Timothy Kelly

CS-499 Computer Science Capstone

Milestone Two: Enhancement One

July 17, 2021

**Use Case Diagram:**

First diagram is my final creation of streamlined use-case diagram. Second diagram is my initial attempt at creating a use-case diagram.

**A picture containing text, map

Description automatically generated** A picture containing text

Description automatically generated

**Use Case 1:**

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Student Records | ID:  1 | Importance Level:  High |
| Primary Actor:  Staff | Use Case Type:  Detail, Essential | |
| Stakeholders and Interests:  Staff – wants to maintain student records information.  College – wants a website/app for records of student information. | | |
| Brief Description:  This use case describes how staff can access and maintain student information. | | |
| Trigger: Staff wants to access Student Records section.  Type: External | | |
| Relationships   * Association:   Student Records   * Include:   Course Records   * Extend:   Modify Student Records  Register Students   * Generalization:   Manage Student Records and Information | | |
| Normal Flow of Events:   1. Staff member accesses Student Records. 2. Staff member chooses to modify a student record attribute.   If the staff wants to modify the selected attribute,  The S-1: CRUD operations are performed.   1. Staff member continues to modify student information until completed. 2. Staff member selects other options or logouts of system. | | |
| SubFlows:  S-1: CRUD operations   1. Create new entry. 2. Read appropriate student entry. 3. Update existing student information. 4. Delete entry from student record. | | |
| Alternate/Exceptional Flows:  S-1: The staff member selects Registration from the main menu.  S-2: The staff member selects Course Records from the main menu. | | |

**Use Case 2:**

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Registration | ID:  2 | Importance Level:  High |
| Primary Actor:  Staff  Student | Use Case Type:  Detail, Essential | |
| Stakeholders and Interests:  Staff – wants to register students for classes.  Students – wants to register for classes.  College – wants an application for students to register for classes. | | |
| Brief Description:  This use case describes how staff and students can register for classes. | | |
| Trigger: Staff or student selects Registration for classes.  Type: External | | |
| Relationships   * Association:   Registration   * Include:   Course Records   * Extend:   Register for Classes   * Generalization:   Register for Available Courses. | | |
| Normal Flow of Events:   1. Staff/Student selects Registration.   Execute the enrollment entry information.   1. Staff/Student enters proper information for desired course offered. 2. System displays courses offered for desired request. 3. Staff/Student chooses desired course.   If staff/student chooses that course,  The S-1: Class validation for selected class performed.   1. System responds with confirmation message if student was placed into course or not. 2. Repeat steps 2-5 for desired number of courses to be added or deleted. 3. Logout of system | | |
| SubFlows:  S-1: Class Validation   1. The system validates if the student can enroll in the class. 2. A confirmation message is displayed if the student was enrolled into the class or not. | | |
| Alternate/Exceptional Flows:  S-1: Login verification for Registration fails and user is prompted back to Login screen. | | |

**Use Case 3:**

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Course Records | ID:  3 | Importance Level:  High |
| Primary Actor:  Staff  Students | Use Case Type:  Detail, Essential | |
| Stakeholders and Interests:  Staff – wants to access and/or modify Course Records.  Students – wants to access Course Records for registration. | | |
| Brief Description:  This use case describes how staff/students can access Course Records for registration purposes, as well as the ability for staff to modify Course Records. | | |
| Trigger: Staff/Students want to access Course Records.  Type: External | | |
| Relationships   * Association:   Course Records   * Include: * Extend:   Class Records   * Generalization:   Manage Course Records and Register for Classes | | |
| Normal Flow of Events:   1. Staff selects Course Records from Student Records.   Execute Course Records selection.   1. Staff member chooses to modify appropriate attribute under Course Records.   If staff member wants to modify attribute,  The S-1: CRUD operations is performed.   1. Staff member continues to modify attributes until completed. 2. Staff member logs out of Course Records. | | |
| SubFlows:  S-1: CRUD operations   1. Create new entry. 2. Read appropriate course entry. 3. Update existing course information. 4. Delete entry from course record. | | |
| Alternate/Exceptional Flows:  S-1: Staff selects Class Records from Course Records menu. (menu cannot exist if there are no classes) | | |

**Use Case 4:**

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Class Records | ID:  4 | Importance Level:  Medium |
| Primary Actor:  Staff | Use Case Type:  Detail | |
| Stakeholders and Interests:  Staff – wants to maintain class record information. | | |
| Brief Description:  This use case describes how staff can access Class Records to modify information. | | |
| Trigger: Staff wants to access Class Records section.  Type: External | | |
| Relationships   * Association:   Class Records   * Include:   Course Records   * Extend:   Modify Class Records   * Generalization:   Manage Class Records. | | |
| Normal Flow of Events:   1. Staff selects Class Records from Course Records menu.   Execute Class Records selection.   1. Staff member chooses to modify appropriate attribute under Class Records.   If staff member wants to modify attribute,  The S-1: CRUD operations is performed.   1. Staff member continues to modify attributes until completed. 2. Staff member logs out of Class Records. | | |
| SubFlows:  S-1: CRUD operations   1. Create new entry. 2. Read appropriate course entry. 3. Update existing course information. 4. Delete entry from course record. | | |
| Alternate/Exceptional Flows:  S-1: Staff member selects Online or F2F classes to be modified. | | |

1. **Testing:** Verify and validate your use case diagram and use case descriptions against the SIS requirements definition.

To verify my Use Case diagram, I tried to adhere to the principles of the diagrams presented in the textbook as well as the video lessons. I also verified my final Use Case diagram against my CRC cards as well as the Use Case descriptions. I worked through the menu interface of the system to include the proper attributes required by the rubric as well.

1. **Approach Explanation:** Explain your approach to the problem, the decisions you made to arrive at your solution, and how you completed it.

After receiving the feedback from my previous assignments and learning a bit more as the course has continued, I believe I managed to arrive at a proper Use Case diagram for the Student Information System. As I mentioned in the previous question, I really tried to adhere to the principles presented in the textbook as well as the video links. With every case I chose, I compared it against the requirements of the rubric. I feel as though I selected the proper cases and narrowed it down to the main elements of the desired program.

1. **Self-Reflection:** Reflect on this experience and the lessons you learned from it.

The most challenging part about this assignment up until this point has been the required information that should be in the program and where that information should properly reside. I found it easier the more I could compare my results against the examples provided to us. One of the biggest areas where I learned a lot but still feel as though I have much to learn is the Use Case diagram itself. Even though they look simple at first, there is a huge thought process that can go into the diagram to choosing the appropriate elements for each use-case.

**Generate your student information system (SIS) sequence diagram for the Register a Student for Classes use case.**

A screenshot of a cell phone

Description automatically generated

**Generate your SIS communication diagram for the Register a Student for Classes use case.**

A close up of a logo

Description automatically generated

**SIS Method Contract 1 template:**

|  |  |  |
| --- | --- | --- |
| Method Name:  Verification | Class Name:  Registration | ID:  - |
| Clients (Consumers):  Staff/students | | |
| Associated Use Cases:  Student Records  Course Records | | |
| Description of Responsibilities:  Validates whether student can take class if eligible and pre-requisites met. | | |
| Arguments Received:  Registration: Verification | | |
| Type of Value Returned:  String | | |
| Pre-Conditions:  Selection made. Verify selection against GPA and pre-requisites | | |
| Post-Conditions:  Validation successful or not successful. | | |

**SIS Method Contract 2 template:**

|  |  |  |
| --- | --- | --- |
| Method Name:  Search by Class ID | Class Name:  Registration | ID:  - |
| Clients (Consumers):  Staff/students | | |
| Associated Use Cases:  Student Records  Course Records | | |
| Description of Responsibilities:  Search for upcoming classes by Class ID number | | |
| Arguments Received:  Registration: Search by class ID number | | |
| Type of Value Returned:  String | | |
| Pre-Conditions:  Student is enrolled in school. Class exists. Class ID number entered | | |
| Post-Conditions:  Staff or student makes selection | | |

1. Verify and validate your sequence diagram and communication diagram against your SIS functional model and structural model.

I tested my sequence and communication diagram against my other models by making sure the steps included mirrored that of the other diagrams. I tried to make sure all steps were included in the registration and that it matched the flow as if the user was trying to search for and register for those classes assuming they exist.

1. Explain your approach to the problem, the decisions you made to arrive at your solution, and how you completed it.

For my sequence diagram, I took the approach as if I was going through each phase in the registration process. I tried to make sure I hit each class as I moved through the diagram. I tried to imagine if I was registering for classes, what menu selections I would traverse through. My communication diagram was a similar approach in that I selected the steps that would take place as if someone was registering for classes. I tried to keep it simple since I didn’t want any unwanted actions taking place in this process.

1. Reflect on this experience and the lessons you learned from it.

I was constantly going back and forth between each diagram as well as the method contracts and specifications. While I know this is probably a rather simple introduction to the models and diagrams used within the industry, I had no idea just how much prep work goes into designing a program with the appropriate diagrams and models. I felt more confident in the diagrams than I did the method applications though. I felt as though I hit all the appropriate phases and steps in my diagrams although I may be missing some features here and there. The method contracts and specifications were another story. Even after going through the text, I wasn’t sure if I was including the proper information. Hopefully with some feedback I can improve on these as I feel it was almost a trial-by-error if everything was in its proper location. I think one of the bigger things I’m learning from these assignments is to be direct and to the point within detail of each diagram and model respectively.

**Generate your SIS class diagram:**

A screenshot of a cell phone

Description automatically generated

**CRC Card 1:**

|  |  |  |  |
| --- | --- | --- | --- |
| Front | | | |
| Class Name:  Student Records | ID:  1 | | Type:  Concrete |
| Description:  For staff members who need/want to modify student information. | | | Associated Use Cases:  2 |
| Responsibilities:  Add student  Modify student information  Delete student  Register students for classes | | Collaborators:  Registration  Course Records | |
| Back | | | |
| Attributes:  First name (string)  Middle initial(s) (string)  Last name (string)  Date of birth (integer)  Student ID (integer)  Department (string)  GPA (integer) | | | |
| Relationships:  Generalization (a-kind-of):  User  Aggregation (has-parts):  Registration  Course Records  Other Associations:  Class Records | | | |

**CRC Card 2:**

|  |  |  |  |
| --- | --- | --- | --- |
| Front | | | |
| Class Name:  Registration | ID:  2 | | Type:  Property |
| Description:  For staff and students to register for classes | | | Associated Use Cases:  2 |
| Responsibilities:  Allow staff and students to search for and register for classes | | Collaborators:  Student Records  Course Records | |
| Back | | | |
| Attributes:  Enter course ID (int)  Enter class ID (int)  Select course (int)  Pre-requisites (int) | | | |
| Relationships:  Generalization (a-kind-of):  Process  Aggregation (has-parts):  Student Records  Course Records  Other Associations:  Online  Face-to-Face | | | |

**CRC Card 3:**

|  |  |  |  |
| --- | --- | --- | --- |
| Front | | | |
| Class Name:  Course Records | ID:  3 | | Type:  Concrete |
| Description:  For staff members who need to manage course information | | | Associated Use Cases:  3 |
| Responsibilities:  Add course  Modify course  Delete course | | Collaborators:  Student Records  Registration | |
| Back | | | |
| Attributes:  Course ID (integer)  Name (string)  Credit hours (integer)  Description (string) | | | |
| Relationships:  Generalization (a-kind-of):  Class  Aggregation (has-parts):  Registration  Student Records  Other Associations:  Class Records  Online  Face-to-Face | | | |

**CRC Card 4:**

|  |  |  |  |
| --- | --- | --- | --- |
| Front | | | |
| Class Name:  Class Records | ID:  4 | | Type:  Concrete |
| Description:  For staff to manage class records information | | | Associated Use Cases:  3 |
| Responsibilities:  Add online/F2F class  Modify online/F2F class  Delete online/F2F class | | Collaborators:  Course Records | |
| Back | | | |
| Attributes:  Course ID (integer)  Begin date (integer)  End date (integer) | | | |
| Relationships:  Generalization (a-kind-of):  Class  Aggregation (has-parts):  Course Records  Other Associations:  Online  Face-to-Face | | | |

1. Verify and validate your CRC cards and class diagram against your SIS functional model:

To verify my CRC cards and diagram, I first made sure I had the correct number of cards for each one of my class objects in the class diagram. I took the attributes from my diagram and made sure I had each of them included on the CRC cards. I also made sure to include each method from the diagram and listed this under the responsibilities section. I tried to keep my associations in-order and included these under the proper categories of collaborators, aggregations, and other associations respectively.

1. Explain your approach to the problem, the decisions you made to arrive at your solution, and how you completed it:

I first looked at the feedback from my previous model and CRC cards. I went back through my diagram to include information I missed before such as the student GPA which I included under student records. However, I also needed it under the Registration class which it is not directly under the attributes, but it would take this into account under the pre-requisites section. I wrote down all the information that was needed in each class first. I then drew out a few rough drafts and diagrams of the information and where each attribute and method would reside. I would constantly compare my models against the examples presented to us in the textbook as well as the links provided to us within the announcements and modules. I arrived at my diagram solution because I wanted it to represent a logical flow of how a user would move through the menu system in a sequence of events and chose a top-to-bottom method to represent this. I feel as though I generated a more professional looking diagram and CRC cards with because of the feedback and constant comparisons with those examples.

1. Reflect on this experience and the lessons you learned from it:

This assignment was difficult in the regard that I wasn’t sure if I was on the correct path with my previous models and use-case diagram. I was having a hard time trying to figure out the correct information and attributes that needed to be included within each case. I believe the reason for this was not having any prior experience with this area of study. The feedback helped to instill a bit more confidence that I was on somewhat of a correct path towards the proper information and requirements. However, I did enjoy the challenge of the class diagram and connecting each relationship. I enjoyed creating the diagram and trying to find a proper solution and presentation that was easy to understand. One big thing that I took away from this assignment is just how much each design and model works together with each other.