# IT 315 Final Project Part II Solution Submission Template

**Name:** Byron Laferriere

**Date:** 22-Nov-2020

1. Using this class responsibility collaboration (CRC) card template, document each class you identified from your student information system (SIS) functional model.

The following cards are what are referred to as use class, responsibility, and collaboration cards or CRC Cards. These cards are used to develop the structural model and help role play classes to ensure that there is the proper amount responsibilities and attributes assigned to each, without providing too little or too much detail. It is useful to draft these cards out before making the Structural Model to ensure everything is covered and allow for peer feedback. (Dennis, Wixom, & Tegarden, 2015.)

**CRC Card 1:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Front:** | | | |
| Class Name:  MaintainCourseRecords | ID:  CRC-001 | | Type:  Domain |
| Description: This class is intended to manipulate course records that are offered at the University. Responsible for providing CRUD capabilities. | | | Associated Use Cases:  Maintain Course Records (FM-002) |
| Responsibilities:  CreateCourseRecord  ReadCourseRecord  UpdateCourseRecord  DeleteCourseRecord | | Collaborators:  N/A | |
| **Back:** | | | |
| Attributes:  CourseIdentification- This is the ID number that can be used to find the Course  CourseName- This is the name of the Course  CourseCreditHours- This is how many credit hours the Course is worth  CourseDescription- This is a brief description of what to expect to learn in the Course | | | |
| Relationships     * Generalization (a-kind-of): N/A      * Aggregation (has-parts): This class composes MaintainClassRecords objects.      * Other Associations: Login Authentication System | | | |
|  |  |  |  |

**CRC Card #2:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Front:** | | | |
| Class Name:  MaintainClassRecords | ID:  CRC-002 | | Type:  Abstract |
| Description: This class is intended to manipulate class records offered at the University. | | | Associated Use Cases:  Maintain Class Records (FM-003) |
| Responsibilities: Class is responsible for providing access and upkeep for itself and child classes.  CreateClassRecords  ReadClassRecords  UpdateClassRecords  DeleteClassRecords | | Collaborators:  MaintainCourseRecords  MaintainOnlineRecords  MaintainF2FRecords | |
| **Back:** | | | |
| Attributes:  ClassIdentification- This is an ID number that can be used to find the Course/Class  ClassBeginDate- This is the day the class begins for the semester  ClassEndDate- This is the day the class ends for the semester | | | |
| Relationships     * Generalization (a-kind-of): N/A      * Aggregation (has-parts): MaintainCourseRecords composes this class      * Other Associations: Login Authentication System (Staff Member access) and Parent class of: MaintainOnlineClassRecords/ MaintainF2FClassRecords | | | |
|  |  |  |  |

**CRC Card #3:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Front:** | | | |
| Class Name:  MaintainOnlineClassRecords | ID:  CRC-003 | | Type:  Domain |
| Description: This class is intended to manipulate Online Class Records | | | Associated Use Cases:  Online Classes (FM-004) |
| Responsibilities:  This class is responsible for providing access and upkeep of all Online Classes offered through Courses at the University. Inherited CRUD capabilities from parent class. | | Collaborators:  MaintainClassRecords | |
| **Back:** | | | |
| Attributes:  ClassURL- This is a URL that contains a hyperlink to the online class  ClassBrowser- This is the homepage for the class | | | |
| Relationships     * Generalization (a-kind-of): MaintainClassRecords      * Aggregation (has-parts): N/A      * Other Associations: N/A | | | |
|  |  |  |  |

**CRC Card #4**

|  |  |  |  |
| --- | --- | --- | --- |
| **Front:** | | | |
| Class Name:  MaintainF2FClassRecords | ID:  CRC-004 | | Type:  Domain |
| Description: This class is intended to manipulate Face to Face Class Records | | | Associated Use Cases:  Face-To-Face Classes (FM-005) |
| Responsibilities: This class is responsible for providing access and upkeep for all On-campus classes offered at the University. It inherits CRUD capabilities from its parent class. | | Collaborators:  MaintainClassRecords | |
| **Back:** | | | |
| Attributes:  ClassBuilding- This is the building location the class is being held in  ClassRoom- This is the room that the class is being held in | | | |
| Relationships     * Generalization (a-kind-of): MaintainClassRecords * Aggregation (has-parts): N/A      * Other Associations: N/A | | | |
|  |  |  |  |

**CRC Card #5**

|  |  |  |  |
| --- | --- | --- | --- |
| **Front:** | | | |
| Class Name:  MaintainStudentRecord | ID:  CRC-005 | | Type:  Domain |
| Description:  This class is responsible for providing access and upkeep capabilities to Enrollment Staff for all students registered at school | | | Associated Use Cases:  Maintain Student Records (FM-008) |
| Responsibilities:  This class is responsible for providing access and upkeep capabilities to Enrollment Staff for all students registered at University.  CreateStudentRecord  ReadStudentRecord  UpdateStudentRecord  DeleteStudentRecord | | Collaborators:  RegisterStudents4Classes  MaintainClassRecords | |
| **Back:** | | | |
| Attributes:  FirstName – This contains the first name of the student registered at the university  MiddleInitials – This contains the middle initial(s) of the student registered  LastName – This contains the last name of the student registered  DateOfBirth – This is the date of birth of the student on file in format (mm/dd/yyyy)  StudentID – This is the identification number assigned to the student  Department – This is the department that the student’s major resides in  CumulativeGPA – This is the student’s cumulative GPA, which decides if they can register for three courses or not. | | | |
| Relationships:  Generalization (a-kind-of): N/A  Aggregation (has-parts): N/A  Other Associations: Login Authentication System to check for Enrollment Staff credentials. There is a dependent relationship with RegisterStudent4Classes attached to this class. | | | |

**CRC Card #6**

|  |  |  |  |
| --- | --- | --- | --- |
| **Front:** | | | |
| Class Name:  RegisterStudent4Classes | ID:  CRC-006 | | Type:  Domain |
| Description:  This class is intended to provide manipulation of class registration to students and enrollment staff. | | | Associated Use Cases:  Register Student For Classes (FM-009) |
| Responsibilities:  This class is intended to provide manipulation of class registration to students and enrollment staff.  Responsibilities (*Doing*):  BlockDuplicateRegistrations  RequiredWorkAcknowledgement  CheckStudentGPA  If: GPA>3.5  Allow3ClassRegistration  ELSE:  EnforceRegistrationLimits  END  AllowStudentSelfRegistration  AllowEnrollmentStaffRegistration | | Collaborators:  MaintainStudentRecords | |
| **Back:** | | | |
| Attributes:  N/A | | | |
| Relationships:  Generalization (a-kind-of): N/A  Aggregation (has-parts): N/A  Other Associations: There is a dependency for this class with three other classes. Those classes are MaintainStudentRecord, OnlineClasses, F2FClasses. | | | |

**CRC Card #1:**

This class can be considered a ‘stand-alone’ class in our Structural Model. While the only association it has is with the Login Authentication System, the rest of the system relies on this to operate. The class will provide Staff Members with CRUD abilities for all attributes listed.

**CRC Card #2:**

This class inherits from MaintainCourseRecords, as no Class can exist without the Course. This class is also a ‘parent’ class to Online Classes and Face-to-Face Classes. This class also needs to provide Staff Members with CRUD abilities for the inherited attributes, as well as the independent attributes specific to the class.

**CRC Card #3:**

This class inherits from MaintainClassRecords, as no Online Class can exist without the Class. This class is considered a ‘child’ class because of this and inherits all attributes and responsibilities from MaintainClassRecords. It also introduces new independent attributes that are specific to Online Classes only.

**CRC Card #4:**

This class also inherits from MaintainClassRecords, as no Face-To-Face Class can exist with the Class. This class is also considered to be a ‘child’ class because of this and inherits all attributes and responsibilities from MaintainClassRecords. It also introduces new independent variables that are specific to Face-To-Face Classes only.

**CRC Card #5:**

This class breaks from the previous inherited classes to create a new class that will provide necessary information for its dependently related class RegisterStudent4Classes. It is provided with its own CRUD capabilities, as well as independent attributes related to individual student information.

**CRC Card #6:**

This class is meant to act alongside MaintainStudentRecords and carries many of the doing responsibilities required for student registration. It has a dependency relationship with MaintainStudentRecords which means that it requires the class for instantiation and cannot operate without it.

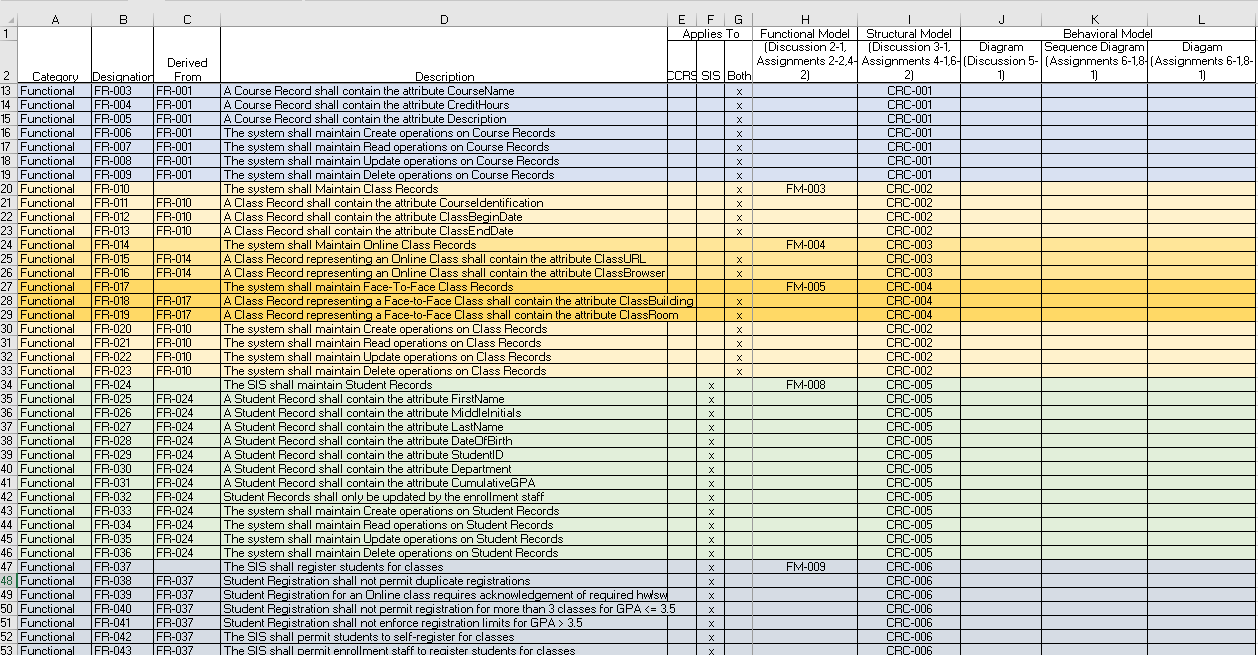
*Generate your SIS class diagram:*

The SIS Structural Model included below, is a compilation of the information from the CRC Cards previously seen. This model takes the information from these cards and lays out the relationships in a manner that explains the flow of the system and how each class interacts with each other. This specific diagram is meant to be read in left-right (verticality) form and up-down (linearity). The numbers (called multiplicity) explain how each instance interacts with others. This diagram has two different instances of these relationships. There are 1 to 1…\*, which is read as “1 to 1 to many” and the other is 1 to 0..\*, which can be read as “1 to 0 to many.” When we see a 1 to 1 to many combination, it refers to the fact that there may be a single instance on one end and the other end may have at least one instance but many more than one as well. The 1 to 0 to many instance infers the same beginning but does not require that there be an instance on the other end, yet allows for as many as possible if there were to be one.



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

The best way to validate and verify that the CRC Cards contained all required use cases from the Courses and Classes Requirements Definitions page, was to supply a spreadsheet containing the requested use cases. The attached Requirements Traceability Verification Matrix (RTVM) contains these use cases, as well as additional information about where these needs were met and, on which model they were satisfied. In column H, the RTVM displays where the use cases can be found on the Functional Model. When viewing the Functional Model, look for the corresponding values assigned in this column, which can be matched back to use cases being defined and constructed on the Functional Model. In column I on the RTVM, the use cases have similar corresponding values that align with the Structural Model. Each use case identified in the Course and Classes Records System has been met and assigned to a Use Class, Responsibilities, and Collaboration Card or CRC Card.



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

To create these CRC Cards, I went back through previously defined documentation to find the information needed. I used the CCRS Requirements Definitions document that was provided, the Functional Model, and the RTVM. Once the CRC Cards had been filled in with the information provided to me, I was able to better ensure the Structural Model would be consistent throughout. To stay true to this mentality, when designing the CRC Cards, I also included the Functional Model ID’s to make cross-referencing easier for the viewer. An aggregation relationship (composition) was chosen to represent the connection between MaintainClassRecords and MaintainCourseRecords, as it cannot exist without MaintainCourseRecords (Dennis, Wixom, & Tegarden, 2015) and MaintainCourseRecord composes MaintainClassRecords. There was a generalization relationship chosen between the MaintainClassRecords and MaintainOnlineClassRecords and MaintainF2FClassRecords because it was adding new functionality to the system through MaintainClassRecords. An aggregation relationship would not be appropriate in this situation because it is meant to be used as a composition of functionalities previously defined (Dennis, Wixom, & Tegarden, 2015) and the generalization relationship is used when inheritance is needed between two classes (Dennis, Wixom, & Tegarden, 2015). Each class definition was chosen based on the class properties, with only MaintainClassRecords being an abstract (parent) class. This is so, because of the attributes that it passes along to its child classes, MaintainOnlineClassRecords and MaintainF2FClassRecords (Dennis, Wixom, & Tegarden, 2015). The other five classes were Domain classes.

When creating the SIS Structural model, I needed to extend the model previously created by adding on two more classes that were defined this week. I supplied the class names as well as which CRC Card they could be found on for clarification of information if needed. There are different elements present on this model that I will explain how I chose to represent the system. First, the attributes and the methods (responsibilities) are divided by a dotted line and have two noticeable differences. There are ‘+’ or ‘-‘ attached to these terms in this diagram. The ‘+’ is meant to be viewed as a public or open definition and can be used for either attributes or methods (Dennis, Wixom, & Tegarden, 2015). For this model, we will use this for our methods. The other symbol ‘-‘ is viewed as a private indicator, where restrictions are applied on CRUD capability on the system. For this system, we will use ‘-‘ to define our attributes. Next, you will notice that different types of connections exist between the boxes. These were chosen by the relationships expressed in the CRC Cards and define class interaction or inheritances. There is one closed diamond shaped line connecting CRC-002 to CRC-001, which is used to represent a composition relationship. This specific type of relationship is a special type of association relationship, that represents a physical a-part-of relationship (Dennis, Wixom, & Tegarden, 2015). The second type of connection chosen for this diagram was the generalization relationship between OnlineClasses (CRC-003), F2FClasses (CRC-004) and their parent class ClassRecords (CRC-002). The generalization relationship is indicated by a line with an open arrow pointing back to the parent class that the attributes are being inherited from. This type of class is used to represent a-type-of relationship within structural models (Dennis, Wixom, & Tegarden, 2015). Lastly, the third connection chosen between the classes on this diagram was a dependency relationship. A dependency relationship is used when one class cannot be instantiated without the presence of another class within the system (Dennis, Wixom, & Tegarden, 2015). These relationships are notated by a dotted line with an unfilled arrow on the end that points to the class where the dependency lies. The dependency relationship was the best choice for this class because of the fact that if a change was made to one of these classes, like availability, it would affect and change the class RegisterStudent4Classes.

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

The Structural Model was a building block, that required a proper Functional Model to be established. Once I had properly defined and shaped the Functional Model, it provided a much clearer picture of what was required in the CRC Cards and Structural Model. This seems to be a redundant message throughout the modeling process. The most important lesson I learned was to find proper definitions before the creation of the process began. It was time-consuming and frustrating when I needed to back track to change things that I assumed to be true. If I can learn to perfect this step, it will only help my creation of the Behavioral Model in the next steps of the process. Taking what I have learned from this project, I will be sure to identify correct definitions from use cases that will help make this process smoother. I have read that the Behavioral Model is the most detailed and time-consuming piece of systems design, therefore I intend to find as many ways as possible to help eliminate wasting time by repeating steps.

References:

Dennis, A., Wixom, B. H., & Tegarden, D. (2015). VitalSource Bookshelf Online. Retrieved November 07, 2020, from <https://mbsdirect.vitalsource.com/>