**CS3716 - Assignment 1**

**System Use Cases**

Friday, September 19, 2014

Christopher Doyle, 200833176

Kayla Blain, 201105103

Instructor: Adrian Fiech

USE CASE #1 – Initializing Group Parameters

Description:

In this case the instructor will use the system to setup his/her classes and group parameters. The Instructor has the choice to have groups filtered by student’s grades, preference for partners and/or self-evaluation of relevant assignment skills. Once the instructor has specified the criteria on how groups will be created, the student will then be able to enter all of their personal data.

This use case is beneficial because it allows the client to interact with the system and make sure that it works in the way that they want it to. This will also identify potential errors in programming and functionality. It also allows the client to manipulate the system, showing us how user friendly our program is.

Actors:

Instructor/Professor

Basic Flow:

1. The instructor enters the class list.
2. Instructor initializes assignment.
3. Instructor enters maximum number of group members
4. Instructor enters maximum number of choices that a student can make for preferred partners
5. Instructor enters deadline for student input.
6. The system is ready for student preference input.

Alternative Flow:

1. The instructor knows which students he wants and does not want to pair together:  
   4.1. The instructor enters those student pairings that he wants/doesn’t want.  
   5.1. Instructor enters deadline for student input  
   6.1. The system is ready for student input
2. The Instructor wants to balance groups based on students’ evaluation of skills:  
   4.2. The instructor sets self-evaluation criteria for the student to fill out base on

main principles of assignment (i.e. Object Oriented Programming, experience

with Java, C++, Prolog, etc…)   
5.2. Instructor enters deadline for student input  
6.2. The system is ready for student input

1. The Instructor wants to balance groups based on student’s grades  
   4.3. The instructor tells the system that it wants the groups together based on

grades (i.e. highest grades to lowest grades, have every group equal to a

similar average, etc…)  
5.3. Instructor enters deadline for student input  
6.3. The system is ready for student input

1. The instructor wants to balance groups based on all possible criteria.   
   4.4. The instructor enters those student pairings that he wants/doesn’t want.  
   5.4. The instructor sets self-evaluation criteria for the student to fill out base on

main principles of assignment (i.e. Object Oriented Programming, experience

with Java, C++, Prolog, etc…)   
6.4. The instructor tells the system that it wants the groups together based on

grades (i.e. highest grades to lowest grades, have every group equal to a

similar average, etc…)  
7.4. Instructor enters deadline for student input  
8.4. The system is ready for student input

Preconditions:

1. User interface was created.
2. User interface includes fields for student preferences that the professor can choose to use:
   1. Self-Evaluation
   2. Partner preference
3. System is programmed to read students grades based on the class list he implements.

Post conditions:

1. Produces a form for students to enter data based on Instructors specifications and assignment requirements
2. Allows students a fair outcome for partner generation based on allotted time available and skill set

Use Case #2: Acquiring Student Data

Description:

In this use case students are given a chance to use the system individually. Each student will input the data required for the system to work properly. This data includes things such as class schedule, personal schedule and/or activities, transcript, etc. Once that data has been collected, the system will assign each student a score based on the provided information. The student will then be asked to specify students that they have a desire to work with as well as students that they wish to avoid working with. After all the students have been given a chance to use the system and input their data, the system will create a list of the assignment groups. The list will be based on the provided data, as well as any other relevant constraints and specifications that may have been outlined by the instructor.

This use case is beneficial to the overall project because it will allow all the students to gain some familiarity with the system before it is completed as well as updating the system with all the required information from the students. It might also help us identify some errors with the system. These errors include:

* Accepting bad/wrong information from students (i.e. accepting ‘8q’ as a grade)
* Not taking into account all the provided data ( i.e. scoring students strictly on grades)
* Assigning scores to students that don’t make sense (i.e. perhaps the range is too large)

Overall, this use case will provide us with essential information about the system and help us move forward with the project.

Actors:

The individual students that are registered for the course.

Main Path:

1. Student opens up the system.
2. Student selects their desired university, class, and assignment.
3. Student inputs his/her strengths and weakness and any information which they feel is relevant to the assignment. (Their skillset, class schedule, personal activities)
4. Student provides their individual grades to the system via transcript.
5. A score is assigned to the student based on 3 and 4 (above).
6. Student enters names of students (3-5) that they wish to be partnered with.
7. Student enters names of students (3-5) that they wish to avoid being partnered with.
8. System is ready to create the list of assignment groups.

Alternate Paths:

1. The system already knows the data that it must acquire from the student:

4.1. System prompts the student to input the required data.

5.1. Student provides their individual grades.

6.1. A score is assigned to the student based on 4.1 and 5.1.

7.1. Student enters names of students (3-5) that they wish to be partnered with.

8.1. Student enters names of students (3-5) that they wish to avoid being partnered

with.

9.1. System is ready to create the list of assignment groups.

1. The system already contains a database of the individual student transcripts(grades):

5.1. The database grades are paired with the student provided data form 4.

6.1. A score is assigned to the student based on 4 and 5.1.

7.1. Student enters names of students (3-5) that they wish to be partnered with.

8.1. Student enters names of students (3-5) that they wish to avoid being partnered

with.

9.1. System is ready to create the list of assignment groups.

1. Students are not asked to provide lists of desirable/undesirable partners because the instructor wishes that certain students work together and that others don’t:

7.1. Students are informed that their instructor has chosen their partner(s) and/or

that they will not be working with certain individuals.

* 1. Those students the instructor has not specified partners for, will be randomly paired among other students that are also unrestricted.

9.1. System is ready to create the list of assignment groups.

Preconditions:

1. The student must be registered for the University.
2. The student must be registered for the course.
3. The student acknowledges that he/she will provide information that is valid and true, to the best of his/her knowledge.
4. Instructor should have indicated whether or not they have decided to pair some students himself/herself and/or if certain students are not to be paired together.
5. Instructor must have provides the grades of the individual students to the system database if the students won’t be asked to this themselves.

Post-conditions:

1. Provides a list of the assignment groups.
2. Verify that the system has correctly assigned all students to a group.
3. Verify that the end result meets the outlined constraints.
4. Check that both students and the instructor are satisfied with the groups.

USE CASE #3 – Generating Group List

Description:

In this use case the instructor will use the system to generate the list of assignment groups. The system should already be loaded with any required student information as well as grades. The grades will either be provided by the student or by the instructor. The instructor will then ask the system to produce the list of groups and verify that it is sufficient for what he/she wants.

This use case is beneficial because it allows us to verify the functionality of the system. Once the list of groups has been created, we can check and see that the groups are organized according to the outlined restrictions and specifications. The system should generate the best possible group outcomes and be able to adjust them according to alternative paths.

Actors:

Instructor/Professor

Basic Path:

1. The instructor starts up the system.
2. The system generates a list of groups.
3. Instructor approves the groups.
4. Students are informed of their group partners.

Alternative Paths:

1. The instructor overrides a student pairing  
   3.1. The instructor enters those student pairings that he wants/doesn’t want.  
   4.1. Instructor uses system to generate a new list.

5.1. Instructor approves the groups.

6.1. Students are informed of their group partners.

1. The system generates a student pairing whose timetables’ conflict.

3.1. Instructor identifies the error in the pairing.

4.1. Instructor updates the system such that those two students cannot be paired

together.

5.1. Instructor uses the system to generate a new list.

6.1. Instructor approves the groups.

7.1. Students are informed of their group partners.

1. The groups that the system generates are not properly balanced:  
   3.1. Instructor notices that the groups are not balanced based on either the size of

the groups or the skillset/cumulative score of the groups.

* 1. Instructor asks us to come up with a new method of scoring.

5.1. The system is updated with a new scoring method.

6.1. Instructor uses the system to generate a new list.

7.1. Instructor approves the groups.

8.1. Students are informed of their group partners.

Preconditions:

1. The system is already equipped with all the student information.
2. The system is already equipped with the database of the student grades (if necessary).
3. The instructor has already provided any restrictions on student pairings to the system.
4. The instructor has been familiarized with the system prior to starting it.

Post conditions:

1. The list of groups that was generated is acceptable.
2. The list of groups is consistent with the constraints and specifications outlined by the instructor.
3. The groups that the system generates are balanced accordingly.

the instructor and the students are satisfied with the groups.