Autonomous Pool Playing Robot

Low-Level Software Design

Eric Le Fort
leforte@mcmaster.ca
1308609

Max Moore
moorem8@mcmaster.ca
1320009

January 11, 2017

Contents

List of Tables

List of Figures

Date	Revision #	Comments	Authors
25/12/2016	0	- Initial document creation	Eric Le Fort

Table 1: Revision History

1 Introduction

This document will outline the low-level software design for a autonomous pool-playing robot. The purpose of this document will be to document the decisions made concerning the system's design as well as provide enough detail so that the programming of the system can be as trivial as possible.

1.1 System Description

A system description can be found in the /textitHigh-Level Software Design document for this system.

1.2 Overview

This document will begin by providing a detailed class diagram of the classes in the system. Then, each module will be covered in more detail such as the module's responsibilities, secrets, Interface Specification (MIS), and Internal Design (MID). Lastly, the document will discuss the scheduling of tasks and provide state charts and sequence diagrams to help illustrate the scheduling.

1.3 Naming Conventions & Definitions

This section outlines the various definitions, acronyms and abbreviations that will be used throughout this document in order to familiarize the reader prior to reading.

1.3.1 Definitions

Table 2 lists the definitions used in this document. The definitions given below are specific to this document and may not be identical to definitions of these terms in common use. The purpose of this section is to assist the user in understanding the requirements for the system.

Table 2: Definitions

Term	Meaning
X-axis	Distance along the length of the pool table
Y-axis	Distance across the width of the pool table
Z-axis	Height above the pool table
End-effector	The end of the arm that will strike the cue ball
θ	Rotational angle of end-effector
Cue	End-effector
Personal Computer	A laptop that will be used to run the more involved computational tasks such as visual recognition and the shot selection algorithm
Camera	Some form of image capture device (e.g. a digital camera, smartphone with a camera, etc.)
Table State	The current positions of all the balls on the table
Entity	Classes that have a state, behaviour and identity (e.g. Book, Car, Person, etc.)
Boundary	Classes that interact with users or external systems

1.3.2 Acronyms & Abbreviations

Table 3 lists the acronyms and abbreviations used in this document.

Table 3: Acronyms and Abbreviations

Acronym/Abbreviation	Meaning
VR	Visual Recognition
PC	Personal Computer
μC	Micro-Controller
CRC	Class Responsibility Collaboration

2 Detailed Class Diagram

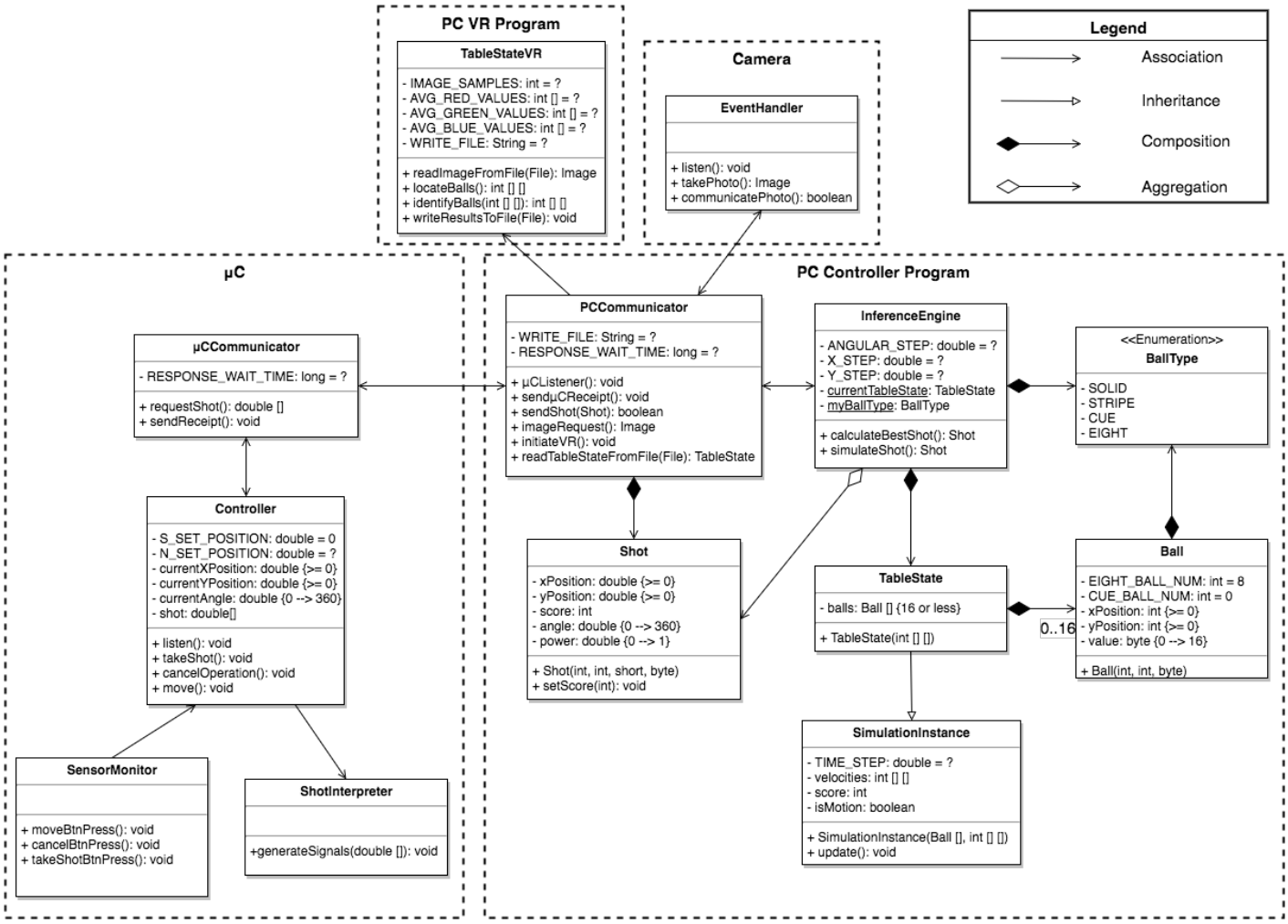

Figure 1: The system's detailed class diagram.

3 Module Guide

This section discusses the various modules that this system is comprised of. The modules are divided based on which program they belong to. For each module, its responsibilities, secrets, MIS, and MID will be outlined.

3.1 Camera Modules

The following is the module contained within the Camera subsystem.

3.1.1 EventHandler

Responsibilities

Secrets

MIS

MID

3.2 PC VR Program Modules

The following is the module contained within the PC VR subsystem.

3.2.1 TableStateVR

Responsibilities

Secrets

MIS

MID

3.3 PC Controller Modules

The following are the modules contained within the PC Controller subsystem.

3.3.1 InferenceEngine

Responsibilities

Secrets

MIS

MID

3.3.2 PCCommunicator

Responsibilities

Secrets

MIS

MID

3.3.3 SimulationInstance

Responsibilities

Secrets

MIS

MID

3.4 μC Modules

The following are the modules contained within the μC subsystem.

3.4.1 Controller

Responsibilities

Secrets

MIS

MID

3.4.2 SensorMonitor

Responsibilities

Secrets

MIS

MID

3.4.3 ShotInterpreter

Responsibilities

Secrets

MIS

MID

3.4.4 μCCommunicator

Responsibilities

Secrets

MIS

MID

4 Scheduling of Tasks

The goal of this section is to outline the ordering, maximum allowable time frames, and the prioritization of tasks in this program.

4.1 State Charts

The following charts illustrate the lifecycle of all relevant classes in this system. This section is meant to depict a more isolated picture of how each class will operate.

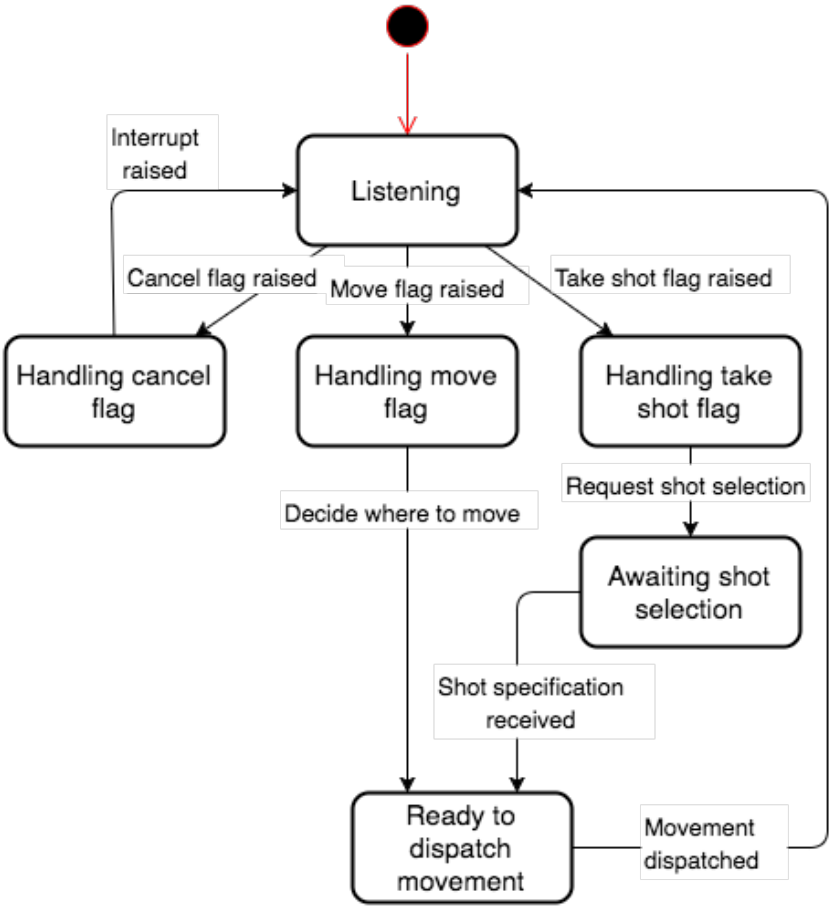

Figure 2: A state chart for the Controller class.

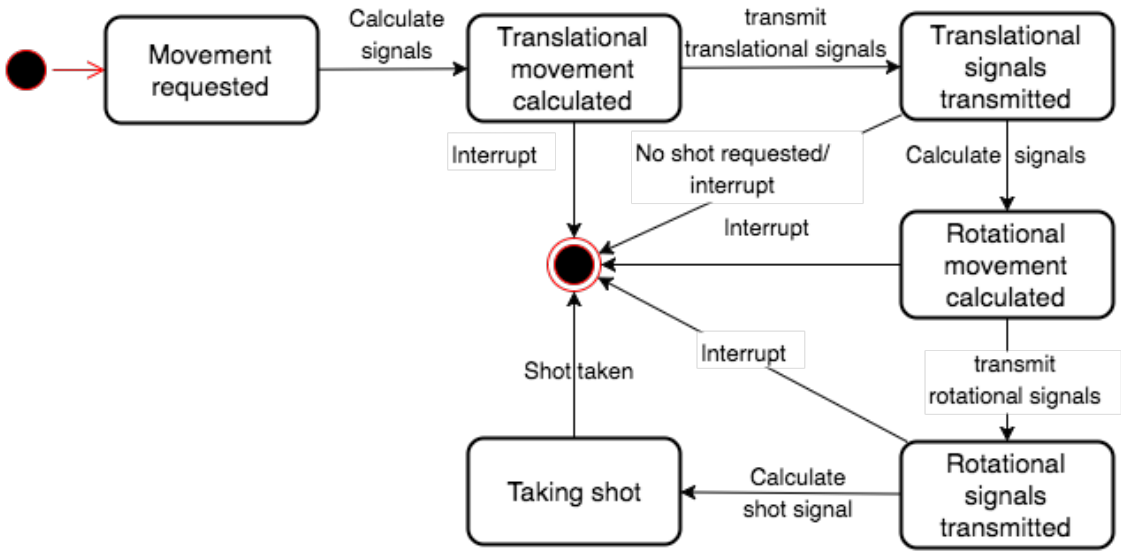

Figure 3: A state chart for the ShotInterpreter class.

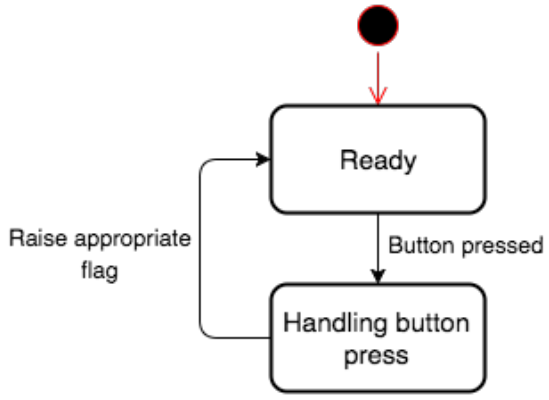

Figure 4: A state chart for the SensorMonitor class.

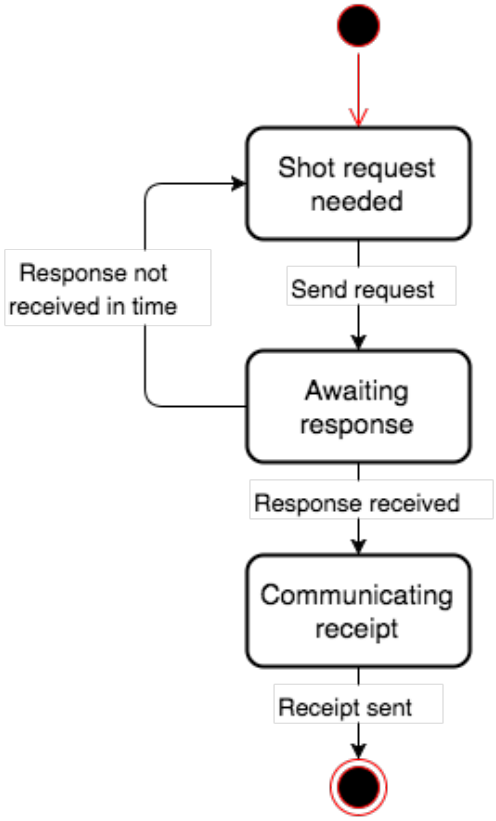

Figure 5: A state chart for the μCCommunicator class.

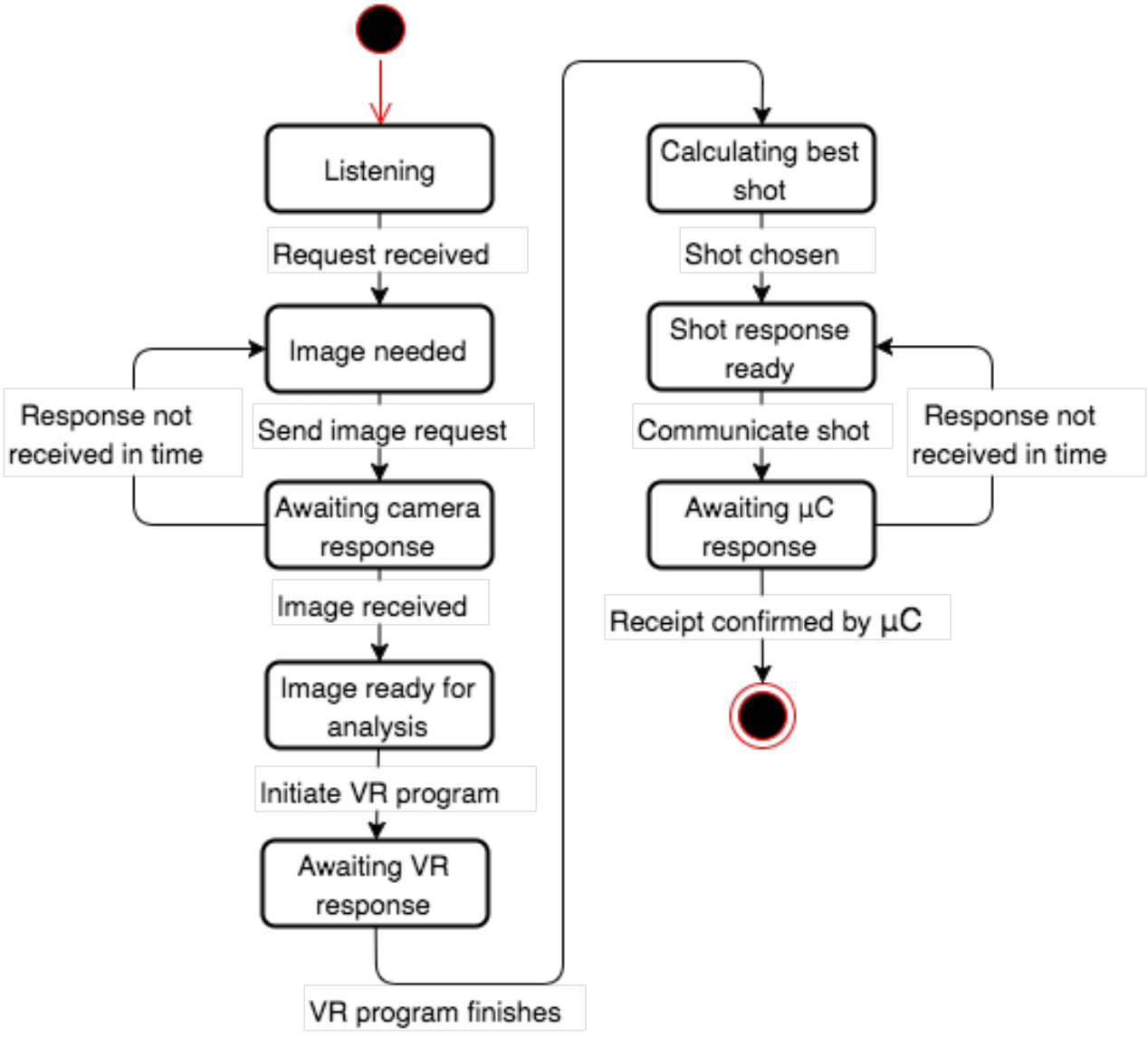

Figure 6: A state chart for the PCCommunicator class.

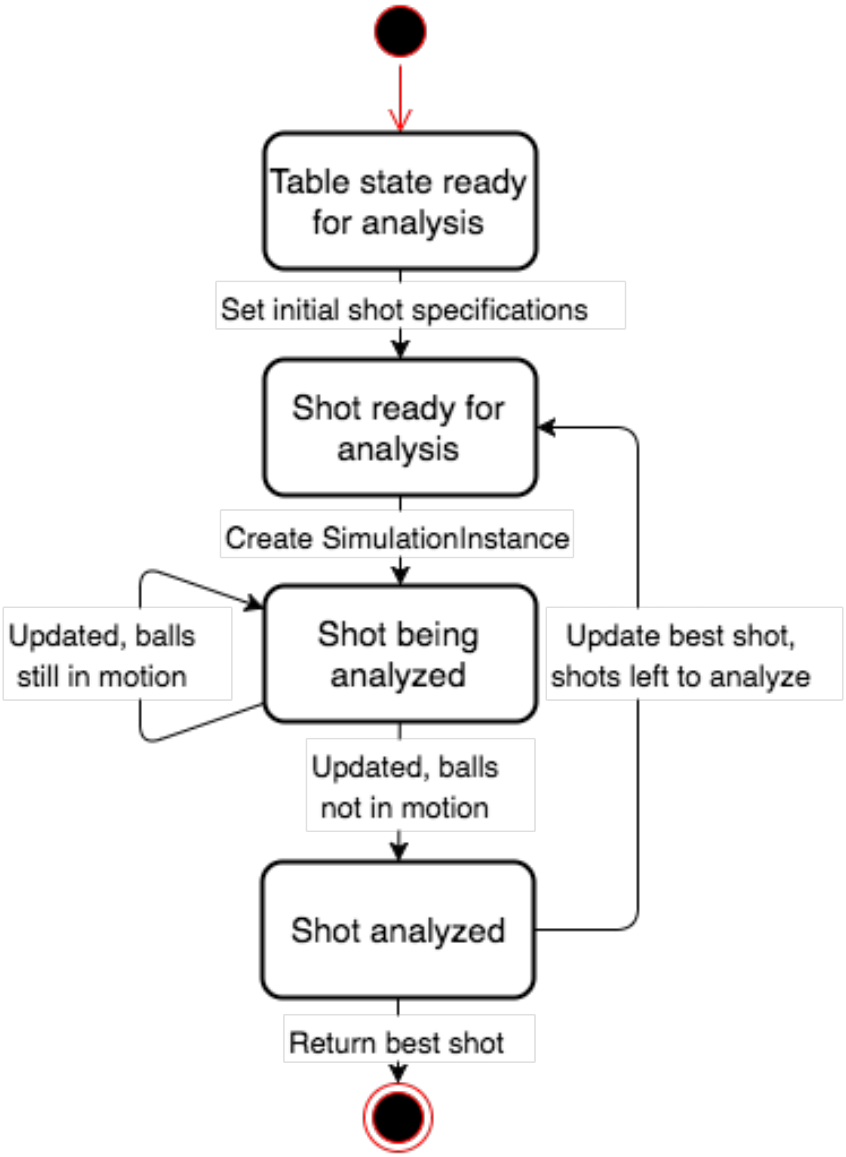

Figure 7: A state chart for the InferenceEngine class.

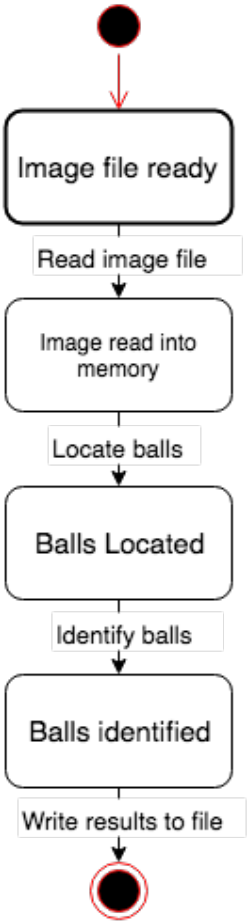

Figure 8: A state chart for the TableStateVR class.

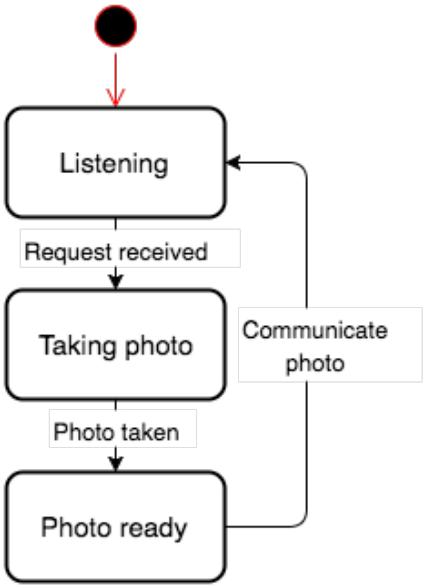

Figure 9: A state chart for the EventHandler class.

4.2 Sequence Diagrams

The following are various sequence diagrams for different actions the system is required to perform. These diagrams are meant to provide better context for how the classes interact with each other to perform certain tasks.

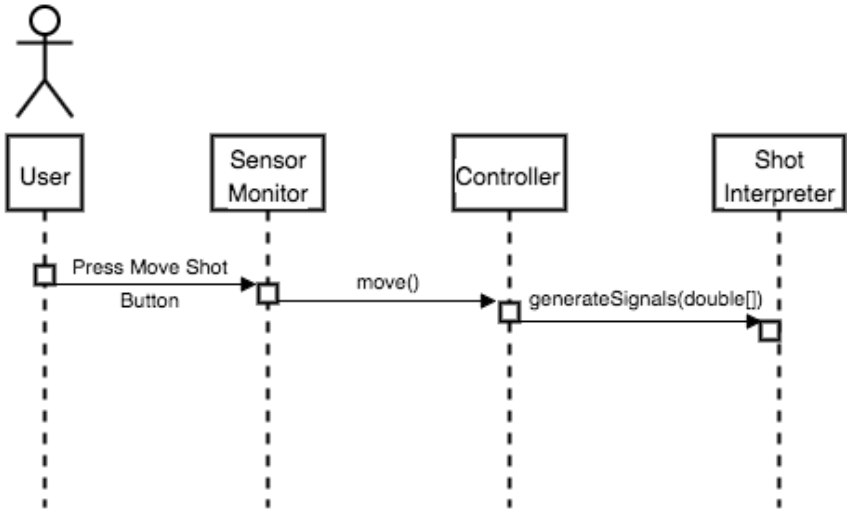

Figure 10: A sequence diagram for the "move" operation.

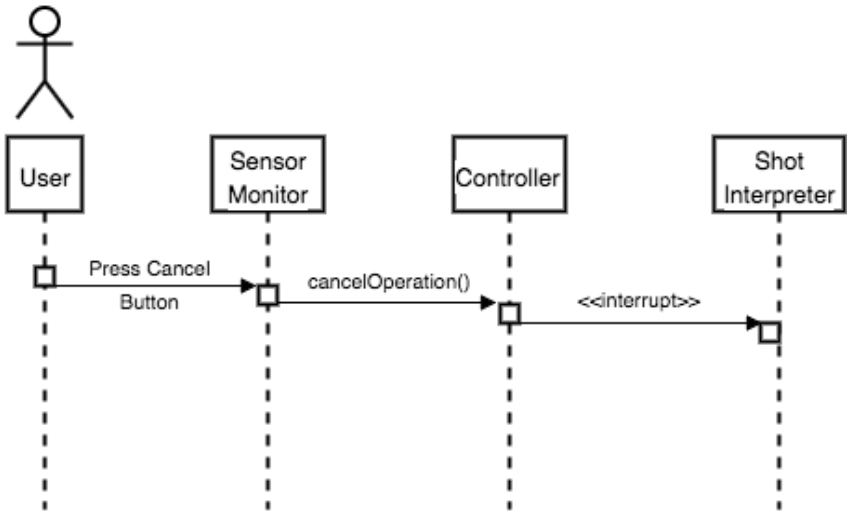

Figure 11: A sequence diagram for the "cancel" operation.

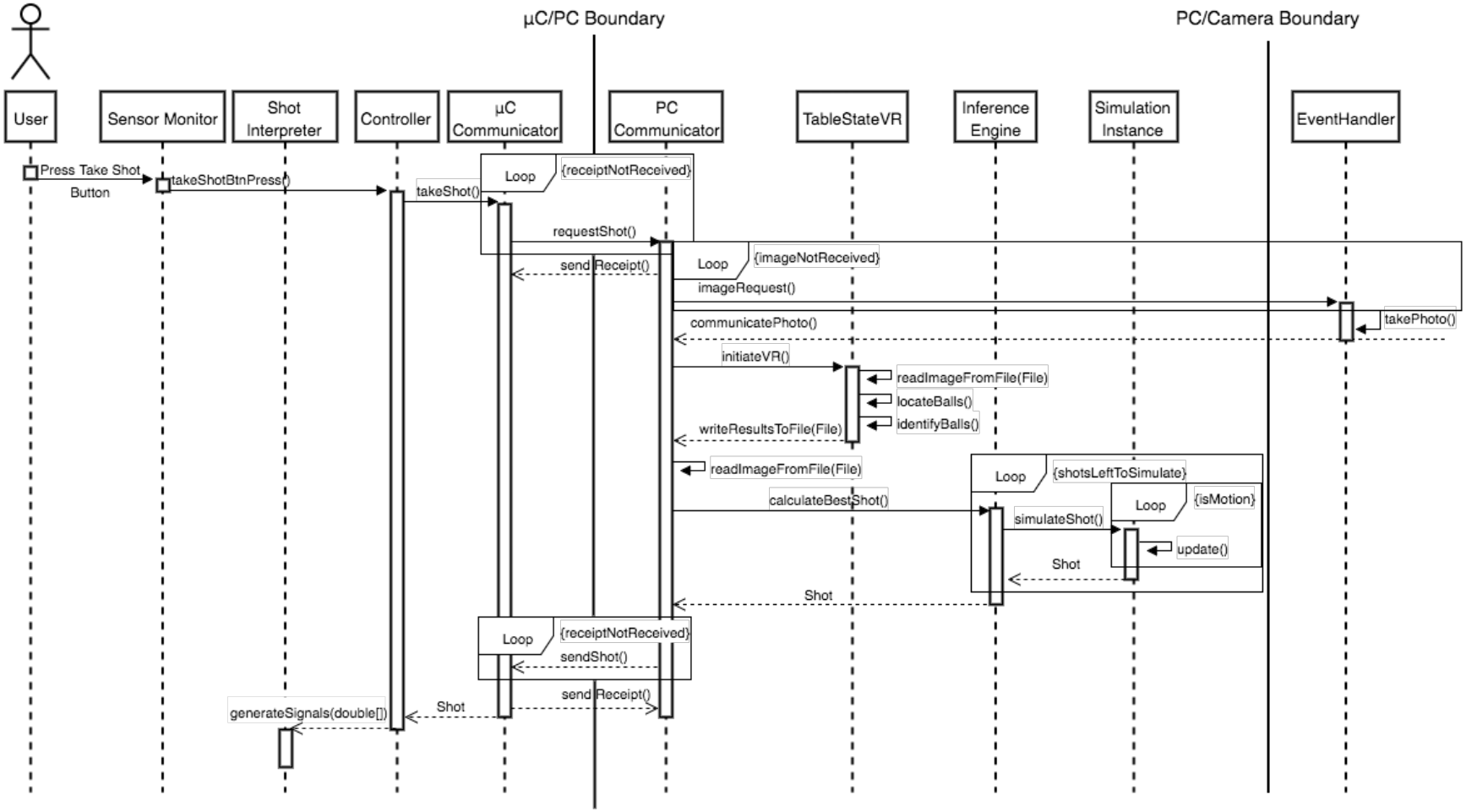

Figure 12: A sequence diagram for the "take shot" operation.