

**CZ2002**

**Object Oriented Programming and Design**

**Lab Assignment Report**

**Student Course Registration and Mark Entry Application (SCRAME)**

|  |  |
| --- | --- |
| **Name:** | **Matriculation No.:** |
| **Cheok Jia De** | **U1121561B** |
| **June Quak Ren Feng** | **U1121476E** |
| **Sri Hartati** | **U1123011H** |

**Submission Date: 12 November 2012**

**Table of Contents**

1. Class Diagram 1

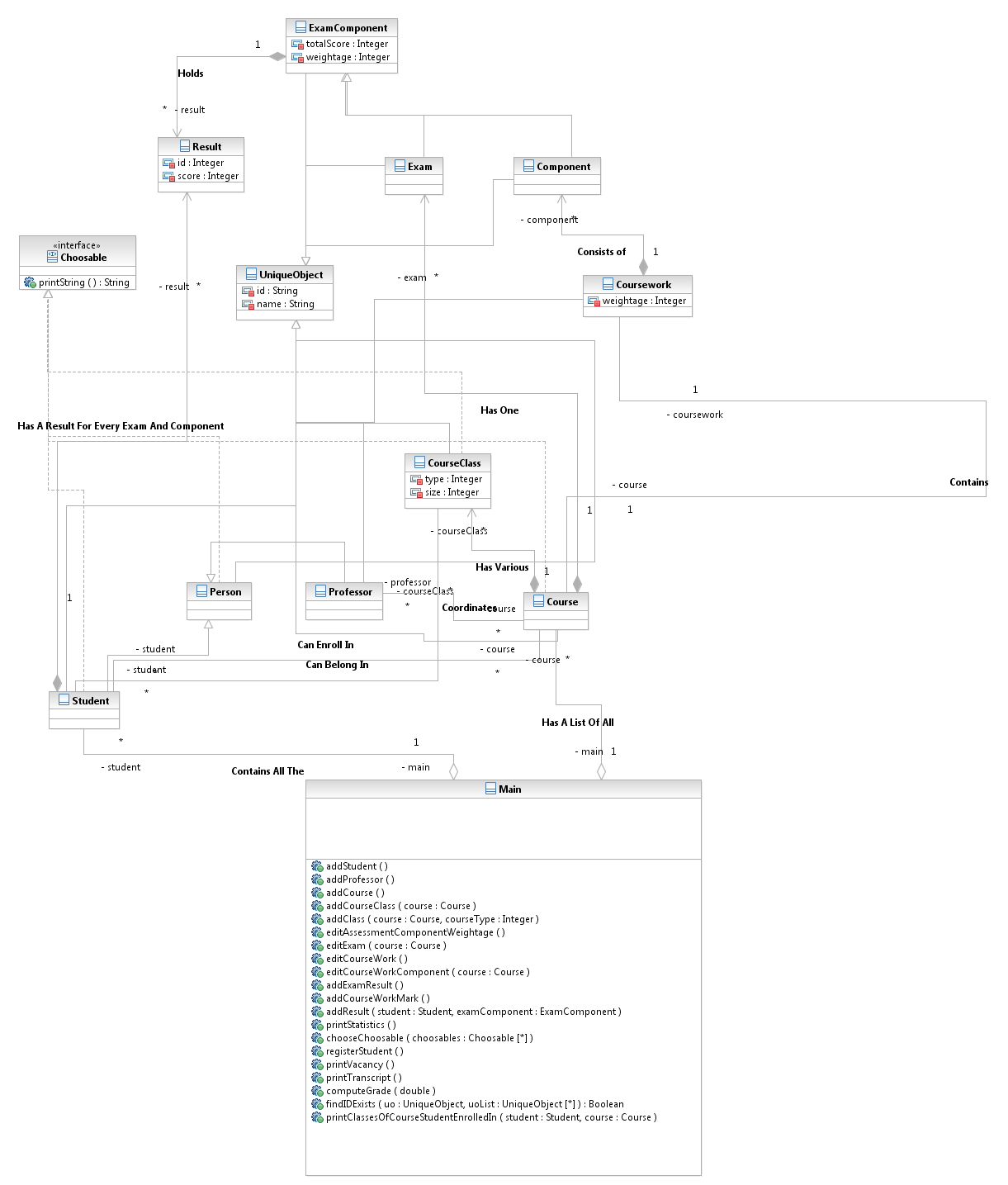
2. Sequence Diagram for ‘Print Student Transcript’ 2

3. Design considerations & use of OO concepts 2

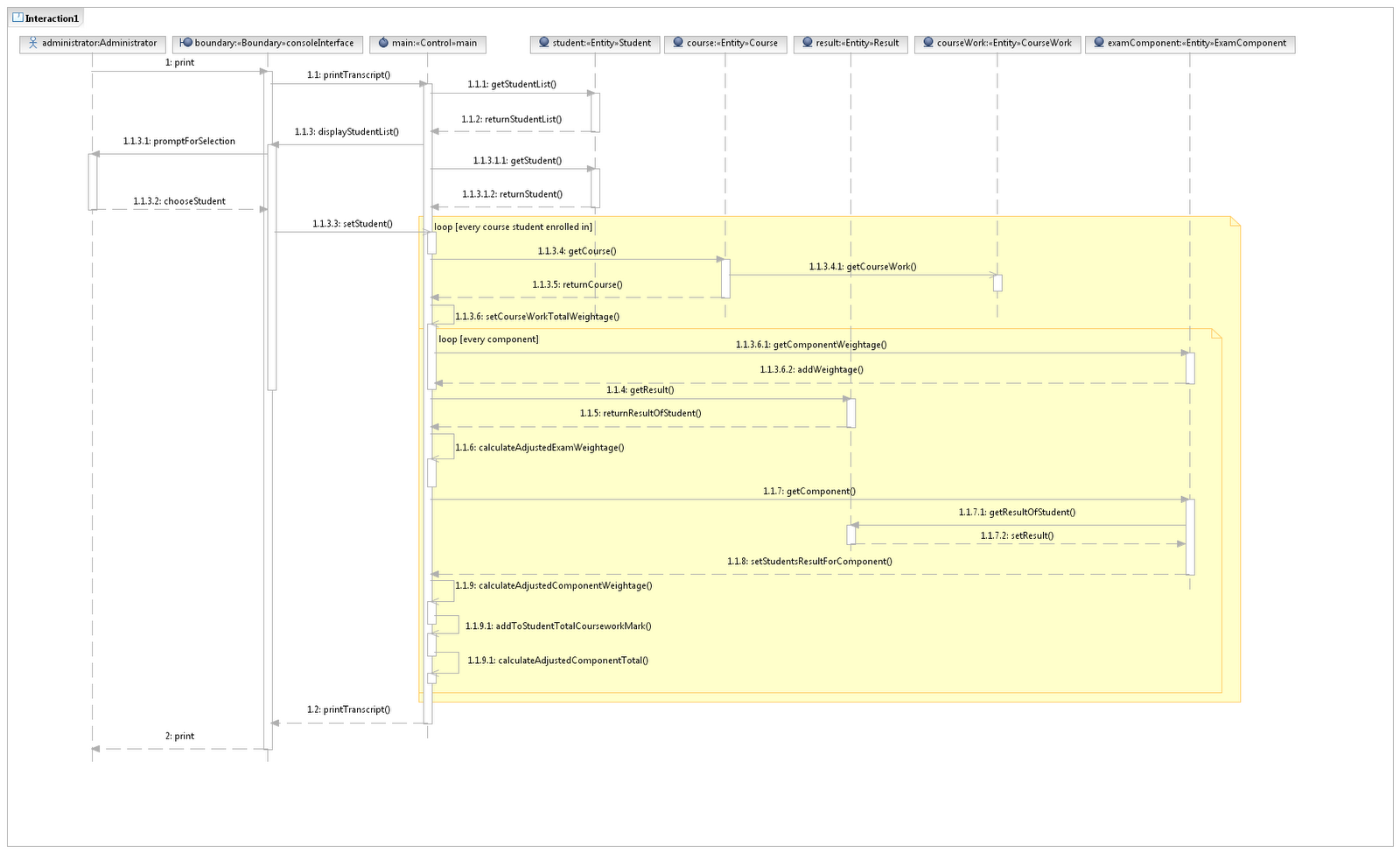
4. Test Case 4

5. Declaration of Original Work for CE/CZ2002 Assignment 9

Class Diagram



Sequence Diagram for ‘Print Student Transcript’



Design considerations & use of OO concepts

As the focus of this course is object oriented programming, we have decided to use an object database(db4o) for persistence of our objects as this would preserve the uniqueness of our objects without needing a unique key attribute for each object. Using an object database has also allowed us to focus on the logic of the program as opposed to wasting time on data management. Also this would allow us to not be accused of plagiarism as any code using a file reader would likely trigger turnitin’s plagiarism algorithms.

Due to this course also focusing on OOP and not on data entry, we have decided to minimized the amount of data needed to create an object and instead just have a name an ID. Requiring less attributes in an object will also allow us to demonstrate the program in a timely fashion and still showing the capabilities. Furthermore, adding additional attributes would be easy to implement in the future if required as there is full separation between entity and business logic.

Although the recommended data structure for an aggregation is a set due to having only unique elements, we have used a list as the set interface does not provide a method for getting an element by position due to the nature of a set. Thus, we have included constraints to prevent duplicate objects.

For our course classes we chose to use a single class to represent any class with a type attribute defining the class type. This allows us to easily add or remove class types as required instead of using Java reflection or creating 3 sub classes that have the only difference in the value of a type attribute and class name.

For validation and invalid input, we have tried hard to disallow invalid input as opposed to handling invalid input. For example, when choosing a single unique object, the user can select the object from an easy-to-use list or search for the object by predefined attributes.

In the first case, a single method lets the user select a single object from any array of objects that implement a <choosable> interface. If the user decides not to select any object, the method returns null and the user is taken back to the main menu.

For validity of input like limiting names to [aA-zZ ], we have decided that as NTU is a globalized university, we will not restrict anything to an Anglicized preconceived notion of naming but instead allow any input supported by Unicode other than a carriage return. The only exception to this is NTU's internal identification like course code and student ID.

For exam and coursework components, due to them having mostly similar attributes and functions, we have extended them from an abstract <ExamComponent> class. The only difference in the 2 classes is that a single exam is owned by a course while many coursework components are owned by the coursework.

Furthermore, for exam and coursework, we have chosen to use a weighted percentage system to compute the final percentage as opposed to using a hard percentage of 100. We feel this be more intuitive for users as in test cases suggested, if a percentage of 50% was entered for the coursework, the user had to make sure the coursework components added up to 50%. Whereas in a weighted percentage, the user just needs to make sure they are in the correct ratios. Our program will take care of making sure they add up correctly in terms of the weights given.

For our result class, it has a one-to-many relationship with both the the student and ExamComponent. Therefore, the uniqueness of a result object is defined its student and ExamComponent objects (or foreign keys if in a relational database). Our system validates and maintains this uniqueness at data creation time.

For the use of OO concepts, we have tried to the SOLID principles where possible. All our objects have a Single responsibility – that of storing data and connections to other objects. This leads to us to the Open/closed principle where the entities can be extended but modifying their attributes would lead to the program crashing. For the Liskov substitution principle, our child objects can easily replace their parent objects. In following the Interface segregation principle, we created an interface that specific objects use instead of using general interfaces already implemented.

Test Case

|  |  |
| --- | --- |
| **1.1 Adding a new student** | **1.2 Adding an existing student** |
| Expected outcome: Program should allow adding of student and print out list of students after added successfully. | Expected outcome: Program should prompt error message that the student with matriculation number already existed. |
| C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 5.57.07 PM.png | C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 6.00.09 PM.png |

|  |  |
| --- | --- |
| **2.1 Adding a new course** | **2.2 Adding to an existing course** |
| Expected outcome: Program should allow adding of new course together with the professor in-charge as well as the number of classes for each class type. | Expected outcome: Program should prompt error message that the course index already existed thus the course is not added. |
|  | C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 6.07.10 PM.png |

|  |  |
| --- | --- |
| **3.1 Add a student to course with vacancies in Tut / Lab** | **3.2 Add a student to course with no vacancies in Tut / Lab** |
| Expected outcome: Program should allow student to be enrolled in the courses together with its corresponding tutorial and laboratory classes. | Expected outcome: Program should prompt that the particular class selected already full and prompt to select another class that has vacancy available. |
|  |  |

|  |  |
| --- | --- |
| **4.1 Check vacancies for classes** | **5.1 Print Student List by Class Type (Lecture)** |
| Expected outcome: Program should display the vacancies available with the total maximum vacancies allowed in the particular class selected. | Expected outcome: Program should display the student list with name and matric no. who has enrolled in the particular lecture class selected. |
| C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 7.04.55 PM.png | C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 7.36.43 PM.png |
| **6.1. Enter Course Component Weightage (Exam + 1 main coursework component w/o sub-components)** | **6.2. Enter Course Component Weightage (Exam + coursework with 2 sub-components)** |
| Expected outcome: Program should allow adding of exam and coursework weightage | Expected outcome: Program should allow adding of exam and coursework weightage together with the coursework components |
| C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 7.54.46 PM.png |  |

|  |  |  |
| --- | --- | --- |
| **7.1. Enter Coursework Marks inclusive of its components** | | **7.2. Enter Coursework Marks with 2 components** |
| Expected outcome: Program should allow adding of student’s coursework marks together with the marks of the corresponding coursework components | | Expected outcome: Program should allow adding of student’s coursework marks together with the marks of the corresponding coursework components |
| C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 8.22.47 PM.png | | C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 8.22.58 PM.png |
| **8.1. Enter Exam Marks** | **8.2 Enter Exam Marks with a course with no student enrolled in** | |
| Expected outcome: Program should allow adding of exam marks for a particular course | Expected outcome: Program should prompt that the course selected has no student enrolled in. | |
| C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 8.01.56 PM.png | C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 8.03.46 PM.png | |

|  |  |
| --- | --- |
| **8.3. Enter Exam Marks with a mark that exceeded the total exam mark and will be set to the highest marks** | **9.1 Print Course Statistics (with overall grade percentage for exam and coursework and its corresponding exam and coursework only)** |
| Expected outcome: Program should automatically set the exam marks to the highest marks allowed. | Expected outcome: Program should display the overall grade percentage for exam + coursework for a course together with each of the exams n coursework components |
| C:\Users\Nobody\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 8.09.35 PM.png | C:\Users\Sri\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 9.11.39 PM.png |
| **10.1. Print transcript (Select Student to print the results)** | **10.2. Print transcript that shows results for all courses registered by a particular students (enrolled in 4 courses)** |
| Expected outcome: Program should allow to select a student to show the student’s results. | Expected outcome: Program should display the results for all courses the student has enrolled in, together with the grade of that student for that course. |
| C:\Users\Sri\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 9.29.16 PM.png | C:\Users\Sri\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 9.29.16 PM.pngC:\Users\Sri\Dropbox\2002_Proj\Screen Shot 2012-11-10 at 9.29.33 PM.png |

Declaration of Original Work for CE/CZ2002 Assignment

We hereby declare that the attached group assignment has been researched, undertaken, completed and submitted as a collective effort by the group members listed below.

We have honoured the principles of academic integrity and have upheld Student Code of Academic Conduct in the completion of this work.

We understand that if plagiarism is found in the assignment, then lower marks or no marks will be awarded for the assessed work.

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | Course | Lab Group | Signature/Date |
| Cheok Jia De | CZ2002 | SSP6 |  |
| June Quak Ren Feng | CZ2002 | SSP6 |  |
| Sri Hartati | CZ2002 | SSP6 |  |