CARLETON UNIVERSITY

Requirement Analysis Document

Prepared by Team

GAME OF CODES

Submitted to:
Dr. Christine Laurendeau
COMP 3004 Object-Oriented Software Engineering
School of Computer Science Carleton University

Oct-14-2015

1. Introduction/ Overview

Doesn't matter what industry you work in, unless you are working completely by yourself chances are you will end up working with co-workers, managers etc. Team-work is very important in today's work-place and also in school. It can provide you a glimpse of what to expect when interaction becomes absolutely necessary in the so-called "real world", and more importantly it exposes you to ideas, thoughts and solutions to a problem you might have never even given a thought to. Team-work also provides added benefit of emotional support from your team-members. Being able to work efficiently with a group is a great interpersonal skill one can possess for their success, especially while they are still in school.

Universities are incorporating group work to help students acquire these skills. But a problem many groups face is having a team of incompatible people. This could manifest as a group being strong in one aspect but weak in another. To help build well managed and cohesive teams, a system was devised to sort students in optimal groups, called the Carleton University Project Partner Identifier (cuPID). The system generates teams by matching compatibility or different students to ensure a well balanced team.

cuPID is a system where administrators can create projects and students can sign up into the projects. The cuPID system has a very unique algorithm called the "Project Partner Identifier" (PPID). The Algorithm groups students together based on a questionnaire that each student is required to complete during registration into the cuPID system.

The questionnaire for the students has two parts. First part focuses on, the registering student's strengths, weaknesses and study habits. The second part focuses on what the registering student is looking for in their desired team-member(s). Students do not have access to other students' profile, all the student profiles are kept private and their academic records are kept confidential.

The administrator is the "Instructor" of any given course and is responsible for creating projects for students to register in. The administrator fills all the necessary description and information about the project. Once the students have registered into their desired projects, the administrator sets the group size and runs the PPID algorithm.

Our algorithm facilitates building a team of compatible students well equipped to complete the task proficiently.

The enclosing document gives a detailed over-view of the functionality and highlights requirements of the system to accomplish the desired task.

2. Functional Requirements

Functional Requirements for the cuPID system allows its two different users, the **Administrators** and the **Students**, to have different functionality. The **students** are allowed to register in projects and keep track of the projects they are registered in. The **Administrators** are provided a way to create projects, keep track of its created projects and form **student** groups for each project.

Table 1 – Functional Requirements

F-01	**Administrator** can manage project properties
F-01-01	Create project
F-01-02	Change existing project's team sizes
F-01-03	Change existing project's name
F-01-04	**Administrator** can view a project
F-02	**Administrator** can run the PPID algorithm to create groups
F-03	**Administrator** can modify the PPID
F-03-01	Change the weighting of each question asked
F-04	**Student** can create a profile and register in the cuPID system
F-05	**Student** can register in a project
F-06	**Student** can edit own profile
F-07	**Student** can edit project partner profile
F-08	**Student** can view list of existing projects
F-09	cuPID system can store a list of projects
F-10	cuPID system can store a list of registered **students** and their profiles
F-11	cuPID system can match **students** based on their profile answers and team sizes
F-12	cuPID system can output a "Summary Print" listing all **students** in their teams
F-13	cuPID system can output a "detailed print" which includes a "Summary Print" plus data supporting results
F-14	**Student** can view **Project**
F-15	**Student** can view profile

3. Non-Functional Requirements

Non-functional requirements for the cuPID system provide an elegant visual interface for the users and reliable storage for the data associated with the **Students**, Projects and **Administrators**. It also allows high level of security and privacy on how the data is shared among users.

Table 2 – Non-Functional Requirements

NF-01	Usability: Menu based on qualitative questions
NF-02	Usability: Every window has instructions on how to use current menu
NF-03	Usability: **Administrators** cannot access **student** profiles
NF-04	Usability: **Students** cannot see each other profiles
NF-05	Usability: "Summary Printouts" do not reveal confidential information
NF-06	Usability: Provide a detailed and clear help menu of instructions
NF-07	Usability: Provide confirmation of save and cancel to the user
NF-08	Usability: Provide the user with a detailed error message with a cause of the error
NF-09	Usability: Error messages will stop the system from advancing until corrected
NF-10	Usability: **Students** access their profiles by the **student** ID
NF-11	Reliability: **Student** profiles will be saved if the system crashes while PPID is running
NF-12	Reliability: Project will be saved once fully created
NF-13	Reliability: **Student** profiles will be saved once fully created
NF-14	Performance: The PPID algorithm will run within 5 minutes for every 100 **students**
NF-15	Supportability: The system will be portable on any Linux system
NF-16	Supportability: System is extensible to another GUI with only changing the UI classes
NF-17	Implementation: System will be written in C++ in QT
NF-18	Packaging: system will be available for download as a collection of object files with a MAKE file for easy compilation
NF-19	Interface: **Students** are not allowed access to **Administrator's** interface.
NF-20	Interface: **Administrator** is not allowed access to **Students**' interface.
NF- 21	Operation: Errors are accurately display to the users.
NF-22	Legal: All the user information is kept private from other user's.
NF- 23	Legal: All user information collection into the system will not be shared with or sold to any Third-party.
NF-24	Operation: The system will not proceed unless and until the error is corrected by the user.

4. Use Case Model

Use Case Overview

Use cases are diagrams that outline the list of steps required to complete a task in the system. They show the what each Actor is allowed to do. The two actors in the cuPID system are the Student and the Administrator.

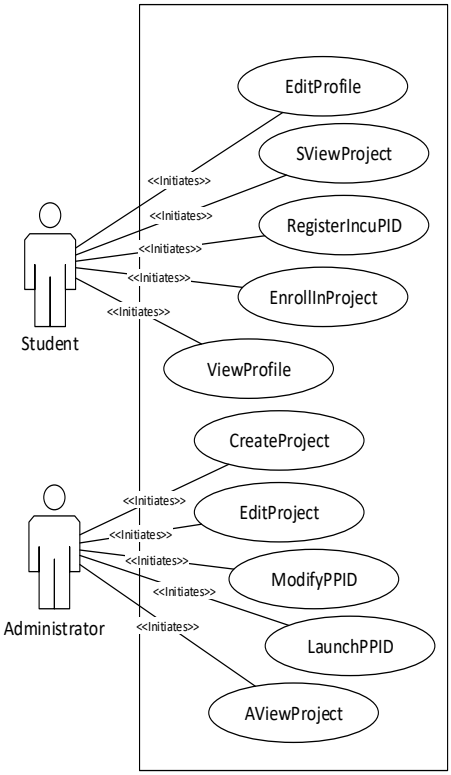


Figure 1 – High-Level Use Case Diagram

Table 3 – High-level Use Case Descriptions

UC-01 CreateProject	**Administrator** creates a new project
UC-02 EditProject	**Administrator** edits or views properties of a project
UC-03 LaunchPPID	**Administrator** runs the PPID on a project with the enrolled students
UC-04 ModifyPPID	**Administrator** modifies properties of the PPID
UC-05 RegisterIncuPID	**Student** registers in the cuPID system
UC-06 EnrollInProject	**Student** enrolls to be part of a project
UC-07 EditProfile	**Student** edits their profile
UC-17 AViewProject	**Administrator** views projects posted
UC-18 ViewProfile	**Student** views their profile
UC-19 SViewProject	**Student** views their Project

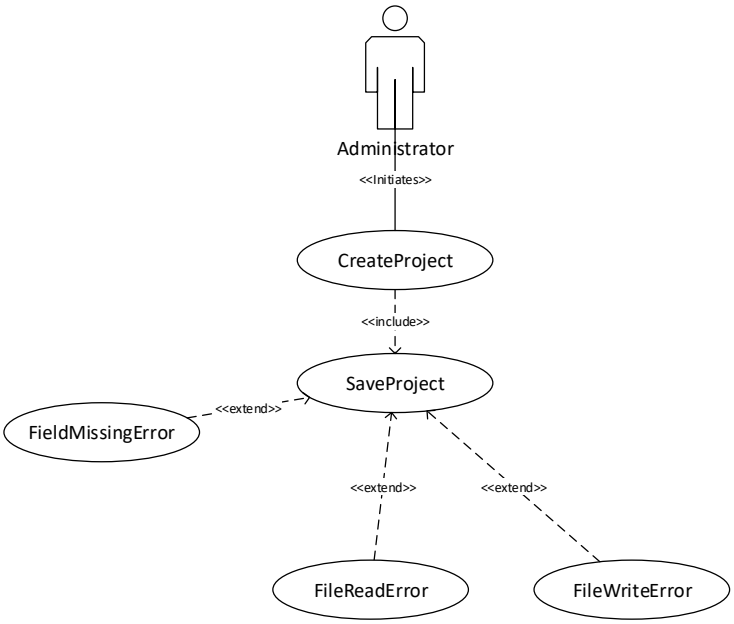


Figure 2 – CreateProject Detailed Use Case Diagram

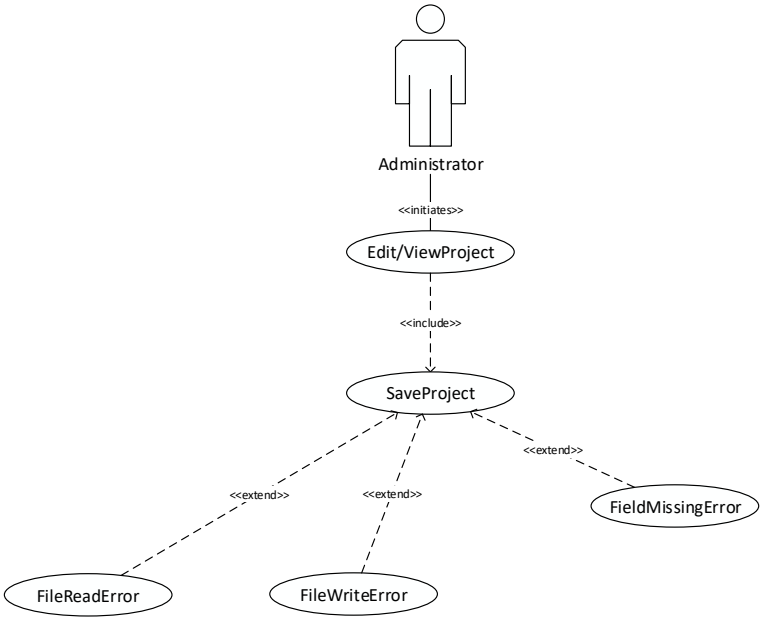


Figure 3 – EditProject Detailed Use Case Diagram

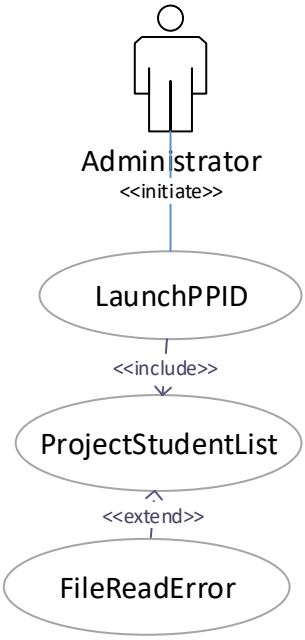


Figure 4 – LaunchPPID Detailed Use Case Diagram

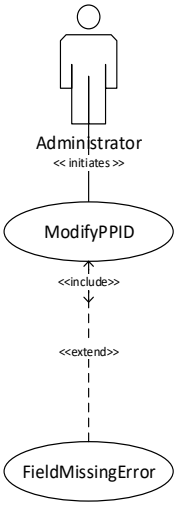


Figure 5 – ModifyPPID Detailed Use Case Diagram

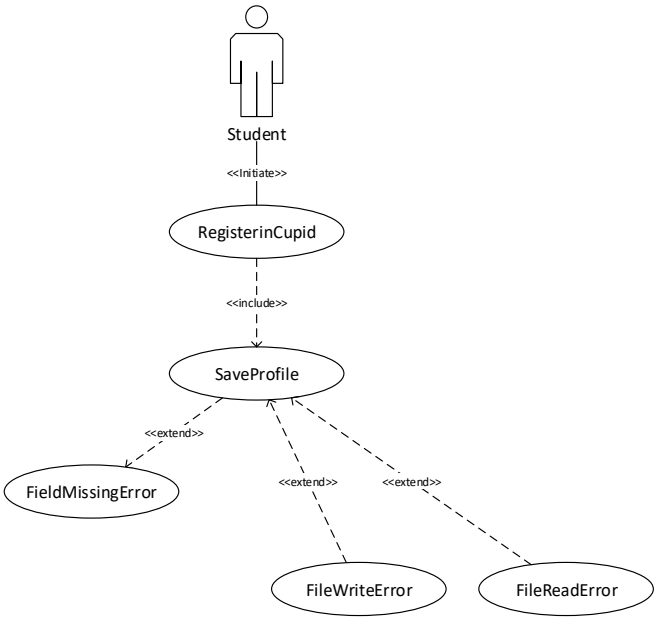


Figure 6 – RegisterIncuPID Detailed Use Case Diagram

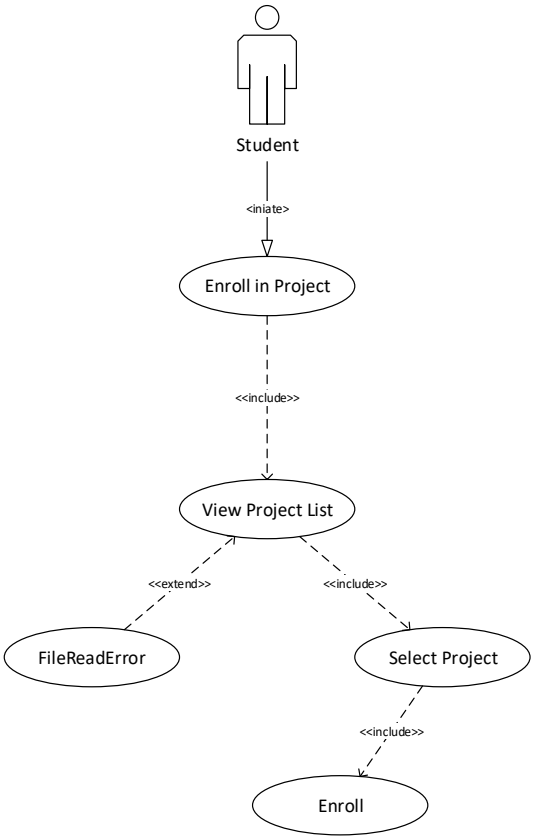


Figure 7 – EnrollInProject Detailed Use Case Diagram

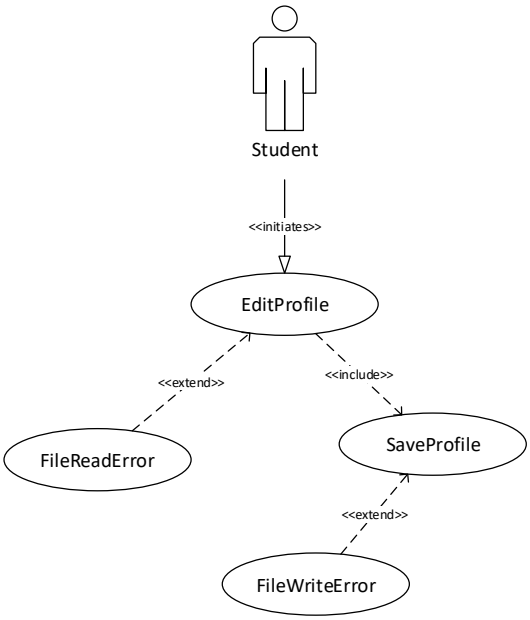


Figure 8 – EditProfile Detailed Use Case Diagram

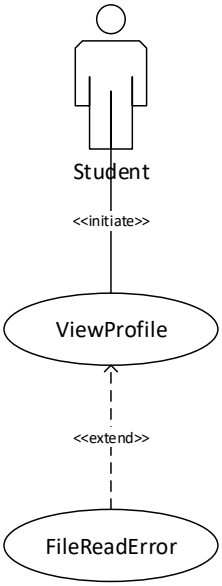


Figure 9 – ViewProfile Detailed Use Case Diagram

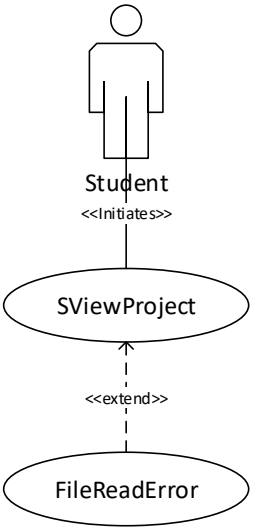


Figure 10 – SViewProject Detailed Use Case Diagram

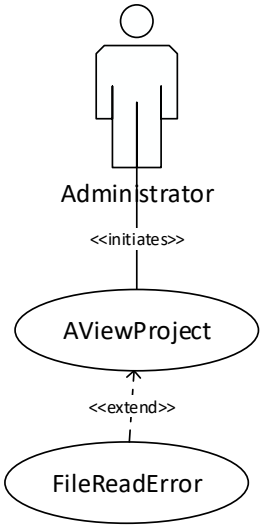


Figure 11 – AViewProject Detailed Use Case Diagram

Table 4 – Detailed Use Case Descriptions

UC-08 FileReadError	File Containing project list could not be opened
UC- 09 FileWriteError	Error writing to the file
UC-10 FieldMissingError	**Administrator** or **Student** tries to save project changes with a field missing
UC-11 ProjectStudentList	PPID accesses list of **students** enrolled in the project
UC-12 SaveProfile	**Student** saves the profile
UC-13 ViewProjectList	**Student** views list of projects to register in
UC-14 SelectProject	**Student** selects project to enroll into
UC-15 Enroll	**Student** Enrolls in select Project
UC-16 SaveProject	**Administrator** saves project

Use Case Flow of Events:

Use Case Identifier	UC-01
Name	Create Project
Participating actors	Initiated by **Administrator**

Flow of Events	• The **Administrator** chooses to create a new project.
	  ○ The system responds by providing a form.
	• The **Administrator** fills out the required field to create a project.
	  ○ The system checks if all the required fields were filled out by **Administrator**
	• The **Administrator** chooses to save the project.
	  ○ The System records the provided information and creates a new project.

Entry Conditions	**Administrator** must be logged in
Exit Conditions	**Administrator** chooses to save project and it was saved successfully
Quality requirements	Only **Administrator** should be able to create projects
Traceability	F-01-01

Use Case Identifier	UC-02
Name	Edit Project
Participating actors	Initiated by **Administrator**

Flow of Events	• The **Administrator** chooses to edit a project
	  ○ The system responds by providing a form
	• The **Administrator** fills out the required field to edit the project.
	  ○ The system checks if all the required fields
	• The **Administrator** chooses to save the project.
	  ○ The System records the provided information and over-writes the information in the project being edited.

Entry Conditions	1. **Administrator** must be logged in 2. Project must already exist
Exit Conditions	**Administrator** chooses to save project and changes are saved successfully
Quality requirements	User must be logged in with **administrator** privileges.
Traceability	F-01-02, F-01-03, F-09, F-10

Use Case Identifier	UC-03
Name	LaunchPPID
Participating actors	Initiated by **Administrator**

Flow of Events	• The **administrator** chooses to launch a PPID on a project.
	  ○ The system responds by providing a form
	• The **administrator** fills out the required fields to configure the PPID
	  ○ The system checks if all the required fields
	• The **administrator** chooses to save the PPID.
	  ○ The System records the provided information
	  ○ The system retrieves the project
	  ○ The system launches the new PPID on a project

Entry Conditions	1. **Administrator** must be logged in 2. Project must already exist upon which the PPID is launched
Exit Conditions	The **administrator** launches the PPID on a project.
Quality requirements	Only **administrator** can launch a PPID.
Traceability	F-02, F-11, F-12, F-13

Use Case Identifier	UC-04
Name	ModifyPPID
Participating actors	Initiated by **Administrator**

Flow of Events	• **Administrator** chooses to modify PPID
	  ○ **Administrator** can chooses between provided algorithm options
	  ○ **Administrator** can choose the group size option on PPID
	  ○

Entry Conditions	**Administrator** must be logged in
Exit Conditions	The **Administrator** would be registered into the project successfully and it would reflect in the **student**'s project list
Quality requirements	Only **Administrator** should be able to modify PPID
Traceability	F-03

Use Case Identifier	UC-05
Name	RegisterIncuPID
Participating actors	Initiated by **Student**

Flow of Events	• **Student** chooses to register into the cuPID program
	  ○ The system responds by providing a form
	• **Student** provides the required fields to register into cupid
	  ○ The system checks if all the required fields
	• **The Student** saves the project
	  ○ The system updates the fields with provided information

Entry Conditions	**Student** must be logged in
Exit Conditions	The **Student** would be registered into the cuPID successfully and the system would update the storage to reflect this change.
Quality requirements	Only **Student** should be able to register in cuPID
Traceability	F-04, F-08

Use Case Identifier	UC-06
Name	EnrollInProject
Participating actors	Initiated by **Student**

Flow of Events	• The **Student** chooses to see the project list.
	  ○ The system acquires the project list from the Storage
	  ○ The System displays project list.
	• **Student** chooses a project from the list.
	  ○ The system provides the option to register in the chosen project.
	• **Student** enrolls in chosen project.
	  ○ The System records the **student** name and **student** number and updates the project data.

Entry Conditions	**Student** must be logged in
Exit Conditions	The **student** would be registered into the project successfully and it would reflect in the **student**'s project list
Quality requirements	Only **Student** should be able to enroll into projects
Traceability	F-05, F-08

Use Case Identifier	UC-07
Name	Edit Profile
Participating actors	Initiated by **Student**

Flow of Events	• The **Student** chooses to edit/view their profile.
	  ○ The system acquires the profile information from the Storage
	  ○ The System displays profile information.
	• **Student** edits the profile if desired.
	• **Student** saves the Profile
	  ○ The System records the changes made to profile and updates students information in storage

Entry Conditions	**Student** must be logged in
Exit Conditions	The **student** would be registered into the cuPID program
Quality requirements	Only **Student** should be able to edit profile
Traceability	F-06 , F-07 , F-10

Use Case Identifier	UC-08
Name	FileReadError
Participating actors	Communicates with **Administrator** and **Student**

Flow of Events	• The **Administrator** chooses to save a newly created project.
	  ○ The system tries to access the project list file from the Storage but is unable to open the file because the file is missing
	• The **Administrator** chooses to save a project that was edited.
	  ○ The system tries to access the project list file from the Storage but is unable to open the file because the file is missing
	• The **Administrator** chooses to run the PPID on a project with enrolled **Students**.
	  ○ The system tries to access the list of **students** enrolled in to the project file but is unable to open it because the file is missing
	• The **Administrator** chooses to save the edited PPID
	  ○ The system tries to access the PPID file but is unable to open it because it is missing
	• The **Student** chooses to register in the system and save the newly created profile
	  ○ The system tries to access the registered **students** file but is unable to open the file because it is missing
	• The **Student** chooses to enroll in a project
	  ○ The system tries to access the file of registered **students** in a project but is unable to because the file is missing

Entry Conditions	**Administrator** or **Student** must be logged in
Exit Conditions	The system will create a new file and save the necessary information
Quality requirements	The system should not be able to proceed without finding or creating the file
Traceability	F-09, F-10, NF-8, NF-9

Use Case Identifier	UC-09
Name	FileWriteError
Participating actors	Communicates with **Administrator** and **Student**

Flow of Events	• The **Administrator** chooses to save a newly created project.
	  ○ The system opens the file from storage that has the list of all the projects
	  ○ The system tries to append to this file the name of the newly created project but is unable to because the project name already exists
	• The **Administrator** chooses to save a project that was edited.
	  ○ The system opens the project file from the Storage
	  ○ The system tries to modify to this file but is unable to due to system write errors
	• The **Administrator** chooses to run the PPID on a project with enrolled **Students**.
	  ○ The system opens the list of **students** enrolled in a project
	  ○ The system runs the PPID on the list of **students** and make groups but is unable to update the individual project's details on storage
	• The **Administrator** chooses to save the edited PPID
	  ○ The system opens the PPID file
	  ○ The system tries to write the modifications to the file but is unable to because certain input files are missing
	• The **Student** chooses to register in the system and save the newly created profile
	  ○ The system opens the file with the list of **students** registered in the system
	  ○ The system tries to write the **student** to the file but is unable to because the **student** number already exists in the file
	• The **Student** chooses to enroll in a project
	  ○ The system opens the file of the registered **students** in a project
	  ○ The system tries to add the **student** to the project but is unable to because the **student** number already exists in the file

Entry Conditions	**Administrator** or **Student** must be logged in
Exit Conditions	The system prompt the user of the error
Quality requirements	The system should not make a duplicate project name of **student** number
Traceability	F-09, F-10, NF-8, NF-9

Use Case Identifier	UC-10
Name	FieldMissingError
Participating actors	Communicates with **Administrator** and **Student**

Flow of Events	• The **Administrator** chooses to save a newly created project.
	  ○ The system checks to make sure all of the required fields have been filled out
	  ○ The system finds a required field that is missing and prompts the **Administrator** to fill it before it can proceed
	• The **Administrator** chooses to save a project that was edited.
	  ○ The system checks to make sure all of the required fields have been filled out
	  ○ The system finds a required field that is missing and prompts the **Administrator** to fill it before it can proceed
	• The **Administrator** chooses to save the edited PPID
	  ○ The system checks to make sure all of the required fields have been filled out
	  ○ The system finds a required field that is missing and prompts the **Administrator** to fill it before it can proceed
	• The **Student** chooses to register in the system and save the newly created profile
	  ○ The system checks to make sure all of the required fields have been filled out
	  ○ The system finds a required field that is missing and prompts the **Student** to fill it before it can proceed

Entry Conditions	**Administrator** or **Student** must be logged in
Exit Conditions	The **Administrator** or **Student** must fill out the missing field or fields
Quality requirements	The system should not be able to proceed when required attributes are missing
Traceability	F-01-01, F-03, F-04, F-06, F-07, F-09, F-10, NF-8, NF-9

Use Case Identifier	UC-11
Name	Project**Student**List
Participating actors	Communicates with the **Administrator**
Flow of Events	• **Administrator** runs the PPID ○ The system uses the PPID algorithm provided by the **Administrator**, using the enrolled list of **students** in the project for making groups
Entry Conditions	**Administrator** must be logged in
Exit Conditions	PPID runs successfully and groups are made
Quality requirements	Only **Administrator** must be allowed to run PPID
Traceability	F-01-01, F-02, F-03, F-05, F-09, F-10, F-11, F-12, F-13, NF-05, NF-14

Use Case Identifier	UC-12
Name	SaveProfile
Participating actors	Communicates with the **Student**

Flow of Events	• The **student** chooses to register into CuPID and create new profile or edit an existing one. Once done the **student** saves the profile created.
	  ○ The system checks to make sure all of the required fields have been filled out
	  ○ The system finds required fields that are missing and prompts the **student** to fill them before it can proceed

Entry Conditions	**Student** must be registered and logged in
Exit Conditions	Profile was complete and successfully saved
Quality requirements	**Student** can only create/edit and save their profiles
Traceability	F-04, F-06, F-07, NF-04, NF-13

Use Case Identifier	UC-13
Name	**ViewProjectList**
Participating actors	Initiated by **Student**

Flow of Events	• **Student** chooses to view project list
	  ○ The system retrieves the project list from the storage
	  ○ The system responds by displaying the project list

Entry Conditions	**Student** must be logged in
Exit Conditions	The **student** is finished viewing the project list and closes the project list
Quality requirements	Only registered **students** should be able to view project list
Traceability	F-08

Use Case Identifier	UC-14
Name	**SelectProject**
Participating actors	Initiated by **Student**

Flow of Events	• **Student** chooses to view project list and selects the project he or she wants to enroll in.
	  ○ The system retrieves the project list from the storage
	  ○ The system responds by displaying the project list for the **student.**

Entry Conditions	**Student** must be logged in
Exit Conditions	The **Student** selects a project that he or she wants to be registered in.
Quality requirements	○ Only a **Students** with a complete and valid student profile and partner profile would be able to view project list
Traceability	F-08, F-14

Use Case Identifier	UC-15
Name	**Enroll**
Participating actors	Initiated by **Student**

Flow of Events	• **Student** chooses to enroll in selected project
	  ○ The system responds by adding the student to the project and adding the project to the student's project list.
	  ○ The system updates storage.

Entry Conditions	**Student** must be logged in
Exit Conditions	The **student** successfully enrolls into the selected project.
Quality requirements	○ Only a **Student** with a complete and valid student profile and partner profiles would be able to enroll into a project.
Traceability	F-05

Use Case Identifier	UC-16
Name	SaveProject
Participating actors	Initiated by **Administrator**

Flow of Events	• **Administrator** creates a project and tries to save it.
	  ○ The system checks if all fields required creating a project is fulfilled.
	  ○ The system responds saving the project and updating the storage.
	  ○ The system updates the **Administrators** information so it reflects the changes made.

Entry Conditions	**Administrator** must be logged in and **administrator** is trying to create a project
Exit Conditions	The System saves the project
Quality requirements	Only **administrator** should be able to create and save a project
Traceability	F-01-01, F-01-02, F-01-03

Use Case Identifier	UC-17
Name	AViewProject
Participating actors	Initiated by **Administrator**
Flow of Events	• The **Administrator** chooses to view a project. ○ The system responds by providing detailed view of the chosen project. • The **Administrator** chooses to close the project view.
Entry Conditions	**Administrator** must be logged in
Exit Conditions	**Administrator** chooses to close the project view.
Quality requirements	The **Administrator** gets to see the title, description and number of **students** enrolled in the **project**.
Traceability	F-01-04

Use Case Identifier	UC-18
Name	ViewProfile
Participating actors	Initiated by **Student**
Flow of Events	• **Student** chooses to view profile. ○ The system responds by showing the view of their profile.
Entry Conditions	**Student** must be logged in and registered.
Exit Conditions	**Student** dismisses the view
Quality requirements	A **Student** can only view their own profile
Traceability	F-15, NF-04. NF-10. NF-22

Use Case Identifier	UC-19
Name	SViewProject
Participating actors	Initiated by **Student**
Flow of Events	• The **Student** chooses to view a **Project**. ○ The system responds by providing a view of the **Project**
Entry Conditions	1. **Student** must be logged in 2. Project must already exist upon view
Exit Conditions	The **Student** dismisses the view.
Quality requirements	**Student** can view **Project** title and description .
Traceability	F-14

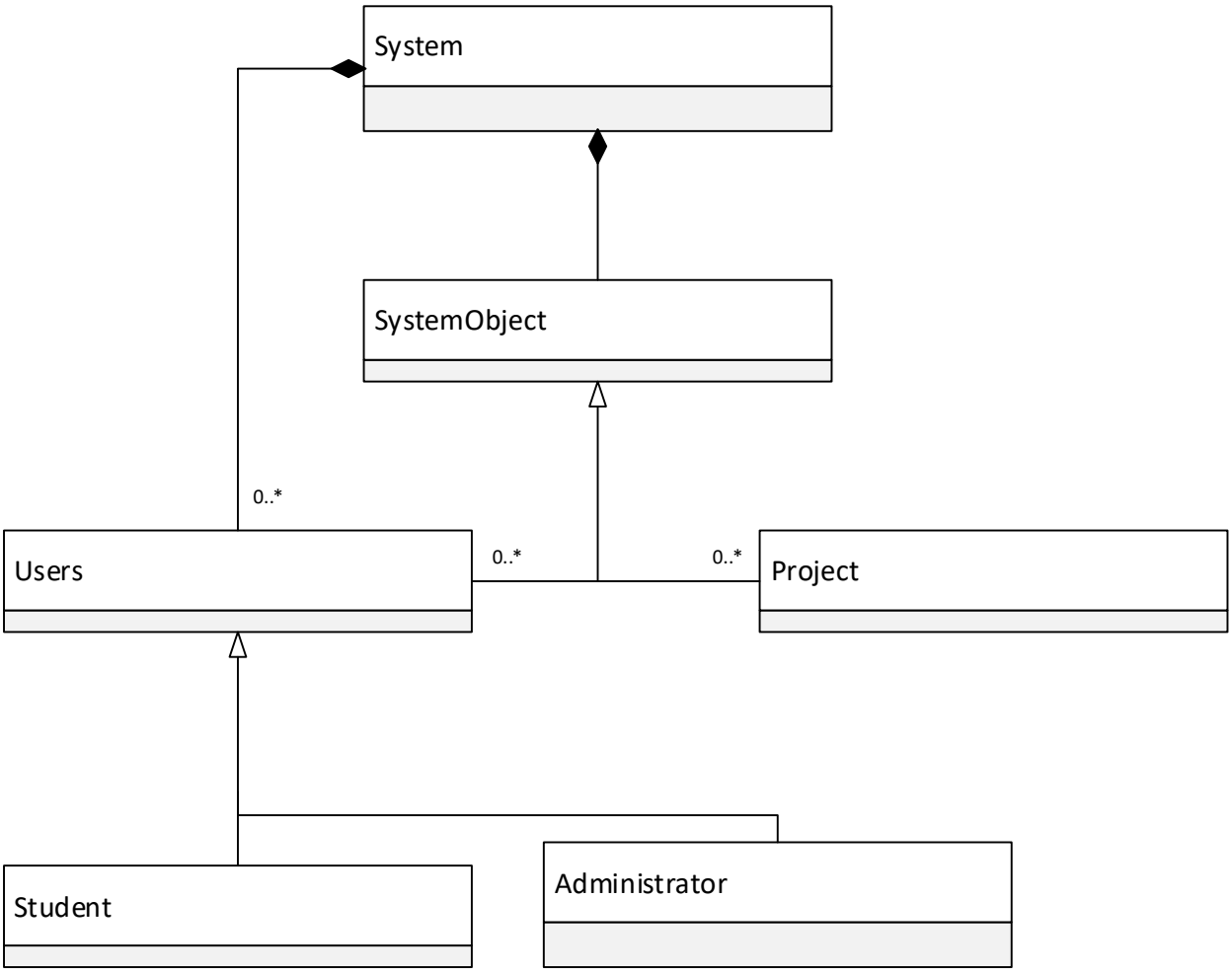
5. Object Model

The object model is focused on the entity objects of the cuPID system. It defines what the entity objects are as well as their associations and attributes.

5.1. Data Dictionary

Entity Object	Attributes / Associations	Definitions
User	UserNumber	• A user is a person who can access the cuPID system. • A user can be either a **student** or an **Administrator.**
	UserName	
Administrator	▪ PPID ▪ ListOfAllThe**Projects**	• The **Administrators** are the user with the highest privileges in the cuPID system. • The **Administrators** can create a new project. • The **Administrators** can edit an existing project. • Only the **Administrators** can access the PPID. • Only the **Administrators** launch the PPID. • Only the **Administrators** can modify the PPID. • Only **Administrators** can view the **students** enrolled in a project.
Student	▪ **Student**Number ▪ **Student**Name ▪ PersonalProfile ▪ PartnerProfile ▪ ListOfEnrolledProjects	• A **student** is required to fill out the partner profile in order to register into cuPID project. • A **student** is required to fill out their own profile in order to register into cuPID project. • A **student** can view the list of the projects available. • **Student** can enroll to one or more available projects. • A **student** can modify their own profile. • A **student** can modify their partner profile.
Projects	▪ ProjectName ▪ **Student**Lists ▪ PPIDAlgorithm ▪ ListOf**Students** ▪ ProjectID	• **Projects** hold description and list of **students** enrolled in it at a time. • **Projects** are created and modified by the **Administrator**. • **Projects** can be viewed and be enrolled into by the **students**. • **Projects** are accessed differently by the **Administrators** and **students.** • An **Administrator** has the highest privileges in the **projects.**

5.2 Class Diagram



6. Dynamic Model

A dynamic model is the representation of how the system behaves overtime. Here the state machine diagrams of the cuPID system. It is represented using two diagrams, the state machine and sequence diagrams. The state machine diagrams show the different state an entity object can have whereas the sequences diagram show the procedural behavior of the use-case. The sequence diagram shows how actors boundary objects, control objects and entity objects interact with each other.

6.1 State Machine Diagrams

State machine for Student entity object (UC – 05, UC-06):

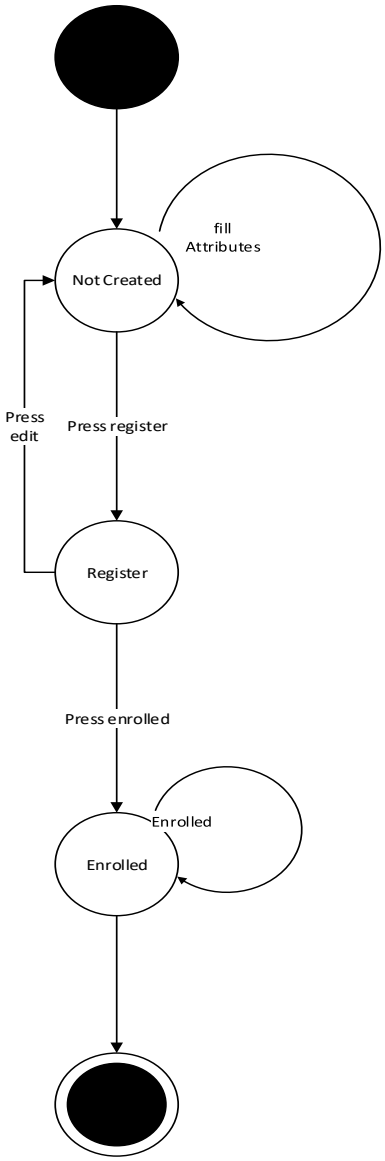


Figure 9 – Student RegistersincuPID and Enrolls in Project

State machine for Student entity object (UC – 05):

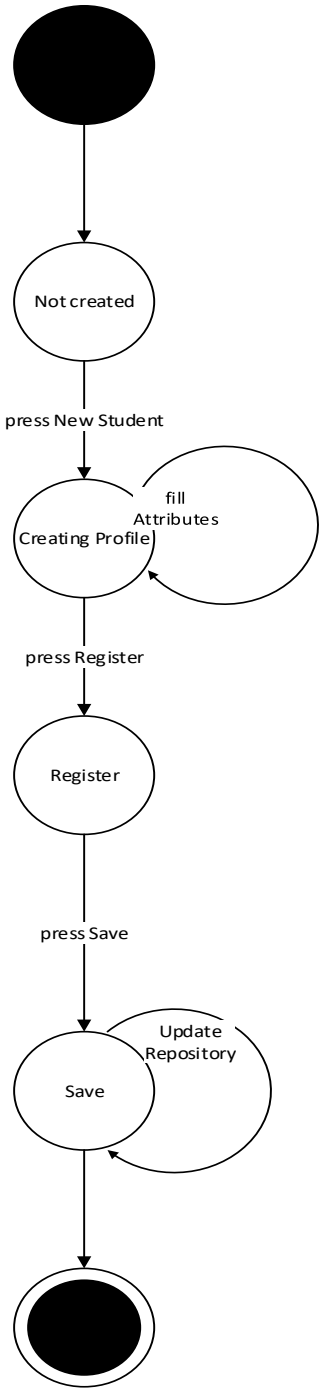


Figure 10 – Student Registers in cuPID

State machine for Student entity object (UC – 07):

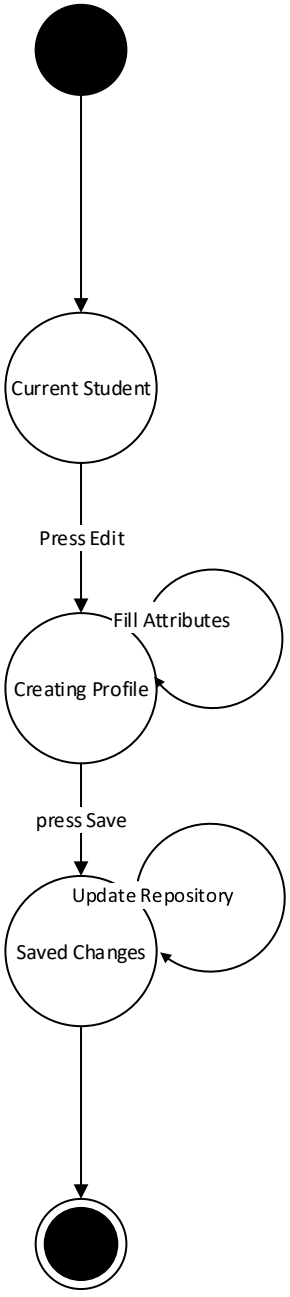


Figure 11 – Current Student Edits Profile

State machine for Administrator entity object (UC – 01):

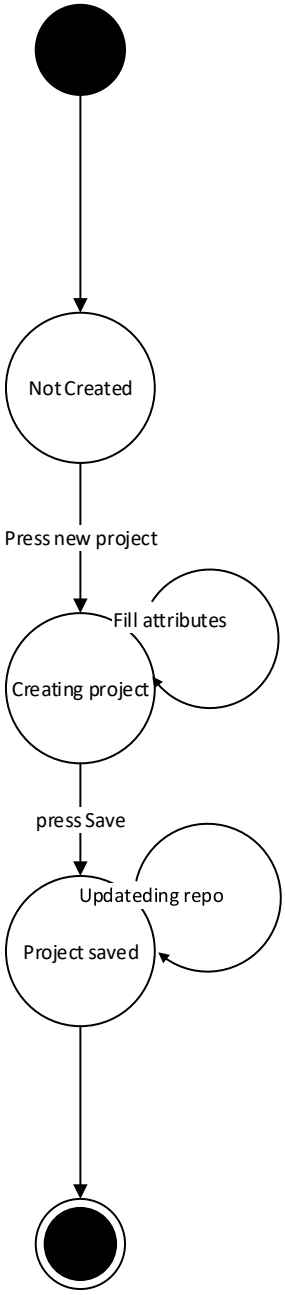


Figure 12 – CreateNewProject

State machine for Administrator entity object (UC – 02):

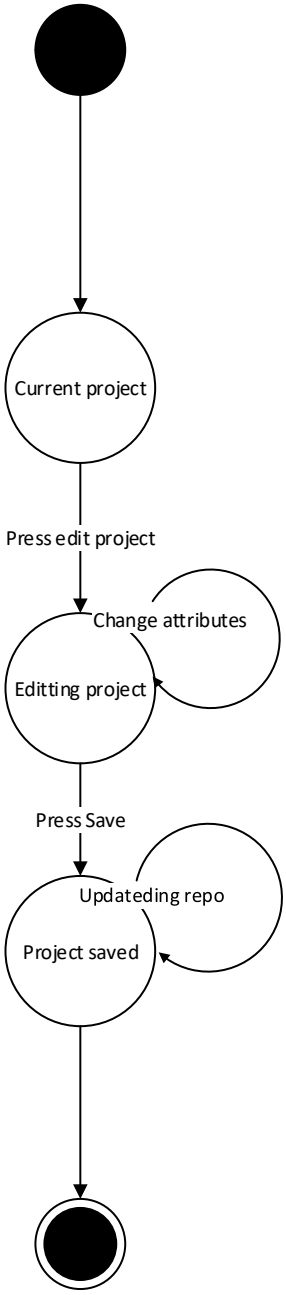


Figure 13 – Edit Project

6.2 Sequence Diagrams

Sequence Diagram for CreateProject (UC – 01):

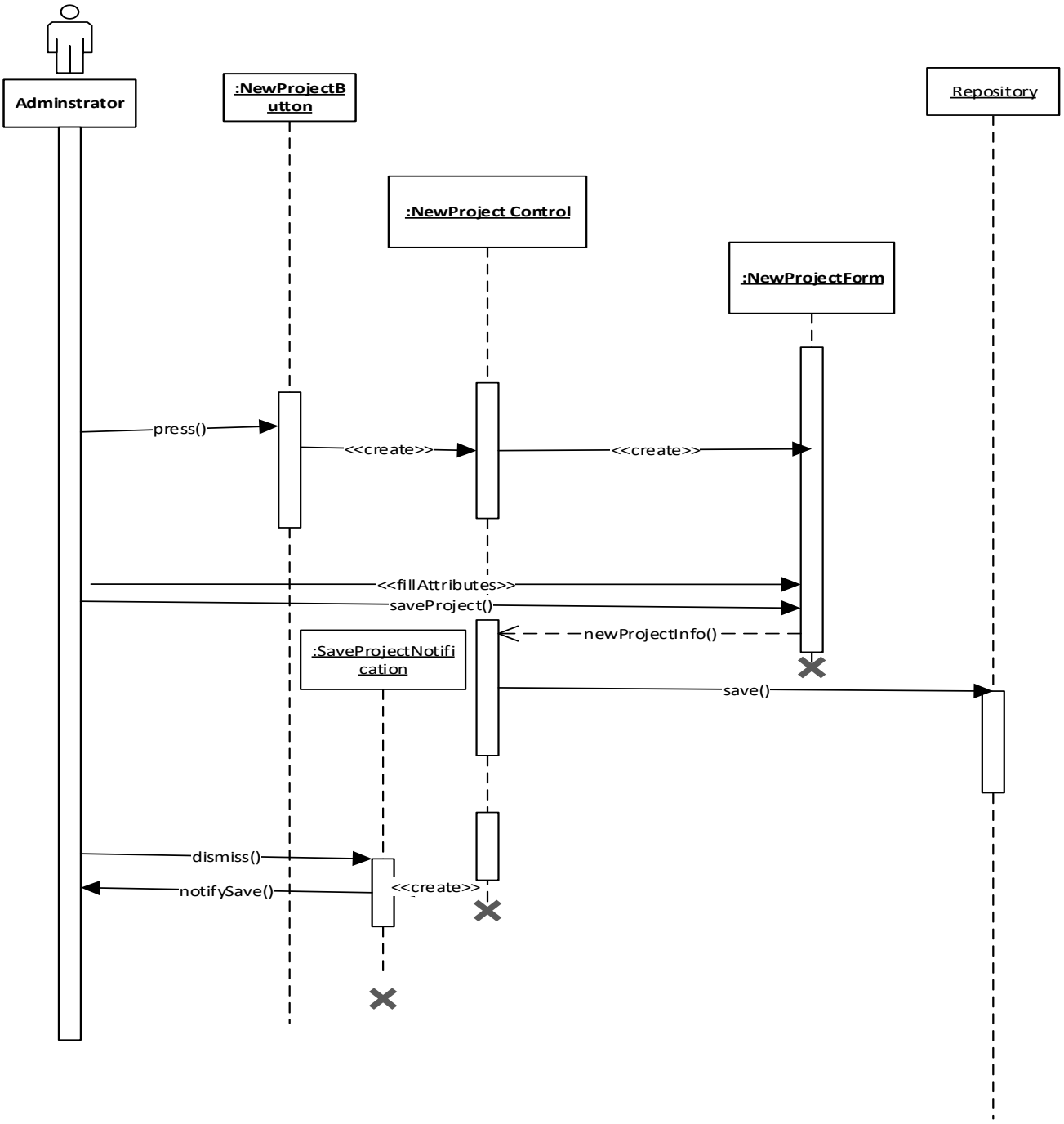


Figure 14 – CreateProject Sequence Diagram

Sequence Diagram for EditProject (UC –02):

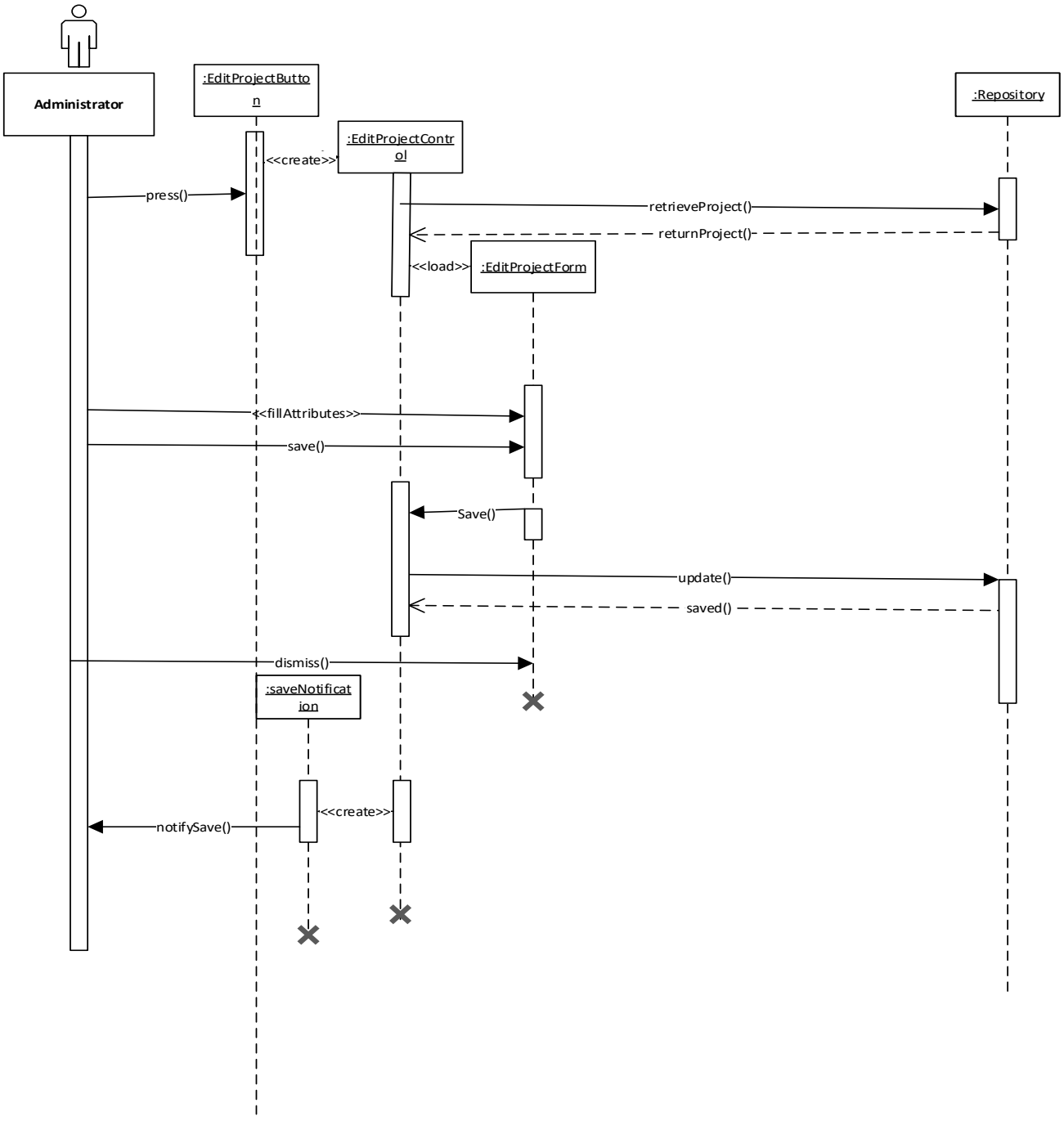


Figure 15 – EditProject Sequence Diagram

Sequence Diagram for ViewProject (UC – 17):

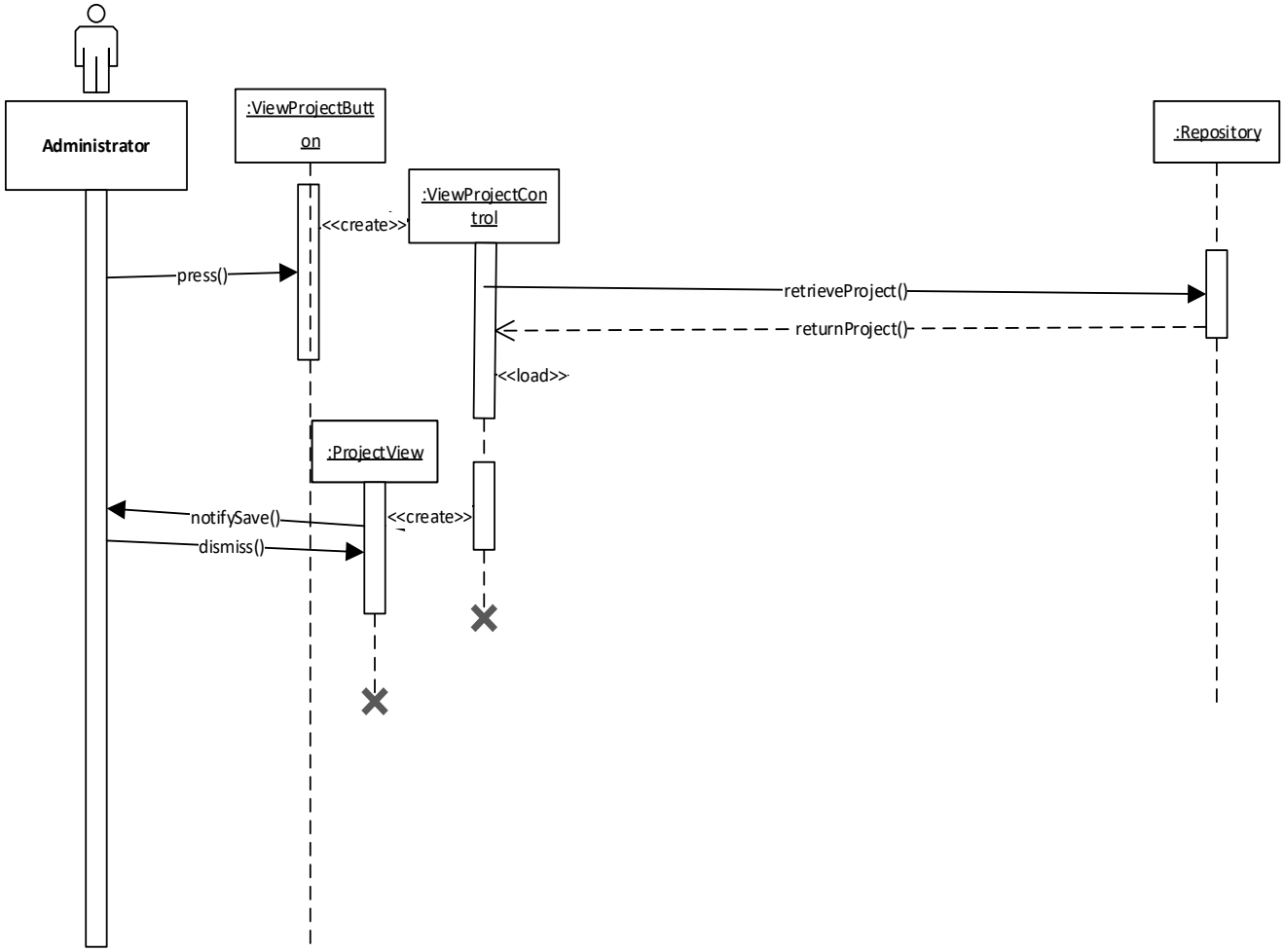


Figure 16 – ViewProject Sequence Diagram for Administrator

Sequence Diagram for LaunchPPID (UC – 03):

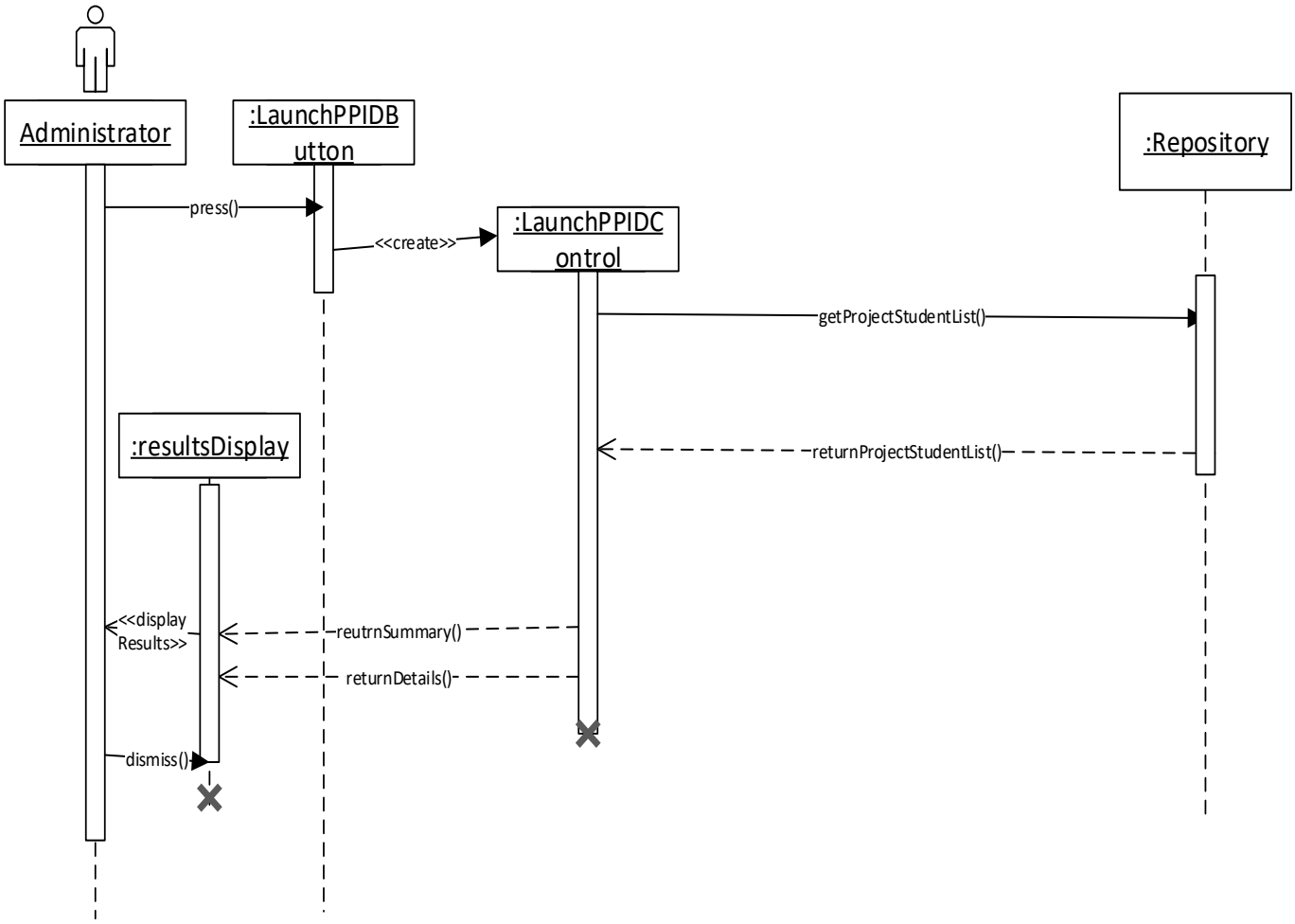


Figure 17 – LaunchPPID Sequence Diagram

Sequence Diagram for ModifyPPID (UC – 04):

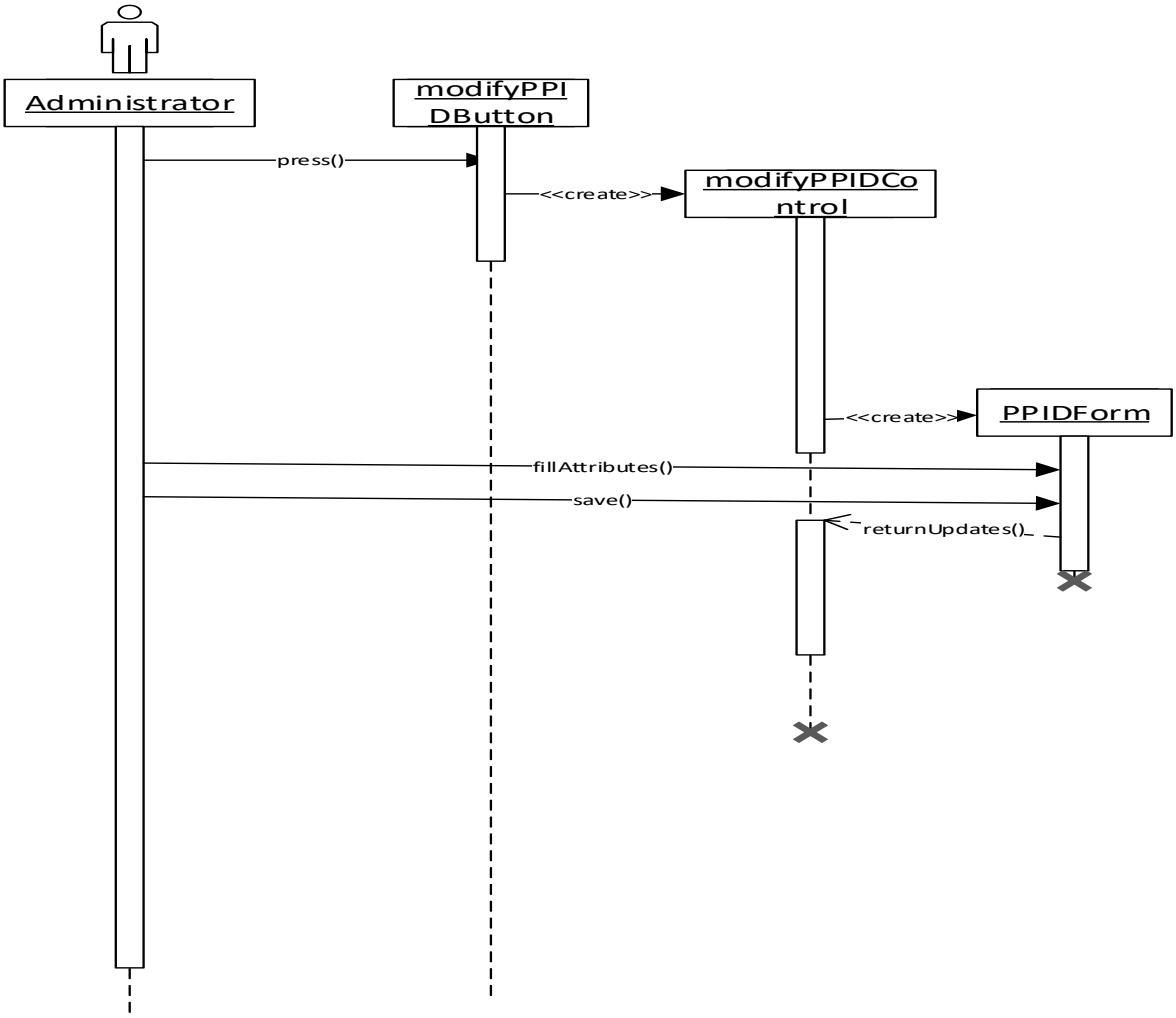


Figure 18 – ModifyPPID Sequence Diagram

Sequence Diagram for RegisterInCupid (UC – 05):

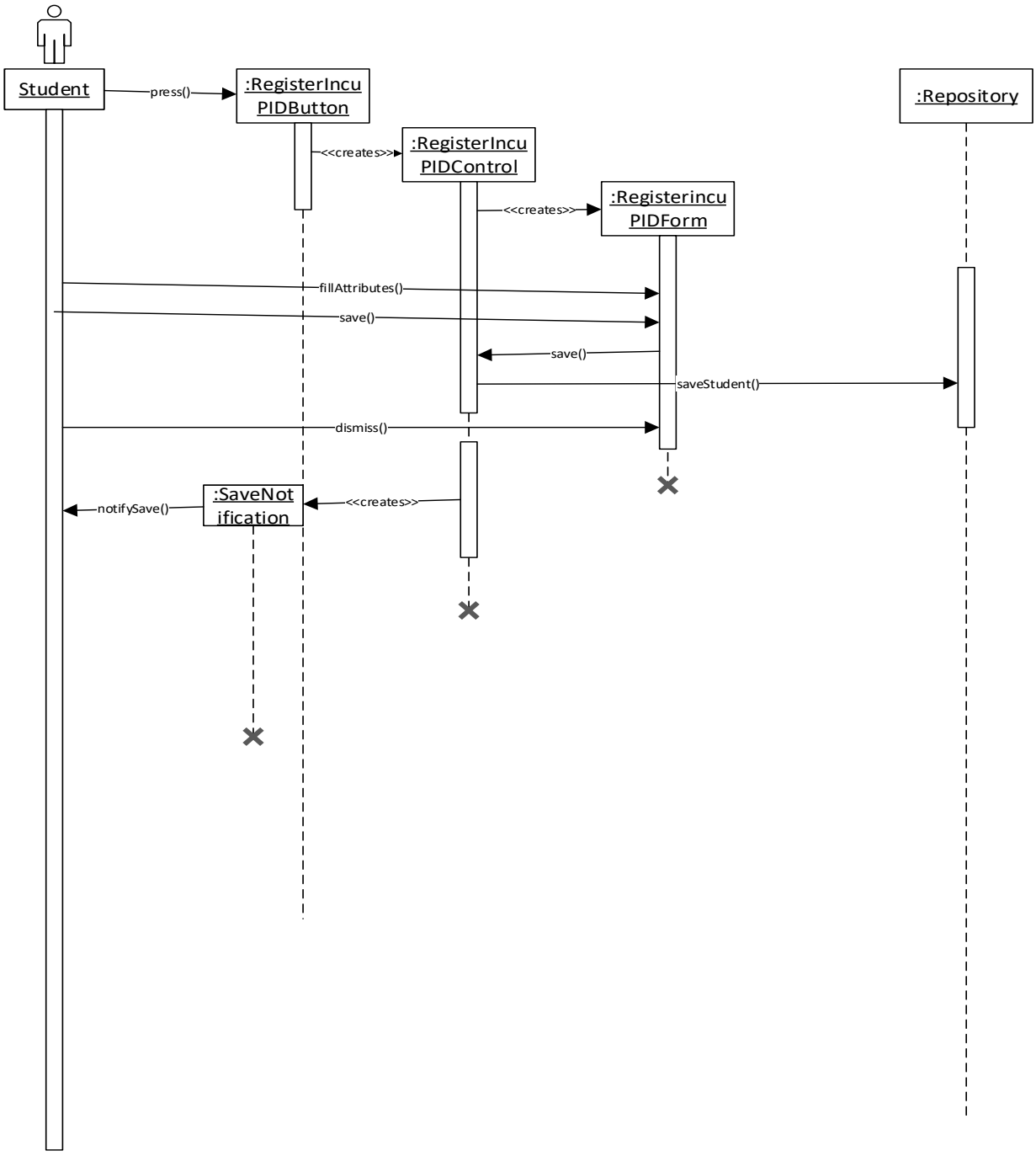


Figure 19 – RegisterInCupid Sequence Diagram

Sequence Diagram for EnrollInProject (UC – 06):

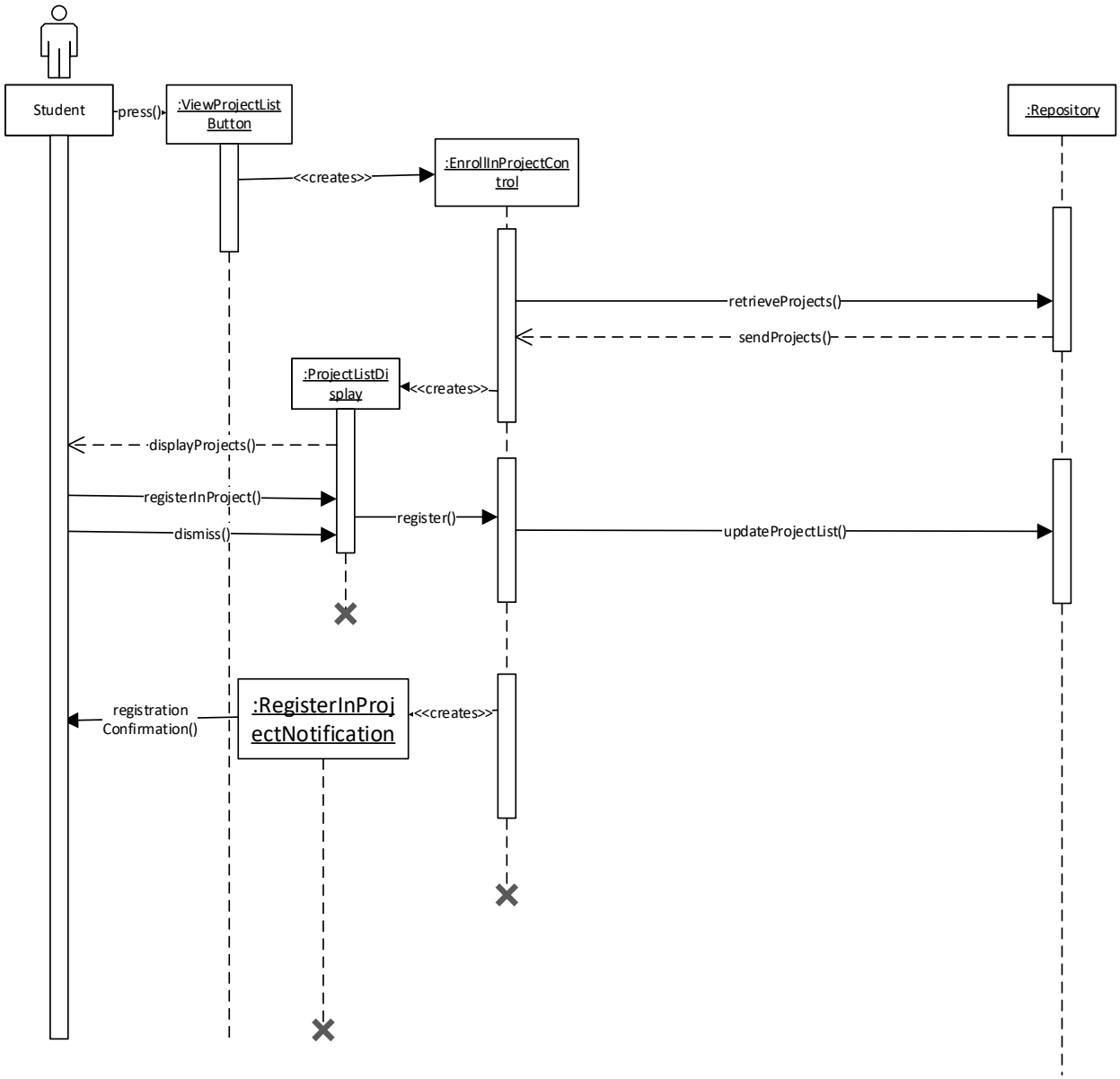


Figure 20 – EnrollInProject Sequence Diagram

Sequence Diagram for EditProfile (UC – 07):

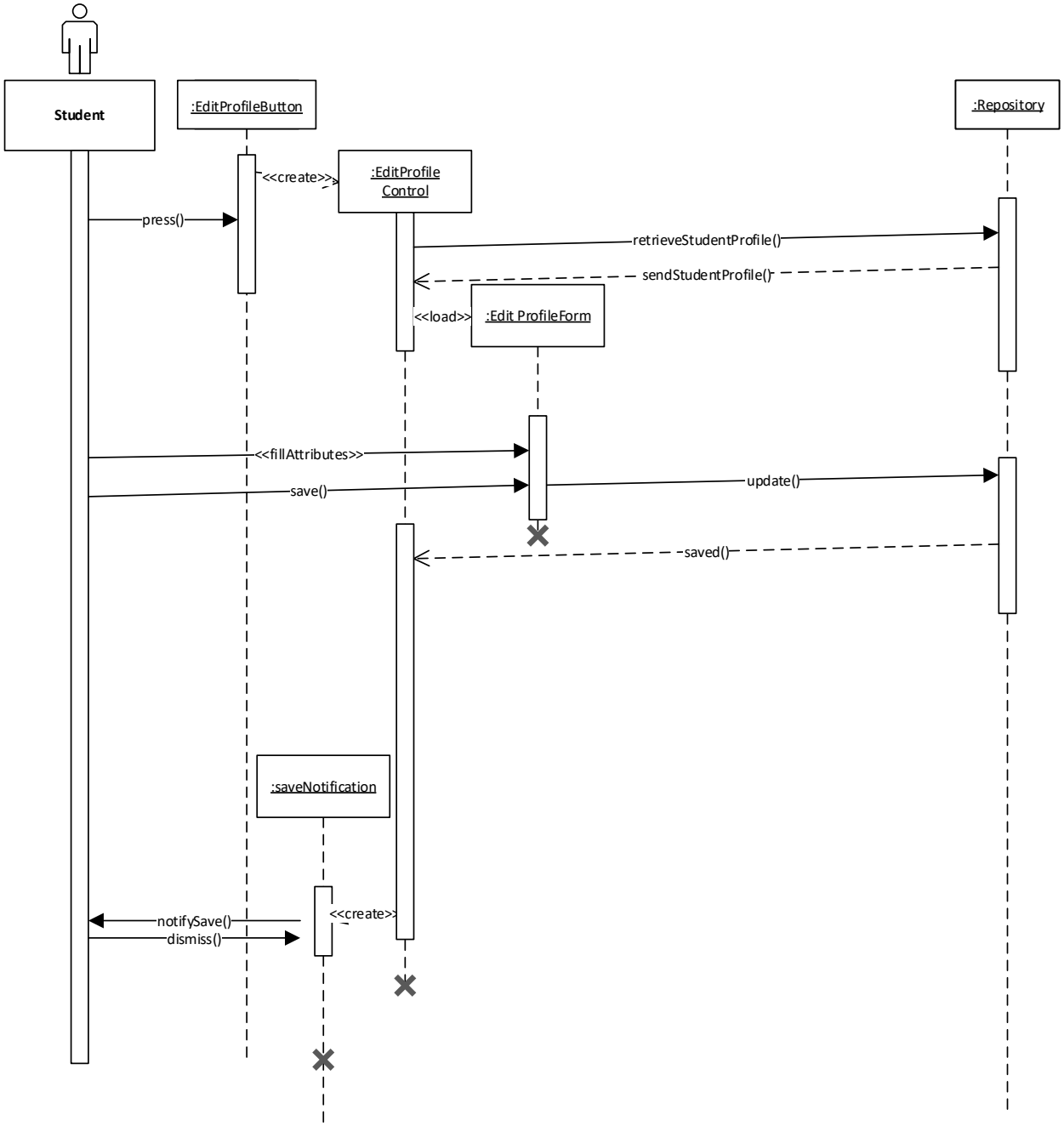


Figure 21 – EditProfile Sequence Diagram

Sequence Diagram for ViewProfile (UC – 18):

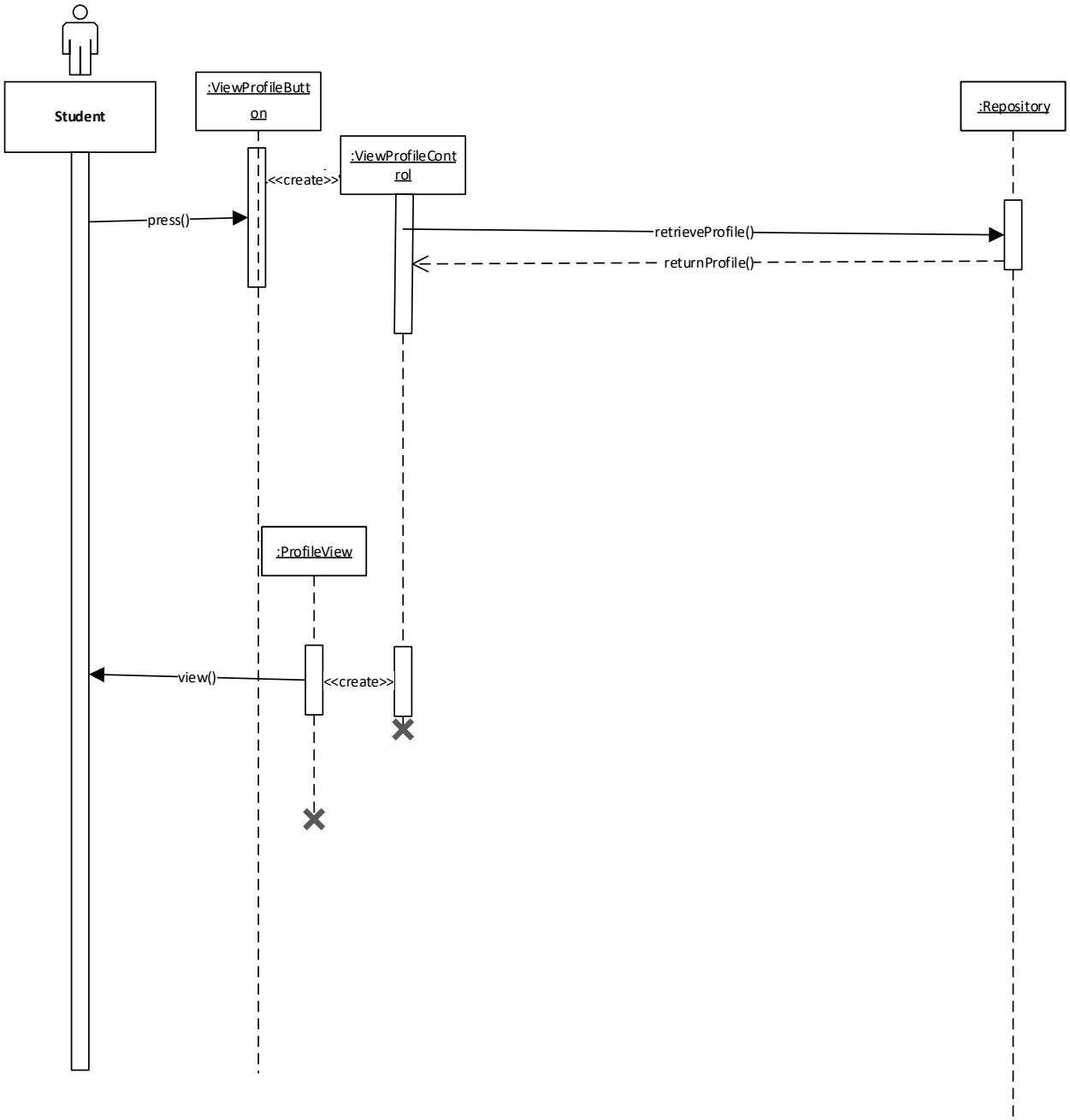


Figure 22 – ViewProfile Sequence Diagram

Sequence Diagram for ViewProject (UC – 19):

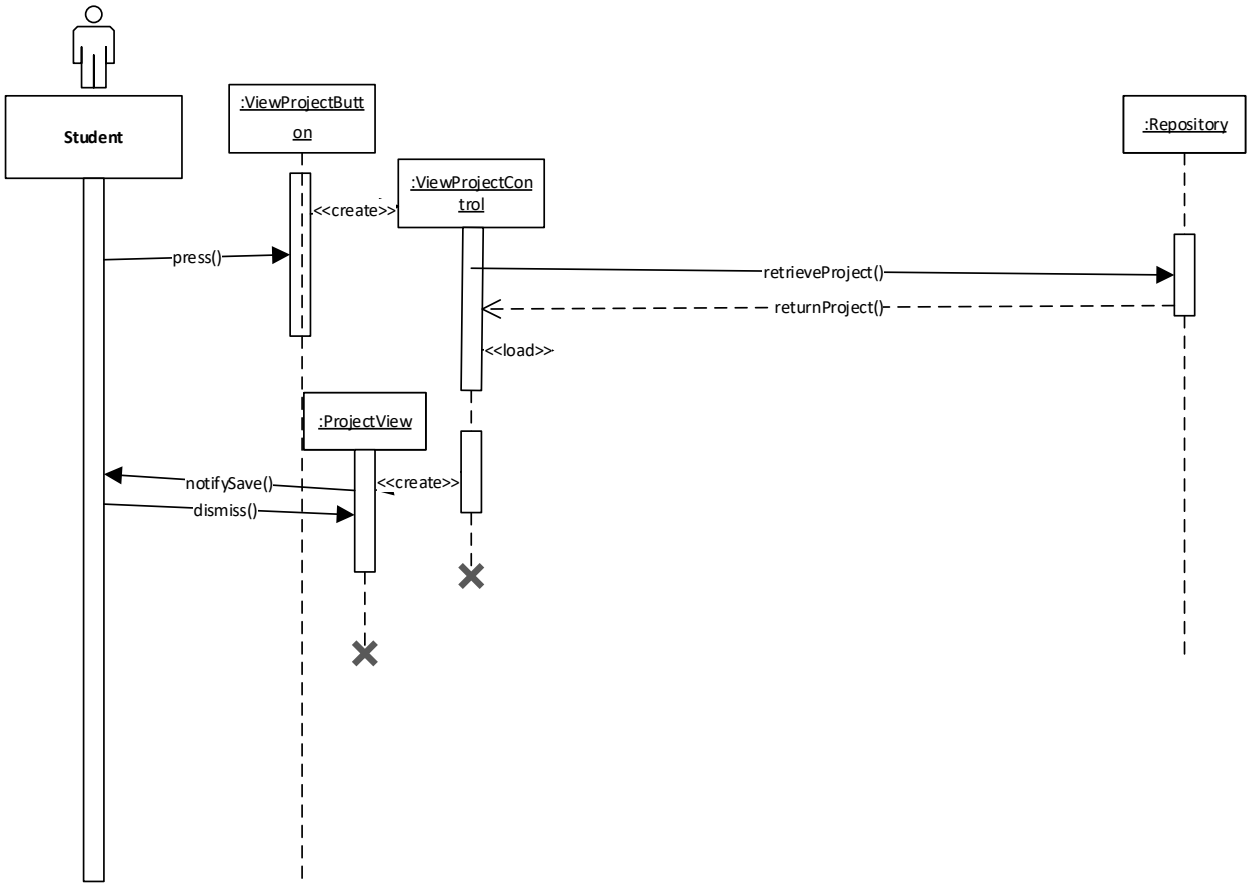


Figure 23 – ViewProject Sequence Diagram for Student

Sequence Diagram for EditProjectReadError (UC – 08, UC - 02):

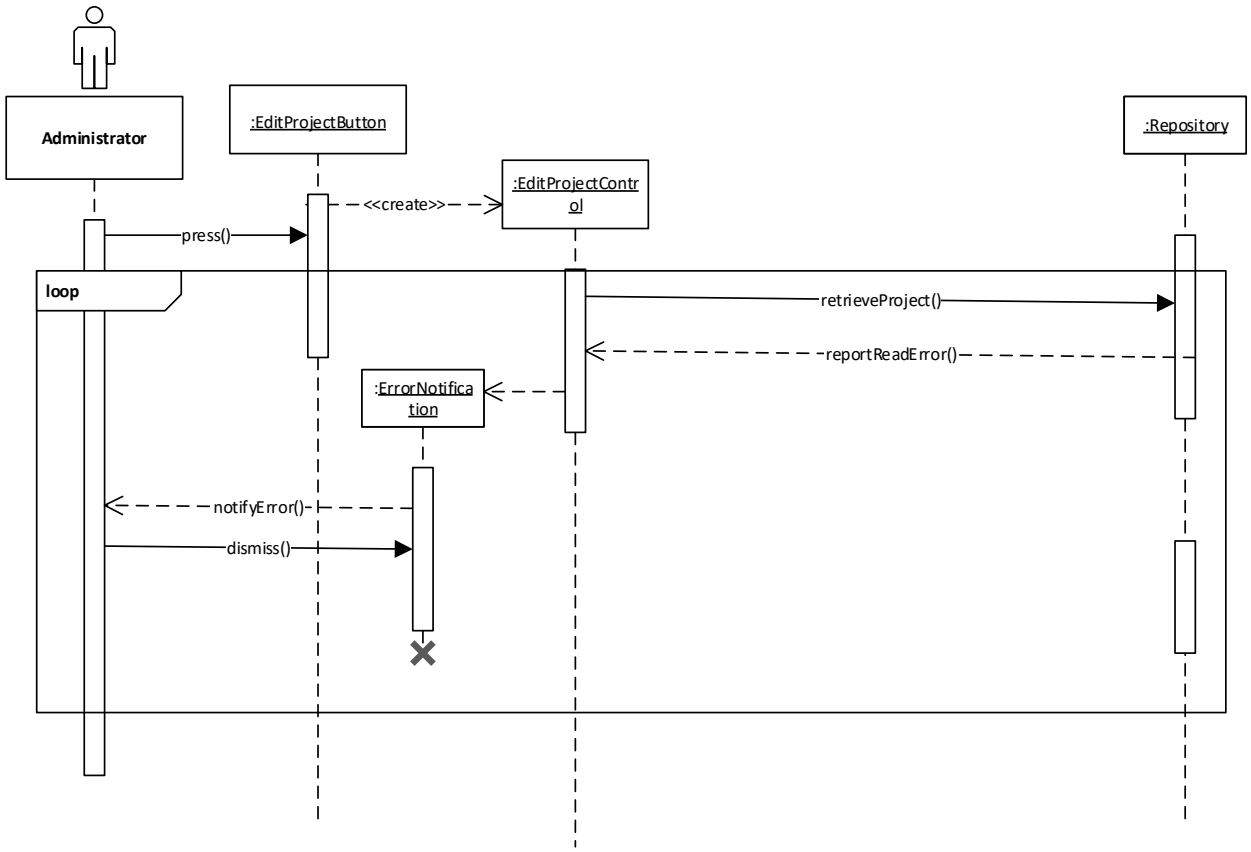


Figure 24 – EditProjectReadError Sequence Diagram

Sequence Diagram for LaunchPPIDReadError (UC – 08, UC - 03):

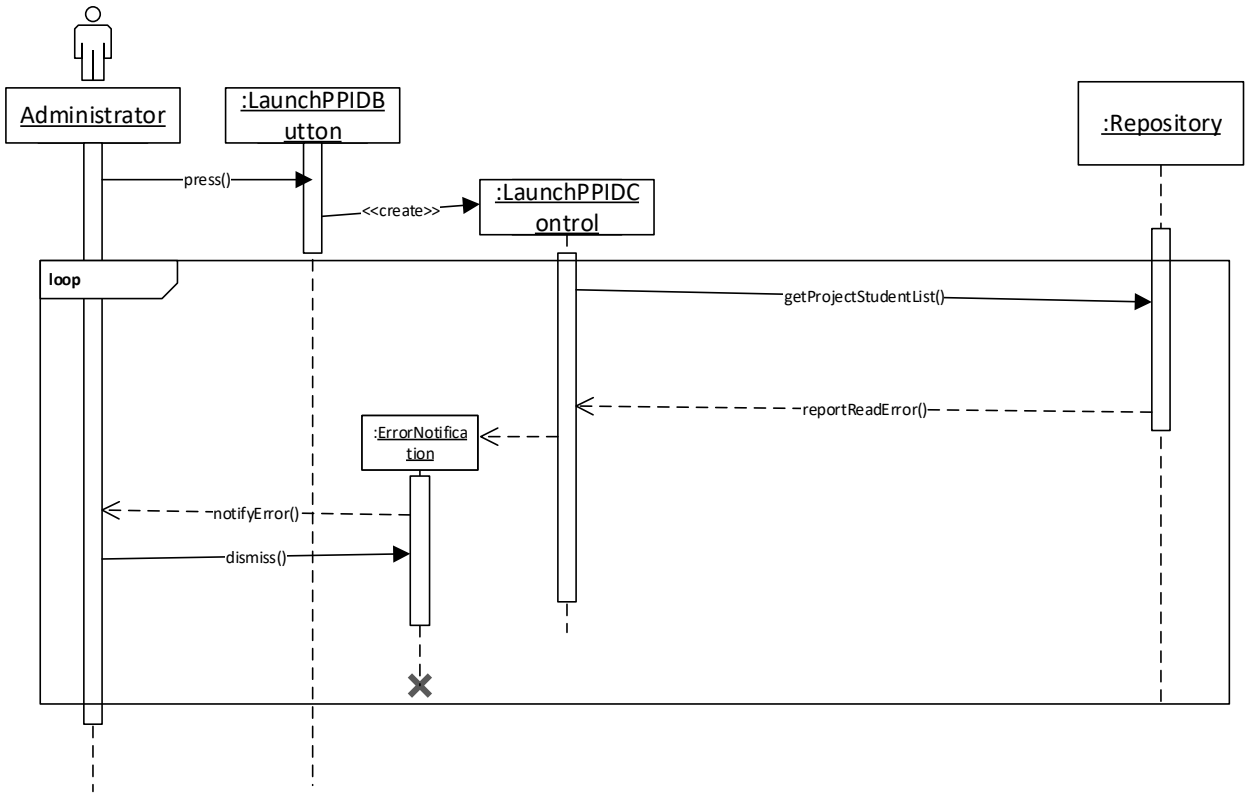


Figure 25 – LaunchPPIDReadError Sequence Diagram

Sequence Diagram for EnrollProjectReadError (UC – 08, UC - 06):

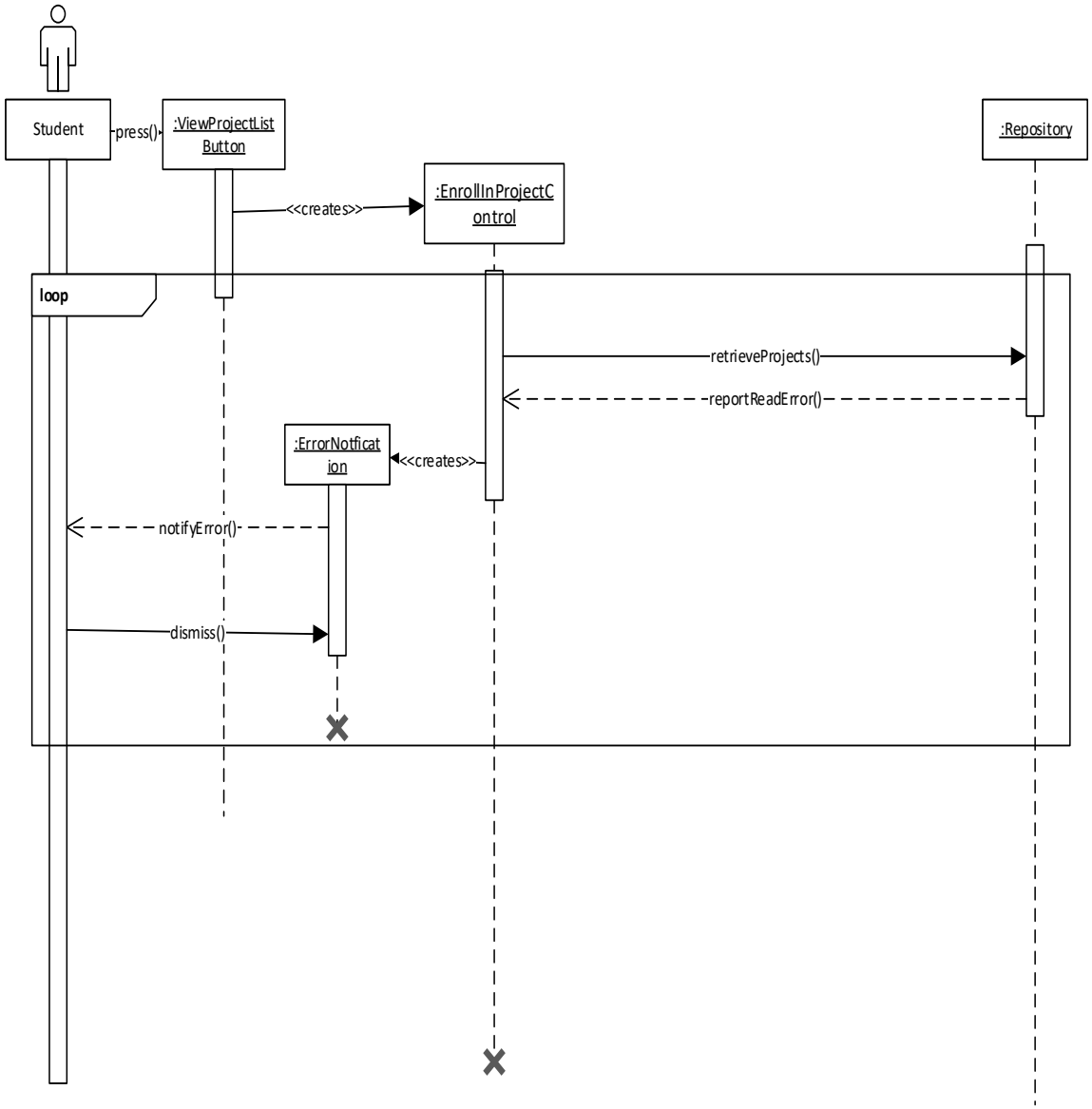


Figure 26 – EnrollInProjectReadError Sequence Diagram

Sequence Diagram for EditProfileReadError (UC – 08, UC - 07):

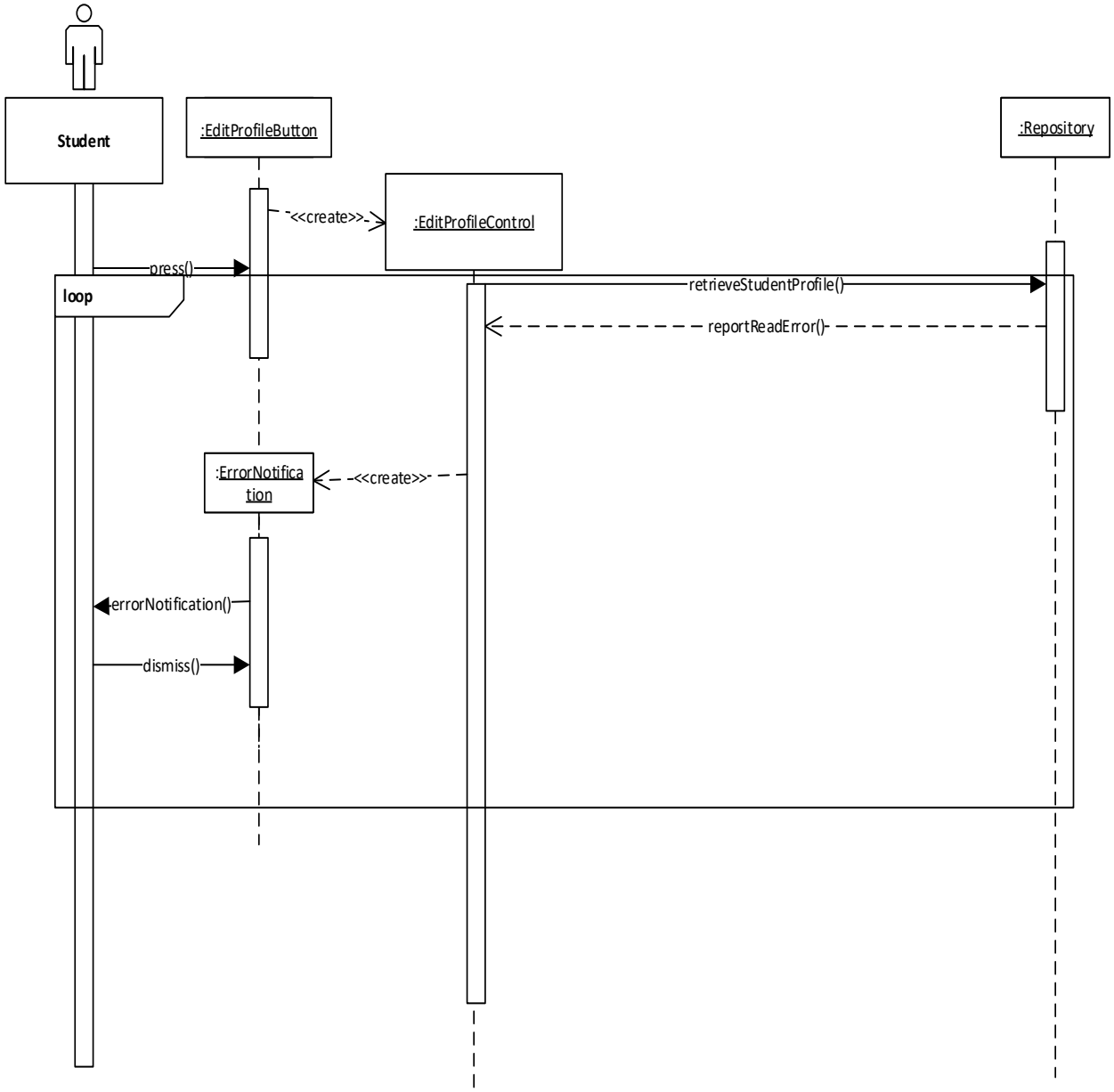


Figure 27– EditProfileReadError Sequence Diagram

Sequence Diagram for CreateProjectWriteError (UC – 09, UC - 01):

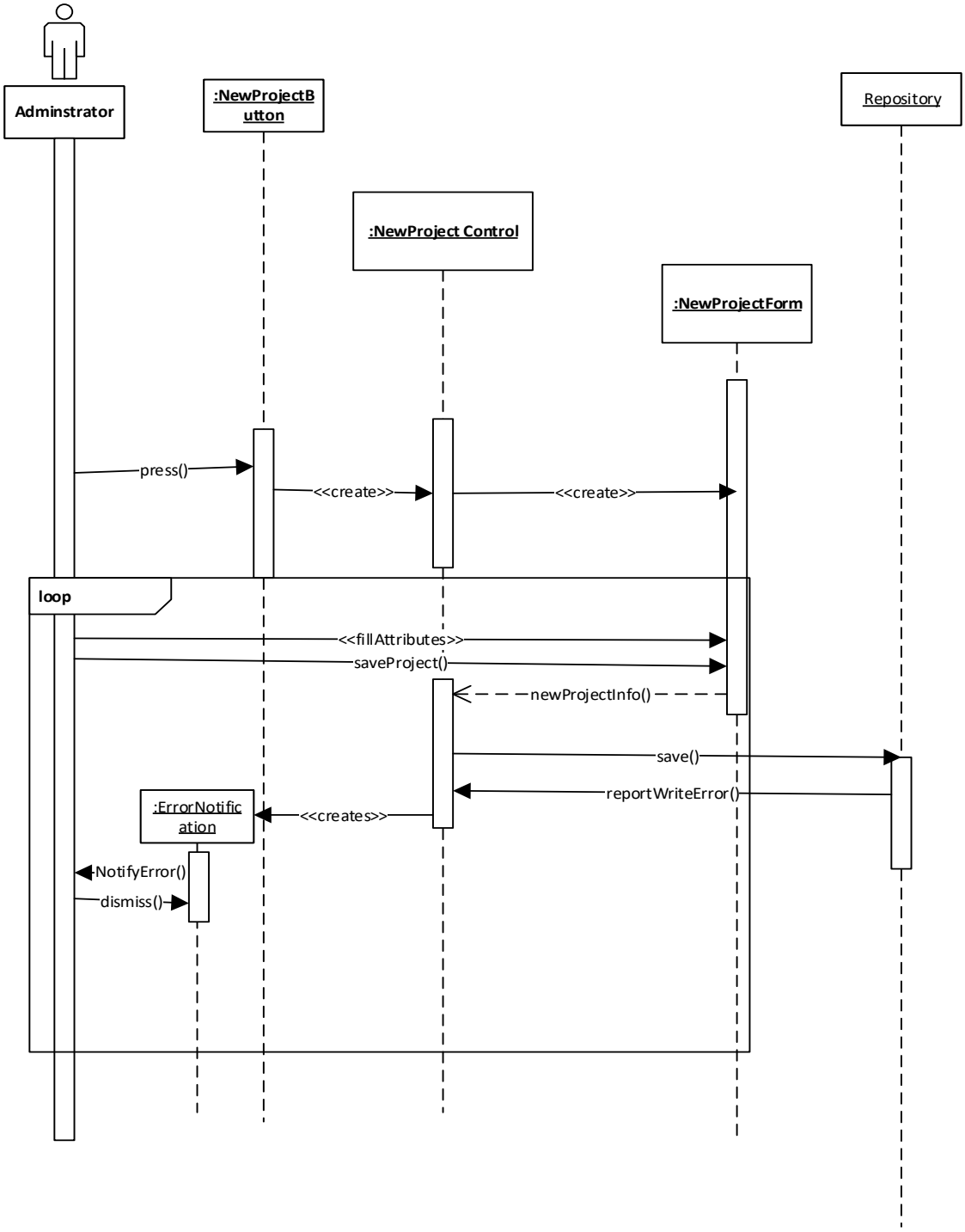


Figure 28– CreateProjectWtriteError Sequence Diagram

Sequence Diagram for EditProjectWriteError (UC – 09, UC - 02):

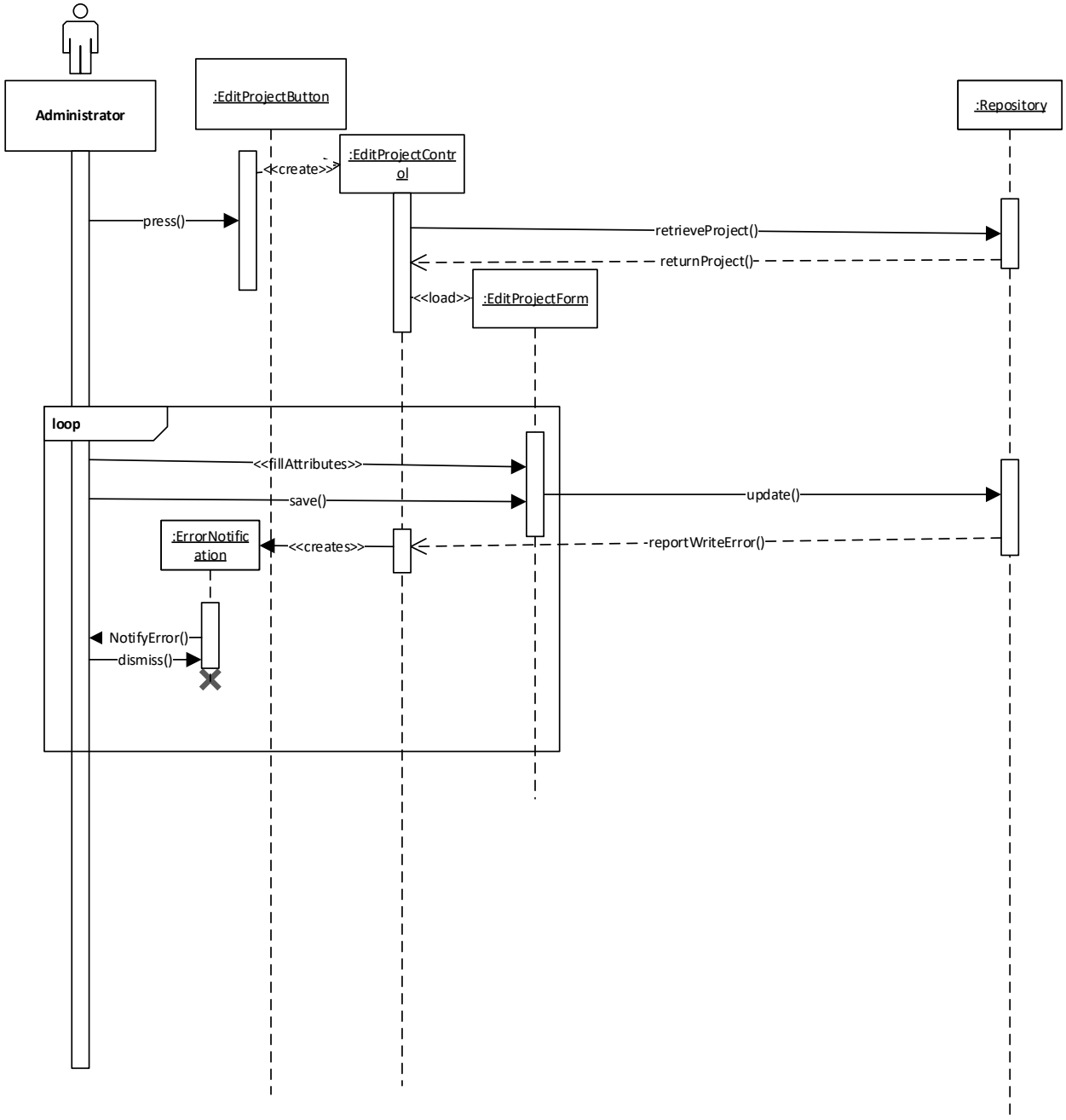


Figure 29– EditProjectWriteError Sequence Diagram

Sequence Diagram for RegisterInCUPIDWriteError (UC – 09, UC - 05):

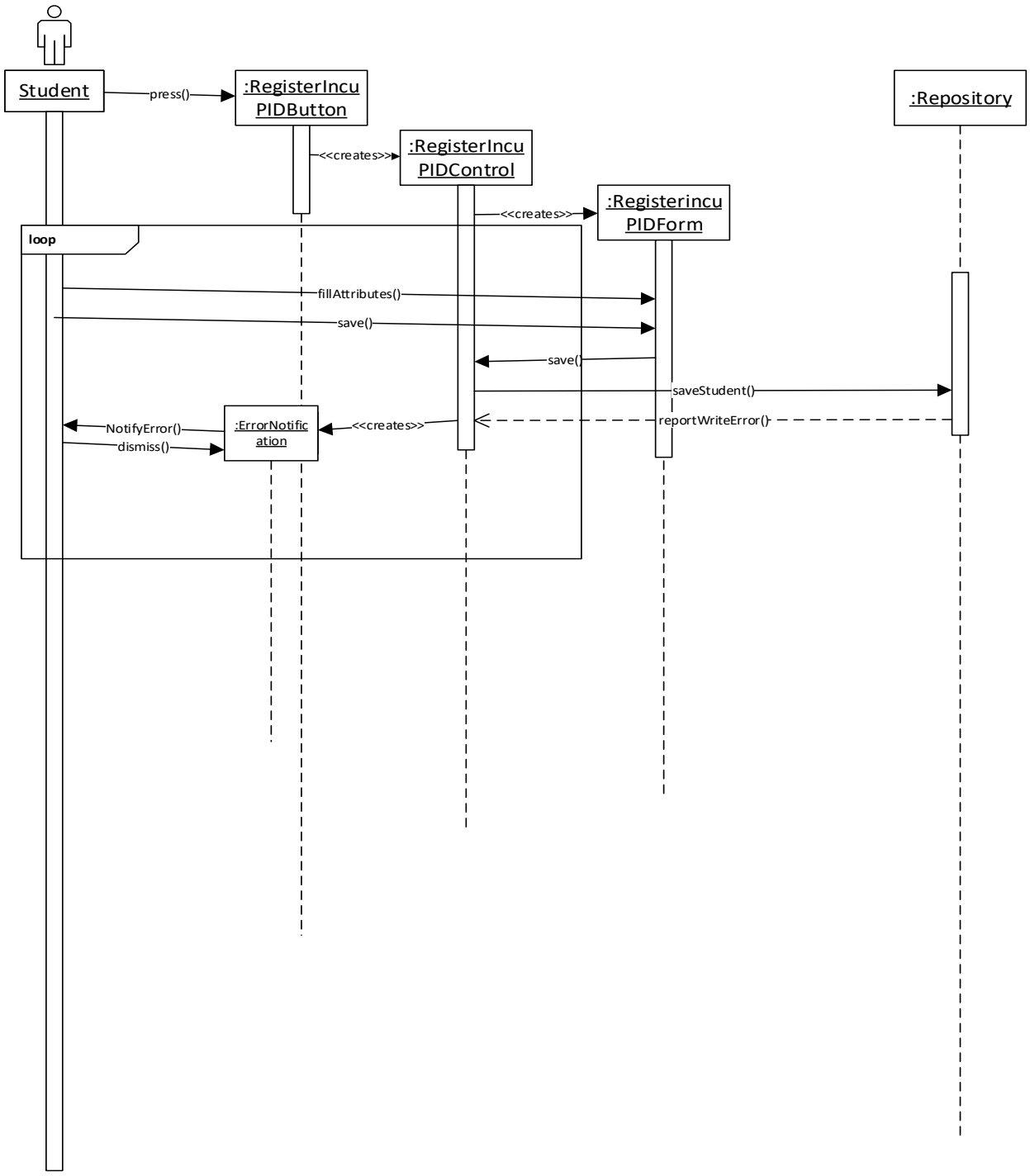


Figure 30– RegisterInCUPIDWtriteError Sequence Diagram

Sequence Diagram for EnrollInProjectWtriteError (UC – 09, UC - 06):

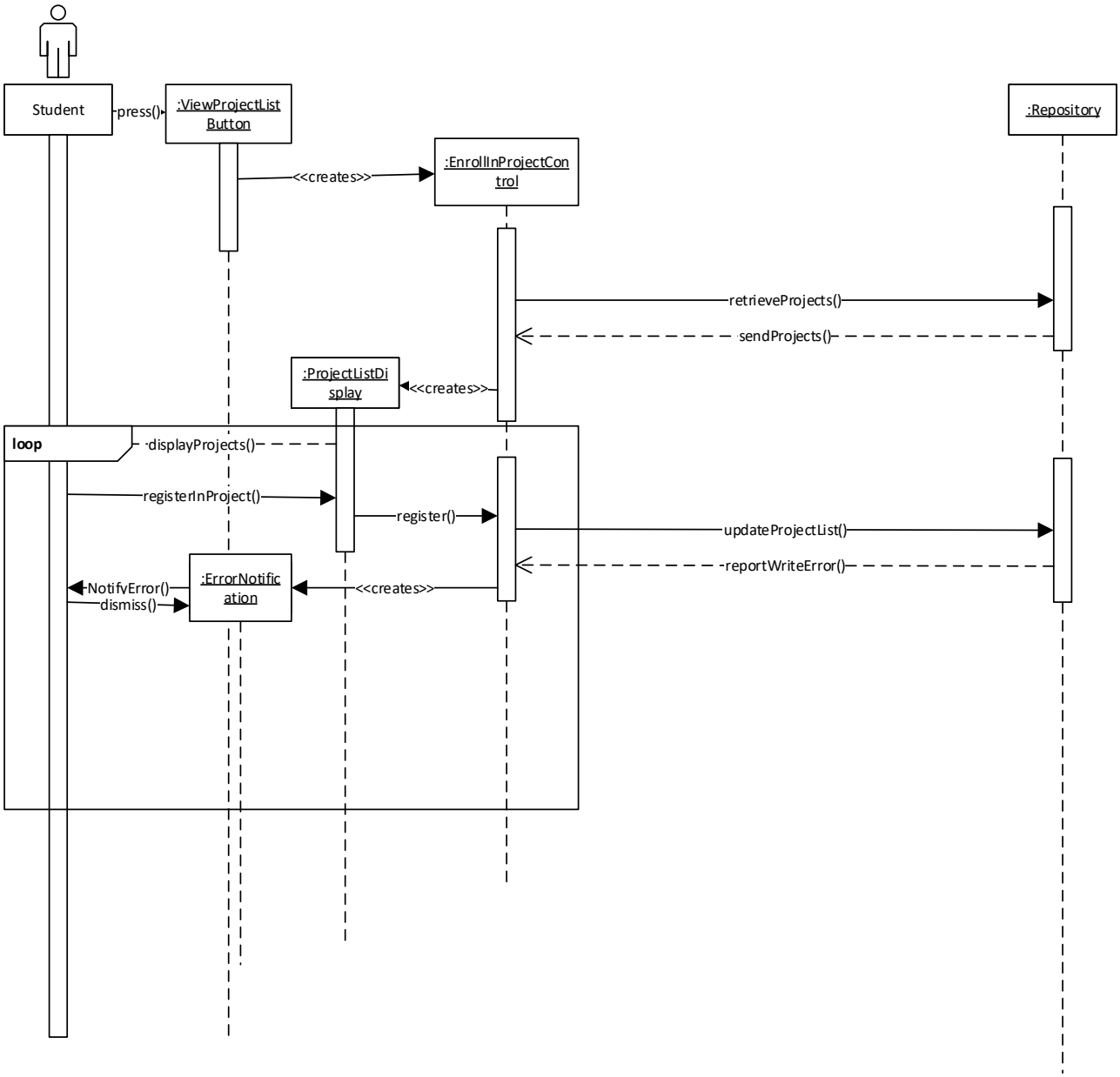


Figure 31– EnrollInProjectWtriteError Sequence Diagram

Sequence Diagram for EditProfileWtriteError (UC – 09, UC - 07):

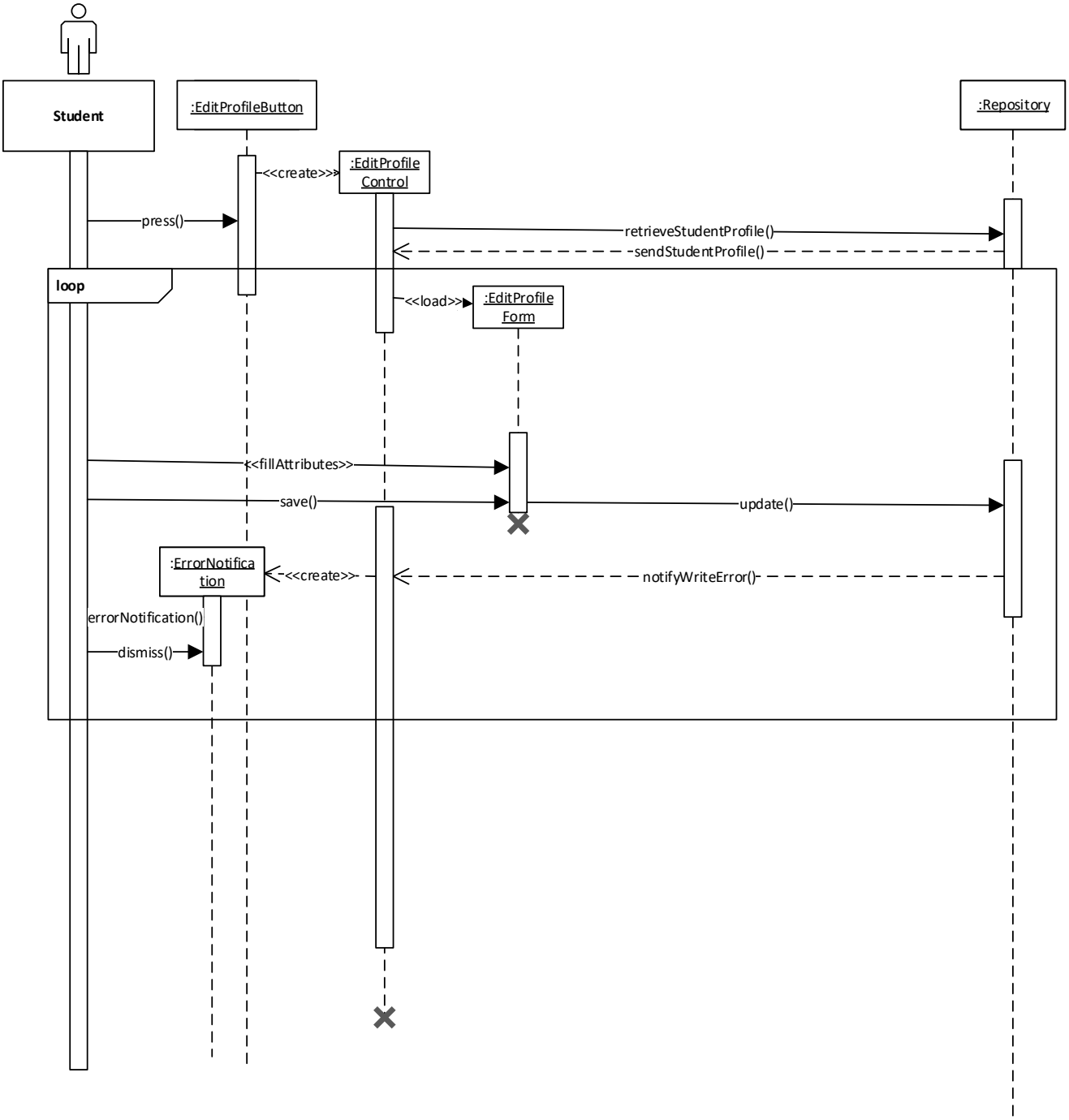


Figure 32– EditProfiletWtriteError Sequence Diagram

Sequence Diagram for CreateNewProjectMissingFieldError (UC – 10, UC - 01):

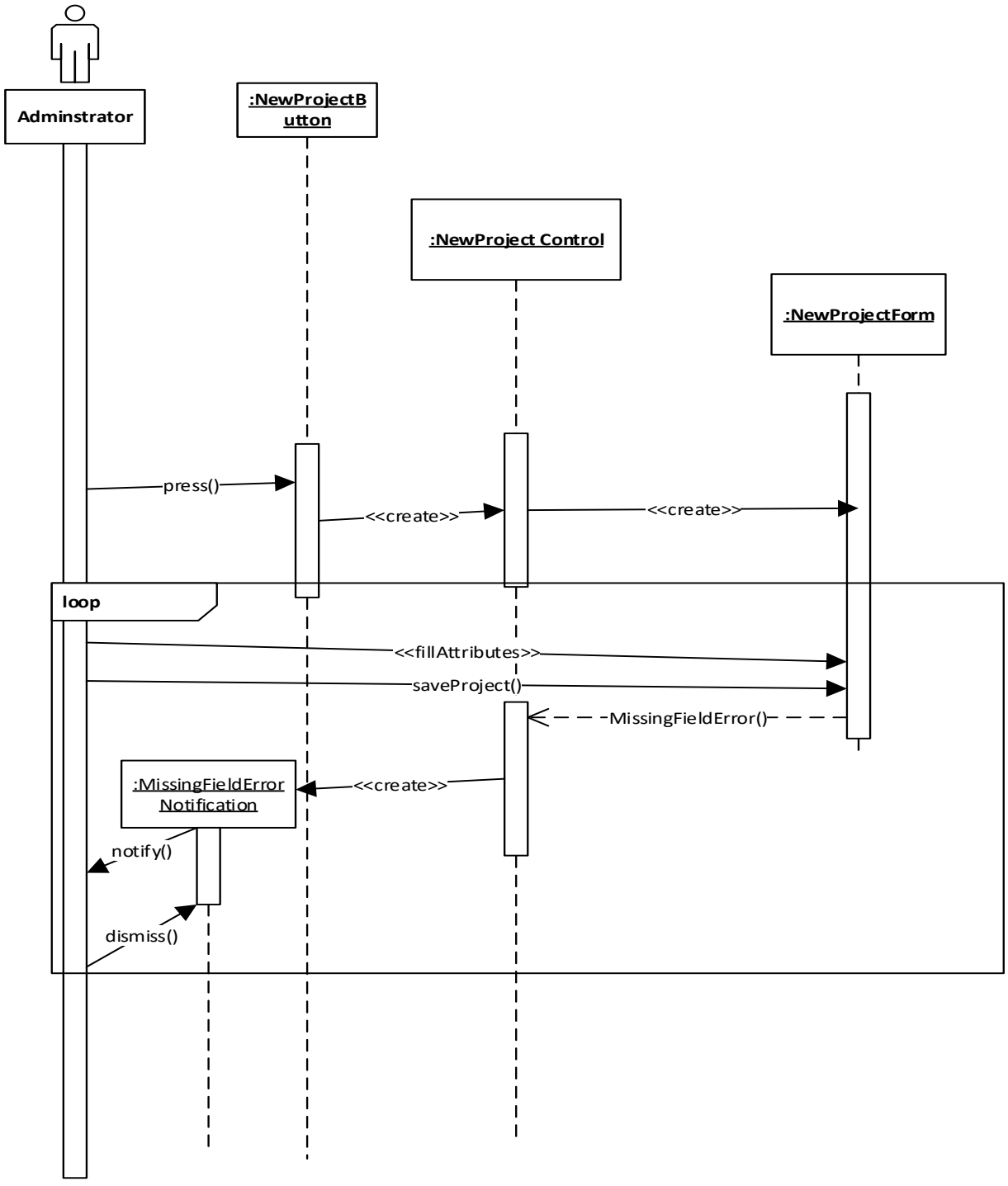


Figure 33– CreateNewProjectMissingFieldError Sequence Diagram

Sequence Diagram for EditProjectMissingFieldAttributeError (UC – 10, UC - 02):

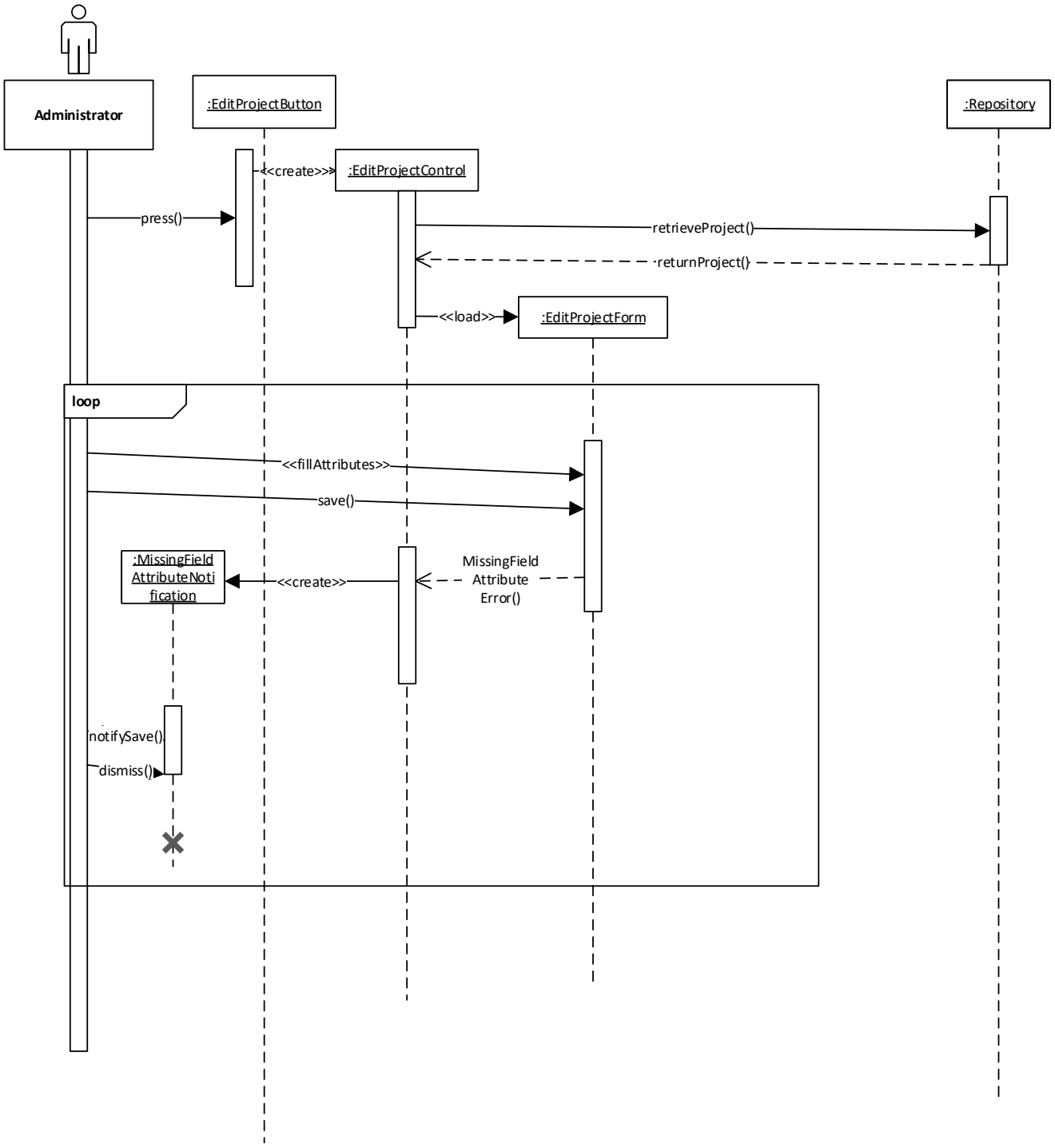


Figure 34– EditProjectMissingFieldAttributeError Sequence Diagram

Sequence Diagram for ModifyPPIDMissingFieldAttributeError (UC – 10, UC - 04):

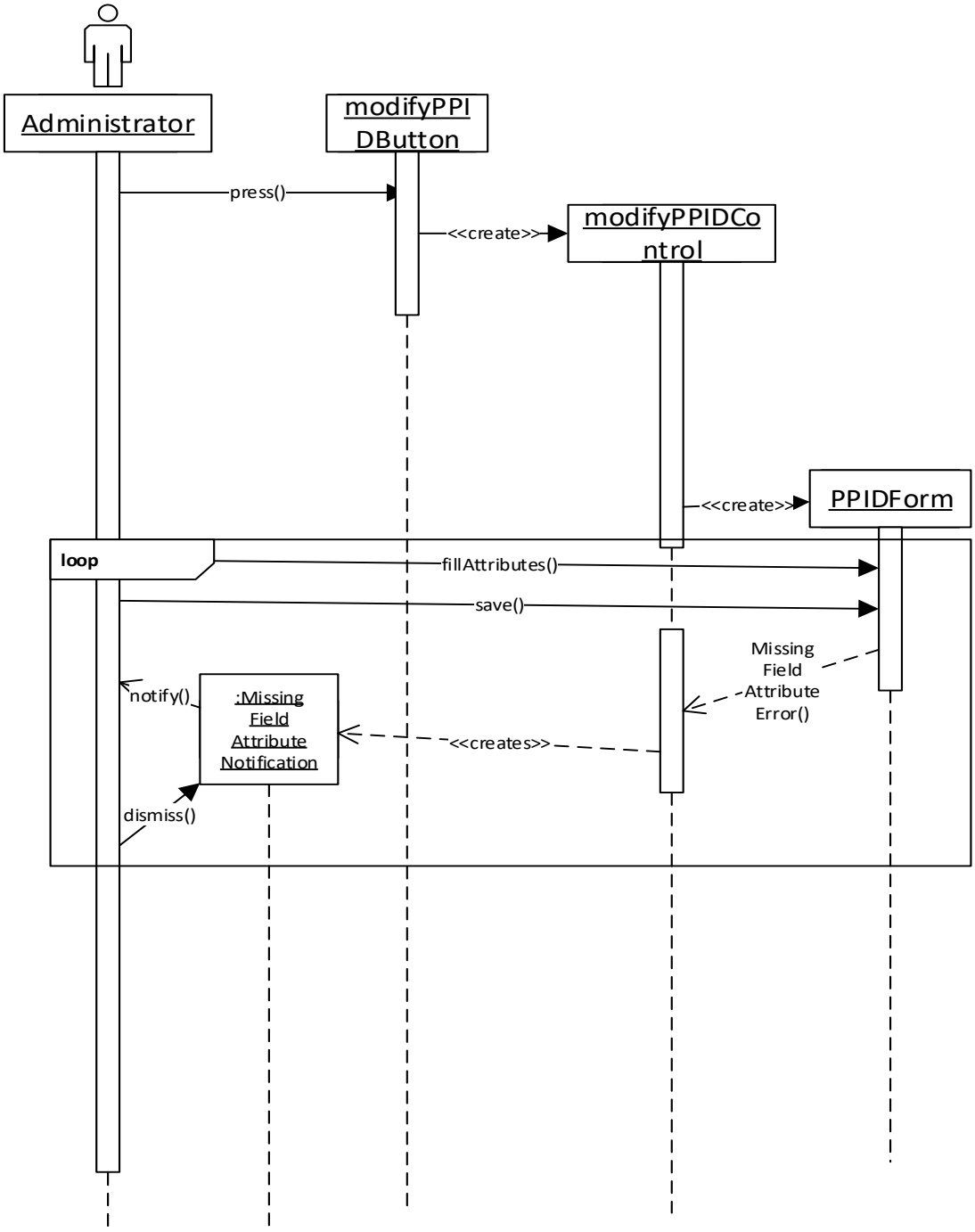


Figure 35– ModifyPPIDMissingFieldAttributeError Sequence Diagram

Sequence Diagram for RegisterInCUPIDMissingFieldAttributeError (UC – 10, UC - 05):

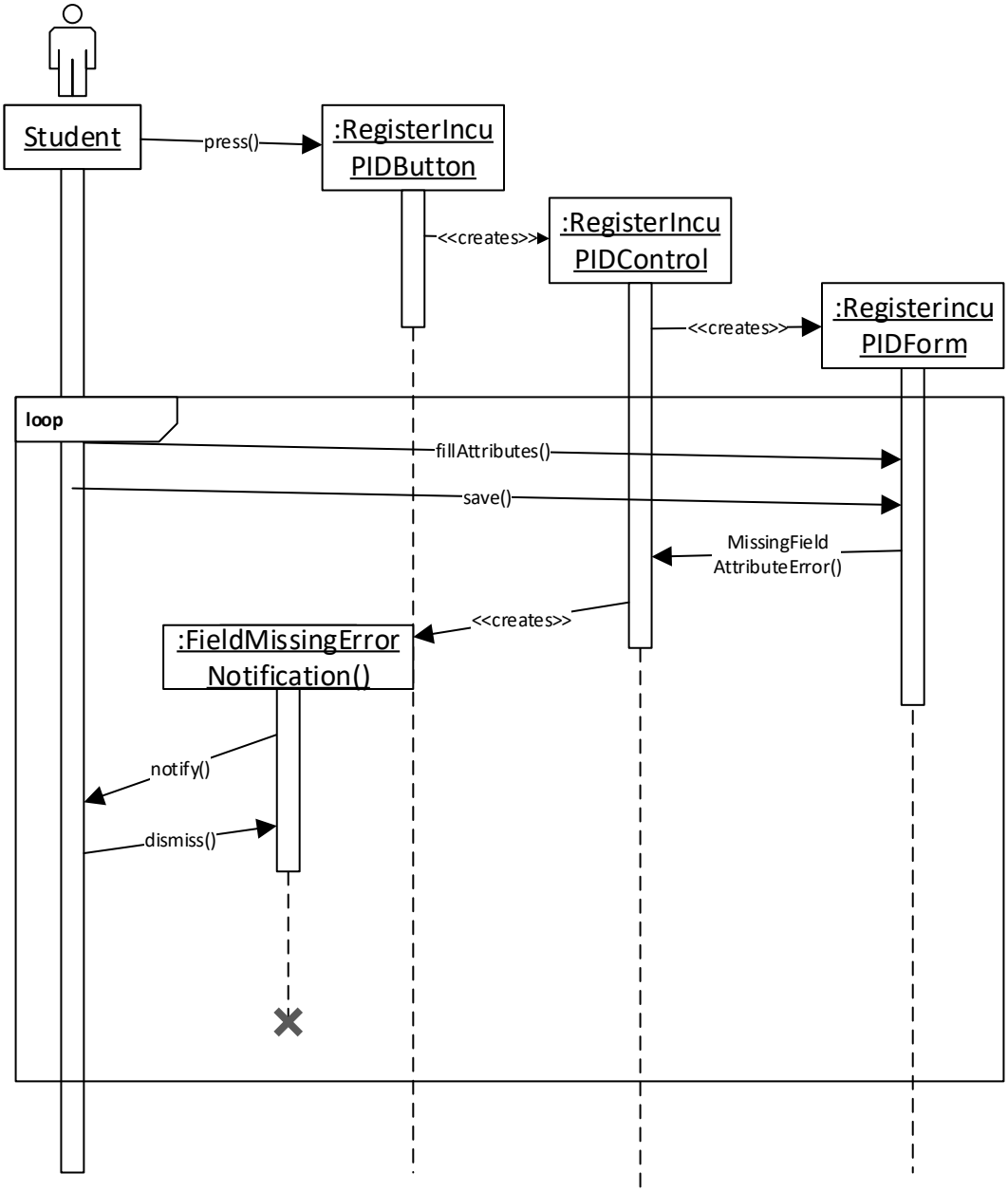


Figure 36– RegisterInCUPIDMissingFieldAttributeError Sequence Diagram

Sequence Diagram for EditProfileMissingFieldAttributeError (UC – 10, UC - 07):

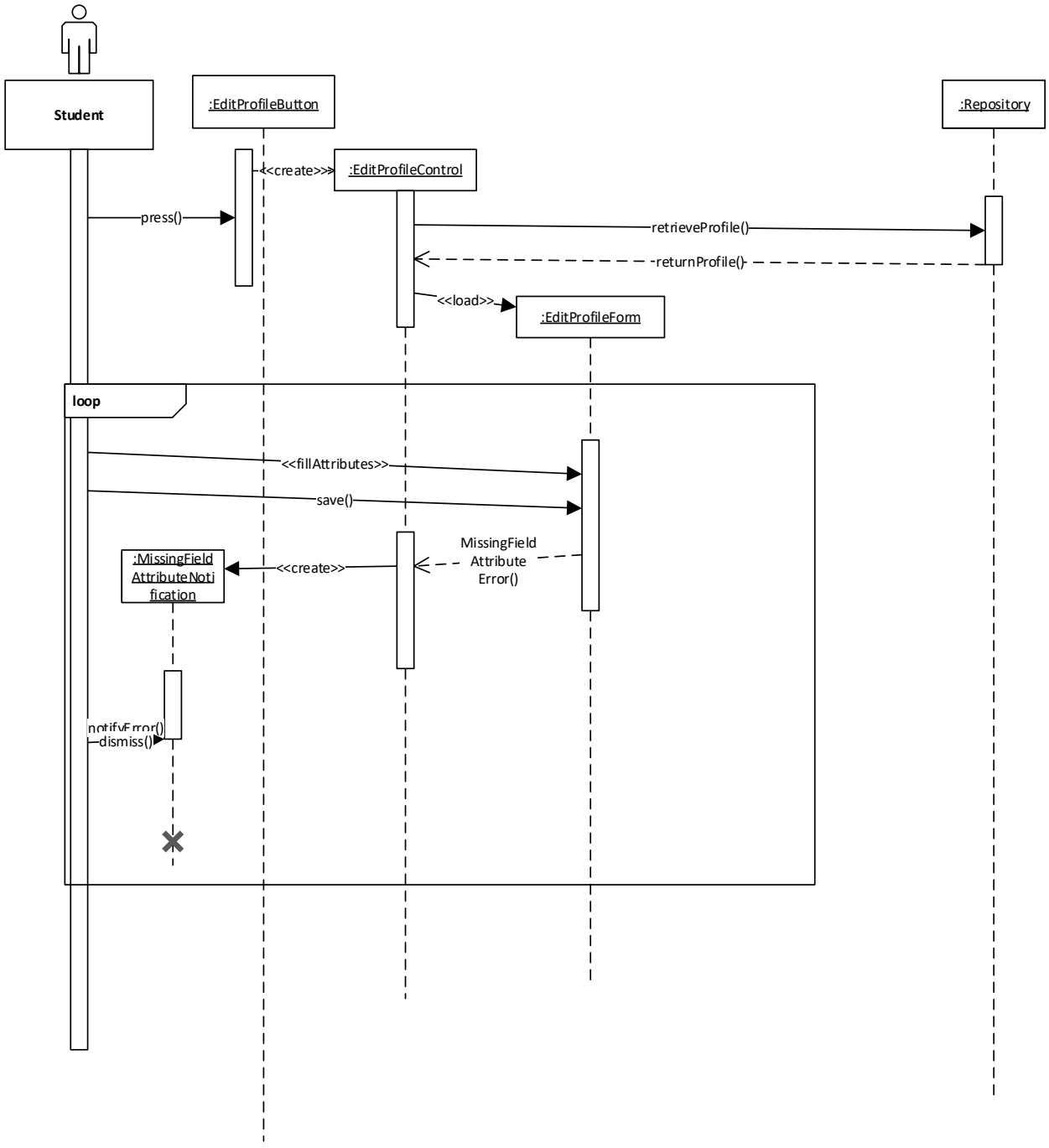


Figure 37– EditProfileMissingFieldAttributeError Sequence Diagram

Sequence Diagram for SaveRegisterIncuPID (UC – 12, UC - 05):

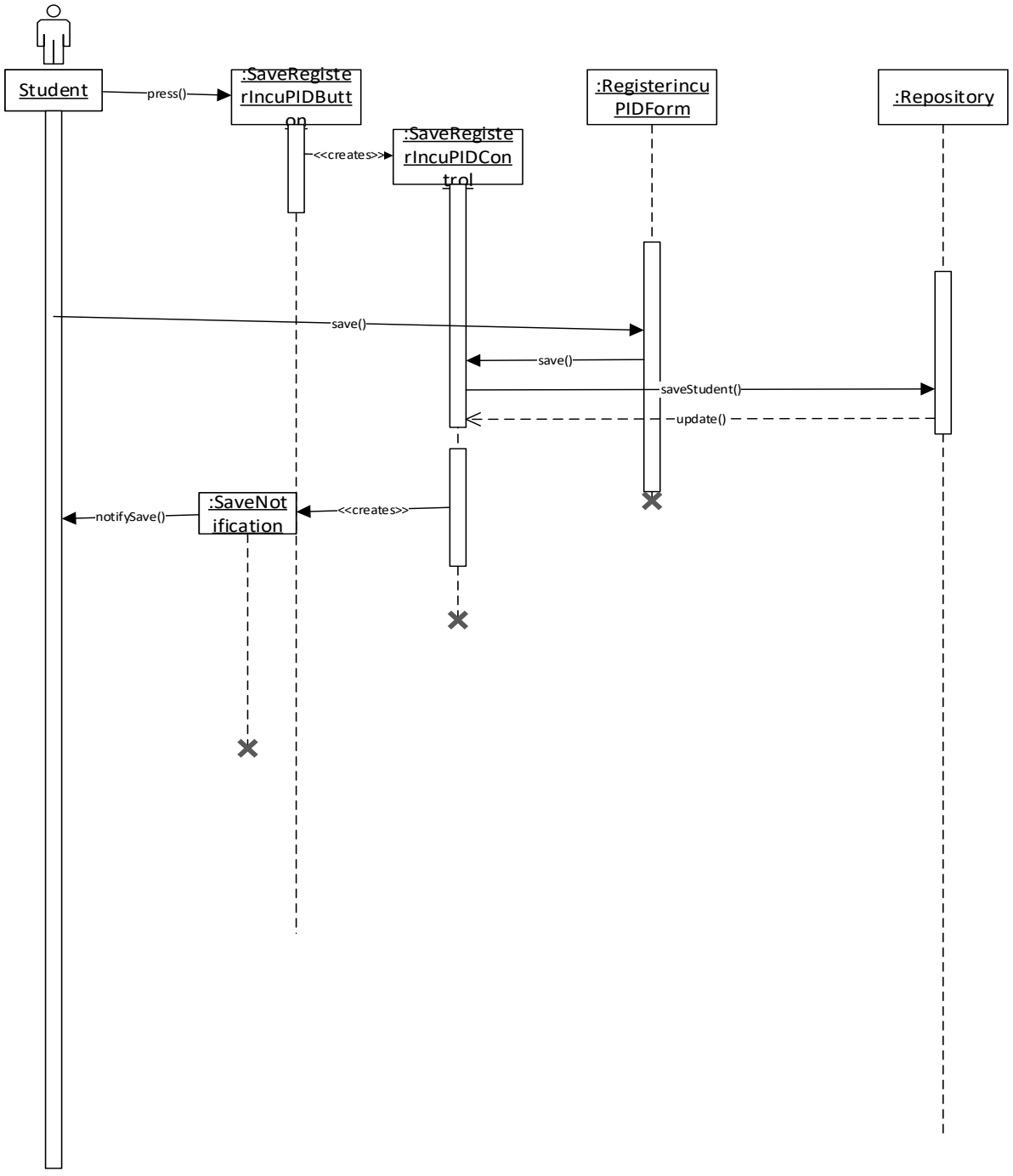


Figure 38–SaveRegisterIncuPID Sequence Diagram

Sequence Diagram for SaveCreateProject (UC – 16, UC - 01):

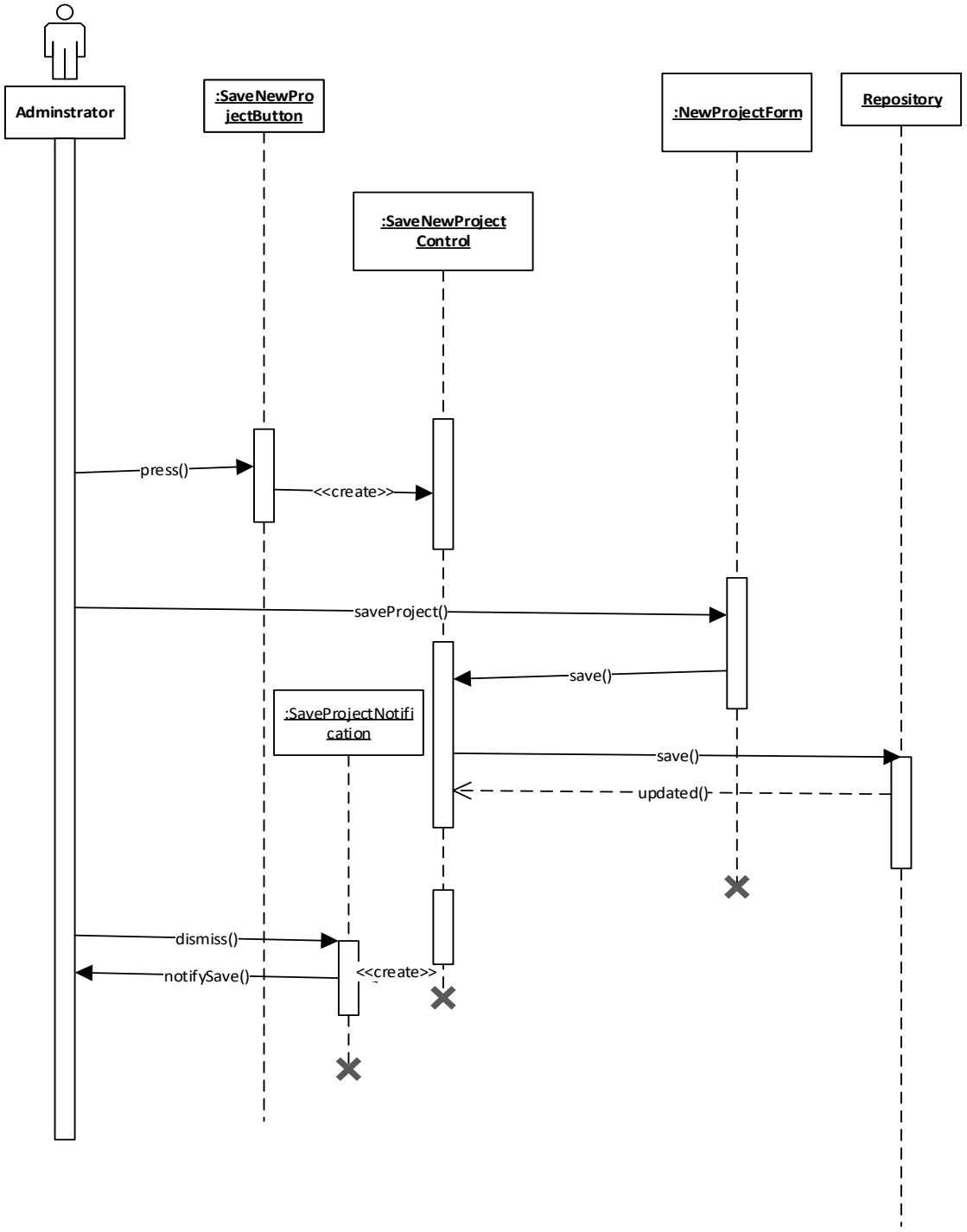


Figure 39–SaveCreateProject Sequence Diagram

Sequence Diagram for SaveEditProfile (UC – 12, UC -07):

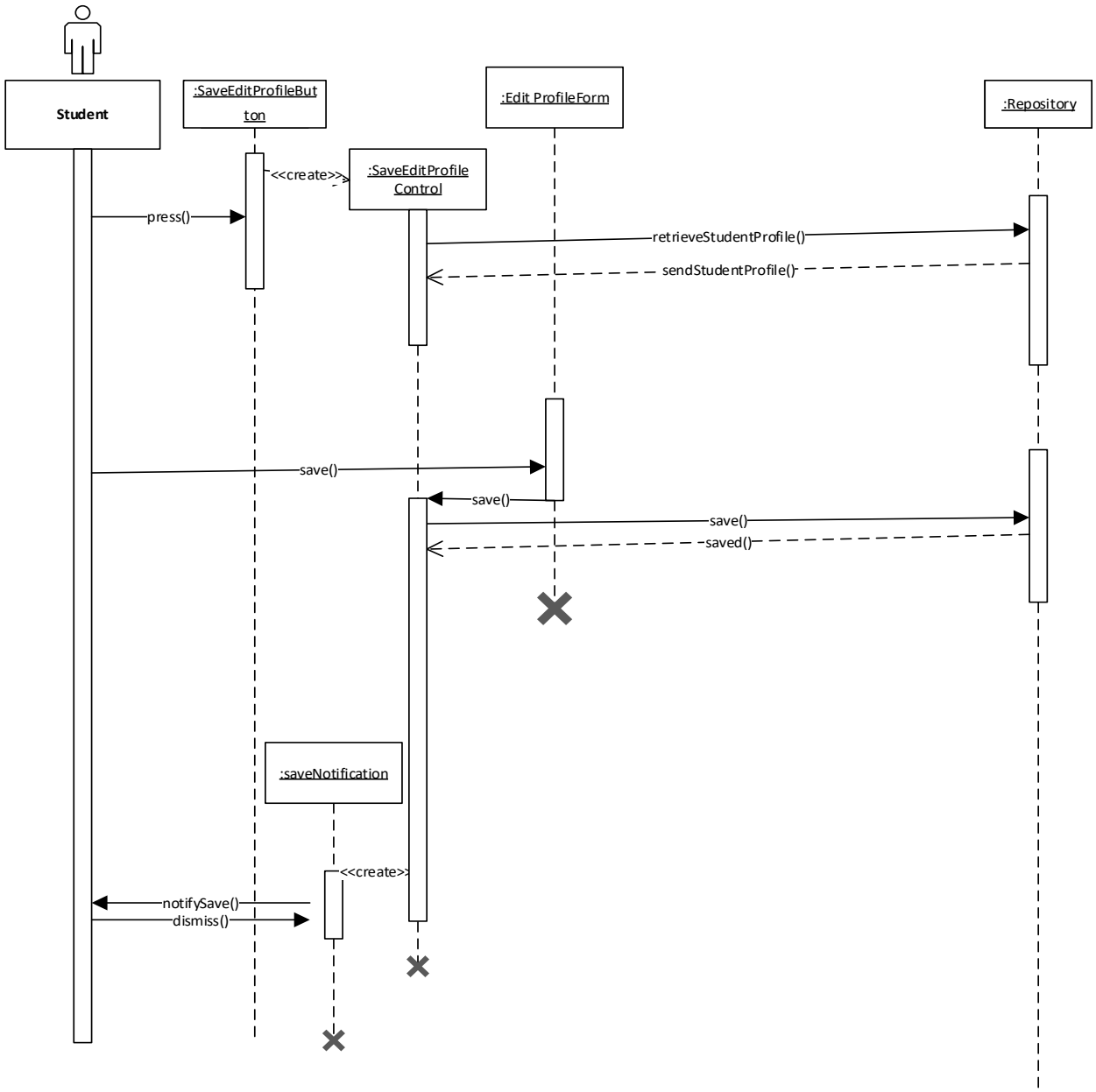


Figure 40–SaveEditProfile Sequence Diagram

Sequence Diagram for SaveEditProject (UC – 16, UC -02):

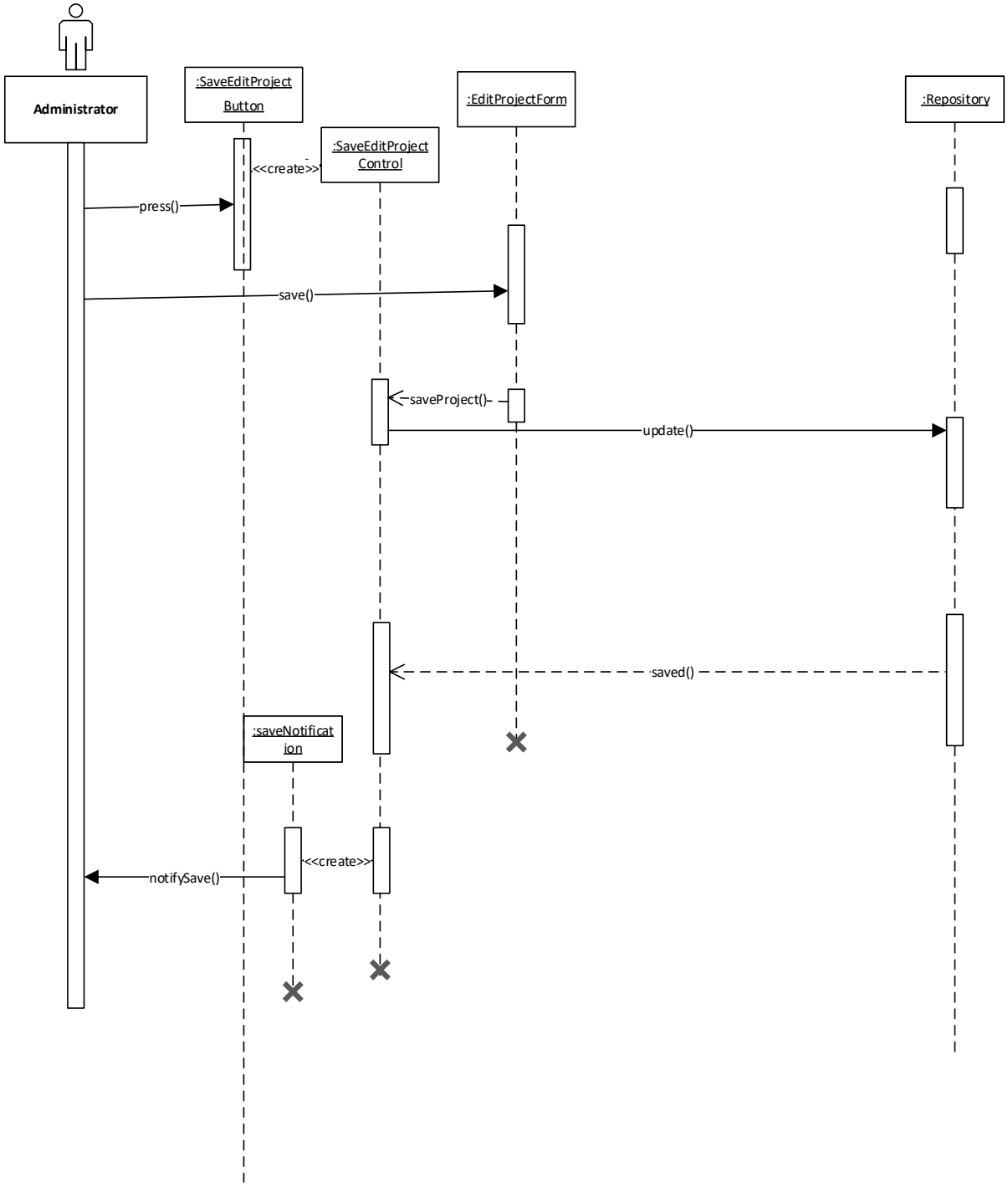


Figure 41–SaveEditProject Sequence Diagram

Sequence Diagram for ViewProjectReadError (UC – 16, UC -17):

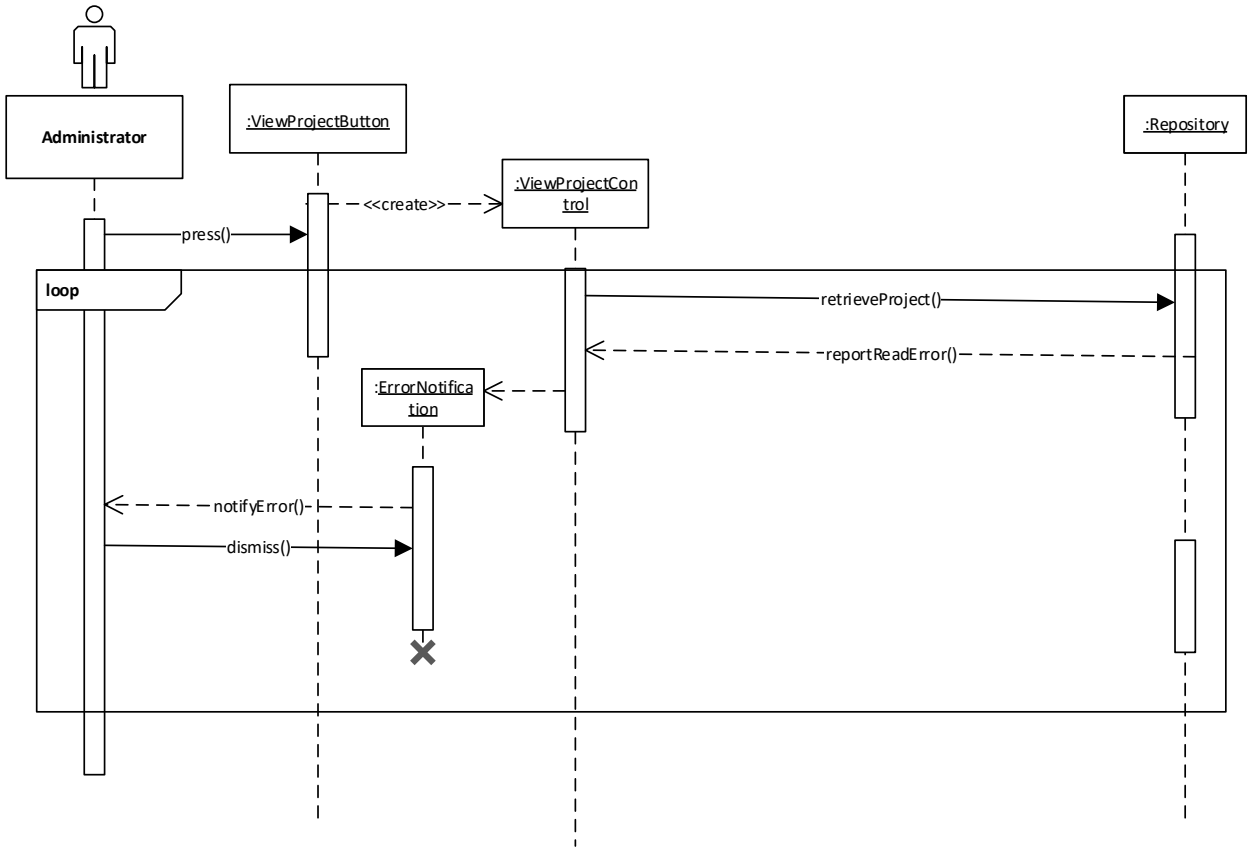


Figure 42–ViewProjectReadError Sequence Diagram

Sequence Diagram for ViewProfileReadError (UC – 16, UC -18):

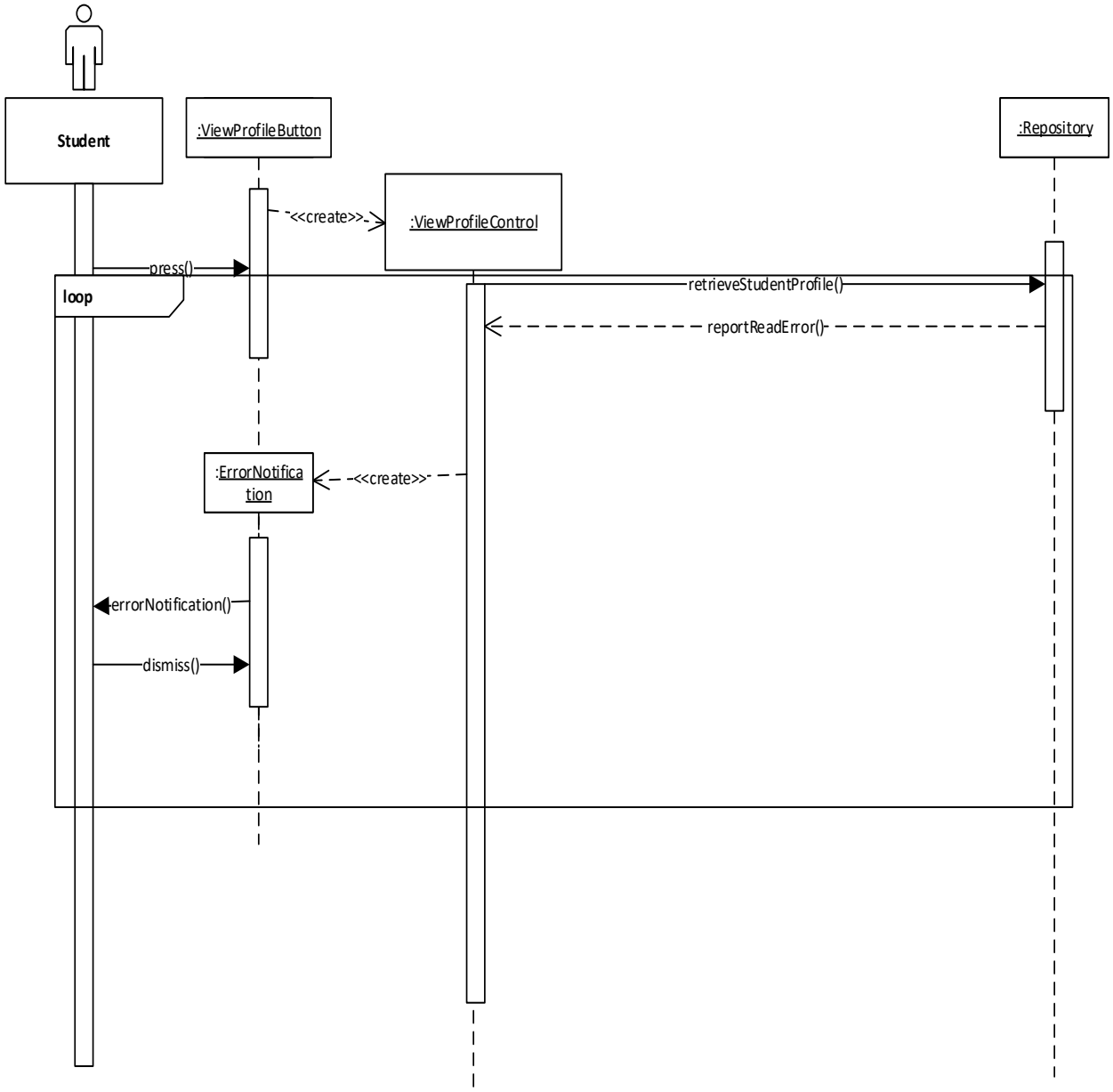


Figure 43–ViewProfileReadError Sequence Diagram