CALAMITY: A Disaster Preparedness Application
Analysis Model

Submitted to:

Asst. Prof. Ma. Rowena C. Solamo
Faculty Member
Department of Computer Science
College of Engineering
University of the Philippines, Diliman

Submitted by:

Dalisay, Nephia Bianca
Tan, Gene Audrey
Wee, Filbert Heinrich

In partial fulfillment of Academic Requirements
for the course
CS 191 Software Engineering I
of the
1st Semester, AY 2019-2020

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

Unique Reference:

The documents are stored in the https://github.com/geneaudrey/Disaster-Preparedness/tree/master/02-Requirements%20Engineering/Project%20Deliverables referenced with Group 6 - Disaster Preparedness App - Analysis Model.pdf.

Purpose:

The purpose of this document is to go into more detail about the attributes and operations of the classes to be used in the system (analysis model) and to describe how the group of objects work together per scenario, as well as the order in which they work (behavioral model). The boundary, entity, and control classes are also further described in the document.

Audience:

The following are part of the target audience:

- People who like or are interested in playing games
- People who have little to average knowledge on disaster preparedness, or people who would like to refresh their knowledge on disaster preparedness
- People who are interested in the development of the application

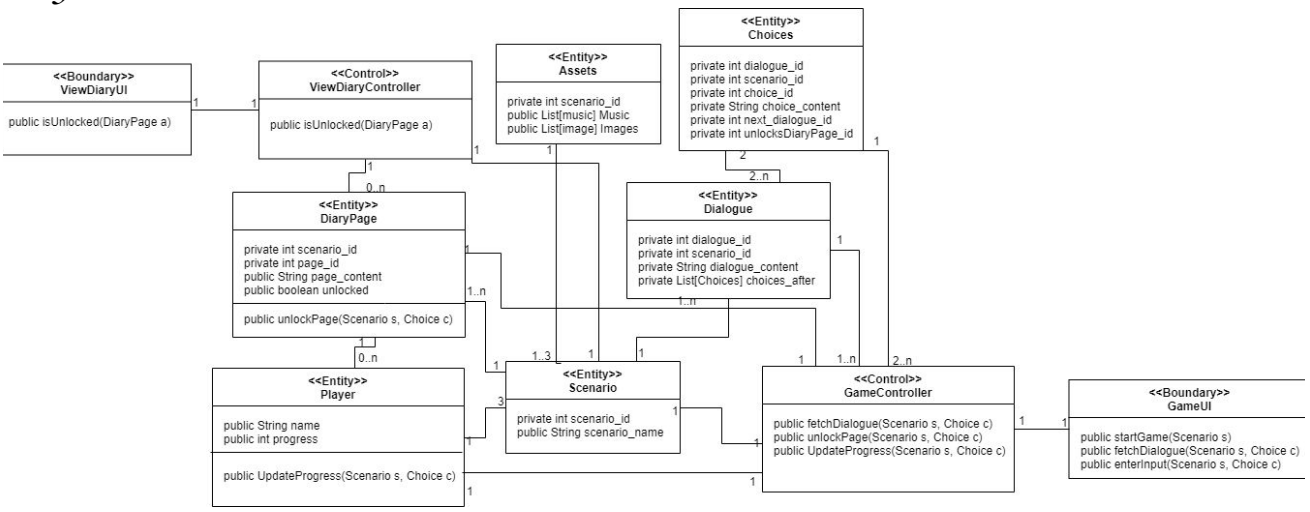
Revision Control:

Revision Date	*Person Responsible*	*Version Number*	*Modification*
10/03/19	Gene Tan	1.0	Initial Document Added Purpose, Audience, Unique Reference Added System Name, Description
10/04/19	Filbert Wee	2.0	Added Analysis Model Added Boundary Classes Class Name and Description Added Entity Classes Class Name and Description Added Control Classes Class Name and Description
10/04/19	Gene Tan	3.0	Added Use Case Name and Description Added Behavioral Models for scenarios 1, 2, 3, 4, 5, 13, 14
10/04/19	Nephia Dalisay	4.0	Added Behavioral Models for scenarios 6, 7, 8, 9, 10, 11, 12

***System Name*:** Disaster Preparedness Gameplay System

Description: The system is an application that aims to encourage disaster preparedness in a fun, unique, and memorable way. With this, a game will be the central feature of the system. More specifically, it will be a game that teaches players what to do in certain moments of disaster (i.e: fires, earthquakes, floods, and storm surges) by forcing them to make choices. Different combinations of decisions may lead to different endings. Additionally, a "diary" feature will also be included within the system. This is to act as "achievements" or bonuses for the players that are unlocked when they reach certain checkpoints in the game. The diary will be containing more information on the disasters featured in the game.

***Analysis Model*:**



Boundary Classes:

Class Name	**Description**
GameUI	This is the interface of the player to the game. *Responsibilities*: public startGame(Scenario s): return void public fetchDialogue(Scenario s, Choice c): return Dialogue public enterInput(Scenario s, Choice c): return void
ViewDiaryUI	This is the interface of the game that shows the player on the progress so far that has been unlocked in the diary. *Responsibilities*: public isUnlocked(DiaryPage a): return boolean

Control Classes:

Class Name	**Description**
GameController	This is the control that manages the player progress, game progress and flow. *Responsibilities*: public fetchDialogue(Scenario s, Choice c): return Dialogue public unlockPage(Scenario s, Choice c): return void public UpdateProgress(Scenario s, Choice c): return void
ViewDiaryController	This is the control that selects which pages of the diary has been unlocked and returns them. *Responsibilities*: public isUnlocked(DiaryPage a): return boolean

Entity Classes:

Class Name	Description
Player	This is the entity class player, which contains the data and the progress of the player. *Attributes*: public String name public int progress *Responsibilities* public UpdateProgress(Scenario s, Choice c): return void
Scenario	This is the entity class scenario, which contains the details of a scenario or path. *Attributes*: private int scenario_id public String scenario_name
Dialogue	This is the entity class dialogue, which contains the details of each dialogue and the choices that follow it. *Attributes*: private int dialogue_id private int scenario_id private String dialogue_content private List[Choices] choices_after
Choices	This is the entity class, choices, which contains the corresponding choices for a certain dialogue, together with the id of the dialogues that succeed it. *Attributes*: private int dialogue_id private int scenario_id private int choice_id private String choice_content private int next_dialogue_id private int unlocksDiaryPage_id
Assets	This is the entity class assets, which contains images, music, etc. to be used in a scenario. *Attributes*: private int scenario_id public List[music] Music public List[image] Images
DiaryPage	This is the entity class diary page, which is the progress and achievements of the player within a scenario. *Attributes*: private int scenario_id private int page_id public String page_content

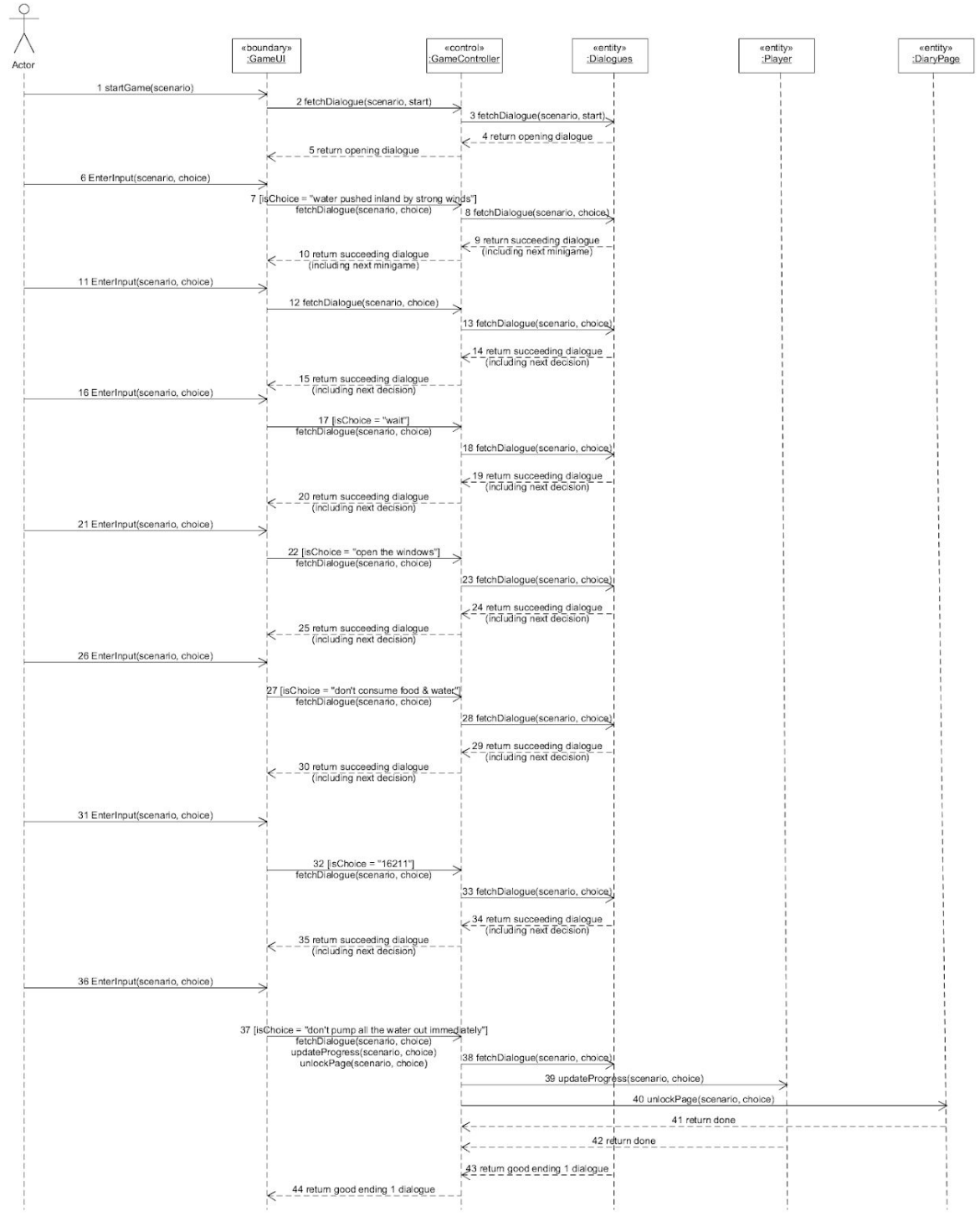
	public boolean unlocked *Responsibilities* public unlockPage(Scenario s, Choice c): return void

Behavioral Model:

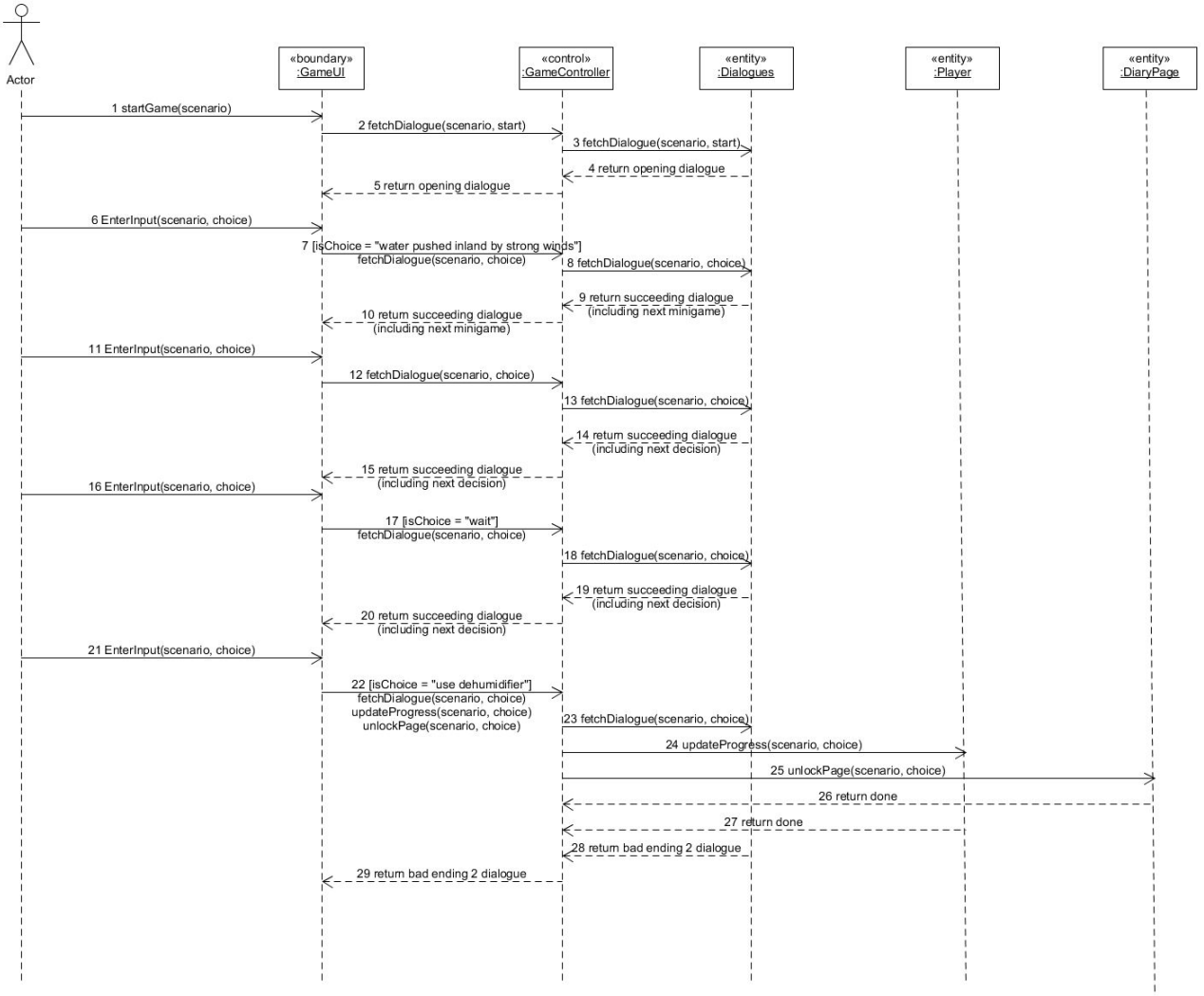
***Use-Case Name*:** 2.3 Play Storm Surge and Flood Scenario

Description: This use-case allows the player to play the flood and storm surge scenario. More specifically, it allows the player to make decisions to help his/ her friend survive the storm surge and flood. The story starts off with the player chatting with his friends through a messaging app. Somewhere in the middle of the conversation, a storm surge will occur in the area where one of his friends is staying. Depending on the choices made by the player, his/her friend may or may not survive. Certain endings in the flood and storm surge scenario will unlock corresponding diary pages.

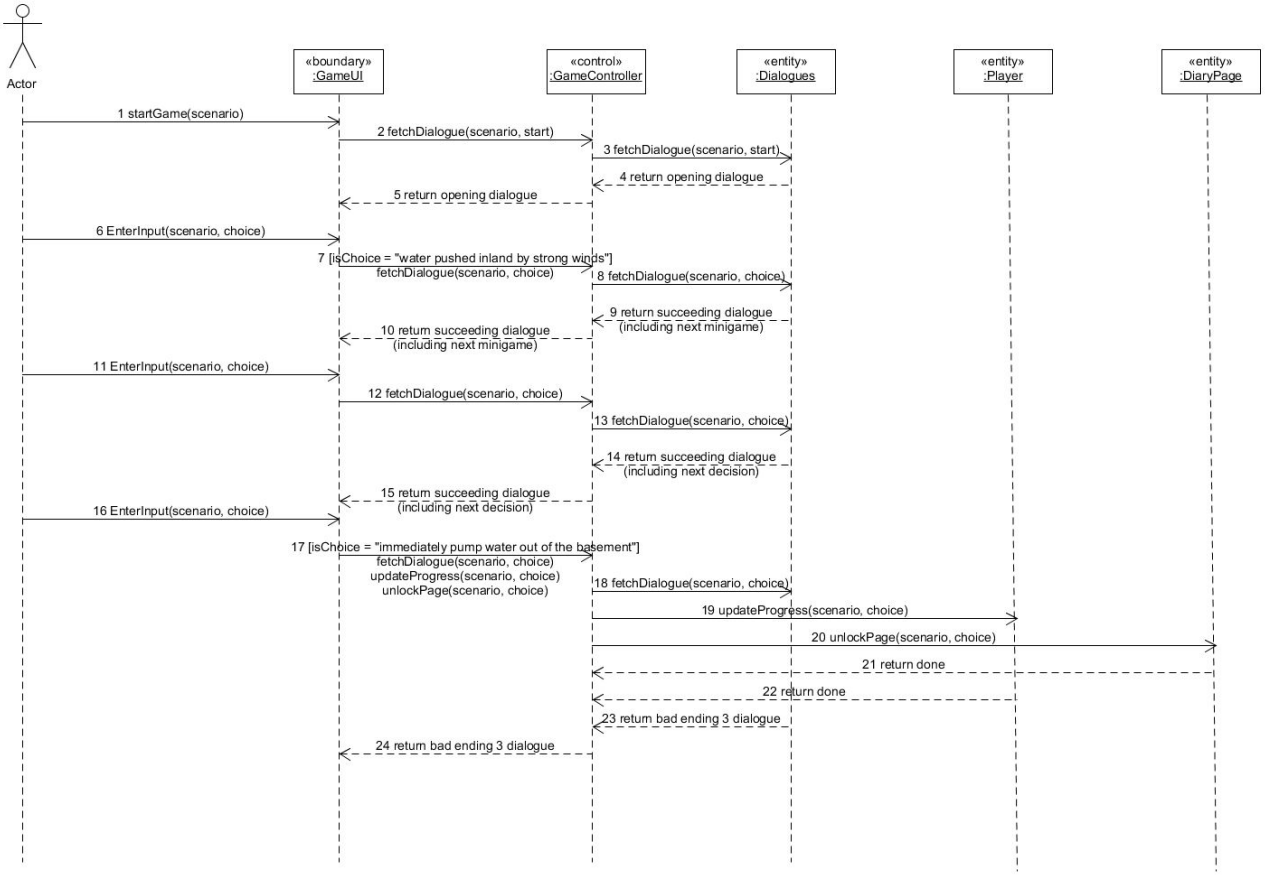
Scenario 1: Good Ending 1, character survives by evacuating.



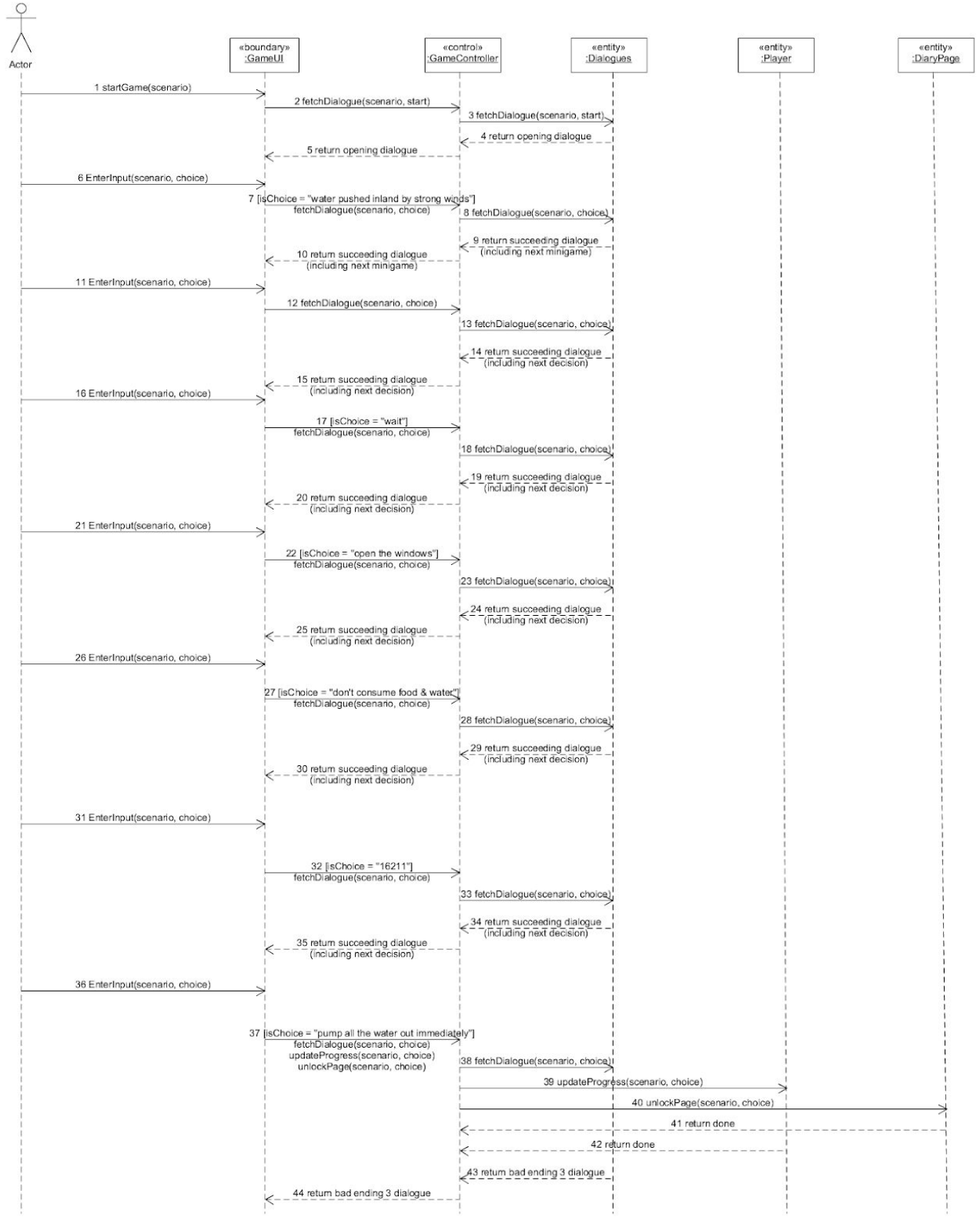
Scenario 2: Bad Ending 2, character gets electrocuted by dehumidifier.



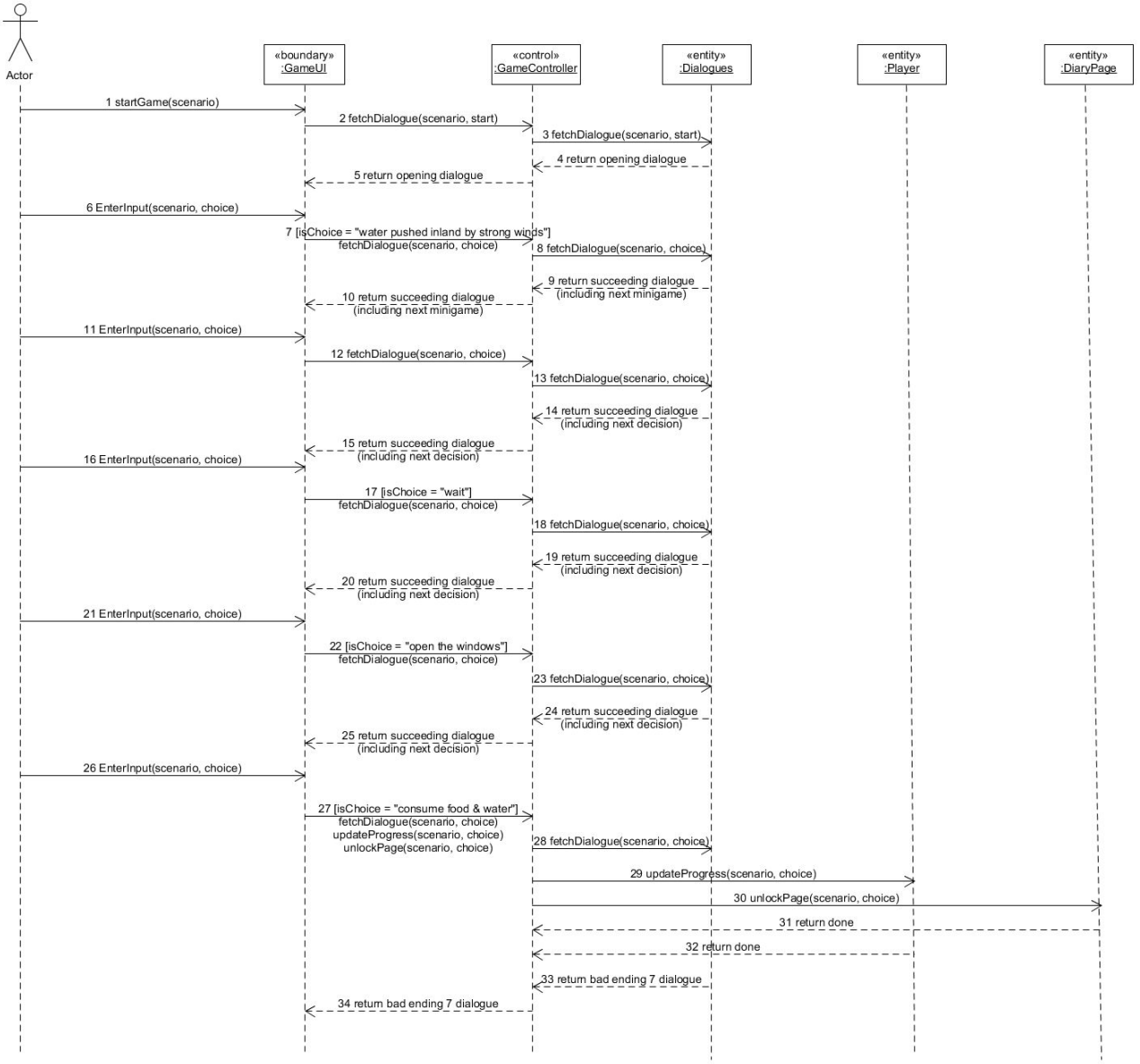
Scenario 3: Bad Ending 3, basement floor & walls crack & collapse.



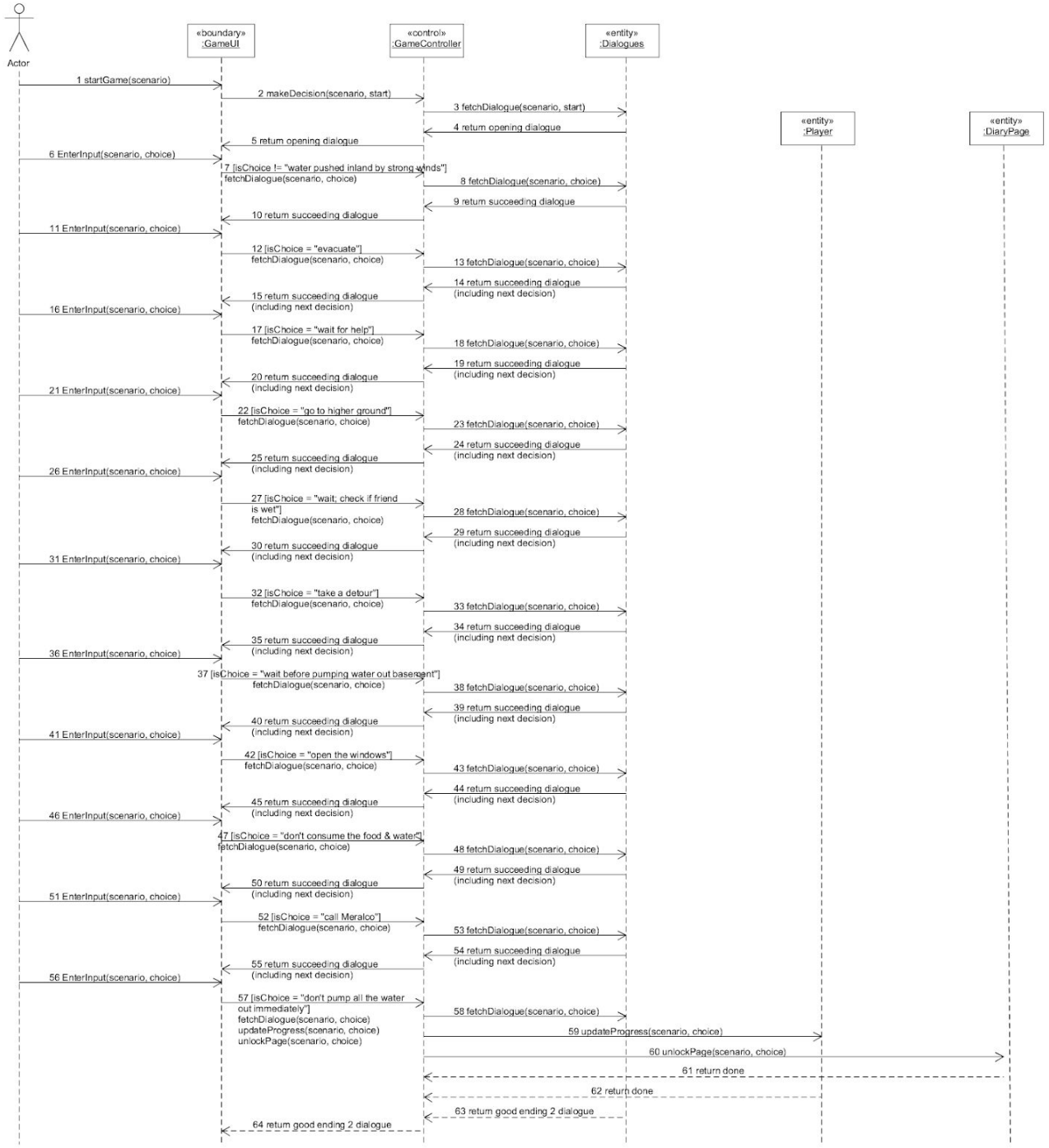
Scenario 4: Alternative way of getting Bad Ending 3, basement floor & walls crack & collapse.



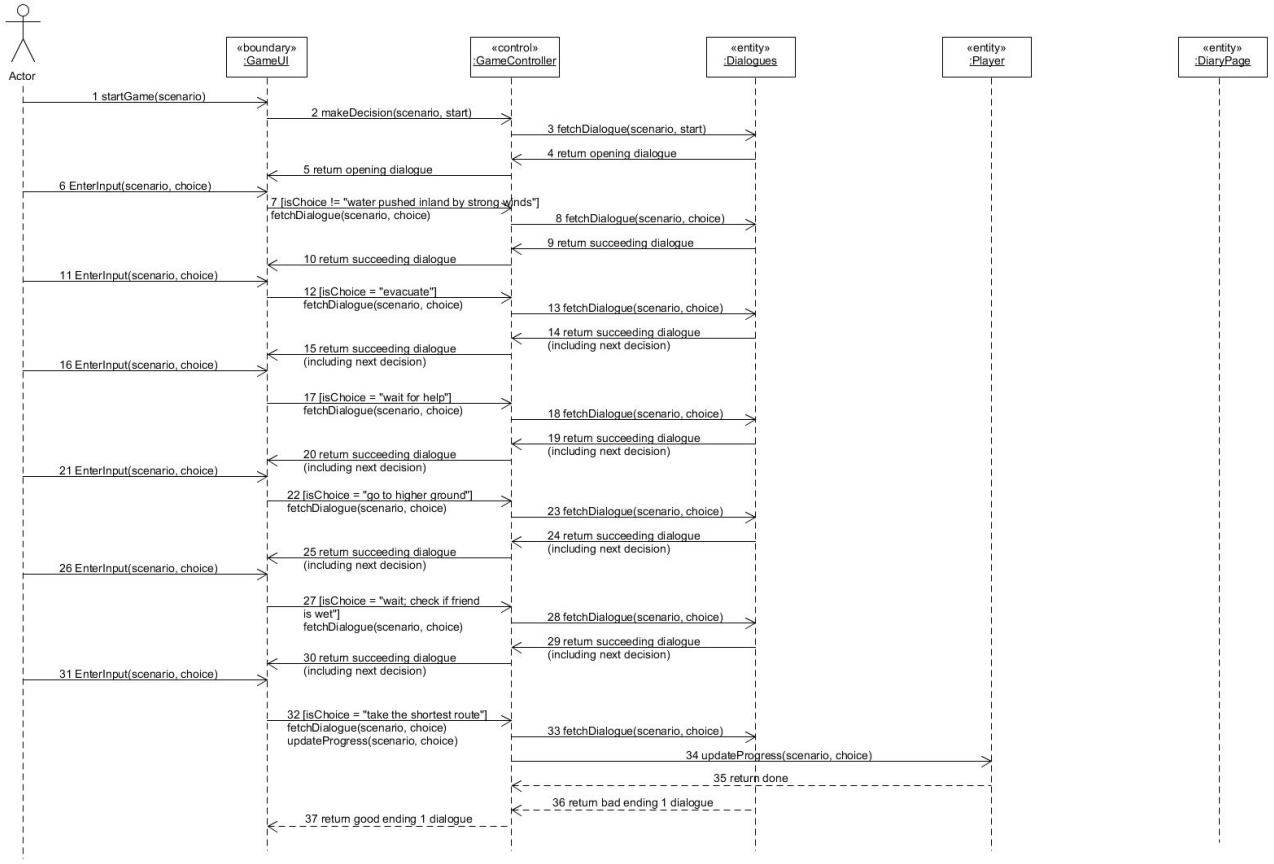
Scenario 5: Bad ending 7, character gets poisoned.



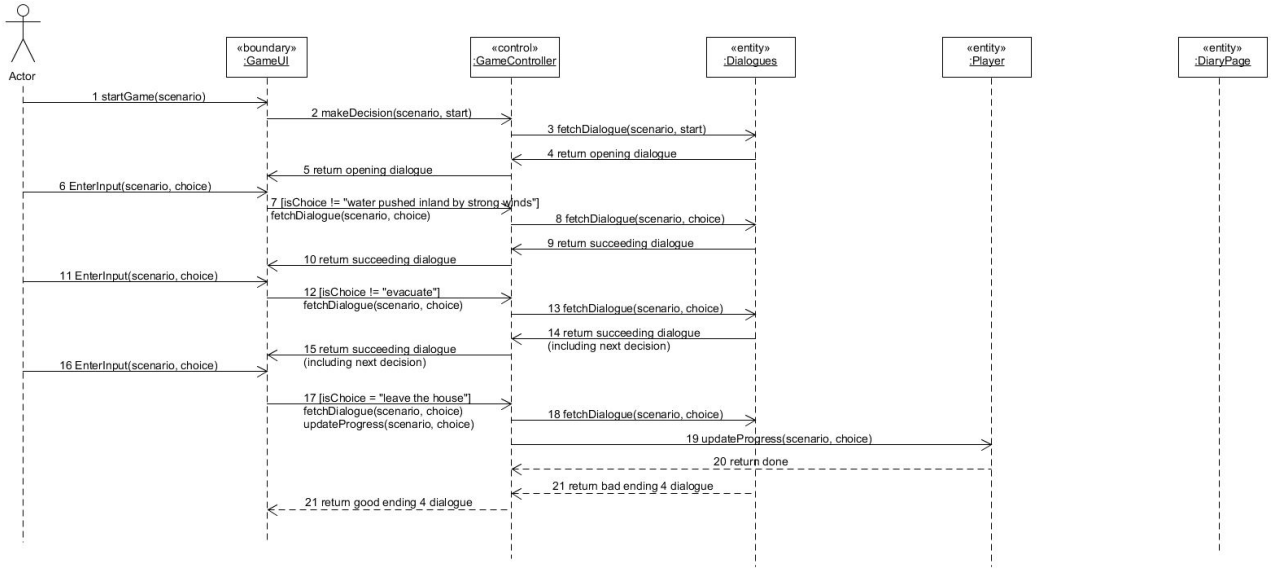
Scenario 6 Good Ending 2, character survives despite not evacuating.



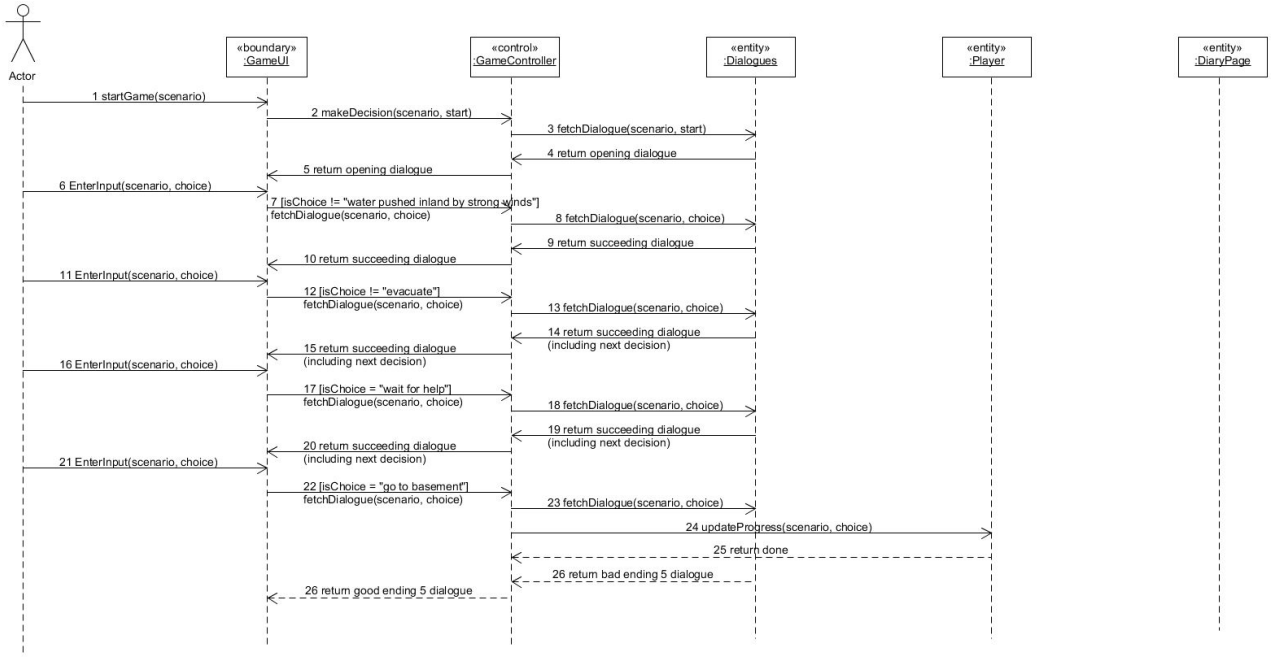
Scenario 7 Bad Ending 1, character gets stuck/falls down a ditch.



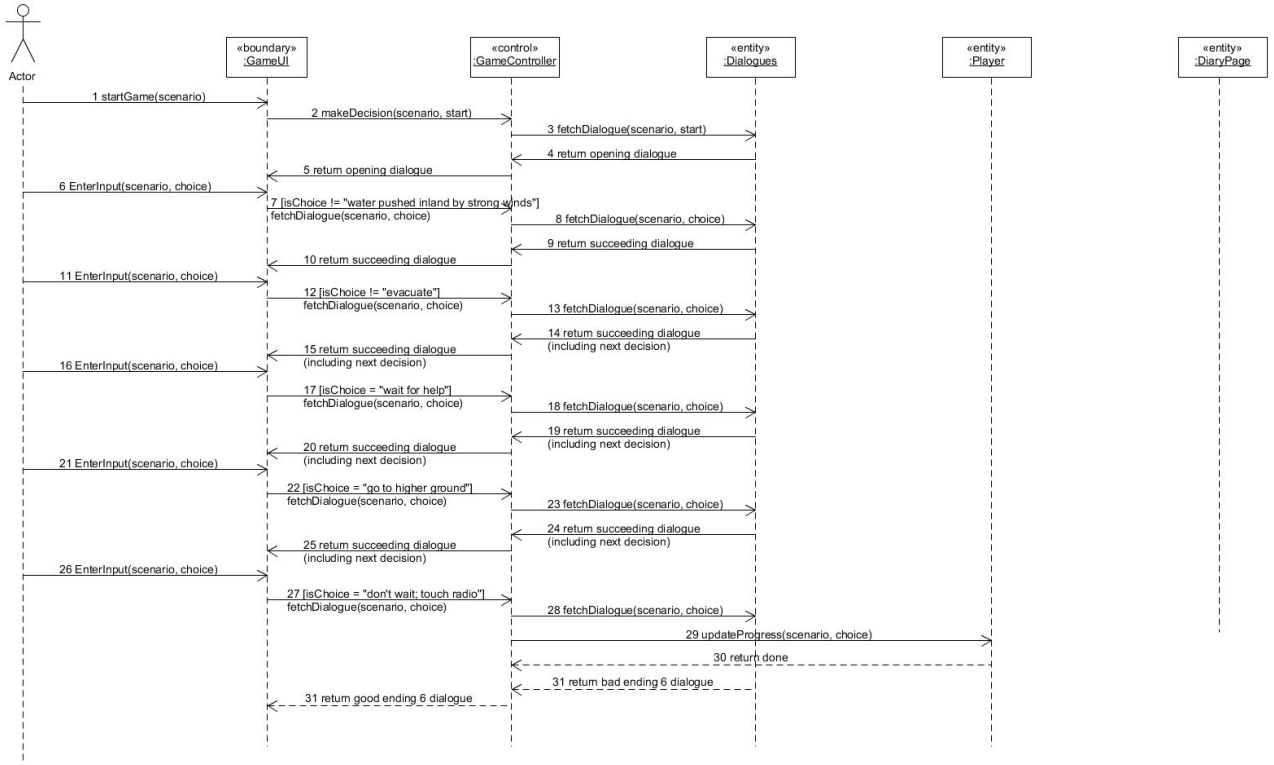
Scenario 8 Bad Ending 4, character gets washed away by the flood.



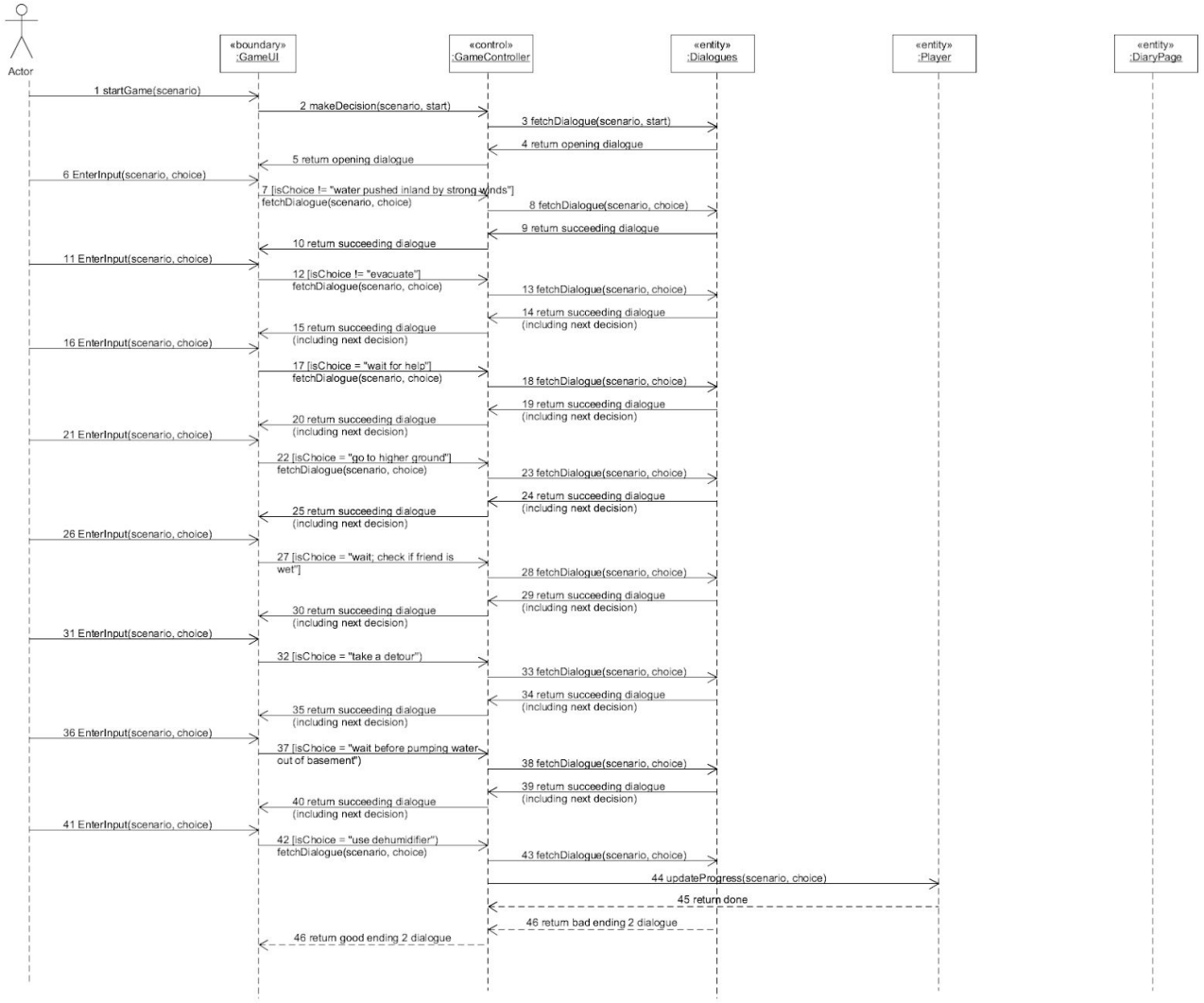
Scenario 9 Bad Ending 5, character gets stuck in the basement and drowns.



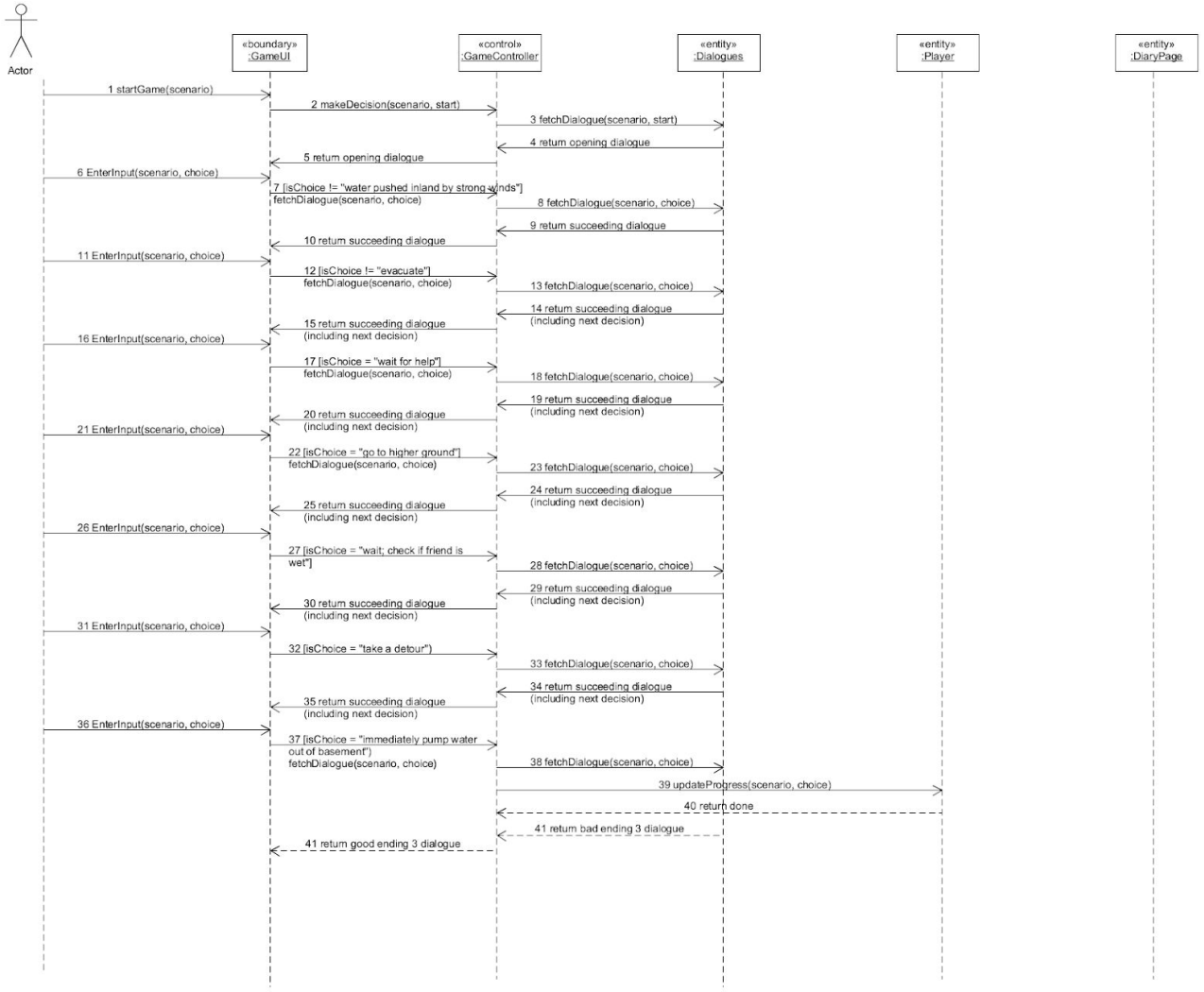
Scenario 10 Bad Ending 6, character gets electrocuted by using the radio.



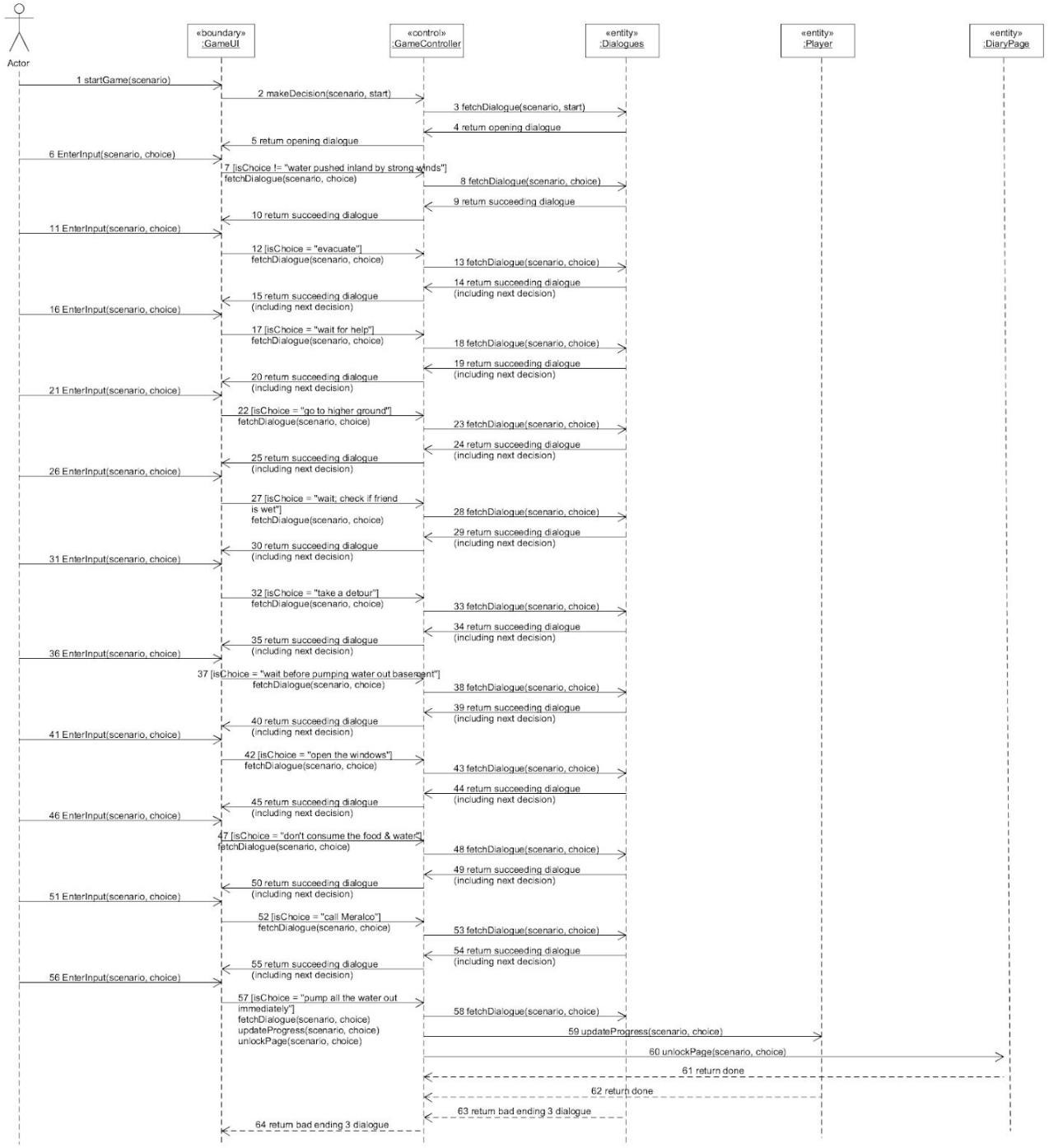
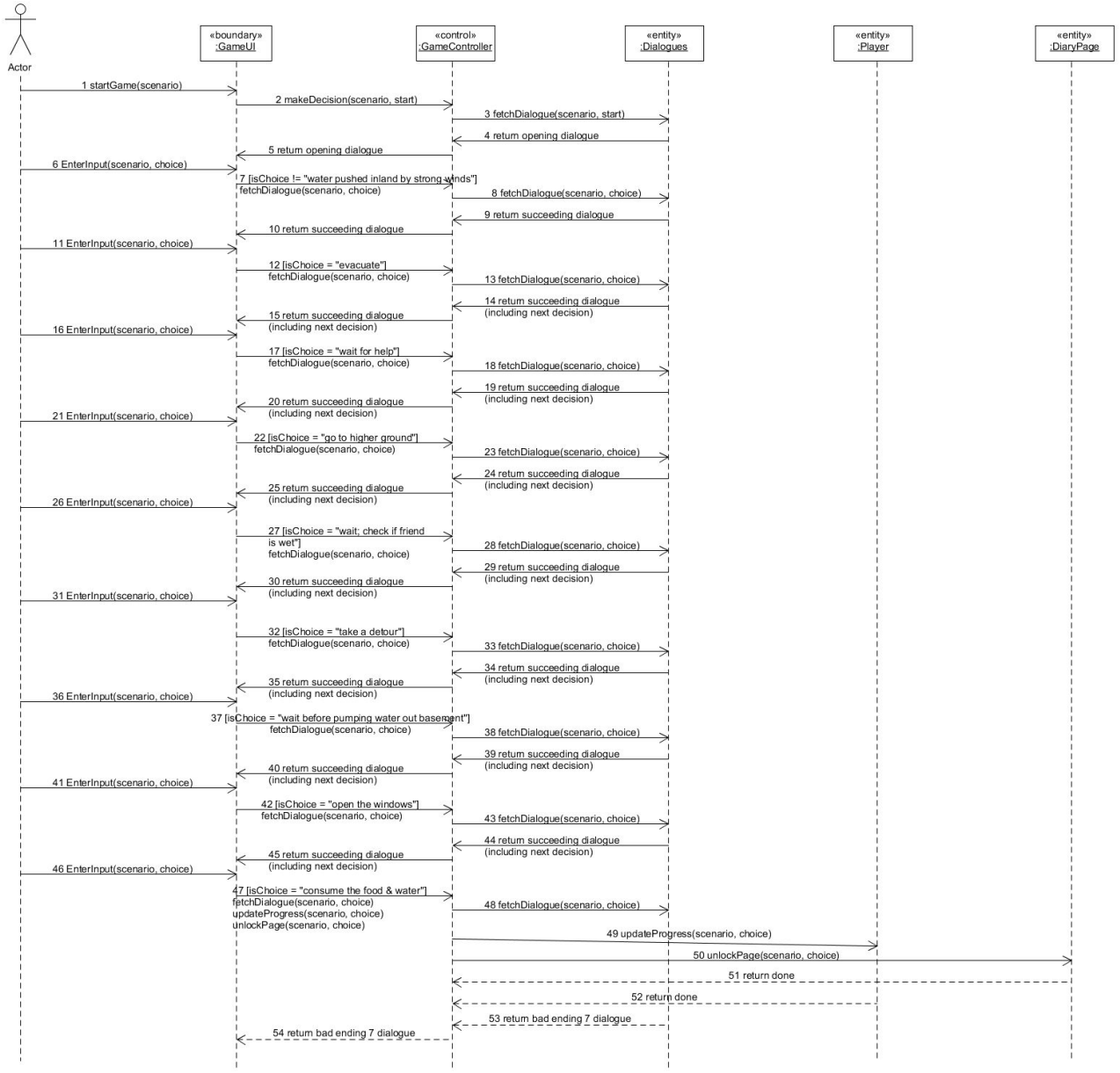
Scenario 11 Alternative way of getting Bad Ending 2, character gets electrocuted by dehumidifier.



Scenario 12 Alternative way of getting Bad Ending 3, basement floor & walls crack & collapse.



Scenario 13 Alternative way of getting Bad Ending 3, basement floor & walls crack & collapse.



Scenario 14 Alternative way of getting Bad ending 7, character gets poisoned.