reaching capacity due

to their student population. This has led to the problem of the current class enrollment system becoming

insufficient for handling course overflows.

The current enrollment system used by ODU is Banner. While most of the

management system is good, it lacks an efficient wait-list system. The wait-list system causes many

problems for not only ODU's Administration but also the students, faculty, advisors, and schedulers.

The current course wait-list system results in excess student communication with faculty and

Administration in order to facilitate the student's desire to be enrolled in a specific course when it is full.

While the student and faculty body are attempting to resolve this issue, they are taken away from their

other tasks. Implementing an improved wait-list feature to the current working course add/drop system

will greatly reduce the number of these avoidable communications.

Understanding the issue, 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 put on an improved wait-list which they can monitor its status and get notifications. This will increase the rate at which students can join a class if a space becomes available. The prototype is only for the CS department, however, it can be adopted to any department.

2   CORSICA PRODUCT DESCRIPTION (NICK)

CORSICA is a wait-list management system that will provide students a simple, fair, and effective way to wait for an available seat in a course. CORSICA will be available to Administrators, Advisors, Schedulers, Students, and Non-Logged in users as a service provided over the internet. CORSICA will send a notification to students on the wait-list when a seat becomes available for a course. This will reduce the stress that students would otherwise face when waiting for an available seat. In order to receive a notification, a course must be full and students must place their name on the wait-list. Once a seat becomes available, by another student dropping the course or having a course's capacity raised, the system will send a notification to the first student on the wait-list. Notifications are sent via two ways, e-mail and/or text message. According to USA TODAY, the number of unattended voice-mails was down 8% from July 2011 to July 2012. Michael Tempora, Senior Vice President of Product Management at Vonage, stated "They hate the whole voice-mail introduction, prompts, having to listen to them in chronological order." The company has responded to the trend with a new voice-mail transcription service that converts voice messages for delivery as e-mail or text message (Yu, 2012). CORSICA will use email and/or text messages for the fastest and most efficient delivery.

2.1   Key Product Features and Capabilities (NICK)

CORSICA's main goal is to quickly and efficiently notify students when a seat for a course has become available. As students approach graduation, the importance to get into certain courses is very high and it is often these same courses that become full. Many students have to delay graduation simply because they were unable to get into a course or promptly get on the wait-list. With this scenario in mind, CORSICA has one objective: to provide students the opportunity to graduate on time.

One of the key features of CORSICA is the quick and efficient manner of the notification that students receive when a seat for a course becomes available. This notification can come in one or two ways, e-mail, and/or text message. After receiving this notification, students will have a 24 hour window to respond. This allows students enough time to respond if they still need a seat in that course. However, if they do not respond within 24 hours they will be removed from the wait-list, allowing the next student the opportunity to respond. The automated removal of students from the wait-list allows for the next student to be notified faster. If the Scheduler decides to close a wait-list, CORSICA will notify all the students on the wait-list that the wait-list has been closed. Another feature of CORSICA is that the Graphical User Interface (GUI) is handled over the internet. This allows users that have different web browsers the ability to use CORSICA. Other features include: the ability for students to sign up for as many wait-lists as they'd like, students can go onto the CORSICA website and view their position on a wait-list, and students can add or remove themselves from a wait-list. All of these features aid students in knowing when a seat for a course becomes available.

A major capability of CORSICA is to be able to handle courses that have lectures, labs and recitations. CORSICA will allow students to register for notifications when a seat for any lecture, lab

and recitation opens; or students can register for a specific Course Registration Number (CRN) for each lecture, lab and recitation. The student will be given an error if he or she tries to register for one part of a course's wait-list that offers three parts, which includes: lecture, lab and recitation. Therefore, CORSICA will make the student register for three wait-lists or none of them. Another capability that CORSICA offers is allowing different levels of access depending on the credentials entered at log-in. The different types of users offered are Administrators, Advisors, Schedulers, Students, and Non-Logged in. Administrators have full control over the system. This is important to have if a scheduler or advisor is using CORSICA in an unacceptable way, because the administrator can then delete one or both of them. Schedulers are important to CORSICA because they view course wait-list statistics for analytical purposes. Schedulers can view these statistics to assist them with knowing whether they should increase course capacity or decrease course capacity for a particular course. They can also manually open or close a wait-list for a course. Advisors have many different tasks including: the ability to view a student's position on a wait-list, the ability to add or remove a student on a wait-list, as well as the ability to move a student up or down on a wait-list. Students can view courses that have wait-lists, the number of students on the wait-list, as well as the wait-lists they are registered for. Non-Logged in users have limited functionality in CORSICA, unless they log-in. They will be able to view static pages, including: home page, about CORSICA, contact CORSICA, and CORSICA faqs.

2.2   Major Components (Hardware/Software) (LATIMER)

In order to implement CORSICA, both hardware and software components will be required; let us first examine the necessary hardware. As illustrated in Figure 1 below, we will need three main

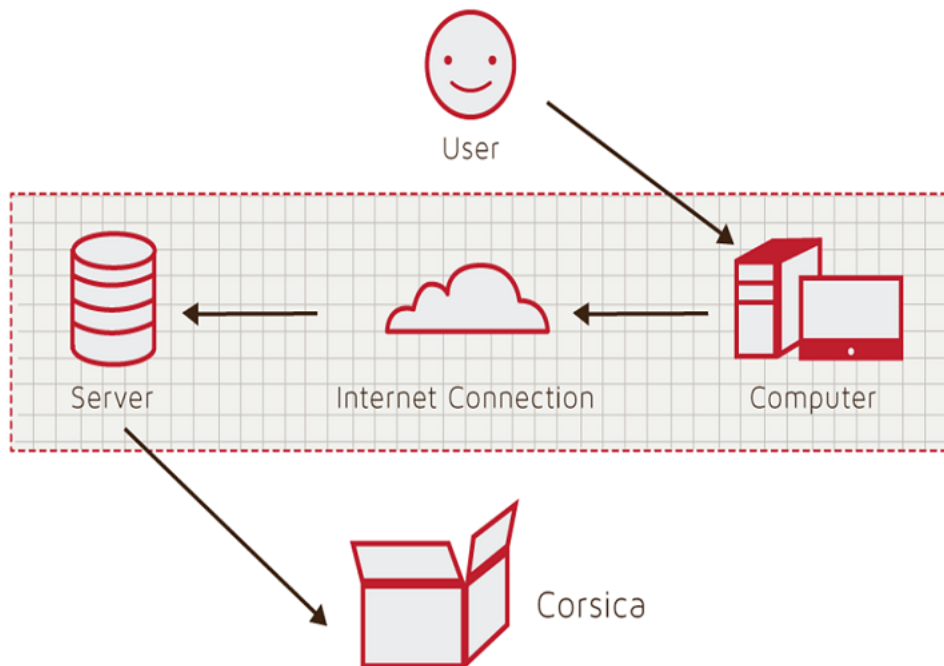pieces: a computer, an internet connection, and a server.

Figure 1

Hardware Components

The computer and internet connection are required in order to view/use and connect to the internet, respectively. The server will consist of both a web and database server - although this could potentially be broken up into two (or more) separate servers. The web server will be used to display and deliver web pages, while the database server will hold and manage the database.

For the software components, we will first need a database that contains all of the information relevant to CORSICA, such as student and course information. The database could possibly be created with MySQL, and manipulated using PHP, SQL, javascript, and HTML. We would require a number of interfaces for our different users, such as students or advisors; these GUIs would utilize HTML, CSS,

and javascript. We would need a notification system in order to let students know by email and text that a seat has opened up in their desired class. We would require some kind of system (or algorithm) to handle the labs and recitations. And finally, we would need algorithms to make CORSICA work, from identifying open seats to loading an enrollment data file. The algorithms will be implemented with C++, and would consist of the following: Increase Course Capacity, Close Course, Notification, Add Student to Wait-list, Drop Student from Wait-list, Check for Open Seats, and Load Enrollment Data Files.

3   IDENTIFICATION OF CASE STUDY (LATIMER)

We are first developing CORSICA for the Old Dominion University Computer Science department with the assistance of our faculty mentor, Dr. Irwin Levinstein. Dr. Levinstein is an Associate Professor at ODU in Computer Science, as well as the course scheduler for the Computer Science department. The analysis component of CORSICA could potentially be of assistance to Dr. Levinstein in that it could determine how many seats should be added (or removed) from the different Computer Science courses. In addition to being of assistance to Dr. Levinstein, CORSICA aims to alleviate some of the headaches related to course registration time. For advisors and faculty, this would mean no longer being barraged with emails from students requesting to be allowed into classes that are full, and students would no longer have to keep a constant vigil on the number of available seats in a class.

If CORSICA functions as we envision and is a success, it could be used in other departments at ODU and eventually throughout the entire school. In addition, it could potentially be used at other universities, as well.

4   CORSICA PRODUCT PROTOTYPE DESCRIPTION (TONY)

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 during the demonstration as access will have not yet been granted. The Seat Analysis System will be omitted in the prototype as it does not contribute essential functionality. Simulation will show how CORSICA operates for various users: Administrator, Scheduler, Student, Advisor, and Non-Logged in personnel.

The CORSICA prototype will demonstrate the following functional goals and objectives: operate as a mainly automated system, make efficient use of current notification system (will be simulated), fair enrollment process, reserve seats for student users on course wait-lists, and trigger alerts. CORSICA will operate as an automated system while still accepting user requests / actions. Allowing the system to be a mainly automated ensures the greatest ease of use for all users. We intend to tie into Old Dominion's current alert system; specifically the text and email notification features. The current alert system sends text messages and emails to students in the event of school closures. Students are required to opt into the system by providing their contact information. In an effort to ensure fairness during course enrollment periods, CORSICA will reserve seats for student users on a course wait-list and block sign up attempts from students who are not. Finally, alerts will be triggered, by means of a pop up textbox, when an Administrator user or Scheduler changes course information to ensure all changes are intentional. Below, Figure 2 shows a comparison between the real world 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

Figure 2
Real World Product and Prototype Comparison

4.1 Prototype Architecture (Hardware/Software) (TONY)

Figure 3 shows the CORSICA prototype will be comprised of a four major components: a course database, back-end algorithms, a notification system, and a user interfa. 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 above information will be arbitrarily generated as CORSICA will not be granted access to actual course information during the prototype demonstration.
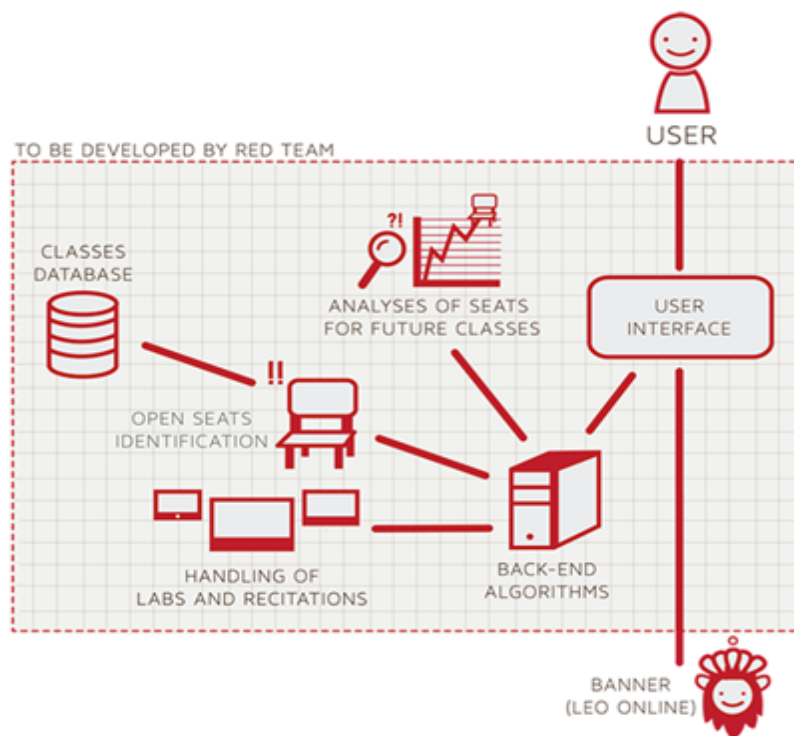

Figure 3

Prototype Major Functional Components Diagram

Various back-end algorithms will be used for the prototype. They include the following: Load Enrollment Data File, Open Course, Check for Open Seats, Add Student to Wait-list, Notification, Drop Student from Wait-list, Increase Course Capacity, and Close Course. Below in Figure 4 are 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 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
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**

Figure 4

Description of Algorithms

CORSICA will integrate with Old Dominion University's current notification system. More information on the current system may be found here. The prototype will simulate the notification process with a prompt alerting the user that students have been notified.

4.2   Prototype Features and Capabilities (Bitaseme)

The CORSICA prototype will aim to show consumers an example of the product's functionality. The audience of users can be separated into three categories: an Administrator, an Advisor, and a Student. How each audience can use CORSICA will be described below.

The following describes how an administrator can use CORSICA. Upon logging into CORSICA, the user will be greeted with a list of courses for the current semester and the

corresponding wait-lists. The user will search for a student name to see their wait-list history. In this particular case, Student Latimer Gerle has been on 4 different wait-lists. The user can also search for a specific course. In this case, CS 250 is the searched class. As you can see, the course description has populated along with the current wait-list. The user can view the contents of the wait-list and manipulate it by changing the position of the students on it as well as removing users from the wait-list or even the wait-list itself. This marks the end of the administrator prototype demonstration.

The following shows how an advisor can use CORSICA. Upon logging into CORSICA, the user will be greeted with the wait-lists list for the current semester. The user will search for a student name to see their wait-list history. In this particular case, Student Latimer Gerle has been on 4 different wait-lists. This search also returns Latimer's student information which consists of a name, UIN, ODU email address, phone number, the number of wait-lists that she is currently on, and the number of wait-lists that she has been on in the past. Like the administrator, the user can search for a specific course and view who is on the wait-list. The user can add, remove, and change the position of the students on the wait-list like the administrator. Unlike the administrator, the user cannot delete a wait-list from a course. This marks the end of the advisor prototype demonstration.

The following shows how a student can use CORSICA. Upon logging into CORSICA, the user will be greeted with the wait-list(s) that they are currently active on. The user can search the last name of a instructor to return results of all the wait-lists that belong to the course(s) that the instructor is currently teaching. The user can choose a course and then sign up to be on the wait-list. CORSICA will pop up an alert box stating the course, the semester, and the spot on the wait-list that the user is attempting to

sign up for. The user accepts the confirmation and is successfully added to the wait-list. This marks the end of the student prototype demonstration.

These three prototype demonstrations are significant in showing how the problem is solved because potential customers can better make a decision on whether or not to buy the product. The risks for CORSICA are listed below in Figure 5:

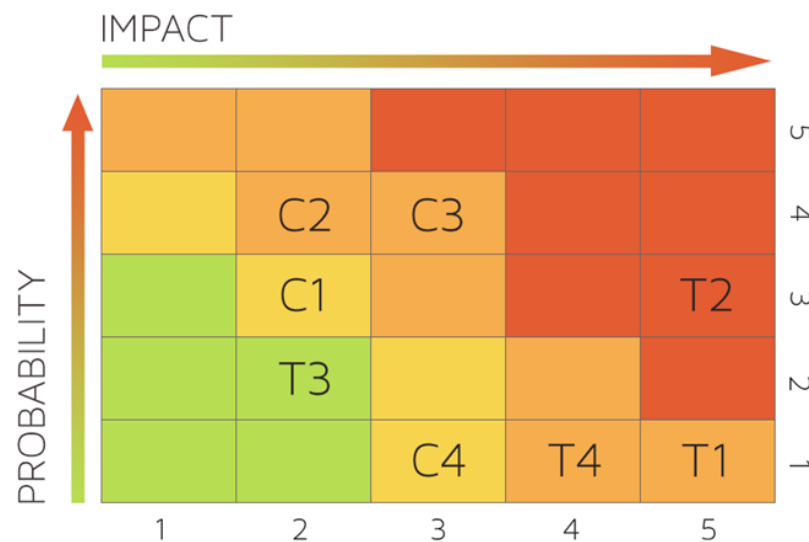| Customer Risks	Technical Risks
C1: Department Use Rejection	T1: Ability to Integrate with Banner
C2: Transition to New GUI	T2: Software Upgrades
C3: Cost of Product	T3: Availability of Server Storage
C4: Product Interest	T4: Security Vulnerability

Figure 5

Risk Matrix

Prototype Risk Mitigation Addressing: The prototype displayed it's functionality to mitigate the customer risks listed above by showing ease-of-use and product effectiveness. The risks that were not covered were the Technical Risks. Technical risks are dynamic with respect to the particular university. For T1, CORSICA is dependent on Banner so this risk should not be a major issue (if an issue at all). An update feature in the prototype was absent however it was not necessary to show the functionality. T3 and T4 are risks that are dependent upon the University using CORSICA and as such would be impossible to cover for every University around the world.

4.3 Prototype Development Challenges (Lookmai)

Objectives of the CORSICA prototype were listed in Section 4 in the comparison between the prototype and real-world product. For convenience, here are just the objectives below along with predicted challenges:

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**

Glossary

*Wait-list - to put on a waiting list

*Notification - the act of notifying someone

*E-mail - a system for sending messages from one computer to another computer

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

*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

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

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

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

*Laboratory - a room or building with special equipment for doing scientific experiments and tests

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

*Computer - an electronic machine that can store and work with large amounts of information

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

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

**MySQL - database management system

**PHP - server-side scripting language designed for web development

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

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

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

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

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

[**]**C++ -** general purpose programming language that is free-form and compiled

[***]**Banner -** university's centralized academic and administrative records system

[*]**Prototype -** an original or first model of something from which other forms are copied or developed

[***]**University Identification Number (UIN) -** unique identifier

[*]Found at http://www.merriam-webster.com/
[**] Found at http://en.wikipedia.org/wiki/
[***] Found at https://www.odu.edu

References

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

(March 24, 2014). Employment Projections. *Bureau of Labor Statistics.* Retrieved May 1, 2014,

from http://www.bls.gov/emp/ep_chart_001.htm