



Software Requirements Specification Document

Quantitative Plant Genetics Simulation Application

Version 1.0
February 10, 2006

CPSC 319
Team 7

UBC Botanical Garden and Centre for Plant Research

Dr. Andrew Riseman



ubcbotanicalgarden
& centre for plant research
AT THE UNIVERSITY OF BRITISH COLUMBIA

Table of Contents

1 INTRODUCTION	1
1.1 PURPOSE	1
1.2 SCOPE	1
1.2.1 <i>Name of the software product</i>	1
1.2.2 <i>Summary of the software and its functions</i>	1
1.2.3 <i>Application of the software</i>	1
1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS	2
1.4 REFERENCES	3
1.5 OVERVIEW	3
2. THE OVERALL DESCRIPTION	4
2.1 PRODUCT PERSPECTIVE	4
2.2 PRODUCT FUNCTIONS	5
2.2.1 <i>Administrator Functions</i>	5
2.2.2 <i>Professor Functions</i>	5
2.2.3 <i>TA Functions</i>	5
2.2.4 <i>Student Functions</i>	6
2.3 USER CHARACTERISTICS	6
2.4 CONSTRAINTS	6
2.4.1 <i>Hardware limitations</i>	6
2.4.2 <i>Software Limitations</i>	7
2.4.3 <i>Parallel Operations</i>	7
2.4.4 <i>Audit functions</i>	7
2.4.5 <i>Reliability Requirements</i>	7
2.4.6 <i>Safety and Security considerations</i>	7
2.5 ASSUMPTIONS AND DEPENDENCIES	7
3. SPECIFIC REQUIREMENTS	8
3.1 EXTERNAL INTERFACES	8
3.1.1 <i>Administrator, Professor, TA, Student Interfaces</i>	8
3.1.2 <i>Administrator Interfaces</i>	11
3.1.3 <i>Administrator, Professor Interfaces</i>	11
3.1.4 <i>Student Interfaces</i>	17
3.2 FUNCTIONS	22
3.2.1 <i>Administrator, Professor, TA, Student Functions</i>	22
3.2.2 <i>Administrator, Professor Functions</i>	25
3.2.3 <i>Administrator, Professor, TA Functions</i>	39
3.2.4 <i>Student Functions</i>	40
3.3 PERFORMANCE REQUIREMENTS	42
3.4 LOGICAL DATABASE REQUIREMENTS	43
3.5 SOFTWARE SYSTEM ATTRIBUTES	44
3.5.1 <i>Reliability</i>	44
3.5.2 <i>Availability</i>	44
3.5.3 <i>Security</i>	44
3.5.4 <i>Maintainability</i>	44
3.5.5 <i>Portability</i>	44
4. CHANGE MANAGEMENT PROCESS	45
5. DOCUMENT APPROVALS	46
6. SUPPORTING INFORMATION	47
6.1 CLIENT PROPOSAL	47
6.2 SAMPLE LECTURE SLIDES (QUANTITATIVE GENETICS)	50

1 Introduction

1.1 Purpose

The purpose of this SRS is to identify the specifications and requirements of the product outlined in section 1.2. The intended audiences are the client, Dr. Andrew Riseman (Assistant Professor, University of British Columbia Botanical Garden Center) and the members of the software engineering group, Team 7.

1.2 Scope

1.2.1 Name of the software product

The name of the application will be "XGene 360".

1.2.2 Summary of the software and its functions

The application will simulate plant breeding between diploid plants of the same species under similar environment conditions, employing basic concepts and principles of plant breeding and quantitative genetics. An administrator will have full access to all professor functions and all accounts. A professor will be able to create courses, professors and TAs. A professor will be able to generate a unique problem for each student user by specifying the mean, standard deviation, heritability factor and the number of genes for two independent (non-linked) traits in the parent generation. The professor will also be able to assign problems to courses, students or TAs and view the progress of any student (i.e. see the status of an individual student's problem without the capability to make any changes to it). Teaching assistants (TAs) will also be able to view the progress of any student from a course assigned to them. The students will be able to select any number of offspring for inter-crossing to generate the next generation. Through the manipulation and analysis of progeny, students will be able to determine the problem parameters without any further assistance from the application. The application will operate in a web-based environment and users will require a web browser to interact with the application.

1.2.3 Application of the software

XGene 360 will be a plant breeding simulation application that aims to facilitate a student's understanding of basic concepts and principles in quantitative genetics.



It will be ideal for use in a classroom setting and since the application will be accessible through a web browser, the concepts will be able to be learned at the leisure of the student. This will save a considerable amount of class time. The feature of generating a unique problem for each student with minimal amount of input will also save considerable time for the professor. It will also provide students with experience in working with simulation application and record keeping (as students are required to analyze the output produced by the application).

The application will operate under the GPL (General Public License) open source license.

1.3 Definitions, Acronyms, and Abbreviations

<i>additive effect</i>	A difference in phenotype resulting from the substitution of an 'a' allele for an 'A' allele when averaged over all the individual members of a population.
<i>additive genetic variance</i>	Genetic variance associated with the average effects of substituting one allele for another.
<i>administrator</i>	A user of the system which has all the functionalities of a professor but also has access to all accounts.
<i>allele</i>	One of the different forms of a gene that can exist at single locus.
<i>client</i>	Dr. Andrew Riseman, Assistant Professor, UBC Botanical Garden and Centre for Plant Research.
<i>course</i>	Created by the professor, contains a group of students.
<i>cross</i>	The deliberate mating of two parental types of organisms in genetic analysis.
<i>CSV file</i>	Comma separated value file. (Comma delimited)
<i>diploid</i>	An organism having two chromosome sets in each of its cells.
<i>environmental variance</i>	The variance due to environmental variation.
<i>frequency histogram</i>	A step curve in which the frequencies of various arbitrarily bounded classes are graphed.
<i>genes</i>	The effective factor that is affecting a trait.
<i>generation</i>	An iteration of the genetic algorithm.
<i>genotype</i>	The specific allelic composition of a cell, either of the entire cell or, more commonly, for a certain gene or a set of genes.
<i>mean</i>	The arithmetic average, the sum of data divided by the sample size.
<i>narrow heritability</i>	The proportion of phenotypic variance that can be attributed to additive genetic variance.
<i>phenotype</i>	The detectable outward manifestations of a specific genotype.
<i>problem</i>	Created by the professor, it generates the progeny base on its parameters specified by the professor.
<i>professor</i>	A user of system who is able to create/modify/remove students, teaching assistants, courses and problems for the course he/she is managing.
<i>progeny</i>	The resulting offspring from a cross.
<i>quantitative genetics</i>	The study of genetic basis of continuous variation in phenotype.
<i>selection differential</i>	The difference between the mean of a population and the mean of the individual members selected to be parents of the next generation.
<i>selection response</i>	The amount of change in the average value of some phenotypic character between the parental generation and the offspring generation as a result of the selection of parents.
<i>standard deviation of environment</i>	The square root of the environmental variance.

Software Requirements Specification Document

<i>student</i>	A user of the system who works on his/her assigned problems by stimulating plant breeding.
<i>teaching assistant (TA)</i>	A user of the system who can view a student's progress.
<i>trait</i>	A genetic feature that maybe passed from one generation to the next.

1.4 References

1. Griffiths, Wessler, Lewontin, Gelbart, Suzuki, and Miller. Introduction to Genetic Analysis Eighth Edition. New York: W. H. Freeman and Company, 2005.
2. Bruegge, and Dutoit. Object-Oriented Software Engineering Using UML, Patterns, and Java Second Edition. Upper Saddle River: Prentice Hall, 2004.
3. YellowLeaf Project. "Greengene" University of British Columbia. 2005
<<http://bgcpr128-01.agsci.ubc.ca/~dev7/login.php>>
4. PowerPoint slides on genetics provided by Andrew Riseman. (See section 6, more available upon request.)

1.5 Overview

The rest of the SRS document will contain functional and non-functional requirements for this application. In particular, section 2 will contain an overall description of the product, including the perspective, functions, characteristics, constraints, and assumptions. This section will be most useful for the client, Dr. Riseman.

In section 3, specific and detailed requirements will be laid out, and this information will be most important and relevant to the developers of Team 7. The section will outline the specifications of the external application interface, as well as the application's functions and use cases. Also in this section will be details about performance and database requirements. The application's non-functional attributes will be outlined in section 3.5.

Section 4 will describe how changes to the application specifications are handled after the approval of the SRS. Section 5 will contain formal agreements between participants of the software development. Lastly, section 6 will include supporting information that contributes to the SRS document.

2. The Overall Description

XGene 360 will be an educational tool that simulates the behaviour of genes during plant breeding. Its purpose will be to give students a broad understanding of quantitative genetics. It will be a web-based application, written as a complement to the qualitative genetics simulation application, YellowLeaf.

2.1 Product Perspective

The system is intended to be a web-based application linked to a database. The database will manage all of the data and files associated with the application. The application will be accessible via four roles: administrator, professor, teaching assistant (TA) and student. The professor will be allowed actions such as creating professor and a course. A professor will have full privileges pertaining to the courses they have created. For each course, the corresponding professor will be able to create problems, assign problems, and view progress. Professors will also be able to create students, TAs and professors, as well as assign problems to an individual student or TA. The administrator will be able to access all the accounts in the database. TAs will have access to their corresponding accounts, as well as courses and problems assigned to them and students assigned to them. Student users will only have access to their corresponding accounts and problems assigned to them.

The application will be written as a counterpart to YellowLeaf, covering the concepts of quantitative genetics. As these two applications will be bundled together, XGene 360 will follow a fair amount of conventions set by YellowLeaf for the sake of consistency. Such conventions include portions of YellowLeaf's layout and design. Also, components of the database will be available for both applications. For example, identical logins and passwords will be used across both applications.

2.2 Product Functions

2.2.1 Administrator Functions

1. The application will start when the administrator installs it. The administrator will need to fill in a username and password when starting up the application in order to access the administrator functionality.
2. The administrator will be able to login to the application using a username and password.
3. All functionality available to the professor user will be available to the administrator.

2.2.2 Professor Functions

1. The professor will be able to login to the application using a username and password.
2. The professor will be able to create, delete and modify professors, TAs and students.
3. To create a professor, the professor will have to open the page for generating a new professor profile. A first name and last name will be required as input.
4. Courses will be created, deleted and modified by the professor. To create a course, a professor will have to enter the course ID in the appropriate field on the course creation page. Once a course is created, a professor will be able to assign students and TAs to the course.
5. The application will allow professors to create, modify and delete problems. Professors will be able to assign a problem to individual TAs, individual students or whole courses.
6. To create a problem, the professor will have to enter in mean values, standard deviations, heritability values, a base value, a unit, and the number of genes for two non-linked traits. Other parameters will include the percent range of acceptance (for student results), the number of generated plants, the number of displayed plants, and problem properties such as deadlines, starting dates and number of iterations allowed.
7. Professors will be able to display a list of the problems they have created. For each problem, a professor will be able to see the progress of each student assigned to it.
8. For each student, the professor will be able to view the student's current results for a problem. The current results will look similar to what a student will see when doing a problem. For problems a student has handed in, the professor will be able to view the final results, view the system's computed evaluation of the results and write feedback to the student.
9. A list of students will be available for viewing for the professor who created them.

2.2.3 TA Functions

1. The TA will be able to login to the application using a username and password.
2. TAs will be able to see a list of problems assigned to them and the students assigned to the problem.
3. For each student in an assigned course, the TA will be able to view the student's current results for a problem. The current results will look similar to what a student will see when doing a problem.

2.2.4 Student Functions

1. The student will be able to login to the application using a username and password.
2. The student will be able to modify his or her own password.
3. Student will be able to see a list of problems assigned to them.
4. Students will be able to work on a problem once it appears on the list of problems.
5. For each problem, the student will see a grid of displayed plants and histograms representing the population for a certain trait. The student will be able to select plants by selecting an area in the histogram, which will highlight certain parts of the grid. From the grid, the student will be able to select seeds to breed the next generation. Students will be able to select seeds from previous generations via a history list display.
6. After a specified number of breeding iteration, the student will be able to fill out their concluded means, standard deviations, heritability values and number of effective factors and hand it in for evaluation. The student will also be able to upload an optional file report for hand in.

2.3 User Characteristics

The intended users will be the administrator, the professors, the TAs, and the students.

It will be assumed that the administrator and professors have basic knowledge of how to use the computer, particularly the web browser. It will also be assumed that the administrator and professors have decent knowledge of plant genetics, particularly quantitative plant genetics, in order to understand the application's nomenclature and interface.

It will be assumed that the students and TAs have basic knowledge of how to use the computer, the web browser and the Internet. The students will be required to have enough knowledge of plant genetics in order to understand and work on the problem assigned to them. The TAs will be required to have enough knowledge of plant genetics in order to understand and evaluate problems assigned to them.

2.4 Constraints

2.4.1 Hardware limitations

The system will be designed to run on a Macintosh G4 running on Mac OS X 10.2.8.

2.4.2 Software Limitations

The system will be developed in an interpretive language. It will also be platform independent.



The following software will be required in order to run the application:

- Apache 1.3.33
- MySQL 3.23
- PHP 4.2.3

Installation of the application will require minimal computer knowledge and effort.

The application will load student login information from the corresponding application, YellowLeaf.

2.4.3 Parallel Operations

There will be no software limitation on the number of concurrent users of the system. The limitation will be based solely on hardware resources and server configuration.



2.4.4 Audit functions

Errors will be logged along with timestamps.



2.4.5 Reliability Requirements

Failures in the application will not cause any error in the physical server and other applications running on the server. The application will be able to function reliably for a minimum of two days.

2.4.6 Safety and Security considerations

The administrator, professors, students and TAs will not be able to access the backend database directly.

2.5 Assumptions and Dependencies

The client will be able to provide a server for application testing, with the server constraints as stated in section 2.4. The application will depend on PHP, MySQL, and an Apache server as mentioned. The students will have access to computers with internet browsers and will be provided with the URL for the client's server to use this application.

3. Specific Requirements

3.1 External Interfaces

The low-fidelity prototype screenshots will be displaying only selected interfaces.

3.1.1 Administrator, Professor, TA, Student Interfaces

Figure 1 shows a low-fidelity prototype of a login interface. The main window displays "Welcome to XGene 360!". In the center, there is a "Login" dialog box. This dialog box contains two input fields: "User:" and "Password:", each followed by a text box. Below these fields is an "OK" button. At the bottom of the main window, there is a placeholder text "<Acknowledgements go here>".

Figure 1: Login

<i>Name</i>	Login
<i>Description</i>	The first page a user sees. It will allow a user to login to the application.
<i>Inputs</i>	1. Username 2. Password
<i>Outputs</i>	The application accepts login information and brings the user to the corresponding starting page.
<i>Possible Errors</i>	An invalid login will result in a message indicating that the information is incorrect.

Software Requirements Specification Document

<i>Name</i>	Logoff
<i>Description</i>	After the user clicks the logoff button on any page, the user will see this page as verification of logoff.
<i>Inputs</i>	None
<i>Outputs</i>	The application accepts logoff and sends user to corresponding page to show after logging off.
<i>Possible Errors</i>	None

<i>Name</i>	ChangeOptions
<i>Description</i>	After the users clicks a link directing them to the options page, the user can change personal information.
<i>Possible Inputs</i>	1. New password 2. Number of entries per page of listing
<i>Outputs</i>	The application accepts changed information and brings the user back to the previous page.
<i>Possible Errors</i>	An invalid field will result in a descriptive message indicating that the information is incorrect.

Hi <Professor Name>!

Here is a list of problems assigned to you:

Problem	Course	Start Date	Due Date	Options/Views
#316: Trait A and B	BIOL334	05/01/06	10/01/06	<div>Change ProblemView Students</div>
#341: Trait C and D	BIOL334	16/01/06	25/01/06	<div>Change ProblemView Students</div>

Figure 2: View Problems for Professor

Hi <Student Name>!
Here is a list of problems assigned to you:





Problem	Course	Start Date	Due Date	Recent Iteration	Problem Solution
<u>#316: Trait A and B</u>	BIOL334	05/01/06	10/01/06	  [21/30]	<input type="button" value="Edit"/> <input type="button" value="Hand In"/>
<u>#341: Trait C and D</u>	BIOL334	16/01/06	25/01/06	  [11/30]	<input type="button" value="Edit"/> <input type="button" value="Hand In"/>

Figure 3: View Problems for Student

<i>Name</i>	ViewProblems
<i>Description</i>	Where a user can see a list of problems assigned/created by him/her. Details for the problem will be viewable via a button click. User-specific options will be available as well.
<i>Inputs</i>	None
<i>Possible Outputs</i>	Certain button clicks will bring you to the following possible pages: <ol style="list-style-type: none"> 1. Changing problem parameters 2. Viewing students and/or TAs assigned to problem 3. Editing problem solution 4. Handing in problem solution
<i>Possible Errors</i>	None

<i>Name</i>	ViewStudentProgress
<i>Description</i>	Where a user can see a student's progress. Administrator, professors and TAs will be able to see the progress of all the students assigned to them.
<i>Inputs</i>	None
<i>Outputs</i>	None
<i>Possible Errors</i>	None

3.1.2 Administrator Interfaces

<i>Name</i>	CreateAdministrator
<i>Description</i>	Creates a new administrator. This only happens once in the application's lifetime.
<i>Inputs</i>	1. First name 2. Last name 3. Password
<i>Outputs</i>	After verifying the inputs, the application will output a username and password.
<i>Possible Errors</i>	Invalid information will result in a message indicating that the information is incorrect.

3.1.3 Administrator, Professor Interfaces

<i>Name</i>	CreateProfessor
<i>Description</i>	Creates a new professor.
<i>Inputs</i>	1. First name 2. Last name 3. Password
<i>Outputs</i>	After verifying the inputs, the application will output a username and password.
<i>Possible Errors</i>	Invalid information will result in a message indicating that the information is incorrect.

<i>Name</i>	DeleteProfessor
<i>Description</i>	Deletes an existing professor.
<i>Inputs</i>	None
<i>Outputs</i>	After verifying the existence of the professor account, the application deletes the account from the database and tells the user that deletion has been successful.
<i>Possible Errors</i>	Professor does not exist, and a message will appear indicating that the delete failed.

<i>Name</i>	ModifyProfessor
<i>Description</i>	Modifies an existing professor.
<i>Inputs</i>	New password
<i>Outputs</i>	A page will pop up verifying the modification.
<i>Possible Errors</i>	Password is invalid, so the application will output a message indicating that another password should be entered.

<i>Name</i>	CreateCourse
<i>Description</i>	Creates a new course.
<i>Inputs</i>	ID of course (i.e. BIOL121)
<i>Outputs</i>	After verifying the input, the application will output a message indicating success.
<i>Possible Errors</i>	Invalid/duplicate course will result in a error message.

Software Requirements Specification Document

<i>Name</i>	DeleteCourse
<i>Description</i>	Deletes an existing course.
<i>Inputs</i>	None
<i>Outputs</i>	After verifying the existence and information for the course, the application deletes the course from the database and tells the user that deletion has been successful.
<i>Possible Errors</i>	If the course is being used, the application will ask the user whether he/she is sure of deletion.
<i>Name</i>	ModifyCourse
<i>Description</i>	Modifies an existing course.
<i>Inputs</i>	New course ID
<i>Outputs</i>	A page will pop up verifying the modification.
<i>Possible Errors</i>	Input is invalid (i.e. duplicate), so the application will output a message indicating that valid input should be used.
<i>Name</i>	ViewCourses
<i>Description</i>	Where a user can see a list of courses. The modification page for a particular course will be viewable via a button click. Courses will be able to be deleted via a button click as well.
<i>Inputs</i>	None
<i>Possible Outputs</i>	Certain button clicks will bring you to the following possible pages: 1. Modifying a course. 2. Deleting a course.
<i>Possible Errors</i>	None
<i>Name</i>	CreateTA
<i>Description</i>	Creates a new TA.
<i>Inputs</i>	1. First name 2. Last name 3. Password 4. Assigned courses (optional)
<i>Outputs</i>	After verifying the inputs, the application will output a username and password.
<i>Possible Errors</i>	Invalid information will result in a message indicating that the information is incorrect.
<i>Name</i>	DeleteTA
<i>Description</i>	Deletes an existing TA.
<i>Inputs</i>	None
<i>Outputs</i>	After verifying the existence of the TA account, the application deletes the account from the database and tells the user that deletion has been successful.
<i>Possible Errors</i>	TA does not exist, and a message will appear indicating that the delete failed.

Software Requirements Specification Document

<i>Name</i>	ModifyTA
<i>Description</i>	Modifies an existing professor.
<i>Possible Inputs</i>	1. New password 2. Assigned courses
<i>Outputs</i>	A page will pop up verifying the modification.
<i>Possible Errors</i>	Input is invalid, so the application will output a message indicating that valid input should be used.

<i>Name</i>	ViewTAs
<i>Description</i>	Where a user can see a list of TAs assigned/created by him/her. The modification page for a particular TA will be viewable via a button click. TAs will be able to be deleted via a button click as well.
<i>Inputs</i>	None
<i>Possible Outputs</i>	Certain button clicks will bring you to the following possible pages: 1. Modifying a TA. 2. Deleting a TA.
<i>Possible Errors</i>	None

Create Student Account

First Name:

Last Name:

Student Number:

Create Account

- OR -

Upload File:
[Help?]

Browse...

Start Batch Job

Figure 4: Create Student

Software Requirements Specification Document

<i>Name</i>	CreateStudent
<i>Description</i>	Creates a new student.
<i>Inputs</i>	<ol style="list-style-type: none"> 1. First name 2. Last name 3. Password 4. Assigned courses (optional)
<i>Outputs</i>	<p>The user will also be able to upload a text file with a list of student information.</p> <p>After verifying the inputs, the application will output a username and password. If the user provided an excel sheet, then a list of usernames and passwords will be provided.</p>
<i>Possible Errors</i>	Invalid information will result in a message indicating that the information is incorrect.
<i>Name</i>	DeleteStudent
<i>Description</i>	Deletes an existing student.
<i>Inputs</i>	None
<i>Outputs</i>	After verifying the existence of the student account, the application deletes the account from the database and tells the user that deletion has been successful.
<i>Possible Errors</i>	Student does not exist, and a message will appear indicating that the delete failed.
<i>Name</i>	ModifyStudent
<i>Description</i>	Modifies an existing student.
<i>Possible Inputs</i>	<ol style="list-style-type: none"> 1. New password 2. Assigned courses
<i>Outputs</i>	A page will pop up verifying the modification.
<i>Possible Errors</i>	Input is invalid, so the application will output a message indicating that valid input should be used.
<i>Name</i>	ViewStudents
<i>Description</i>	Where a user can see a list of students assigned/created by him/her. The modification page for a particular student will be viewable via a button click. Students will be able to be deleted via a button click as well.
<i>Inputs</i>	None
<i>Possible Outputs</i>	<p>Certain button clicks will bring you to the following possible pages:</p> <ol style="list-style-type: none"> 1. Modifying a student. 2. Deleting a student.
<i>Possible Errors</i>	None

Create Problem

of Plants Generated:
of Plants Displayed:
% Range of Acceptance:
Iteration:

Problem ID: **316**
Problem Name:
Start Date:
Due Date:
Course:

Trait A:
Mean of P1:
Mean of P2:
SD of P1:
SD of P2:
Base Value:
Unit:
h²:
of Genes:

Trait B:
Mean of P1:
Mean of P2:
SD of P1:
SD of P2:
Base Value:
Unit:
h²:
of Genes:

Figure 5: Create Problem

<i>Name</i>	CreateProblem
<i>Description</i>	Creates a new problem.
<i>Inputs</i>	Indicated by Figure 5.
<i>Outputs</i>	After verifying the input, the application will output a message indicating success.
<i>Possible Errors</i>	Invalid field inputs will result in an error message.
<i>Name</i>	DeleteProblem
<i>Description</i>	Deletes an existing problem.
<i>Inputs</i>	None
<i>Outputs</i>	After verifying information for the problem, the application deletes the problem from the database and tells the user that deletion has been successful.
<i>Possible Errors</i>	If the problem is being used, the application will ask the user whether he/she is sure of deletion.
<i>Name</i>	ModifyProblem
<i>Description</i>	Modifies an existing problem.
<i>Inputs</i>	Any of the inputs indicated by Figure 5 will be able to be modified.
<i>Outputs</i>	A page will pop up verifying the modification.
<i>Possible Errors</i>	Input is invalid (i.e. duplicate), so the application will output a message indicating that valid input should be used.

Student: John Doe Student ID: 00000000 Problem: 316 Due Date: 10/01/06		Problem Solution: % Range of Acceptance: 20%	
Student Solution:		Trait A: Height	
Mean of P1:	39.2 cm	Mean of P1:	40 cm
Mean of P2:	50.2 cm	Mean of P2:	50 cm
SD of P1:	2.5 cm	SD of P1:	5 cm
SD of P2:	3.5 cm	SD of P2:	4 cm
h2:	20	h2:	25
# of Effective Factors:	10	# Factors:	12
Trait B: Amount of blue		Trait B: Blue	
Mean of P1:	30%	Mean of P1:	25%
Mean of P2:	40%	Mean of P2:	45%
SD of P1:	2.5%	SD of P1:	12%
SD of P2:	3.5%	SD of P2:	10%
h2:	10	h2:	10
# of Effective Factors:	12	# Factors:	15
Student File Report:		Upload Feedback:	
<input type="button" value="View Report"/>		<input type="button" value="Browse..."/>	
		<input type="button" value="Back"/> <input type="button" value="Send Feedback"/>	

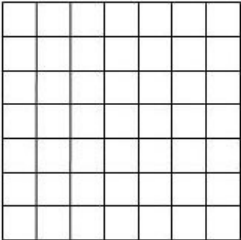
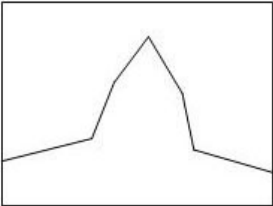
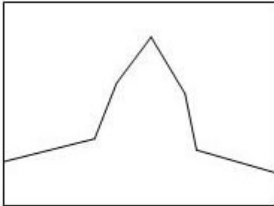
Figure 6: Evaluate Student

<i>Name</i>	EvaluateStudent
<i>Description</i>	The user will be able to view student results for a problem and upload an evaluation/feedback file.
<i>Inputs</i>	An evaluation/feedback file upload that will be optional.
<i>Outputs</i>	A page will pop up verifying the success of sending the evaluation.
<i>Possible Errors</i>	Evaluation file is invalid, so the application will output a message indicating that a valid file should be used.

3.1.4 Student Interfaces

Student: John Doe
Student ID: 00000000
Problem: 316
Due Date: 10/01/06

Iteration: 0/30

System Status:

Generated 1000 plants ...
Chosen 100 plants ...
Graphed histograms ...
Creating next generation ...

Instructions:


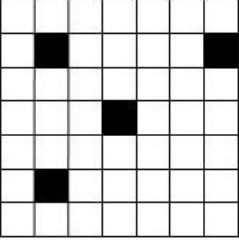
Wait until generation is created ...


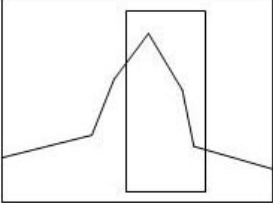
Figure 7: Begin Problem

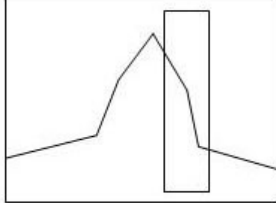
<i>Name</i>	BeginProblem
<i>Description</i>	The user will start the problem and view the beginning problem parameters.
<i>Inputs</i>	None
<i>Outputs</i>	The next page in the problem will appear, allowing the student to continue.
<i>Possible Errors</i>	None

Student: John Doe
 Student ID: 00000000
 Problem: 316
 Due Date: 10/01/06

Iteration: 1/30








System Status:

Generated 1000 plants ...
 Chosen 100 plants ...
 Graphed histograms ...
 Generated 1000 plants ...
 Chosen 100 plants ...
 Waiting ...

Instructions:

1. Select an area on left histogram to select plants.
 2. Select an area on right histogram to select plants.
 3. Press 'Breed' to breed plants.

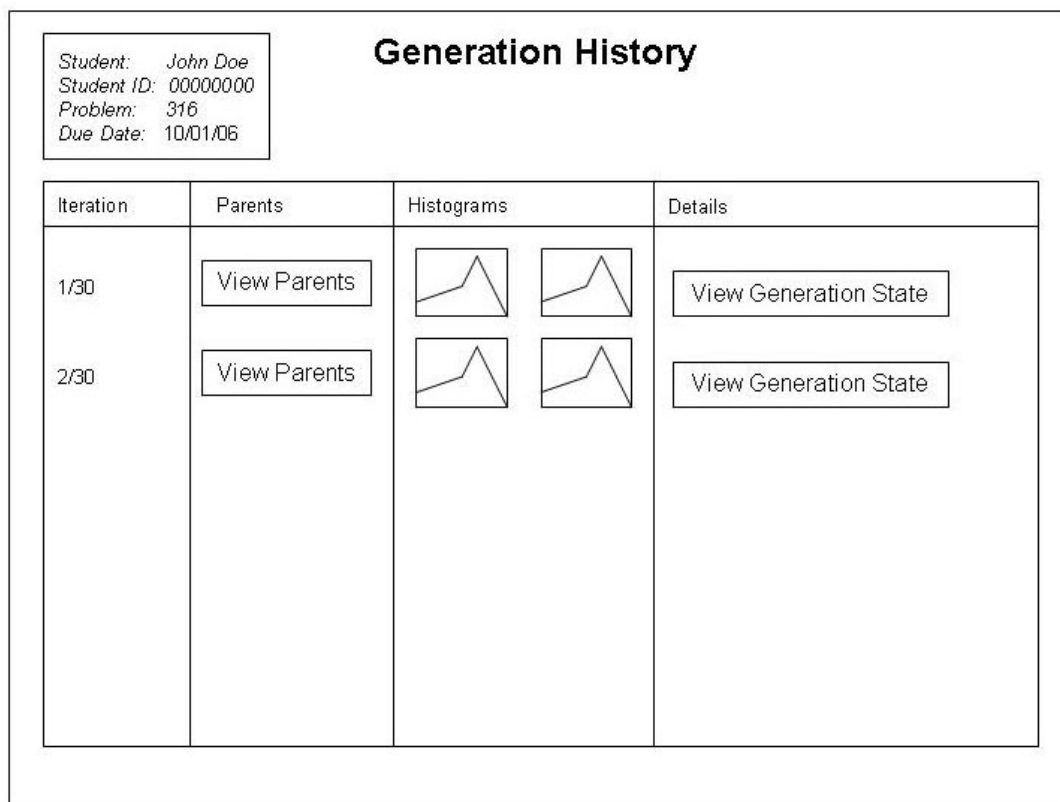


Breed

View/Select from History

Figure 8: Breed Generation

<i>Name</i>	BreedGeneration
<i>Description</i>	Based on the current state of the problem, the user will be able to select plants for breeding.
<i>Inputs</i>	1. An area in each histogram 2. Seeds from the highlighted areas on the grid 3. As an option, seeds from a previous generation
<i>Outputs</i>	Breeding will occur, and the next page in the problem will appear, allowing the student to continue.
<i>Possible Errors</i>	None

**Figure 9: View Generation History**

<i>Name</i>	ViewGenerationHistory
<i>Description</i>	The user will be able to view a generation history for the current problem.
<i>Inputs</i>	None
<i>Outputs</i>	Certain button clicks will bring you to the following possible pages: <ol style="list-style-type: none"> 1. View the parents for the generation. 2. View the problem state for that generation.
<i>Possible Errors</i>	None

Student: John Doe
 Student ID: 00000000
 Problem: 316
 Due Date: 10/01/06

Iteration: 30/30

Edit Solution:

Trait A:

Mean of P1:

Mean of P2:

SD of P1:

SD of P2:

h^2 :

of Genes:

Trait B:

Mean of P1:

Mean of P2:



SD of P1:

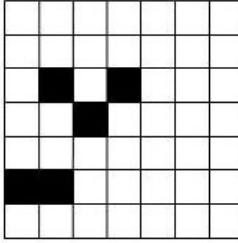
SD of P2:

h^2 :

of Genes:

Upload File Report:



Instructions:

1. Fill in blanks with answers.
 2. Upload a file report for evaluation.

Figure 10: Edit Solution

<i>Name</i>	EditSolution
<i>Description</i>	The user will be able to evaluate the problem and hand in his/her solution.
<i>Inputs</i>	Indicated in Figure 10.
<i>Outputs</i>	Certain button clicks will bring you to the following possible pages: <ol style="list-style-type: none"> 1. View the history for the problem. 2. Save the problem solution. 3. Hand in the problem solution.
<i>Possible Errors</i>	Fields and/or report file is invalid, so the application will output a message indicating that a valid input should be used.

Student: John Doe Student ID: 00000000 Problem: 316 Due Date: 10/01/06		Problem Solution: % Range of Acceptance: 20%	
Student Solution:		Trait A: Height	
Trait A:	Height	Mean of P1:	40 cm
Mean of P1:	39.2 cm	Mean of P2:	50 cm
Mean of P2:	50.2 cm	SD of P1:	5 cm
SD of P1:	2.5 cm	SD of P2:	4 cm
SD of P2:	3.5 cm	h2:	25
h2:	20	# Factors:	12
# of Effective Factors:	10		
Trait B: Amount of blue		Trait B: Blue	
Mean of P1:	30%	Mean of P1:	25%
Mean of P2:	40%	Mean of P2:	45%
SD of P1:	2.5%	SD of P1:	12%
SD of P2:	3.5%	SD of P2:	10%
h2:	10	h2:	10
# of Effective Factors:	12	# Factors:	15
Student File Report: <input type="button" value="View Report"/>		Student Evaluation: <input type="button" value="View Evaluation"/>	
		<input type="button" value="Back"/> <input type="button" value="OK"/>	

Figure 11: View Problem Solution

<i>Name</i>	ViewProblemSolution
<i>Description</i>	The user will be able to view student results for a problem as well as any feedback from the professor.
<i>Inputs</i>	None
<i>Outputs</i>	None
<i>Possible Errors</i>	None

3.2 Functions

3.2.1 Administrator, Professor, TA, Student Functions

<i>Use case name</i>	Login
<i>Initiating actor</i>	Administrator, Professor, TA, Student
<i>Participating</i>	None
<i>Entry condition</i>	The system has started.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor enters necessary fields. 2. The initiating actor clicks the "Login" button. 3. The system performs login authentication process. 4. The initiating actor is redirected to the homepage.
<i>Exit condition</i>	The initiating actor is logged in.
<i>Extension</i>	FailedToLogin
<i>Special requirements</i>	None

<i>Use case name</i>	Logout
<i>Initiating actor</i>	Administrator, Professor, TA, Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "Logout". 2. The system performs the logoff process. 3. The initiating actor is redirected to the login page.
<i>Exit condition</i>	The initiating actor is logged out.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToLogin
<i>Initiating actor</i>	Administrator, Professor, TA, Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to login in with incorrect login information.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor is redirected to the "Login" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Login" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	ChangePassword
<i>Initiating actor</i>	Administrator, Professor, TA, Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor enters necessary fields.2. The initiating actor clicks the "Update" button.3. The system validates the inputs, updates the information in the database, and displays a message indicating the password has been successfully changed.
<i>Exit condition</i>	The information is updated and is saved to the database.
<i>Extension</i>	FailedToChangePassword
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToChangePassword
<i>Initiating actor</i>	Administrator, Professor, TA, Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to change the password but provides invalid information.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor is redirected back to the "Change Password" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Change Password" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ViewListOfProblems
<i>Initiating actor</i>	Administrator, Professor, TA, Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating selects "View List of Problems".2. The system finds all problems associated with the initiating actor.3. The list of problems is displayed.
<i>Exit condition</i>	The list of problems assigned to the initiating actor is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	SelectAProblem
<i>Initiating actor</i>	Administrator, Professor, TA, Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and at least one problem can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor views the list of problems.2. The initiating actor selects a problem from the list.3. The initiating actor clicks "Select" button.
<i>Exit condition</i>	The selected problem is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

3.2.2 Administrator, Professor Functions

<i>Use case name</i>	CreateProfessor
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "Create Professor". 2. The initiating actor enters necessary information. 3. The initiating actor clicks "Create" button. 4. The system validates the input, saves the information to the database, and displays a message indicating why the operation is successful.
<i>Exit condition</i>	The new account is created and is saved to the database.
<i>Extension</i>	FailedToCreateProfessor
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToCreateProfessor
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to create a new account for a professor but enters invalid information.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor is redirected to the "Create Professor" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Create Professor" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	DeleteProfessor
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "Delete Professor". 2. The initiating actor selects a professor from the list to delete. 3. The initiating actor clicks "Delete" button. 4. The system deletes the account from the database and displays a message indicating the account is successfully deleted.
<i>Exit condition</i>	The account is deleted from the database.
<i>Extension</i>	FailedToDeleteProfessor
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	FailedToDeleteProfessor
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to delete an account for a professor but fails.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Delete Professor" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Delete Professor" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ModifyProfessor
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	1. The initiating actor selects "Modify Professor". 2. The initiating actor enters necessary information. 3. The initiating actor clicks "Modify" button. 4. The system validates the input, updates the information in the database, and displays a message indicating the information has been successfully updated.
<i>Exit condition</i>	The information is updated and is saved to the database.
<i>Extension</i>	FailedToModifyProfessor
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToModifyProfessor
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to modify account information for a professor but enters invalid information.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Modify Professor" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Modify Professor" page.
<i>Extension</i>	None
<i>Special requirements</i>	None


Software Requirements Specification Document

<i>Use case name</i>	CreateTA
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Create TA".2. The initiating actor enters necessary information.3. The initiating actor clicks "Create" button.4. The system validates the input, saves the information to the database, and displays a message indicating the new account has been successfully created.
<i>Exit condition</i>	The new account is created and is saved to the database.
<i>Extension</i>	FailedToCreateTA
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToCreateTA
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to create a new account for TA but enters invalid information.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor is redirected to "Create TA" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Create TA" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	DeleteTA
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Delete TA".2. The initiating actor selects a TA to delete from the list.3. The initiating actor clicks "Delete" button.4. The system deletes the account from the database and displays a message indicating the account is successfully deleted.
<i>Exit condition</i>	The account is deleted from the database.
<i>Extension</i>	FailedToDeleteTA
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	FailedToDeleteTA
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to delete an account for a TA but fails.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Delete TA" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Delete TA" page.
<i>Extension</i>	None
<i>Special requirements</i>	None 

<i>Use case name</i>	ModifyTA
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	1. The initiating actor selects "Modify TA". 2. The initiating actor enters necessary information. 3. The initiating actor clicks "Modify" button. 4. The system validates the input, updates the information in the database, and displays a message indicating the information has been successfully updated.
<i>Exit condition</i>	The information is updated and is saved to the database.
<i>Extension</i>	FailedToModifyTA
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToModifyTA
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to modify account information for a TA but enters invalid information.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Modify TA" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Modify TA" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	CreateStudent
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Create Student".2. The initiating actor enters necessary information.3. The initiating actor clicks "Create" button.4. The system validates the input, saves the information to the database, and displays a message indicating the new account has been successfully created.
<i>Exit condition</i>	The new account is created and saved to the database.
<i>Extension</i>	FailedToCreateCourse
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToCreateStudent
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to create a new account for a student but enters invalid information.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor is redirected to the "Create Student" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Create Course" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	DeleteStudent
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Delete Student".2. The initiating actor selects a student to delete from the list.3. The initiating actor clicks "Delete" button.4. The system deletes the account from the database and displays a message indicating the account is successfully deleted.
<i>Exit condition</i>	The account is deleted from the database.
<i>Extension</i>	FailedToDeleteCourse
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	FailedToDeleteStudent
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to delete account information for a student but fails.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Delete Student" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Delete Student" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ModifyStudent
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	1. The initiating actor selects "Modify Student". 2. The initiating actor enters necessary information. 3. The initiating actor clicks "Modify" button. 4. The system validates the input, updates the information in the database, and displays a message indicating the information has been successfully updated.
<i>Exit condition</i>	The information is updated in the database.
<i>Extension</i>	FailToModifyStudent
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToModifyStudent
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to modify account information for a student but enters invalid information.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Modify Course" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Modify Student" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	ImportStudents
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Import Students".2. The initiating actor selects a CSV file to be uploaded to the server for processing.3. The initiating actor clicks "Import" button.4. The system validates the file, creates accounts for each student in the list, saves the information to the database, and displays a message indicating new accounts have been successfully created.
<i>Exit condition</i>	The new accounts have been added in the database.
<i>Extension</i>	FailedToImportStudents
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToImportStudents
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to import list of students but fails.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor is redirected to the "Import Students" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Import Students" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	AssignStudentsToACourse
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one student can be selected, and at least one course can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Assign Students to Course".2. The initiating actor selects list of students.3. The initiating actor selects a course.4. The initiating actor clicks "Add" button.5. The selected students are assigned to the selected course.
<i>Exit condition</i>	The students are assigned to the course.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	AssignStudentsToAProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one student can be selected, and at least one problem can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Assign Students to Problem".2. The initiating actor selects list of students.3. The initiating actor selects a problem.4. The initiating actor clicks "Add" button.5. The selected students are assigned to the selected problem.
<i>Exit condition</i>	The students are assigned to the problem.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	RemoveStudentFromCourses
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one course can be selected, and at least one student is assigned to that course.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "View Student".2. The initiating actor selects list of courses.3. The initiating actor clicks "Remove" button.4. The selected student is removed from the selected course.
<i>Exit condition</i>	The student is removed from the course.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	RemoveStudentFromAProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one problem can be selected, and at least one student is assigned to that problem.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "View Student".2. The initiating actor selects list of problems.3. The initiating actor clicks "Remove" button.4. The selected student is removed from the selected problem.
<i>Exit condition</i>	The student is removed from the problem.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	AssignTAsToACourse
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one TA can be selected, and at least one course can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Assign TAs to Course".2. The initiating actor selects list of TAs.3. The initiating actor selects a course.4. The initiating actor clicks "Add" button.5. The selected TAs are assigned to the selected course.
<i>Exit condition</i>	The TAs are assigned to the course.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	AssignTAsToAProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one TA can be selected, and at least one problem can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "Assign TAs to Problem".2. The initiating actor selects list of TAs.3. The initiating actor selects a problem.4. The initiating actor clicks "Add" button.5. The selected TAs are assigned to the selected problem.
<i>Exit condition</i>	The TAs are assigned to the problem.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	RemoveTAFromCourses
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one course can be selected, and at least one TA is assigned to that course.
<i>Flow of events</i>	<ol style="list-style-type: none">1. The initiating actor selects "View TA".2. The initiating actor selects list of courses.3. The initiating actor clicks "Remove" button.4. The selected TA is removed from the selected course.
<i>Exit condition</i>	The TA is removed from the course.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	RemoveTAFromAProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, at least one problem can be selected, and at least one TA is assigned to that problem.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "View TA". 2. The initiating actor selects list of problems. 3. The initiating actor clicks "Remove" button. 4. The selected TA is removed from the selected problem.
<i>Exit condition</i>	The TA is removed from the problem.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	CreateProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "Create Problem". 2. The initiating actor enters necessary information. 3. The actor clicks "Create" button.
<i>Exit condition</i>	The new problem is created and saved to the database.
<i>Extension</i>	FailedToCreateProblem
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToCreateProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to create a new problem but enters invalid information.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor is redirected to the "Create Problem" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Create Problem" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	DeleteProblems
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and at least one problem can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "View List of Problems". 2. The initiating actor selects problems. 3. The initiating actor clicks "Delete" button.
<i>Exit condition</i>	The selected problems are removed from the database.
<i>Extension</i>	FailedToDeleteProblems
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	FailedToDeleteProblems
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to delete problems but fails.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Delete Problems" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Delete Problems" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ModifyProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and at least one problem can be selected.
<i>Flow of events</i>	1. The initiating actor selects "View List of Problems". 2. The initiating actor selects a problem. 3. The initiating actor clicks "Modify Problem" button. 4. The initiating actor enters necessary information. 5. The initiating actor clicks "Modify" button. 6. The system validates the input, updates the information in the database, and displays a message indicating the information has been successfully updated.
<i>Exit condition</i>	The information is updated in the database.
<i>Extension</i>	FailToModifyProblem
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToModifyProblem
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to modify a problem but enters invalid information.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Modify Problem" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Modify Problem" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	CreateTrait
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "Create Trait". 2. The initiating actor enters necessary information. 3. The initiating actor clicks "Create" button.
<i>Exit condition</i>	The new trait is created and saved to the database.
<i>Extension</i>	FailedToCreateProblem
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToCreateTrait
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to create a new trait but enters invalid information.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor is redirected to the "Create Trait" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Create Trait" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	DeleteTrait
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "Delete Trait". 2. The initiating actor selects a trait to delete from the list. 3. The initiating actor clicks "Delete" button. 4. The system deletes the trait from the database and displays a message indicating the trait is successfully deleted.
<i>Exit condition</i>	The trait is deleted from the database.
<i>Extension</i>	FailedToDeleteTrait
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToDeleteTrait
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to delete trait for a student but fails.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor is redirected to the "Delete Trait" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Delete Trait" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	ModifyTrait
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "Modify Trait". 2. The initiating actor enters necessary information. 3. The initiating actor clicks "Modify" button. 4. The system validates the input, updates the information in the database, and displays a message indicating the information has been successfully updated.
<i>Exit condition</i>	The information is updated in the database.
<i>Extension</i>	FailToModifyStudent
<i>Special requirements</i>	None

<i>Use case name</i>	FailedToModifyTrait
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to modify account information for a student but enters invalid information.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor is redirected to the "Modify Trait" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Modify Trait" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ViewStudentAnswer
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and at least one student has submitted the answer.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "View Student". 2. The initiating actor selects "View List of Problems". 3. The initiating actor selects "View Answer".
<i>Exit condition</i>	The answer submitted by the student is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	SendFeedback
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and at least one student has submitted the answer.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects "View Student Answer". 2. The initiating actor enters necessary information. 3. The initiating actor clicks "Send Feedback" button.
<i>Exit condition</i>	The feedback is sent to the student.
<i>Extension</i>	FailedToSendFeedback
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	FailedToSendFeedback
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to send feedback to a student but enters invalid information.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Send Feedback" page with an error message indicating why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Send Feedback" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ExportPastCourseData
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	1. The initiating actor selects "Export Past Course Data". 2. The initiating actor selects courses from the list of courses. 3. The initiating actor clicks "Export" button. 4. The initiating actor selects where the file is saved to.
<i>Exit condition</i>	The selected course data is saved to the path the initiating actor selects.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	RemovePastCourseData
<i>Initiating actor</i>	Administrator, Professor
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor clicks "Remove Past Course Data". 2. The initiating actor selects courses from the list of courses.. 3. The initiating actor clicks "Remove" button.
<i>Exit condition</i>	The selected course data is removed from the database.
<i>Extension</i>	None
<i>Special requirements</i>	None

3.2.3 Administrator, Professor, TA Functions

<i>Use case name</i>	ViewListOfStudentsAssignedToACourse
<i>Initiating actor</i>	Administrator, Professor, TA
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and at least one course can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects a course from the list. 2. The initiating actor selects "View List of Students". 3. The system finds all students assigned to the course. 4. The list of students is displayed.
<i>Exit condition</i>	The list of students assigned to the course is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ViewListOfStudentsAssignedToAProblem
<i>Initiating actor</i>	Administrator, Professor, TA
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and at least one problem can be selected.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects a problem from the list. 2. The initiating actor selects "View List of Students". 3. The system finds all students assigned to the problem. 4. The list of students is displayed.
<i>Exit condition</i>	The list of students assigned to the problem is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	ViewStudentProgress
<i>Initiating actor</i>	Administrator, Professor, TA
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects a student. 2. The initiating actor selects a problem. 3. The initiating actor clicks "View Progress" button. 4. The system finds the information of the student and the problem.
<i>Exit condition</i>	The progress of the student is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

3.2.4 Student Functions

<i>Use case name</i>	ViewListOfGenerations
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and has selected a problem.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor views the list of generation created for the problem.
<i>Exit condition</i>	List of generations created for the problem is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	SelectGeneration
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and has selected a problem.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor views the list of generations for the selected problem. 2. The initiating actor selects a generation to view.
<i>Exit condition</i>	The generation is displayed.
<i>Extension</i>	None
<i>Special requirements</i>	None

<i>Use case name</i>	SelectGeneDistribution
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, has selected a problem, and has selected a generation.
<i>Flow of events</i>	<ol style="list-style-type: none"> 1. The initiating actor selects a distribution partition from the histogram.
<i>Exit condition</i>	Plants belong to that distribution is highlighted.
<i>Extension</i>	None
<i>Special requirements</i>	None

Software Requirements Specification Document

<i>Use case name</i>	SelectParentPlant
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and a problem is selected.
<i>Flow of events</i>	1. The initiating actor selects target plants from the list of plants displayed.
<i>Exit condition</i>	Selected plants are marked as parent plants.
<i>Extension</i>	None
<i>Special requirements</i>	None
<i>Use case name</i>	CrossPlants
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in, has selected a problem, and has selected the parent plants.
<i>Flow of events</i>	1. The initiating actor clicks "Cross" button.
<i>Exit Condition</i>	The system generates new generation and redirects user to the "List of Generation" page
<i>Extensions</i>	FailedToCrossPlants
<i>Special requirements</i>	None
<i>Use case name</i>	FailedToCrossPlants
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	User tries to cross plants with invalid selection.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Cross Plant" page and error message is displayed to indicate the reason why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Cross Plant" page.
<i>Extension</i>	None
<i>Special requirements</i>	None
<i>Use case name</i>	SubmitAnswer
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor has logged in and has selected a problem.
<i>Flow of events</i>	1. The initiating actor enters necessary information. 2. The initiating actor clicks "Submit" button. 3. The system validates the input, save the information to the database, and evaluates the answer.
<i>Exit condition</i>	The answer is submitted and saved to the database.
<i>Extension</i>	FailedToSubmitAnswer
<i>Special requirements</i>	None

<i>Use case name</i>	FailToSubmitAnswer
<i>Initiating actor</i>	Student
<i>Participating</i>	None
<i>Entry condition</i>	The initiating actor tries to submit answer but fails.
<i>Flow of events</i>	1. The initiating actor is redirected to the "Submit Answer" page and error message is displayed indicating the reason why the operation is unsuccessful.
<i>Exit condition</i>	The initiating actor remains at the "Submit Answer" page.
<i>Extension</i>	None
<i>Special requirements</i>	None

3.3 Performance Requirements

The number of concurrent users and requests per second will be ultimately determined by the physical resources (RAM, disk space, processor speed) of the application server on which the application will be installed as well as the geographic location of the server and its connected internet connection performance.

On the assumptions that the server will be located on campus, the majority of clients will access the server on campus or within a general vicinity of the campus, and the server specifications outlined in section 2.4.1, an estimated average of five requests per second will be handled without affecting the responsiveness of the system.

Disk space requirements will be met in order for the system to function properly. The system will use an estimated 30 megabytes within the MySQL server with per class with specifications of 300 students, 100 plants, 64 genes and 50 crosses. The estimate will increase or decrease and is dependant on the specifications given to a problem as well as class sizes. Each request to the server will indirectly imply one or more operations to the database.

3.4 Logical Database Requirements

All the information stored in the database will be accessed and modified only through the web-based user interface provided by the software.

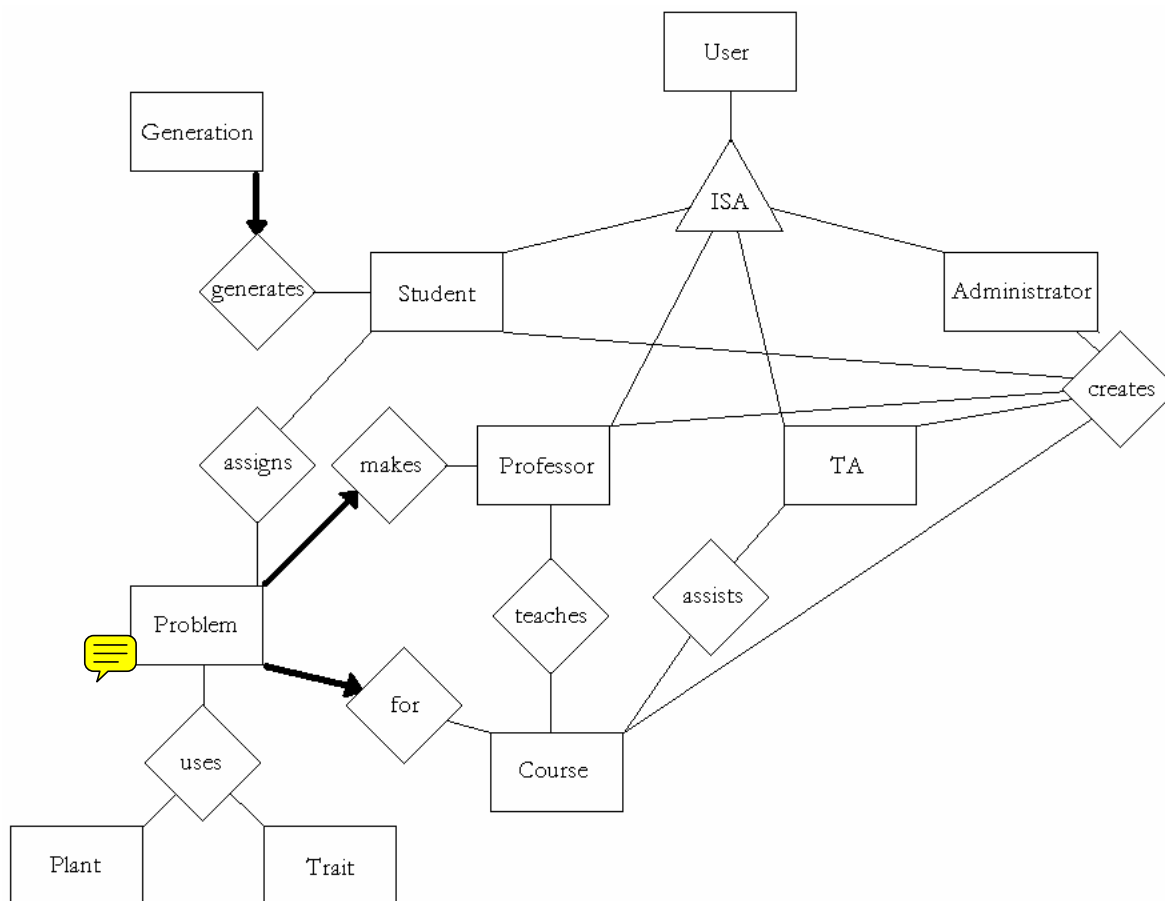


Figure 12: ER Diagram

3.5 Software System Attributes

3.5.1 Reliability

If an error occurs and forces the application to terminate, an error log will be generated in order to trace back to the origin and cause of the error.

3.5.2 Availability

The application will be available online at all times. Access to the application will be through a URL address. It will be assumed that the application will run on the administrator's server, so the actual availability will depend on the server's availability.

Data recovery will be handled by the application's database server. In the case of a server restart, the application will have to be restarted by the administrator unless it is manually specified to run upon server restart.

3.5.3 Security

The database files containing administrator, professor, TA and student data will not be directly accessible by any of the four under any circumstances.

The administrator, professors, TAs, and students will require a password to enter the system online. The administrator will have the highest level of access, which includes the ability to assign user information (username, password) to any professor, TA or student. Professors will be able to assign user information to a professor, TA or student. Passwords will not be encrypted since permission to change or reset passwords will be limited.

Any errors in the application will be logged in a file with timestamps.

3.5.4 Maintainability

Maintenance will be minimal and mainly consist of any backup and repair done by the administrator. The administrator and professors will be allowed the optional action of removing past course data. Past course data will contain problem parameters and student interactions in the system.

3.5.5 Portability

This application will be written in PHP, which is a cross-platform language that allows portability. It also will need a MySQL database server for the storing of student information and transactions. Specifically, it will be able to run on a Mac UNIX platform server. The application will work with browsers that support HTML 4.0 and CSS 1.0.

4. Change Management Process

For any changes requested, either by the client, instructor, or other team members, the following process must be followed:

1. Any requests for change will be presented as a hard-copy request containing a detailed description of the change. This hard-copy can be made by either the person requesting the change or any member of the team who is present at the time of the request.
2. This hard-copy request will be presented to the risks manager who will assess the impact of implementing the request.
3. If the request does not hinder other functions of the program, then the request will be passed to the progress manager. The progress manager will assess the feasibility of including the request with the implementation on the current timeline.
4. Finally, if the request is feasible on the current timeline, then the request will be presented to the project manager for final approval.
5. If necessary, the contact manager will notify the person requesting the status of the request.

5. Document Approvals

By signing in the space provided below, I hereby agree to adhere to the requirements and specifications outlined in this document.

Client's Name

Signature

Date

Team Lead

Signature

Date

6. Supporting Information

6.1 Client Proposal

Expansion of 'Greengene', an Interactive Web-based Plant Breeding Simulation, to Simulate Quantitative Genetics

Facility: UBC Botanical Garden and Centre for Plant Research
Faculty of Land and Food Systems
323 Macmillan Building
UBC

Client: Dr. Andrew Riseman (ariseaman@interchange.ubc.ca)

Introduction:

Greengene is simulation software used to introduce students to some of the concepts relevant to plant breeding and sexual reproduction. Specifically, the program is currently designed to simulate qualitative genetics with respect to dominance (i.e. intra-allelic interaction) where outcomes are based on either no dominance or dominance, linkage (i.e. inter-allelic interaction) where segregation between 2 traits is not random (i.e. not equal proportions of each trait in subsequent generations), and epistasis (i.e. inter-loci interaction) where the interaction of 2 genes affect the expression of a single trait, either with altered ratios or the expression of a new phenotype not observed in the parents.

Currently, Greengene is written in an open-source language and runs as a cross platform server application. I would like to expand Greengene's capacity by including a simulation of quantitative genetics.

Current Program Description:

Each student is given a unique and individual problem (i.e., addressing only qualitative genetics), from which Greengene produces progeny and subsequent generations through adhering to pre-set instructor parameters specific to that student. These parameters are unknown to the student and are, in fact, the objective of the exercise; to determine dominance relationships within a trait, to identify genetic distances between traits, and to identify the presence and form of epistasis.

Greengene currently allows for the manipulation of three traits (through the effects of potentially four genes if epistasis is engaged) simultaneously, acting independently or together, as set-up in the instructor's parameters. For example, Trait 1, Plant Height, may have the phenotypes short, intermediate, or tall. Trait 2, Disease Resistance, may have resistant, moderate, or susceptible as the phenotypes. Trait 3, Flower Color, may have the phenotypes, white, pink, or red for any individual (i.e. a virtual plant within a progeny). Trait 4 may act as an epistatic gene to Trait 3 where Flower Color is affected and expressed as altered ratios or new phenotypes, depending on the allelic configuration of the progeny. Any individual will possess only one phenotype of each of the three traits. Therefore, individual 1 could be tall (Trait 1), moderate (Trait 2), and white (Trait

Software Requirements Specification Document

3) while individual 2 may be short, susceptible, and red. The number of phenotypes is determined based on the dominance parameter and the presence/absence of epistasis. If Trait 1 has dominance (w/o epistasis), only 2 forms can be produced, red or white. If no dominance is present, all 3 forms are possible.

Initially, a student is assigned a 'crop' and given the 1st 50 progeny from a self-pollination of an initial plant created based on the instructor's parameter inputs. The instructor generates this 1st progeny group. The students then need to make a series of virtual crosses from this population to generate subsequent generations. Based on analyses of the resulting progenies, the student is able to determine the dominance relationships within a trait, determine linkage distances between any 2 traits, and identify the presence and type of epistasis.

New Code Description

Quantitative Genetics: This enhancement should enable a separate (i.e., non-qualitative) problem for student access. It will simulate the function of two independent (i.e., non-linked) traits via the action of multiple genes (i.e., quantitative genetics). Problem set-up should involve inputting the trait descriptions (i.e., yield, disease resistance), the number of effective factors for each trait (i.e., number of genes), parent #1 mean values, parent #2 mean values, variance values for selected populations, and heritability values. Additional information on quantitative genetics, variance calculations, and heritability calculations are available upon request.

Further details will evolve from client meetings.

Non-Functional Requirements:

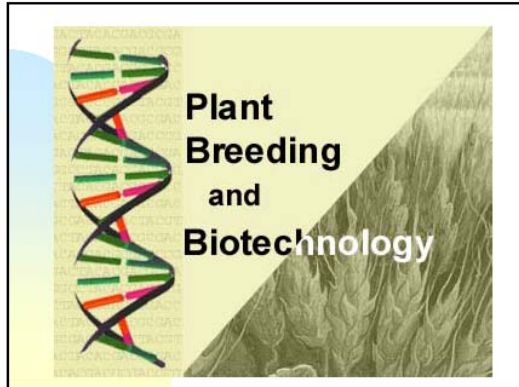
- I. The new code is designed to enhance the functionality of Greengene in the teaching of plant genetics.
 - A. The new code will be minimally integrated into the existing code only when needed.
 - B. The new function will be used by teachers to setup quantitative genetics problems for students, and be used by students to simulate quantitative genetics in plants.
 - C. The genetics logic will come from new code.
 - D. The program is protected by passwords and the new code should continue to use them.
 - E. The program will be accessed through a web interface.
 - F. The program will have built-in Help screens detailing to procedures associated with any particular activity.

Software Requirements Specification Document

Further Information:

1. Write for Linux server only
2. Prefer no Java or compiled code but is acceptable as long as #8-13 below are fulfilled.
3. Continue use of the interpreted language used to write Greengene
4. Can be database or flat files
5. Must install and run on our server, a Mac G4.
6. Error log capable (messages either added to log or displayed on screen)
7. Install instructions need very high degree of detail
8. For demonstration, passing grade is based on successful installation on our machine
9. All groups must install sequentially without moving other programs; must create own directory
10. Self-installing/extracting Zipped file
11. All groups must use common software (only one machine)
12. No dedicated IT support; must be drop and install.

6.2 Sample Lecture Slides (Quantitative Genetics)



Quantitative Genetics Terminology:

Qualitative traits – phenotypes differ by quality.
Exp.: red/white eyes, wrinkled /round peas.

Quantitative traits – phenotypes differ in quantity:
height, yield, flower/fruit size, disease resistance

Polygenic inheritance: more than one gene operating to make a single phenotype.

Phenotypes in qualitative inheritance are all distinct from each other – **discontinuous variation**.

Traits in quantitative inheritance are not as easily categorized into distinct classes – **continuous variation**.

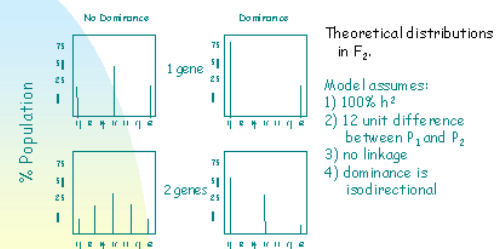
Quantitative Inheritance

- Definition: when genes produce metrical effects of a trait of interest.
- Quantitative traits display continuous variation; controlled by many genes, each with a small effect
- Controlled by many segregating loci whose alleles affect the observed phenotype
- However, both quantitative and qualitative traits follow the same laws of inheritance
 - e.g. dominance, epistasis, linkage, pleiotropism

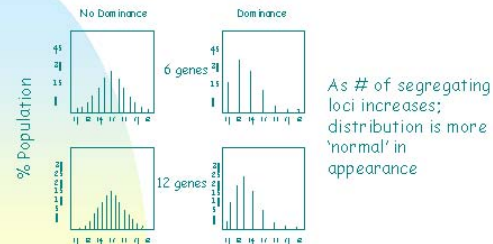
General Considerations:

- Normally assume genes are nuclear
- Normally quantitative traits are more affected by environmental effects than are qualitative traits
- Normally we see continuous variation; however, continuity can be as much a consequence of environmental influence as it is a consequence of the number of segregating loci.

Theoretical Distributions:



Theoretical Distributions:



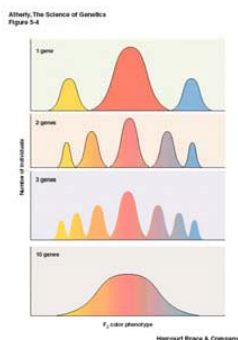
Gene action:

As the number of loci controlling a trait increases, so do the number of genotypes and phenotypes

When the loci produce a cumulative effect, the variation becomes continuous.

Variation increases in a predictable manner

Example: If two gene pairs were involved, five F₂ phenotypes in a 1:4:6:4:1 ratio would be expected.



A mathematical rule can be developed that relates the number of the phenotypes and the genes involved:

$$\text{Number of phenotypes} = (\# \text{ genes} \times 2) + 1$$

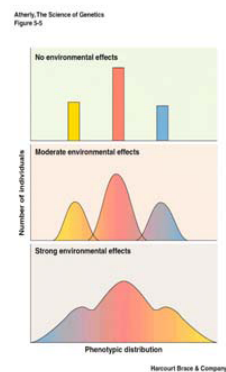
Or, you can predict the number of phenotypic classes by looking at the number of the genes operating (if it's known)

2 genes yield 5 phenotypes
 3 genes yield 7 phenotypes
 4 genes yield 9 phenotypes

Number of genes involved in a quantitative inheritance can also be estimated using the equation:

$$N = \frac{(\bar{P}_1 - \bar{P}_2)^2}{8(V_{F_2} - V_E)}$$

Effect of the environment:
 Most quantitative traits are subject to environmental effects, which can cause individuals with same genotypes to have slightly different phenotypes.



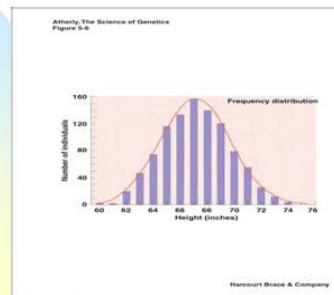
History #1:

Josef Kölreuter in early 19 century crossed tall and dwarf tobacco plants:

- F_1 : all intermediate in height;
- F_2 showed continuous variation in height from tall to short

Results supported **biometricians** theory that all traits were determined by many genes each having small effects

F_2 phenotypes showed "normal distribution" or bell curve distribution.



History #2:

William Bateson (late 1800s - early 1900s): 'Mendelian'

- Multiple-factor or multiple-gene hypothesis (the earliest form of polygenic inheritance theory)
- Quantitative traits were controlled by Mendelian 'genes'.
- Each genes contributes to the same phenotype in a small but quantifiable way.

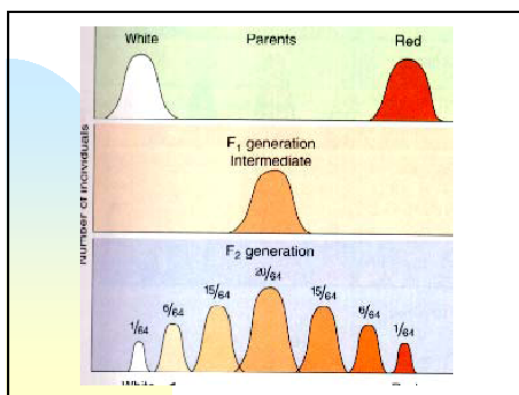
i.e. Quantitative traits are controlled by multiple genes acting together

How to reconcile quantitative traits theory with Mendel's theory?

Nillson-Ehle (1909) made a cross between red-seeded wheat and white-seeded wheat:

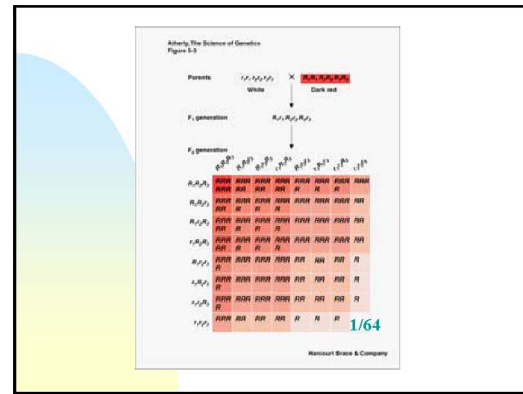
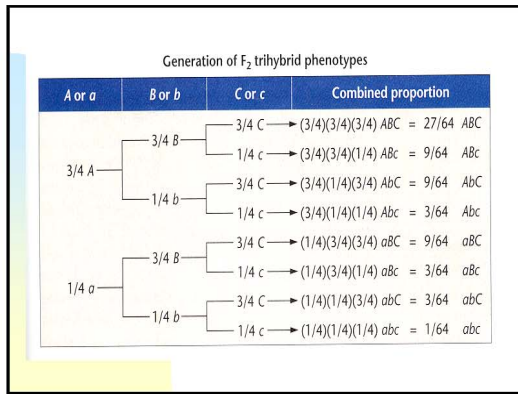
F_1 : All plants showed intermediate seed color.

F_2 : continuous variation in kernel colors between red and white.



Among the continuous expression of colored grain phenotypes, about 1/64 showed completely white kernels (i.e. one of the parental phenotypes), and ~ 1/64 showed dark red kernels (i.e. the other parental phenotype)

Software Requirements Specification Document



1/64 matches the phenotype ratio produced by the combination of three recessive gene in a trihybrid cross.

Conclusion:

Quantitative inheritance also follows Mendel's rule for inheritance;

Wheat grain color is a quantitative trait governed by three pairs of genes.

The cross of white and red wheat involves three gene pairs:



The darkness of red color is proportional to the number of R.

The darkest class, with 6 R alleles, shows the same color as the dark red parental and is one of two classes that breed true.

This means that the effect of allele R is additive

- additive alleles effects are another feature of quantitative traits.

If the contribution of multiple genes to a quantitative trait is additive, then how do we estimate the contribution of each gene?

For example:

Plants with genotype as $A_1A_1A_1A_1A_1A_1$ produce 7 cm high progeny, while those with $A_2A_2A_2A_2A_2A_2$ produce 5 cm height.

(Note: sometimes one of the extreme genotypes produces a base value or base line for the phenotype. In this example, there are no progeny with 0 cm height!)

How much does each allele contribute?

- 1) Calculate difference between the two extremes: $7 - 5 = 2$ cm
- 2) This difference is caused by 6 additive alleles;
- 3) Therefore, each additive allele causes an increase of $2/6 = 0.33$ cm above the base height.

To predict the height of a plant with genotype $A_1A_1A_2A_2A_1A_2$:

$$5 + 3(0.33) = 5.99 \text{ cm height.}$$

But, don't forget about the environmental effects...

Genetic Structure of a Population

- Genetic structure
 - ◆ Self-pollinated vs. Cross-pollinated crops
 - ◆ Homozygous vs. Heterozygous
 - ◆ Homogeneous vs. Heterogeneous
- Allele frequencies
- H-W equilibrium
- "Fixing" traits in a population

General Considerations

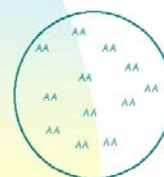
- Self-pollinated crops
 - ◆ Heterogeneous population of homozygotes
- Cross-pollinated crops
 - ◆ Heterogeneous population of heterozygotes

Changes in genotype and allele frequency

- Plant breeding seeks to move the genetic composition of a population toward a predetermined end point
- Accomplished through selection

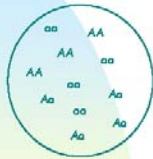
Allelic Frequency

- Genotype Frequency: proportion or % of individuals with a specific genotype



Frequency
 $f(AA)=1$

Allelic Statistics



$$\begin{array}{ccc} \underline{AA} & & \underline{Aa} & & \underline{aa} \\ n_1 & + & n_2 & + & n_3 = N \\ D & & H & & R \end{array}$$

$$f(AA) = n_1/N = D$$

$$f(Aa) = n_2/N = H$$

$$f(aa) = n_3/N = R$$

$$D+H+R=1$$

Hardy-Weinberg Law (1908)

- Predicts probable genetic structure of a population
- In it's simplest form, the law states that 2 alleles, A and a, having frequencies p and q respectively, shall be represented in different genotypes with proportions equal to:

$$p^2 AA + 2pq Aa + q^2 aa$$

based on a specific set of assumptions

H-W Law Assumptions:

- ♦ Population is infinitely large
- ♦ Only random mating; no self-fertilization
- ♦ No selective advantage (no selection)
- ♦ No mutation, migration, or random genetic drift
- ♦ No linkage

Derivation of the H-W Equation

$$\begin{array}{c} A = p \\ a = q \end{array} \quad \begin{array}{c|c|c} & p & q \\ & A & a \\ \hline p & A & \begin{array}{c} AA \\ p^2 \end{array} & \begin{array}{c} Aa \\ pq \end{array} \\ \hline q & a & \begin{array}{c} Aa \\ pq \end{array} & \begin{array}{c} aa \\ q^2 \end{array} \end{array}$$

$$\begin{array}{ccc} AA & Aa & aa \\ p^2 & 2pq & q^2 \end{array}$$

Testing Equilibrium

For allele A:

$$p = \frac{D + 1/2H}{N} \quad \text{or} \quad \frac{2D + H}{2N}$$

Where:

p = Proportion or frequency of allele 'A'
D = # of homozygous dominant individuals
H = # of heterozygous individuals
N = Total population

Testing Equilibrium

For allele 'a':

$$q = \frac{R + 1/2H}{N} \quad \text{or} \quad \frac{2R + H}{2N}$$

Where:

q = Proportion or frequency of allele 'a'
R = # of Homozygous recessive individuals
H = # of heterozygous individuals
N = Total population

Allelic Statistics

Gene Frequency (allele frequency)

Genotype frequency in each population is determined by the allele frequency in the previous generation of the population

In the absence of disruption, gene and genotype frequency will remain constant from generation to generation

Allele Frequency Calculation

Example:

	Resistant	Moderate	Susceptible
Genotype	AA	Aa	aa
#	10	60	30
Genotypic freq	0.1	0.6	0.3

$$f(A) = p = \frac{(2 \times 10) + 60}{100 \times 2} = 0.4$$

$$f(a) = q = \frac{(2 \times 30) + 60}{100 \times 2} = 0.6$$

$$p + q = 1$$

After 1 generation of H-W conditions
Genotypic/Allelic frequency # 1:

$$p^2 + 2pq + q^2$$

$$\frac{AA}{D'} = (0.4)^2 = 0.16$$

$$\frac{Aa}{H'} = 2 \times 0.4 \times 0.6 = 0.48$$

$$\frac{aa}{R'} = (0.6)^2 = 0.36$$

Where:
D', H' and R'
represent the
proportion in the
new population

After 1 generation of H-W conditions
Genotypic/Allelic frequency # 2:

		p	q
		A (.4)	a (.6)
p	A (.4)	AA (.16)	Aa (.24)
q	a (.6)	Aa (.24)	aa (.36)

$$\text{In equilibrium: } .16AA + .48Aa + .36aa = 1$$

Population not in H-W equilibrium:

	AA	Aa	aa	
Obs #	40	80	80	
Obs freq	.2	.4	.4	$f(A) = p = 0.4$
Calc freq	.16	.48	.36	$f(a) = q = 0.6$
Calc #	32	96	72	

Test with Chi-square statistic:

Test with 1 df because we lose 1 df to normal limitation and 1 df for using estimated ratio

$$\chi^2 = 5.56 \text{ w/1df; } p < 0.05; \text{ reject } H_0 \text{ (pop not in HW eq)}$$

Population not in H-W equilibrium will return to equilibrium after one generation of random mating

		.2	.4	.4	
		AA	Aa	aa	
.2	AA	AA x AA (.2)(.2) = .04	AA x Aa (.2)(.4) = .08	AA x aa (.2)(.4) = .08	Random mating = all possible combinations
.4	Aa	Aa x AA .08	Aa x Aa .16	Aa x aa .16	
.4	aa	aa x AA .08	aa x Aa .16	aa x aa .16	

Can Summarize Crossings:

Mating	Freq of mating	Progeny		
		AA	Aa	aa
1 AAxAA	.04	.04		
2 AAxAa	.16	.08	.08	
1 AaxAa	.16	.04	.08	.04
2 AAxaa	.16		.16	
2 Aaxaa	.32		.16	.16
1 aaxaa	.16			.16
w/in 1 generation		.16	.48	.36
Obs freq= calc freq				

Population Statistics

Effect of selection on allelic frequency:

$$q_n = \frac{q_0}{1 + nq_0}$$

Where:
 q_n = allelic freq after n^{th} generations
 q_0 = original allelic frequency
 n = number of generations

Example: changes in allelic frequency after 1 generation of selection away from susceptibility

$$q_n = \frac{q_0}{1 + nq_0}$$

Note: use allelic frequency of the allele selected against

$$q_1 = \frac{0.6}{1 + (1 \cdot 0.6)}$$

$$q_1 = 0.37$$

Therefore $q = 0.37$ and $p(1-q) = 0.63$ after one generation of selection

After 2 generations of selection against susceptibility:

$$q_n = \frac{q_0}{1 + nq_0}$$

$$q_2 = \frac{0.6}{1 + (2 \cdot 0.6)}$$

$$q_2 = 0.27$$

After 3 generations:
 $q_3 = 0.21$ and $p_3 = 0.79$

Genotypic Frequency: Selection for Recessive Traits

1 generation to identify:

F_2	AA	Aa	aa
	1/4	1/2	1/4

# Genes	% F_2 w/aa	Minimum Pop Size
1	$(1/4)^1$ or .25	10
2	$(1/4)^2$ or .0625	46
3	$(1/4)^3$ or .015	190
4	$(1/4)^4$ or .004	765
n	$(1/4)^n$	

Genotypic Frequency: Selection for Recessive Traits

Minimum Population Size:

$$n = \frac{\ln(1-P)}{\ln(1-p)}$$

Where:
 n = population size
 P = probability
 p = freq of genotype

So for aa:

$$n = \frac{\ln(1-.95)}{\ln(1-.25)}$$

n = approx 10 plants

**Genotypic Frequency:
Selection for Dominant Traits**

2 generations to identify:

F_2	$\frac{AA}{1/4}$	$\frac{Aa}{1/2}$	$\frac{aa}{1/4}$	Need to self A- and select non-segregating parents → Progeny Test
	3/4			

So 3/4 can be divided
into 1/3 AA and 2/3 Aa

$$n = \frac{\ln(1-.95)}{\ln(1-.33)}$$

$n = \text{approx } 7 \text{ plants}$