Florida International University
School of Computing and Information Sciences

CIS 4911 - Senior Capstone Project
Software Engineering Focus

Final Deliverable

Project Title:
Urban Decision Theater 1.0

Team Member:
Tkachenko, Olena
Santana, Renan

Product Owner(s):
Mostafavi, Ali
Bobadilla, Leonardo

Mentor(s):
Peeraya, Inyim
Paez, Juan Sotomayor

Instructor: Masoud Sadjadi

Abstract

Extreme events such as flooding and storm surges due to sea-level rise are significant threats to urban areas. The Urban Decision Theater (UDT) is a 3D Unity application that simulates these extreme event scenarios to allow Urban City Planners to practice and model their decisions, with real world constraints, to find the best available outcome for their cities. UDT simulates such things as traffic, city flood pumps, time, budget, sea-level, and tide.The events and decisions are monitored and routed to an external version of the application. The decisions are also stored in an external server to be reviewed by a moderator. The system supports virtual reality to provide an immersive decision experience in evaluating various scenarios.

The main purpose of UDT 1.0 is provide experience to urban-planners to make serious decisions in extreme event scenarios with limited resources like budget and time.

Table of Contents

INTRODUCTION

In the current system, Urban City Planners must solve complicated flooding scenarios and storm surge events due to sea-level rise without robust tools to assist them with decision making. This decision making process is critical for the well-being of the city and it sometimes results in multi-million dollar mistakes made by lack of experience and knowledge of the problem. The main goal in our system is to provide a simulation for Urban City Planners to experience the importance of making serious decisions with limited resources, modeling real world impediments like time and allocated city budgets. Our system will also stream the decisions in real-time to a moderator and persist each simulation to an external server, so that moderator can review the quality of the decisions and compare them to previous attempts.

This document will start-off by describing the current system along with its limitations; then a high-level description of the new system that will add new contributions to the current system. The following section will provide an overview of all of the completed user stories that were implemented along with those that were not. Next, there will be a description of the types of software and hardware configurations used to make this project including a list of sprints, along with the full description of the completed user stories. Subsequent section will detail the Architectural patterns, System and Subsystem Decomposition, Deployment Diagram, and Design Patterns. Finally, we conclude with our system validation section.

Current System

Presently, Urban City Planners (Decision Makers) do not have robust tools to assist them in solving complicated flooding scenarios and storm surge events due to sea-level rise that threaten many urban areas. Typically Decision Makers rely on regional height data gathered from the satellite and geological terrain types from many parts of the urban city. However, there are many unknowns to the geological terrain, since over time there can garbage buildup that can cause toxins to change the permeability of the terrain. This can lead to a persistent flooding, which can cause problems like heavy traffic, health-issues, saltwater intrusion, sewer overflow and many other issues. A popular solution for persistent flooding is the use of expensive city flood pumps and new, efficient sewers to minimize the flooding of the urban city. Even then, the methods to resolving these issues take years to plan and execute. Thus, the limitations to this system includes time constraints on planning and building, budget evaluation, team organization, site evaluation and other details for a large scale project execution. It follows, that to make a less

than optimal layout decision in the placement of the new pumps and sewer systems would waste the city's money while still leaving the underlying problem of flooding to be dealt with.

Purpose of New System

The new system, Urban Decision Theatre 1.0 (UDT), will simulate these extreme flooding scenarios to allow Urban City Planners to practice and model their decisions with limited time and budget to find the best available outcome. The system will have two types of users; a Decision Maker who is also the Urban City Planner and a Moderator who sets the Decision Maker's starting budget and views the decisions made by the Decision Maker. The system will use Oculus Rift for immersive visualization experience, simulate sea-level rise and tide, functional traffic, city improvement budget, city flood pump placement, time, population mood, saltwater damage, and other; the Decision Maker is able to change and influence some of these functionalities. For example, the placement of the pump by the Decision Maker will affect the flooding of the neighboring roads which would influence the traffic on those roads. The Moderator is able to examine the decisions of the Decision Maker and judge how well the Decision Maker responds to the scenario. The Moderator will also be able to compare previous attempts by analyzing the Decision Maker decisions and determine the Decision Maker's progress towards solving the flooding problems of the urban city. In-all, this application provides Urban City Planners with immersive experience and practice for making serious decisions to solve complicated urban flooding problems.

USER STORIES

This section will survey implemented and pending user stories that were outlined throughout this semester. The pending user stories were relegated to lower priority this semester but are slated for future implementation in the second version of Urban Decision Theatre. Each story will contain a brief description of the feature to be implemented. The stories will also be ranked by importance in descending order. There are a total of 19 user stories that were implemented and 8 user stories that are pending.

Implemented User Stories

User Story # 754 - Create Urban City Sectors

As a Moderator, I would like to have the city be divided into sectors or blocks, so that data (traffic, water height, and population emotions) can be generated for the individual sectors.

User Story # 682 - Create Pump Water Removal System

As the Decision Maker, I would like my pumps to remove water in the area around them so that they simulate how real pumps drain flooding.

User Story # 734 - Create Sea Level Manager

As a Moderator, I would like to have the sea-level to rise or fall with the corresponding time, so that tide can simulated.

User Story # 704 - Create Navigation

As the Decision Maker, I would like to have navigation capability so that I can navigate over my city and evaluate my decisions.

User Story # 778 - Add Traffic Intensity

As a Moderator, I would like the road nodes to have a quantity attribute, so that the "cars" can occupy the node.

User Story # 768 - Store Decisions

As a Decision Maker, I would like to store my decisions for each game in a database, so that I have a way to remember and judge my past gaming attempts.

User Story # 782 - Produce Path

As a Moderator, I would like the path to be found through the path request manager, so that the manager can select the request of a path search and not have search conflicts in the grid.

User Story # 763 - Consume Path

As a Moderator, I would like the path to be traversed through a separate thread, so that the many threads can work on separate paths.

User Story # 728 - Display Game Time

As a Moderator, I would like to have the game time be displayed with a distinction b/w game time and real time, so that the game time can be simulated at a different speed.

User Story # 788 - Create View Perspectives

As a Decision Maker, I would like to view many perspectives of my urban city, so that I can easily view my decisions and their consequences to the urban city.

User Story # 787 - Display Traffic

As a Decision Maker, I would like to be able to view the traffic as a grid, so that I'm aware of the amount of traffic happening in the urban city.

User Story # 722 - Display Budget

As a Moderator, I would like to have the budget be displayed on the top of the screen, so that I'm aware of how much I have.

User Story # 717 - Enable Oculus

As a Decision Maker, I would like to use all the Oculus VR features available in Unity while playing Urban Theater, so that I have the best possible VR experience Oculus can provide.

User Story # 799 - Add Traffic Time of Day

As a Moderator, I would like the traffic to be correlated with the time of day, so that the traffic can function and appear more realistically.

User Story # 803 - Create Budget-Pump Transaction

As a Decision Maker, I would like to pay for my pump placement with my budget, so that my Urban City simulates real world pump purchase costs.

User Story # 663 - Select Starting Budget

As a Moderator, I would like to have a budget slider, so that I can determine my starting budget when entering the game.

User Story # 666 - Start Menu

As a Moderator, I would like to have a start menu, so that the when I run the executable I get a menu page.

User Story # 677 - Begin Button

As a Moderator, I would like to have a "Begin" button that loads the simulation scene, so that I can start the simulation for the Decision Maker.

Pending User Stories

User Story # 676 - Create Information Dashboard

As a Decision Maker, I would like to have an information panel, so that I can see number of pumps placed, how many nuisance flooding, population emotion, areas mostly impacted, and other.

User Story # 807 - Add Player Boundary

As a Decision Maker, I would like the player to not go out of bounds in the city, so that player is focused on solving the problem and not explore.

User Story # 676 - Create Introductory Information

As a Decision Maker, I would like to be presented with an introduction information when the Simulation Scene is first loaded, so that I have an understanding of the objectives in the game.

User Story # 808 - Add Sea Level Effect to Traffic Flow

As a Moderator, I would like to have the sea level effect the traffic flow, so that the sea level can cause the traffic to be higher.

User Story # 809 - Create Population Emotion

As a Moderator, I would like to have the population react to environment in the urban city, so that the population in certain sectors can be between happy and mad.

User Story # 810 - Add Pump Pipes

As a Decision Maker, I would like to be able to place pipes that are connected to the pumps and sewers, so that I can make the water flow in the pipe system more efficient.

User Story # 769 - Add Raise Land Functionality

As a Decision Maker, I would like to be able to raise the land, so that I can create a water barrier.

User Story # 769 - Add Traffic-Pump Interaction

As a Decision Maker, I would like my pumps to interact with neighboring traffic, so that the pumps help in easing the city traffic around them.

PROJECT PLAN

Urban Decision Theatre (UDT) 1.0 follows the scrum development model. In this model the semester is divided into seven sprints, each sprint lasting a total of two weeks. For each sprint each developer would pick one or more user stories to finish. There are two developers working on UDT 1.0: Olena Tkachenko and Renan Santana, who worked on the implemented user stories in the above section. Ali Mostafavi, UDT project owner, supervised the direction and features that UDT needed to have. The Gantt chart in *Appendix E* lists, in detail, when and by whom the user stories were implemented from January to May. Orange color marks the user stories implemented by Olena while blue represents the user stories implemented by Renan.

Hardware and Software Resources

The following OS, programming language, framework, and development tool were used to develop this project:

Hardware

- ***Oculus Rift*** *: (Was used in conjunction with Unity to provide a more immersive gaming experience for UDT users. This includes Oculus Rift drivers to connect the device to our application)*

Software

- ***Mingle*** *: (Software development management tool, required by the instructor)*
- ***GitHub*** *: (Web Repository used for control and source code management)*
- ***Git Shell*** *: (Used to communicate / control / manage the GitHub repo server)*
- ***Google Drive*** *: (Repository used to shared documents among the team)*
- ***MongoDB*** *: (Store gaming attempts information on our remote server)*
- ***Microsoft Visual Studio 2016*** *: (Used to debug and write C# scripts)*

- ***Unity 5.3 64-bit*** *: (Game engine used to render our models and run our scripts)*
- ***Skype*** *: (Team communication management)*
- ***Draw.io*** *: (Web based application used to create UML diagrams)*
- ***Microsoft Visio Studio 2016*** *: (Used to create UML diagrams)*
- ***Visual_Paradigm_12.2 CE****: (Used to create UML diagrams)*
- ***Unity Asset Store*** *: (Community driven / created components used to expedite the project)*
- ***Windows 10 Xbox Game DVR*** *: (Used to record demo videos)*
- ***C#*** *: (Used to program the game scripts that Unity will run in the project)*
- ***JavaScript*** *: (Used to program the game scripts that Unity will run in the project)*
- ***Windows 10*** *: (Main OS that runs the Unity application)*
- ***Linux (Ubuntu v12+)*** *: (Virtual machine that hosts the server services)*
- ***Visual Paradigm Community Edition*** *: (Used to create UML diagrams)*
- ***Camtasia*** *: (Used to record project videos)*
- ***Terrain.party*** *: (Web site to get the height map of Miami area)*

Sprints Plan

For a total of 7 Sprints, each sprint lists all the implemented user stories during the two week time frame, in order of descending priority.

Sprint 1
(01/16/2016 - 01/29/2016)

User Story # 663 - Select Starting Budget

Tasks

- 675 Testing
- 674 Research Feature
- 673 Implementation
- 672 Create Documentation

Acceptance Criteria

- Budget range should be [20-160 mil]
- The corresponding value of the slider should be reflected in text above slider as the user moves slider handle.

Modeling
Sequence Diagram: Figure # A.4 - Select Starting Budget
Class Diagram: Figure # A.2 - Minimal Class Diagram.

User Story # 666 - Start Menu

Tasks

- 671 Test Start Menu
- 670 Implementation of starting canvas
- 669 Research Unity Canvas as Menu
- 668 Create Documentation

Acceptance Criteria

- The application will start by an executable.
- The menu will have a white background.
- The menu will take up the entire screen.

Modeling
Sequence Diagram: Figure # A.5 - Start Menu

Class Diagram: Figure # A.2 - Minimal Class Diagram

Sprint 2
(01/30/2016 - 02/12/2016)

User Story # 677 - Begin Button

Tasks

- 690 Create Documentation
- 691 Implementation of Begin Button
- 692 Research Feature
- 693 Testing
- 694 Import Terrain to Environment
- 696 Add Features to terrain
- 697 Import pre-built Urban Model
- 698 Create Skybox
- 710 Create Static Water
- 726 Revision of Document

Acceptance Criteria

- The button needs be named “Begin” and have a 32 font size.
- The button needs to be in the lower right hand side of the canvas.
- The button needs to be a white color with black text.
- After the begin button is clicked, the loaded scene must have :
    - Miami model terrain.
    - Urban model
    - Non functioning water model
    - Day time lighting

- Texture on the terrain model (grass texture / trees)

Modeling
Sequence Diagram: Figure # A.6 - Begin Button
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 704 - Create Navigation

Tasks

- 708 Implement Navigation
- 707 Research Tasks
- 706 Testing
- 705 Create Documentation

Acceptance Criteria

- player can move in the 'city' via keyboard arrows
- player can change the view via mouse movement.

Modeling
Sequence Diagram: Figure # A.8 - Create Navigation
Class Diagram: Figure # A.2 - Minimal Class Diagram

Sprint 3
(02/13/2016 - 02/26/2016)

User Story # 717 - Enable Oculus

Tasks

- 740 Testing
- 739 Research Oculus
- 738 Create Documentation

- 737 Implement player controls for Oculus

Acceptance Criteria

- The player controller is adopted to Oculus.
- The camera in the game scene is adopted to Oculus.

Modeling
Sequence Diagram: Figure # A.9 - Enable Oculus
Class Diagram: Figure # A.2 - Minimal Class Diagram

Sprint 4
(02/27/2016 - 03/11/2016)

User Story # 682 - Create Pump Water Removal System

Tasks

- 770 Create Documentation
- 762 Research
- 761 Implement pump place blocks
- 750 Testing
- 749 Implement Pump Drain Component
- 748 Create Documentation
- 747 Research Mesh on collision in Unity

Acceptance Criteria

- Pump is able to remove water out of the city in a circle area.

Modeling
Sequence Diagram: Figure # A.7 - Create Pump Water Removal System
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 722 - Display Budget

Tasks

- 736 Revision of Document
- 727 Research persistent data and HUD
- 725 Implement Budget HDU
- 724 Test Persistent data is passable b/w scenes
- 723 Create Documentation

Acceptance Criteria

- The budget will be displayed on the top right.
- The budget text must be visible with any background.
- The budget text must be large enough (around size 18 font).

Modeling
Sequence Diagram: Figure # A.10 - Display Budget
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 728 - Display Game Time

Tasks

- 735 Revision of documentation
- 733 Implement game time display
- 732 Test game time speed
- 731 Create Documentation
- 730 Research how to accelerate time
- 729 Implement Game Time

Acceptance Criteria

- Every one minute in real time the game time clock would pass 24 hrs.

- The time is to be displayed at the top center of the screen.

Modeling
Sequence Diagram: Figure # A.11 - Display Game Time
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 734 - Create Sea Level

Tasks

- 746 Revision of documentation
- 745 Test the tide rise/fall
- 744 Research real life Miami tide times
- 742 Implement Sea level rise/fall
- 741 Create Documentation

Acceptance Criteria

- Sea level must rise or fall according to the time
    - The sea level tide must follow a real life tide
        - Use NOAA data as the model

Modeling
Sequence Diagram: Figure # A.12 - Create Sea Level Manager
Class Diagram: Figure # A.2 - Minimal Class Diagram

Sprint 5
(03/19/2016 - 04/01/2016)

User Story # 754 - Create Urban City Sectors

Tasks

- 760 Research how to create a grid with nodes
- 759 Implement sectors in the city

- 758 Create Documentation
- 757 Test grid and nodes

Acceptance Criteria

- The city must be divided into sectors.
- Each sector must be able to **store data** about the condition of itself.
    - Traffic data
    - Population data
    - Water data
- No data generation yet
    - Will be a separate user story

Modeling

Sequence Diagram: Figure # A.13 - Create Urban City Sectors
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 763 Consume Path

Tasks

- 766 Test traffic level
- 765 Implement consumer
- 764 Create Documentation

Acceptance Criteria

- When the thread visits a particular road node in the grid it will:
    - occupy the space by decrementing the traffic allowed through.
    - Start an asynchronous timer; when the timer is elapsed, a method will release the occupied space.

Modeling
Sequence Diagram: Figure # A.14 - Consume Path
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 768 - Store Decisions

Tasks

- 768 Implement Testing cases
- 775 Implement Unity client code
- 774 Implement Database in VM
- 773 Research
- 772 Get VM Access
- 771 Documentation

Acceptance Criteria

- Each Decision Maker is identified by unique user_id.
- Each user_id may contain a list of attached files containing the decisions made by the Decision Maker for a game.

Modeling
Sequence Diagram: Figure # A.15 - Store Decisions
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 778 - Add Traffic Intensity

Tasks

- 781 Test traffic intensity values
- 780 Implement traffic intensity
- 779 Create Documentation

Acceptance Criteria

- Each node will take into account the amount of:

    - neighbor roads (entering and exiting traffic).
    - itself
- The max amount of traffic any node can have is 17.
- The min amount of traffic any node can have is:
    - Zero, if the road is fully being occupied
    - Three, if the road is a corner piece with only one neighbor

Modeling
Sequence Diagram: Figure # A.16 - Add Traffic Intensity
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 782 - Produce Path

Tasks

- 786 Test production of path
- 785 Research path finding
- 784 Implement producer
- 783 Create Documentation

Acceptance Criteria

- Producer is a separate thread
- Produce paths will create a request to the request manager which enqueues the start and end point of a path.
- The request manager dequeues the request and sends it to the FindPath where the path will be found.

Modeling
Sequence Diagram: Figure # A.17 - Produce Path
Class Diagram: Figure # A.2 - Minimal Class Diagram

Sprint 6
(04/02/2016 - 04/15/2016)

User Story # 787 - Display Traffic

Tasks

- 793 Test traffic value in grid
- 792 Research grid display
- 791 Implement traffic display
- 790 Create Documentation

Acceptance Criteria

- Display the traffic if Ctrl key is pressed down; hide the traffic if the ctrl key is not pressed.
- The display should be big enough to see.
- The traffic display should update with in 1-1.5 seconds.

Modeling
Sequence Diagram: Figure # A.18 - Display Traffic
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 788 Create View Perspectives

Tasks

- 798 Research
- 797 Research look to fly
- 796 Testing
- 795 Implementation zoom vision
- 794 Unity Birds Eye View

- 789 Documentation

Acceptance Criteria

- Decision Maker has bird's eye view of the city.
- Decision Maker can use mouse scroll to zoom out of first person perspective to look down on the city.

Modeling
Sequence Diagram: Figure # A.19 - Create View Perspectives
Class Diagram: Figure # A.2 - Minimal Class Diagram

Sprint 7
(04/16/2016 - 04/29/2016)

User Story # 799 - Add Traffic Time of Day

Tasks

- 802 Test the time of day with the traffic
- 801 Create Documentation
- 800 Implement time of day into the traffic

Acceptance Criteria

- There should be more traffic b/w the morning and lunch time.
- There should be more traffic b/w the 3 and 6.
- There should be fewer traffic b/w 7 and 9.
- At 10 pm – 6 am the traffic should be very minimum.

Modeling
Sequence Diagram: Figure # A.20 - Add Traffic Time of Day
Class Diagram: Figure # A.2 - Minimal Class Diagram

User Story # 803 Create Budget-Pump Transactions

Tasks

- 806 Testing
- 805 Implementation budget deduction
- 804 Feature Documentation

Acceptance Criteria

- Each Pump should cost the Decision Maker 5 million $.
- The Decision Maker should not be able to place any pumps if budget is under 5 million.

Modeling

Sequence Diagram: Figure # A.21 - Create Budget-Pump Transactions

Class Diagram: Figure # A.2 - Minimal Class Diagram

SYSTEM DESIGN

This section describes Urban Decision Theatre 1.0 (UDT) high level system design. We begin by describing the Architectural Patterns of UDT. Followed by a discussion of UDT system and subsystem decomposition. Then, a section on UDT system deployment will discuss the types of services and applications that must be running and the environment types that they must be running on. Finally we will identify the design patterns that were used in UDT.

Architectural Patterns

In our project, the Model View Controller (Figure 1) architecture was incorporated in Urban Decision Theatre 1.0 (UDT). MVC was used because Unity is a user-interactive event driven system and UDT was built on Unity Game Engine. This pattern helped us apply C# scripts to the project and also have them work together as a controller / entity. The second architectural pattern used in UDT was the Client-Server. With our understanding on the MVC role in Unity we were able to connect to the Server Database from our application (a.k.a Client).

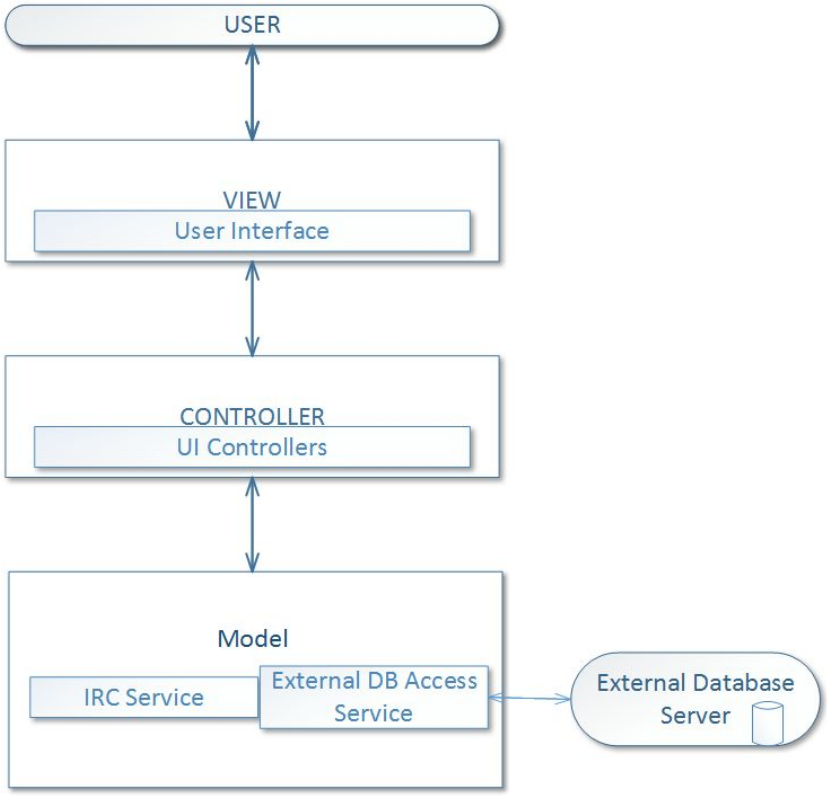


Figure 1 - Architecture Pattern

System and Subsystem Decomposition

Referring to Figure 2, the two main subsystems of UDT are the Unity application and Virtual Machine. The Unity Application houses three subsystems: Model, View, and the Controller. Virtual Machine, is our external server and it contains the services needed to support the features of Unity Application like MongoDB database and IRC service.

Model itself contains three types of subsystems. First, the entities of the game, which are called GameObjects. These entities are used in UDT to manipulate the game objects' position, visibility, scale, and other. The other two subsystems are client services that are needed to communicate and connect to an outside server.

The View subsystem houses the different scenes that in UDT. As well as the user-interaction the scenes provide, such as GUIs.

The Controller subsystem is composed of UDT scripts that contain the logic to manipulate game objects and process user inputs. For example, during a scene load Unity will first loop through all scripts that are attached to a game object and instantiate the script by first calling the MonoBehavior Awake function. Also, during a user interface interaction the system will be able to respond to the requested action of the user through a GUI or Hardware controls.

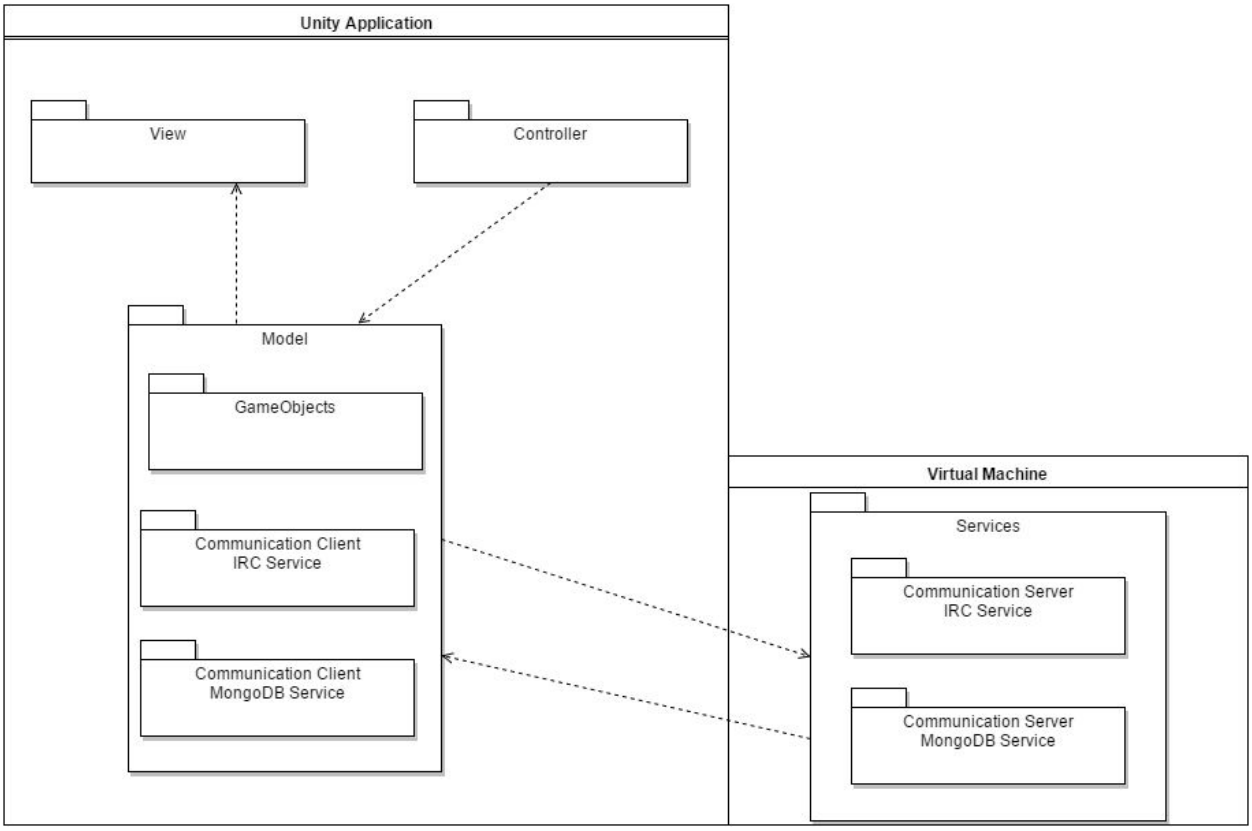


Figure 2 - System Decomposition (Package Diagram)

Deployment Diagram

Referring to Figure 3, the deployment of UDT is composed of two instances of our application running on two different machines and a Linux Virtual Machine.

The Application will run for the Decision Maker (user) locally. This system is connected to the Linux virtual machine server by two types of connector services. The two types of services in the connection are the IRC and the MongoDB, which are routed through a TCP/IP protocol.

The Linux virtual machine is a middleman, its purpose is to store information and relay the decisions of the Decision Maker to the second version of the application.

The second instance of the Application will be a version of the application that the Moderator (user) will control. And again, this system is just like the first instance of the application, but it won't be using storage and relay services for itself. The purpose of this instance is to view or monitor the decisions made by the Decision Maker.

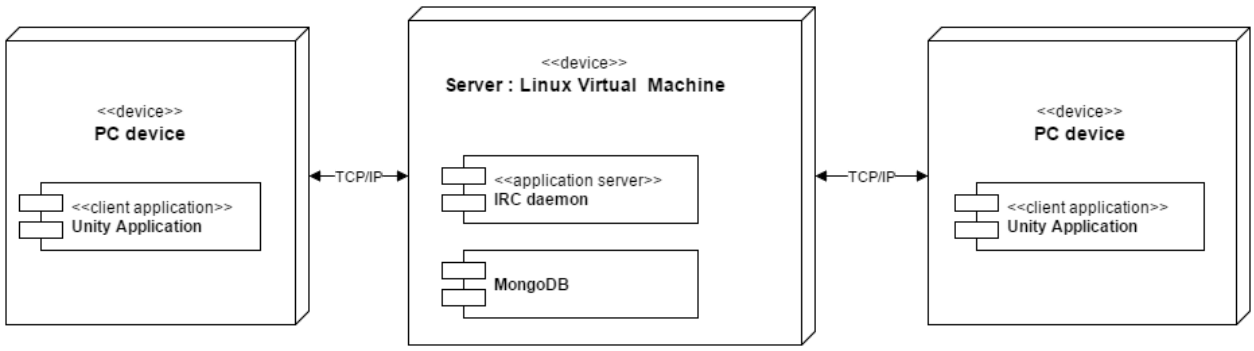


Figure 3 - System Deployment Diagram

Design Patterns

Singleton
In the Model subsystem, where the game objects reside, there is only one instance of a particular object. This design pattern allows the system to find the particular object in a swift manner. Also, the ability of sending individual pump and/or sewer information to the database for storage applies to singleton, since there is only one instance of it in the scene (view).

SYSTEM VALIDATION

This section contains the validation tests performed during Urban Decision Theatre (UDT) development life cycle this semester. There are two types of tests performed: Subsystem, and System. Subsystem Tests verify the individual components of the User Story, they were performed using Unity Unit Testing Tools. System Tests verified correct behavior of the User Story when it was incorporated into the current UDT system, they were done with Unity Integration Testing Tools.

User Story # 663 - Select Starting Budget

Subsystem Tests

- UDT663- Start Menu - Canvas is visible via camera.
- UDT663- Select Starting Budget - Budget slider is initialized with minimal value.

User Story # 666 - Start Menu

System Tests

- UDT-IT666Start Menu SUNNY01-Scene Game objects rendered if active.

Subsystem Tests

- UDT-666-Start Menu SUNNY01- Game objects in the scene are updated.

User Story #677 - Begin Button

System Tests

- UDT-IT677-Begin Button - Ensure that Begin Button is dependent of the Simulation Scene.

Subsystem Tests

- UDT677-Begin Button - Begin Button loads and displays in new scene.

User Story # 682 - Create Pump Water Removal System

System Tests

- UDT_IT682-Pump_City-Test rendering of Pump water drainage in the city flood area.

Subsystem Tests

- UDT682-PumpDrain_SUNNY- Ensure that pump drains water around the area it was placed in terrain.

User Story # 704 - Create Navigation

System Tests

- UDT_IT704-PlayerController-User can navigate City Scene.

Subsystem Tests

- UDT704-PlayerControllerTest_SUNNY- Decision Maker Navigation works as expected.

User Story # 717 - Enable Oculus

System Tests

- UDT_IT717-City_Oculus - Test Oculus head movement inside simulation/city scene.

Subsystem Tests

- UDT717-Navigation_SUNNY- Ensure that head movement works as expected.
- UDT717-Register Oculus_RAINY - Display error message when Oculus is not registered by the game.

User Story # 722 - Display Budget

System Tests

- UDT_IT722- Display Budget-Changing the Budget in the StartMenu will display the value in the next scene (SimulationScene)..

Subsystem Tests

- UDT722- Display Budget_SUNNY- To ensure that the Budget is displayed in the SimulationScene.

User Story # 728 - Display Game Time

Subsystem Tests

- UDT728- Display Game Time SUNNY01 -To ensure that the Game Time is displayed in the SimulationScene.
- UDT728- Display Game Time RAINY01 - To ensure that the Game Time captures the 9999 year boundary.

User Story # 734 - Create Sea Level Manager

System Tests
- UDT_IT734- Create Sea Level Manager - The script is dependent on the water transform and game time.

Subsystem Tests
- UDT734- Create Sea Level Manager_SUNNY01 - To ensure that the water is being displaced with the given tide height.
- UDT734- Create Sea Level Manager_RAINY01 - To ensure that the Sea Level Manager captures the 9999 year boundary.

User Story # 754 - Create Urban City Sectors

System Tests
- UDT_IT754- Create Urban City Sectors - The size of the grid is dependent on the size of the city as-well-as the colliders contained in the city that represent the city sectors.

Subsystem Tests
- UDT754- Create Urban City Sectors_SUNNY01 - To ensure that the city is divided into sectors.

User Story # 763 - Consume Path

System Tests
- UDT_IT763- Consume Path01 - The script uses the path that was produced from the Produce Path.
- UDT_IT763- Consume Path02 - The script depends on the traffic intensity.

Subsystem Tests

- UDT763- Consume Path_SUNNY01 - To ensure that the consumer thread dequeues a path and traverses the path while modifying the traffic intensity.
- UDT763- Consume Path RAINY01 - To ensure that the traffic intensity of a node can not go below 0.

User Story # 768 - Store Decisions

System Tests

- UDT-IT768-GetFile- Upload a scene local file to existing user in the database.

Subsystem Tests

- UDT768-Upload_Sunny01 - Upload a local file to existing user in the database.
- UDT768-Upload_Rainy01 - Upload a local file to new user in the database.

User Story # 778 - Add Traffic Intensity

System Tests

- UDT_IT778- Add Traffic Intensity - The script is dependent on the grid being populated with city-sectors and road nodes.

Subsystem Tests

- UDT778- Add Traffic Intensity_SUNNY01 -To ensure that every road node is accounting its neighbor and itself to the traffic intensity attribute.
- UDT778- Add Traffic Intensit_ RAINY01 - There maybe a set of road nodes that are stranded.

User Story # 782 - Produce Path

System Tests

- UDT_IT782- Produce Path01 - The script is dependent on the grid being populated with city-sectors and road nodes.
- UDT_IT782- Produce Path02 - The script is dependent on the traffic intensity.

Subsystem Tests

- UDT782- Produce Path_SUNNY01 - To ensure that the producer thread requests for a path to be found given a start and end point of the path.

User Story # 787 - Display Traffic

System Tests
- UDT_IT787- Display Traffic - The script is dependent on the grid being populated with city-sectors and road nodes, HUD game object, Timer, and Image.

Subsystem Tests
- UDT787- Display Traffic_SUNNY01-To ensure that the calculated color is applied to the texture and then applied to the image.
- UDT787- Display Traffic_SUNNY02 - To ensure that the UI Image is hidden when the "Ctrl" key is released.

User Story # 788 - Create View Perspectives

System Tests
- UDT-IT788-Place Pump - To help the Decision Maker place pump from another perspective.

Subsystem Tests
- UDT788-ZoomUpward_SUNNY01 -Decision Maker wants to see city from the sky in first person.
- UDT788-ZoomUnderground_RAINY_01 - Decision Maker wants to zoom beneath the city.

User Story # 799 - Add Traffic Time of Day

System Tests
- UDT_IT799- Add Traffic Time Of Day -The script uses the uses the hour and min from the GameTime script.

Subsystem Tests

- UDT799- Add Traffic Time Of Day_SUNNY01 - To ensure that the node gets a delay time from the traffic time of day.
- UDT799- Add Traffic Time Of Day_RAINY01 - To ensure that the node traffic intensity value doesn't go below a zero value.

User Story # 803 - Create Budget-Pump Transactions

System Tests

- UDT-IT803-PumpCostScene - Deduct Pump Cost from Budget.

Subsystem Tests

- UDT803-DeductCost_SUNNY01 - Decision Maker wants to place pump in city.
- UDT803-DeductCost_RAINY01 - Decision Maker wants to place pump in city with low budget.

GLOSSARY - USERS

- ***Decision-Maker*** *- a user that is in control of making decisions in the simulation.*
- ***Moderator*** *- a user that is in control of the menu and all of its feature. The Moderator is able to view the decisions made by the Decision-Maker.*

GLOSSARY - UNITY TERMS

- ***Scene*** *- Scenes contain the objects of your game. They can be used to create a main menu, individual levels, and anything else.*
- ***Component*** *- functional pieces of every GameObject.*
- ***Game Object*** *- hold the different pieces that are required to make up a character, a light, a tree, a sound, or whatever else in the game.*
- ***HUD*** *- Head-up display, otherwise can be know as the non-interactable GUI.*
- ***Decision*** *- a decision is made by the Decision-Maker, which includes placing a pump and / or sewer.*
- ***Node*** *- a piece of a grid component that can represent a road and city sector.*
- ***City Sector*** *- considered as a city block*

APPENDIX

Appendix A - UML Diagrams

Static UML Diagrams

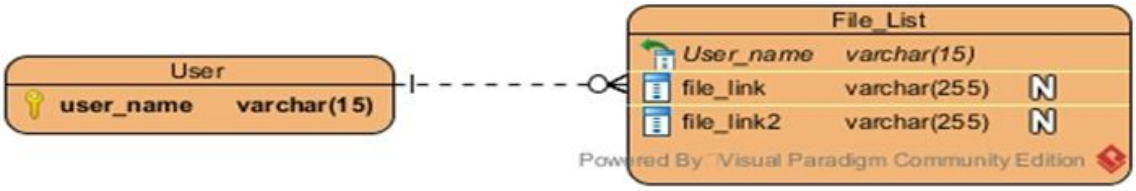


Figure # A.1 - Entity Relationship Diagram

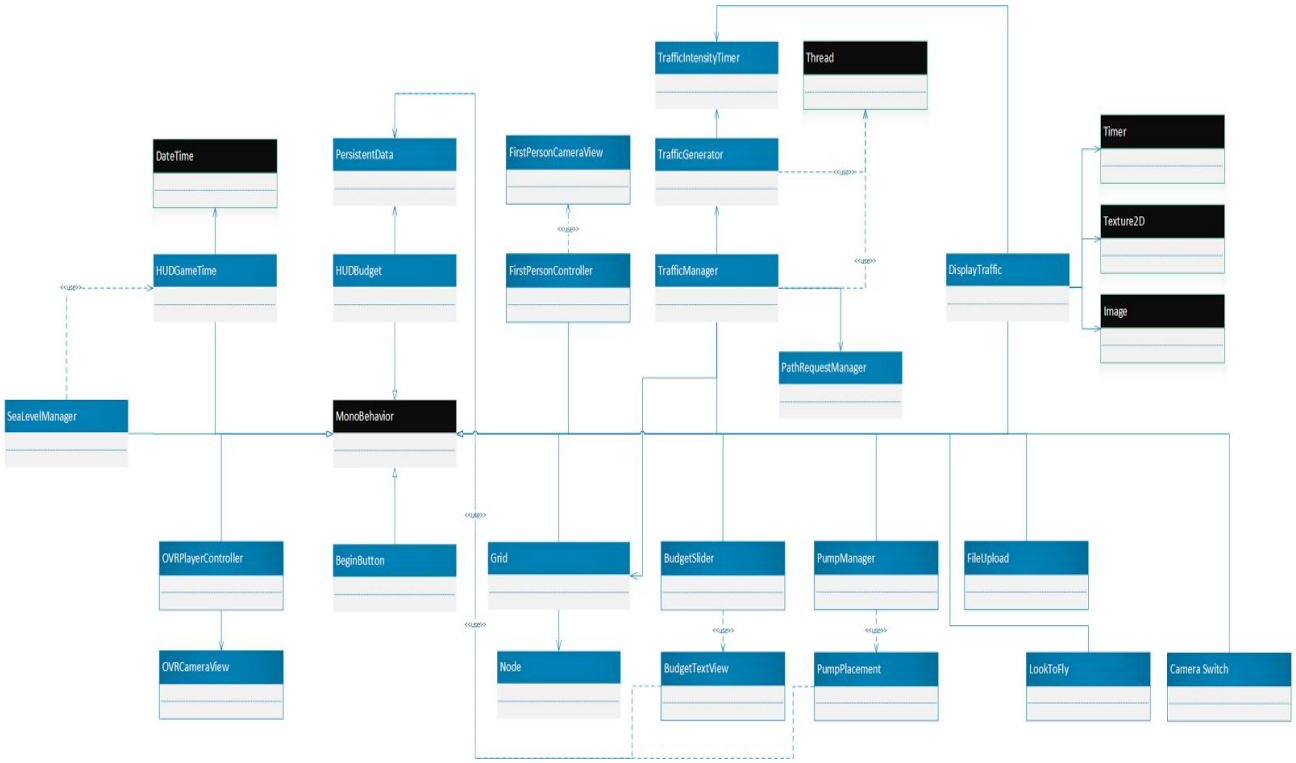


Figure # A.2 - Minimal Class Diagram

Dynamic UML Diagrams

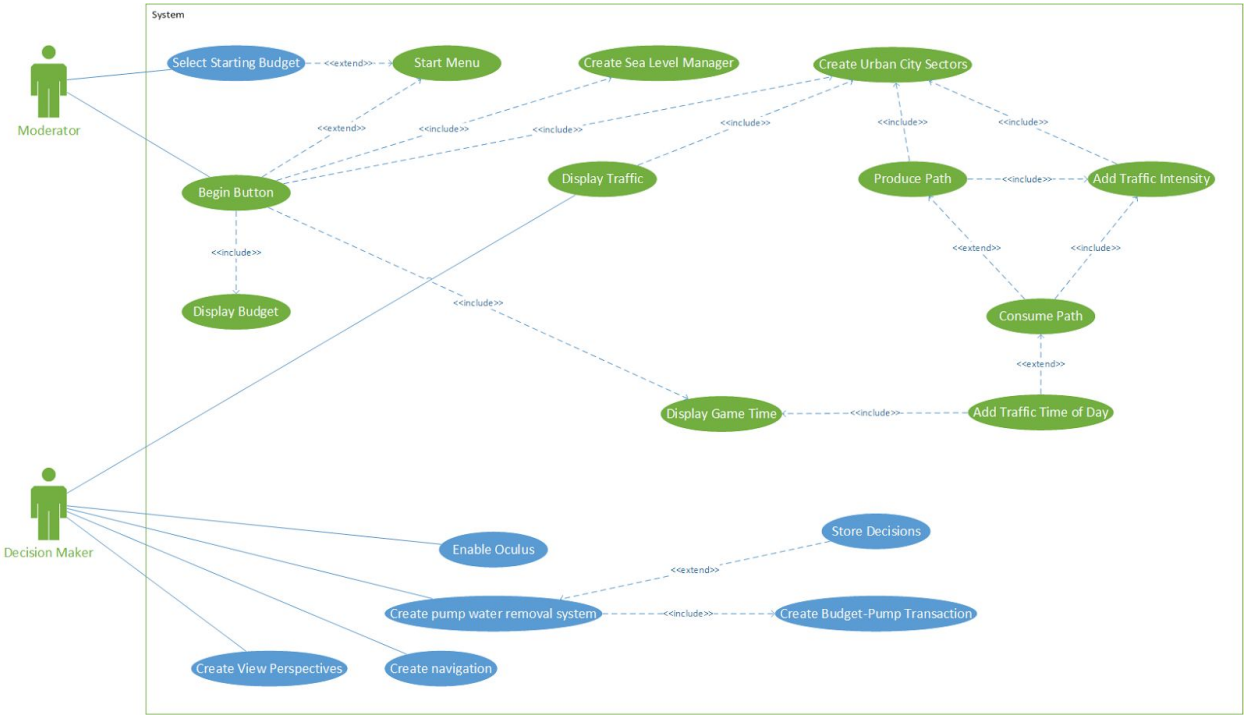


Figure # A.3 - Use Case Model

Sequence Diagrams

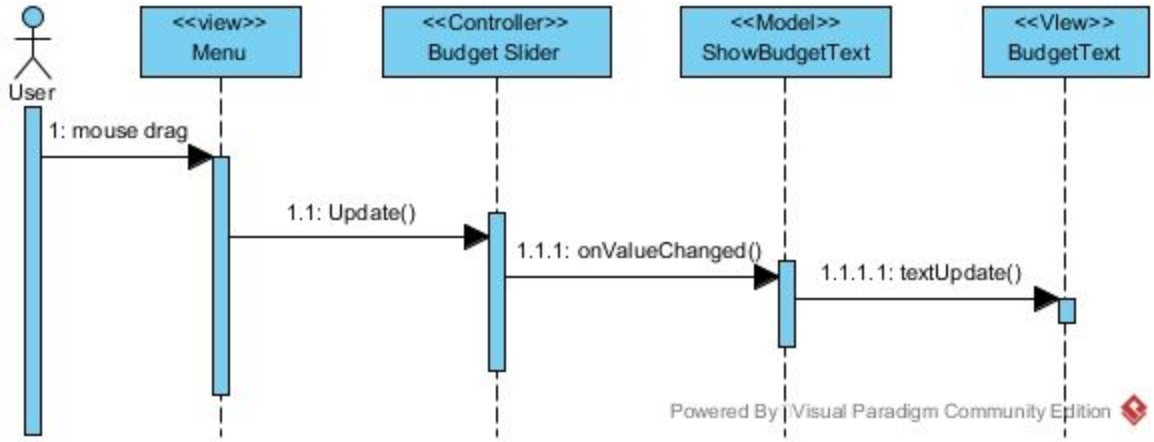


Figure # A.4 - Select Starting Budget

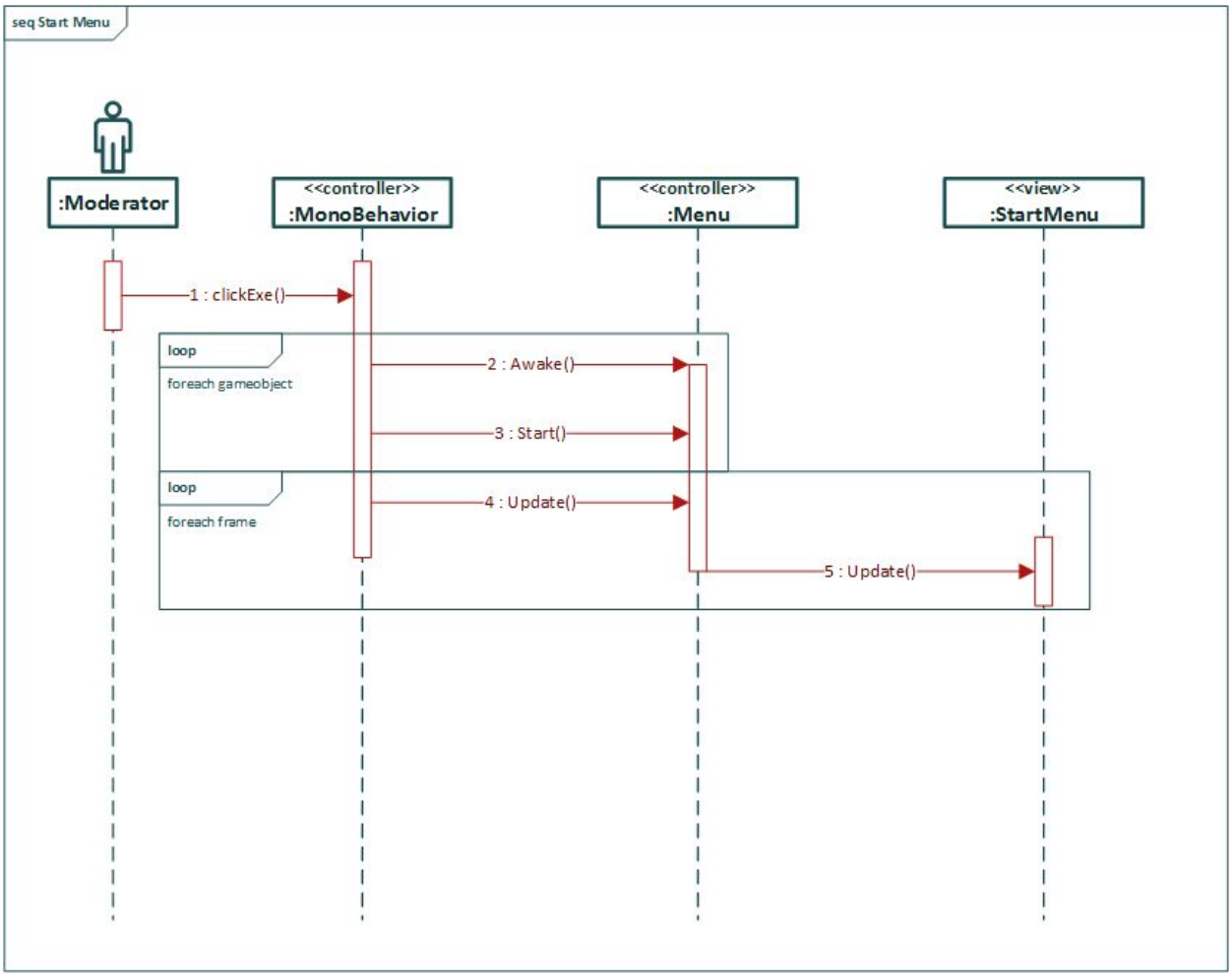


Figure # A.5 - Start Menu

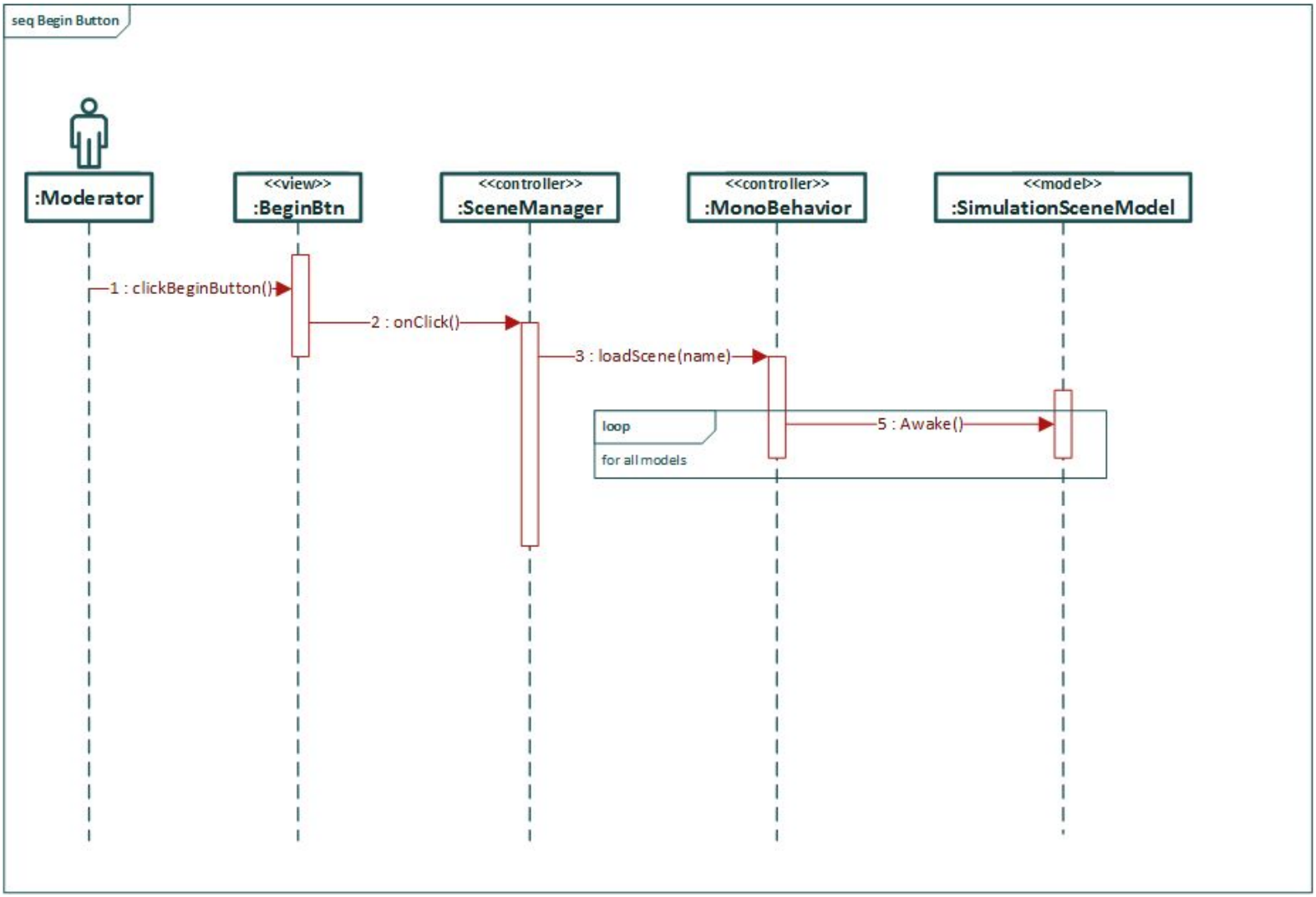


Figure # A.6 - Begin Button

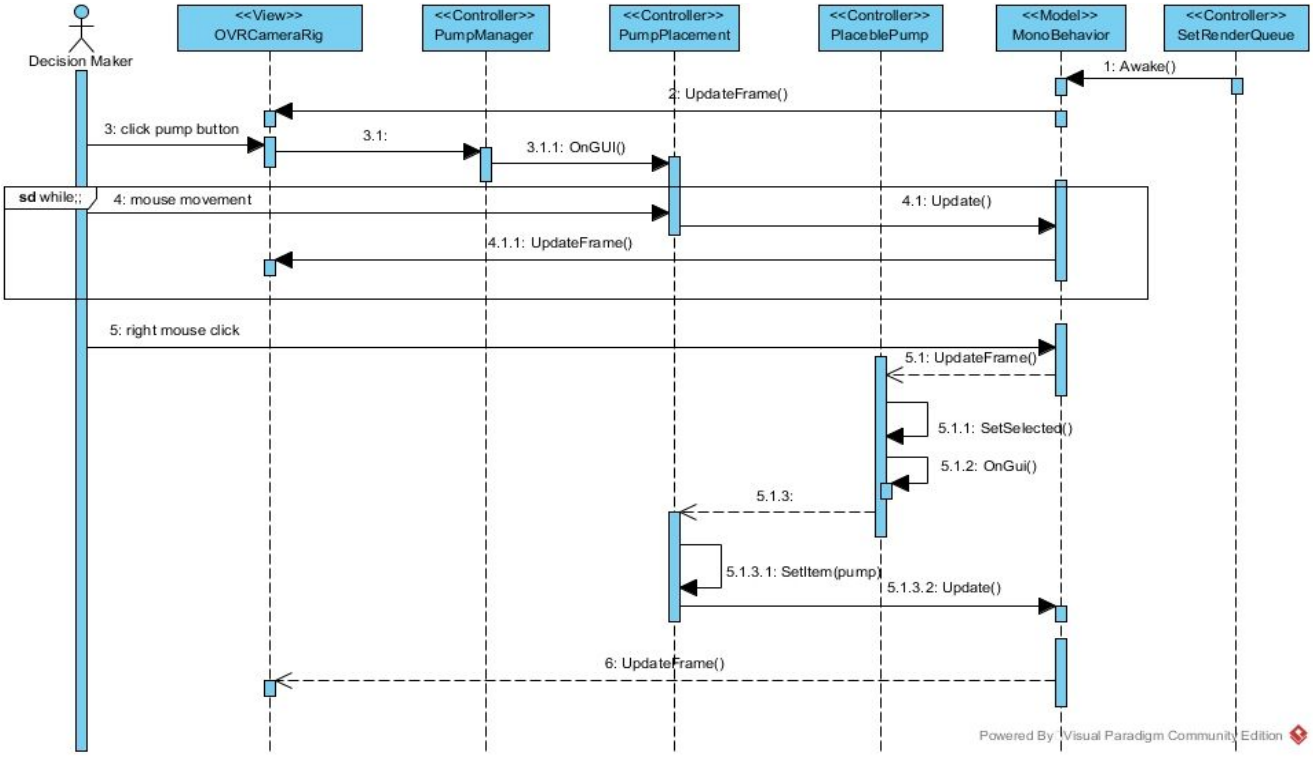


Figure # A.7 - Create Pump Water Removal System

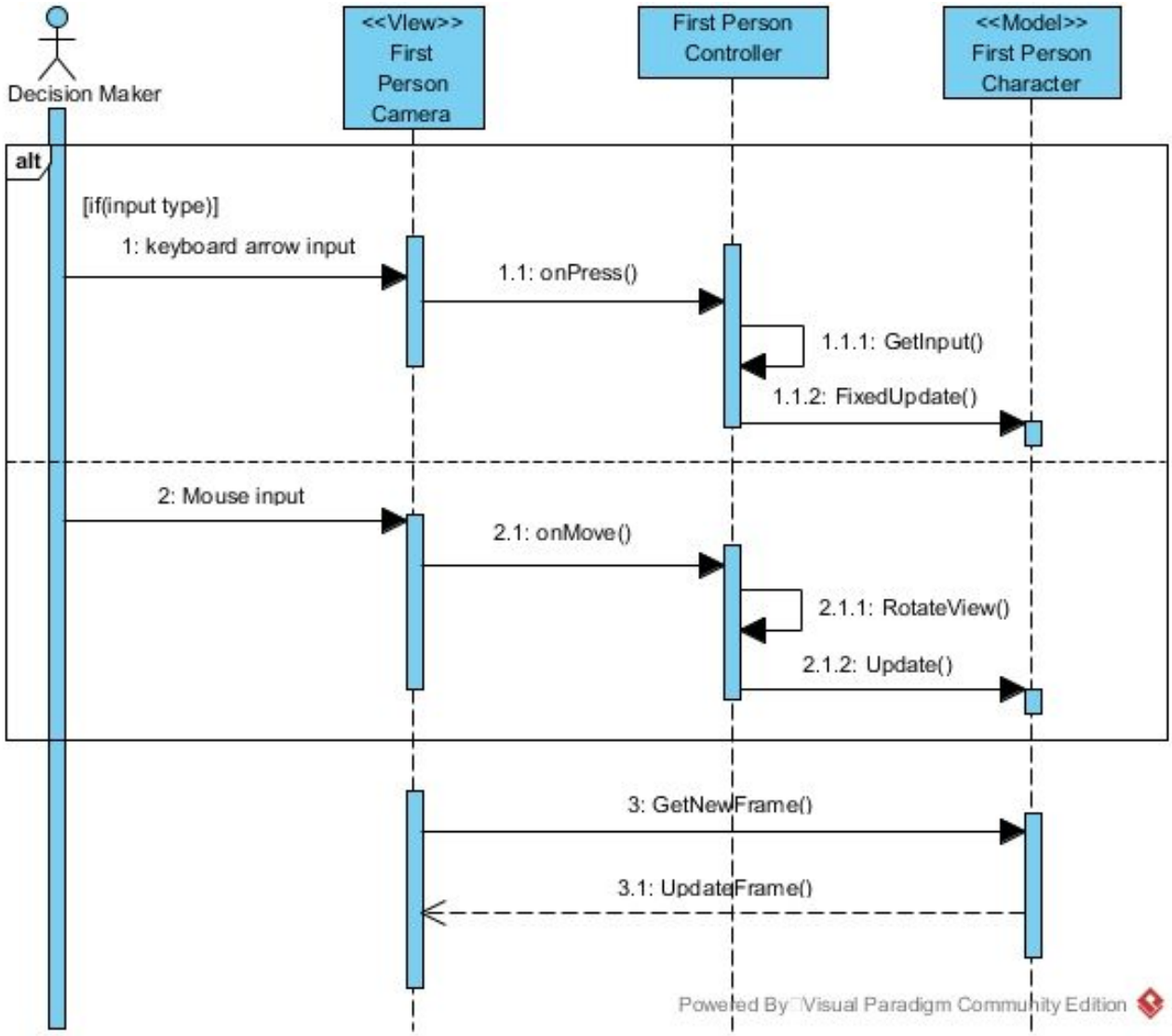


Figure # A.8 - Create Navigation

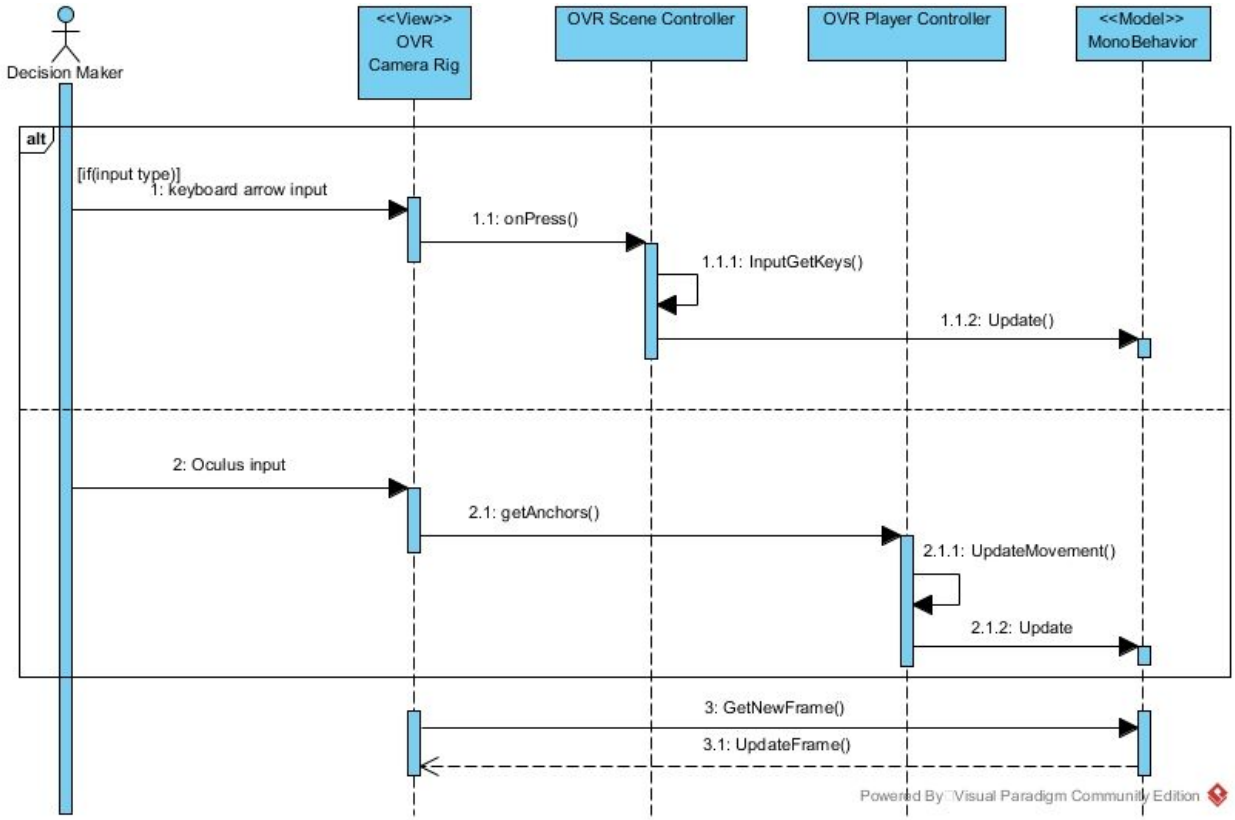


Figure # A.9 - Enable Oculus

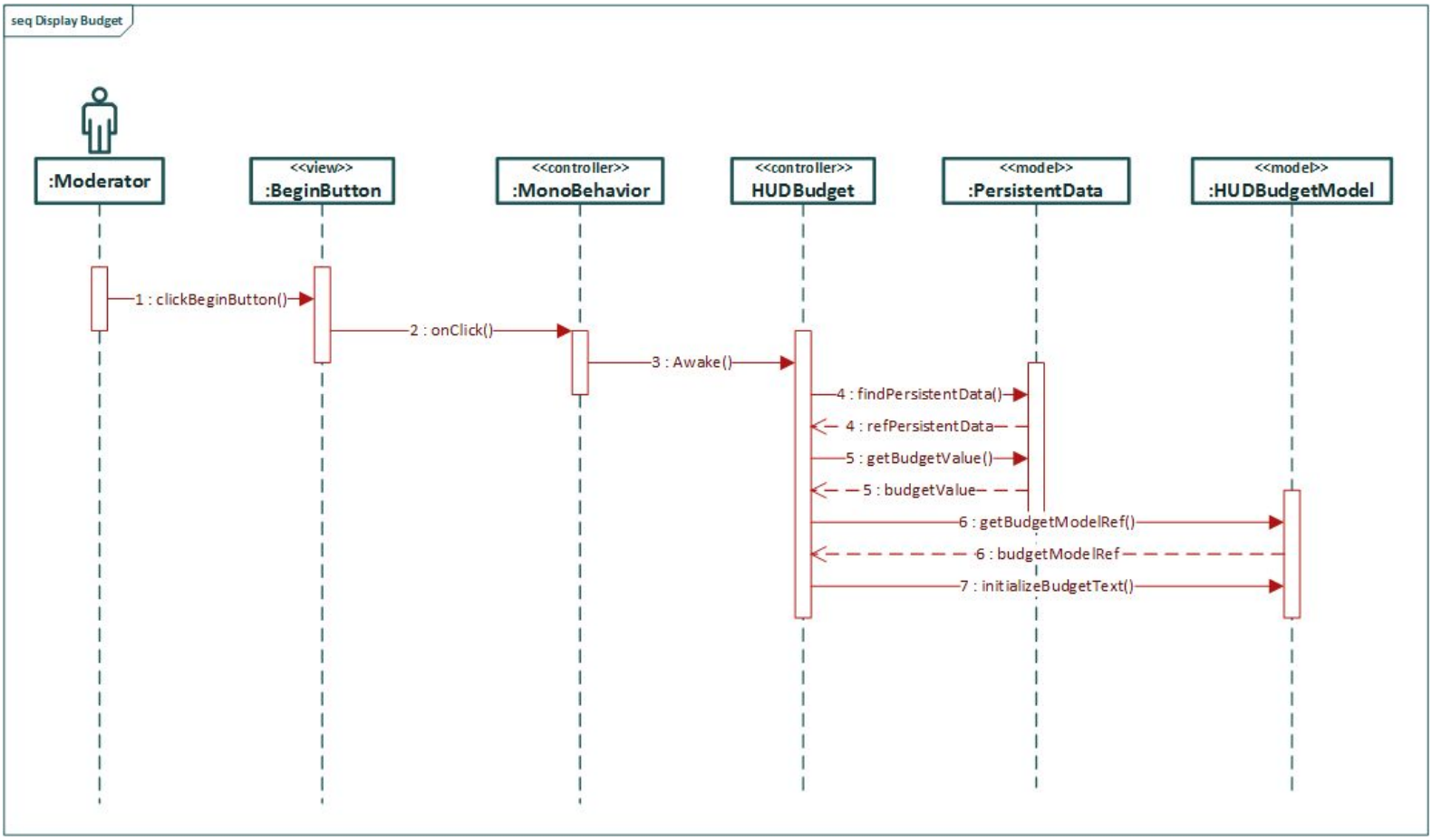


Figure # A.10 - Display Budget

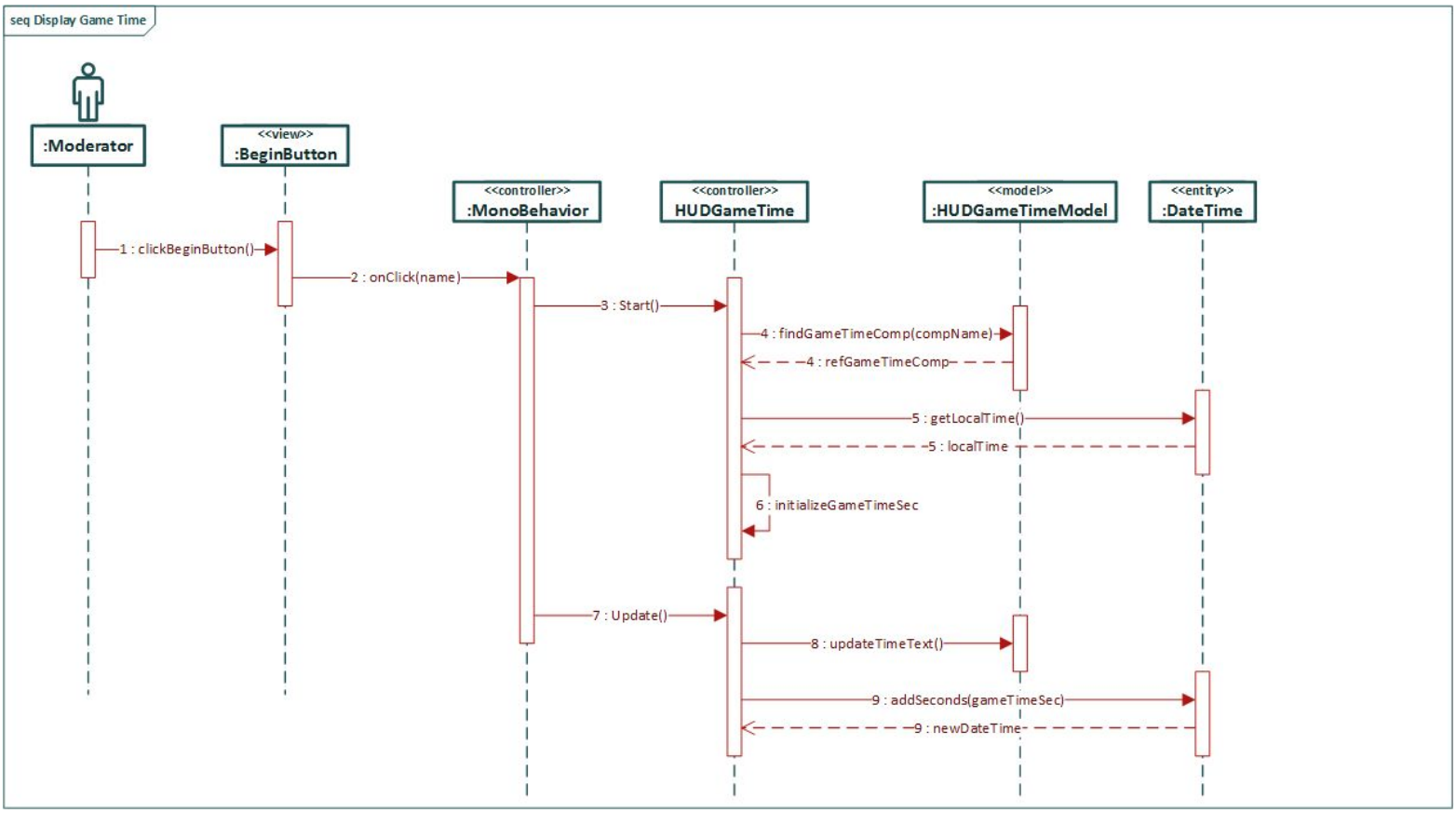


Figure # A.11 - Display Game Time

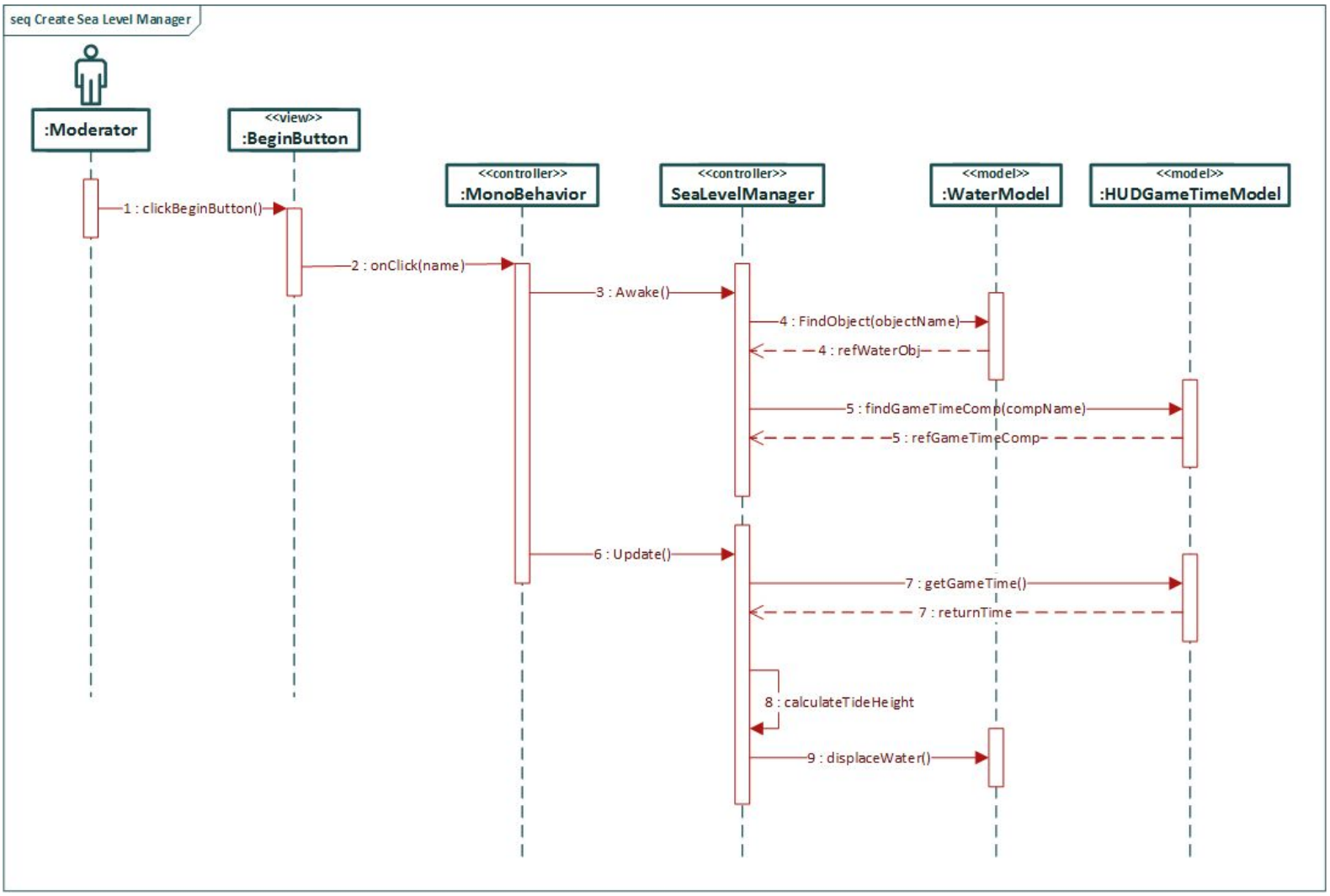


Figure # A.12 - Create Sea Level Manager

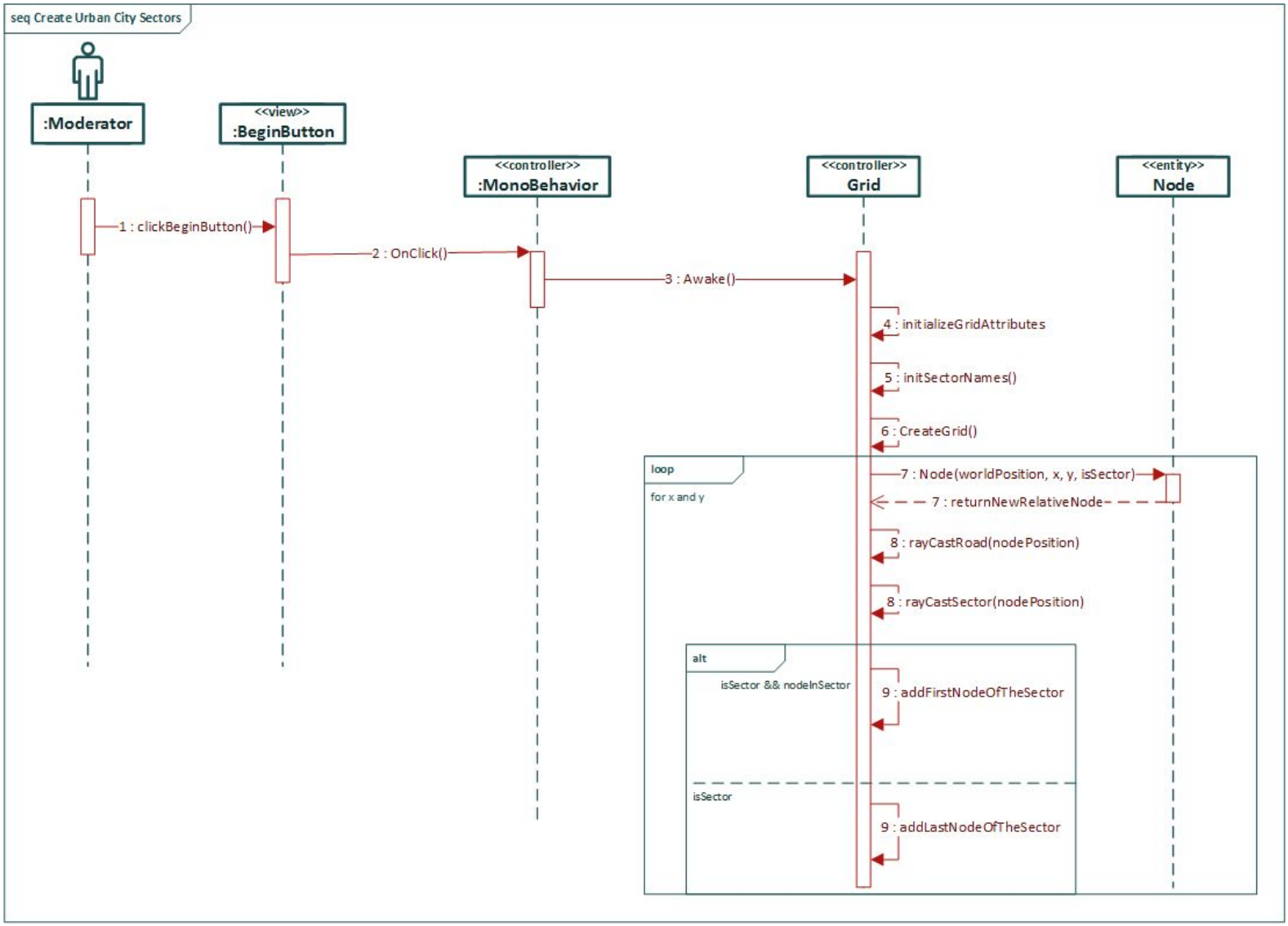


Figure # A.13 - Create Urban City Sectors

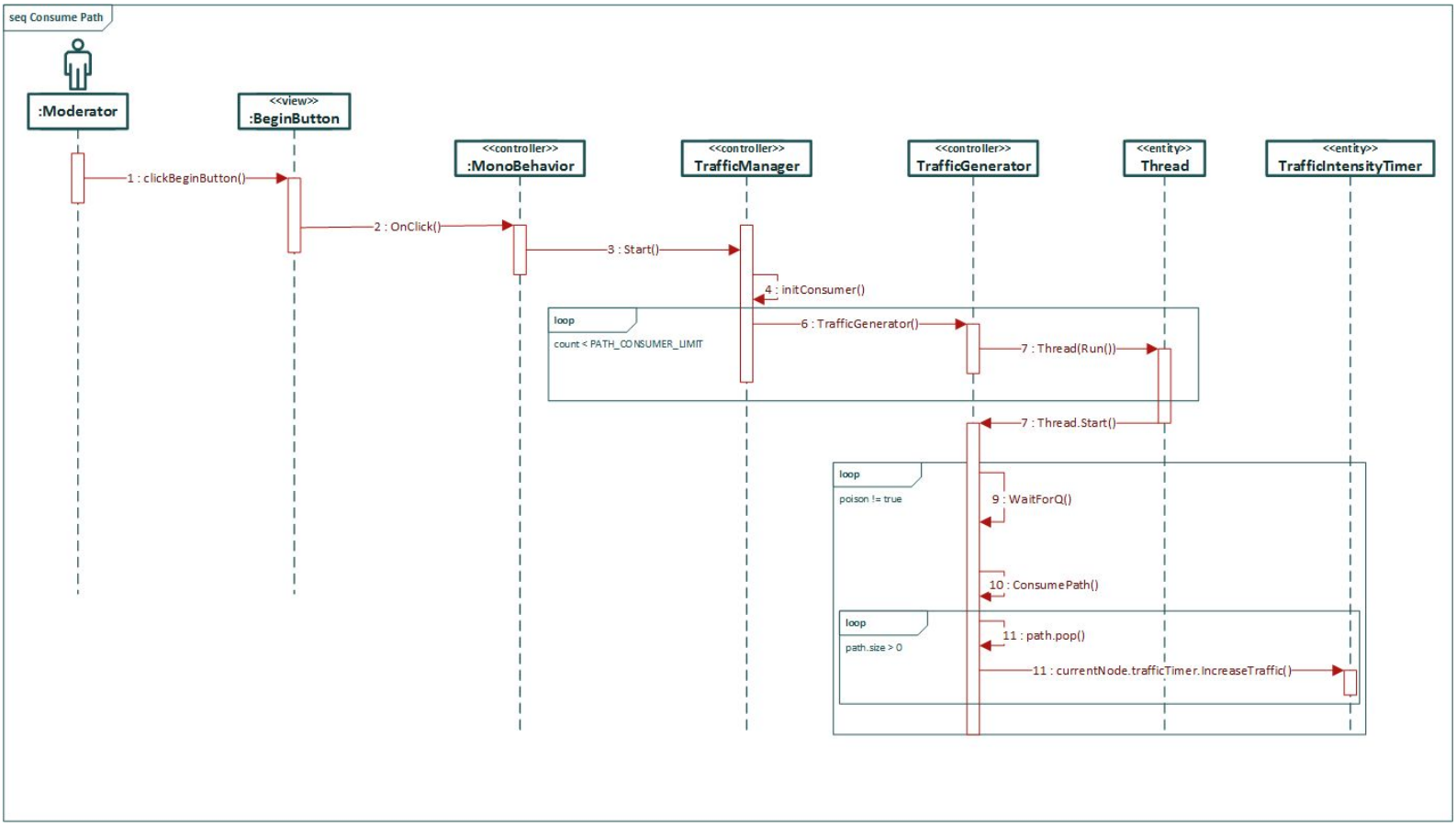


Figure # A.14 - Consume Path

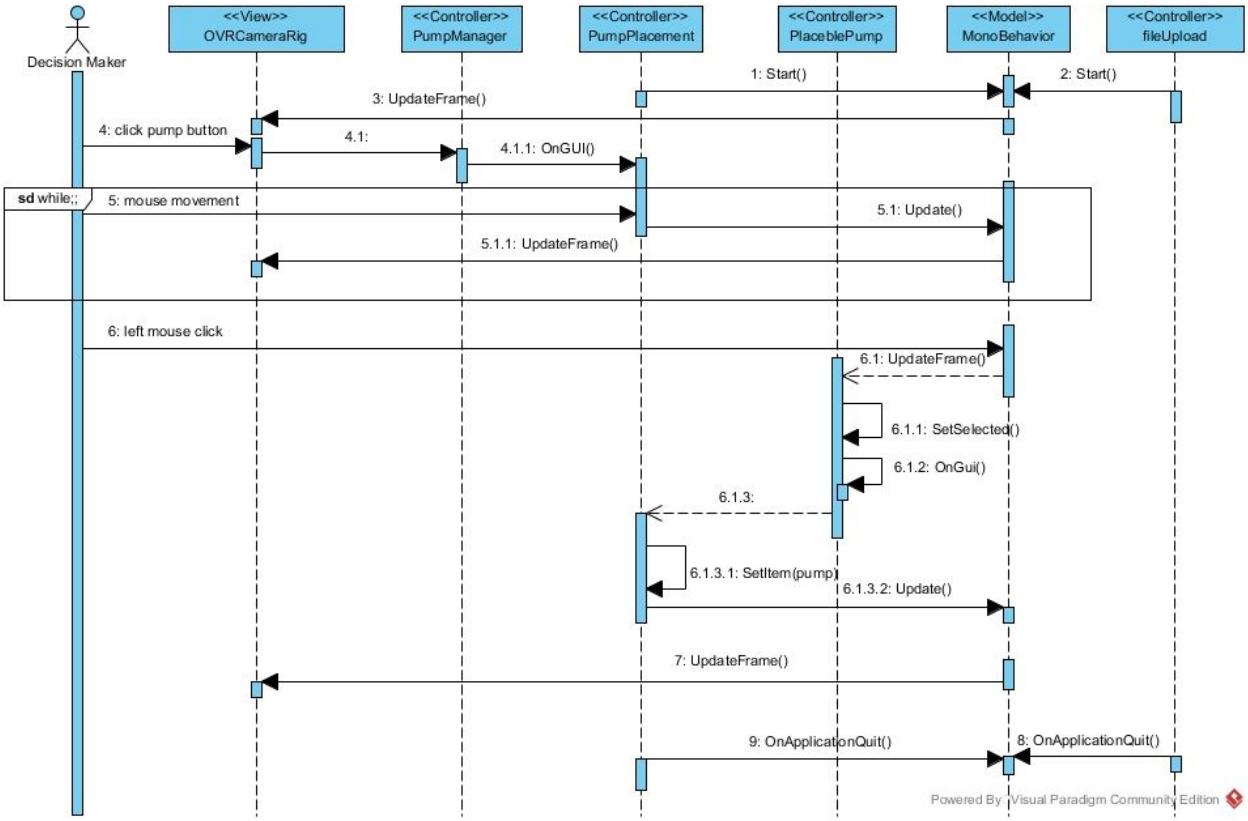


Figure # A.15 - Store Decisions

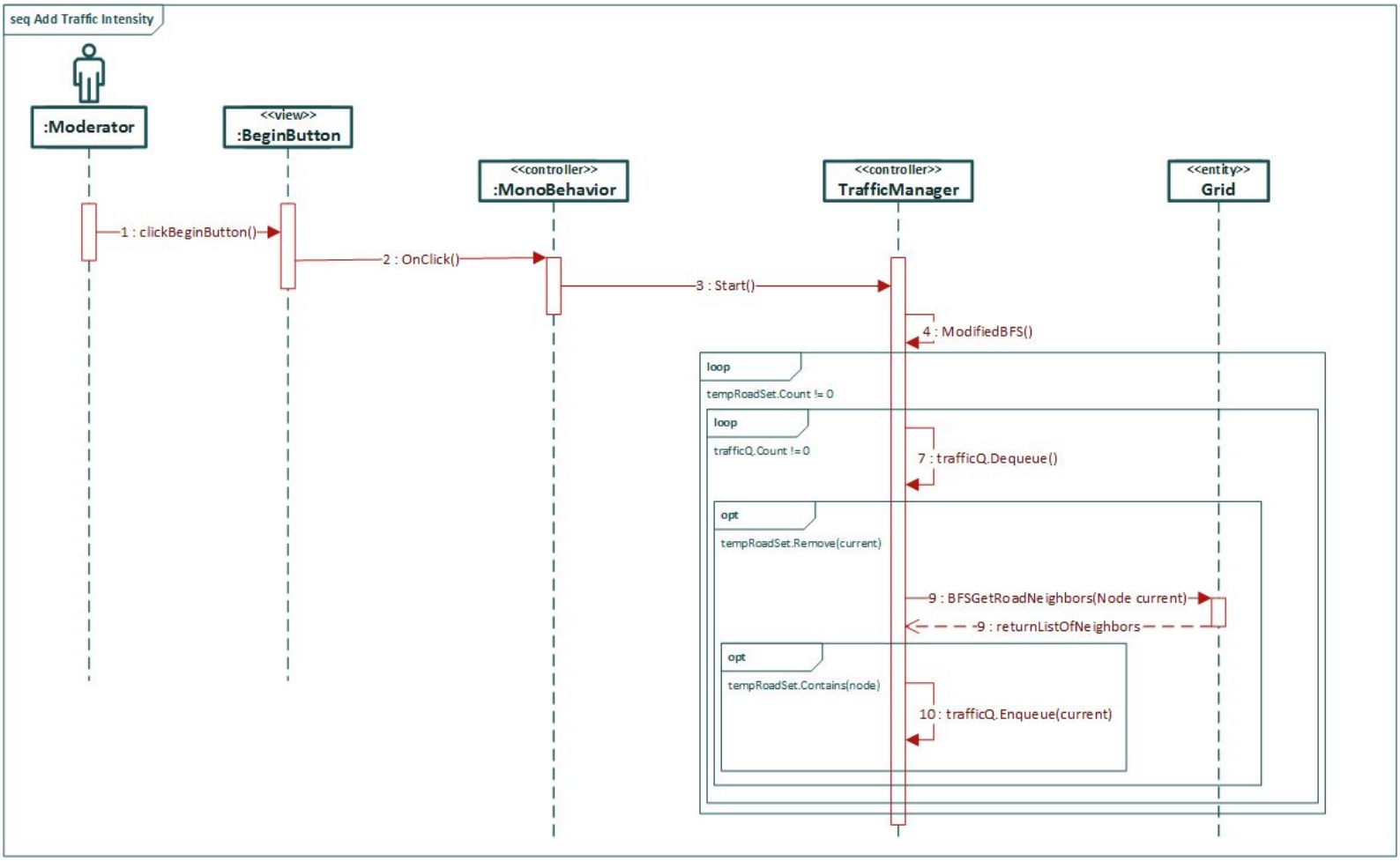


Figure # A.16 - Add Traffic Intensity

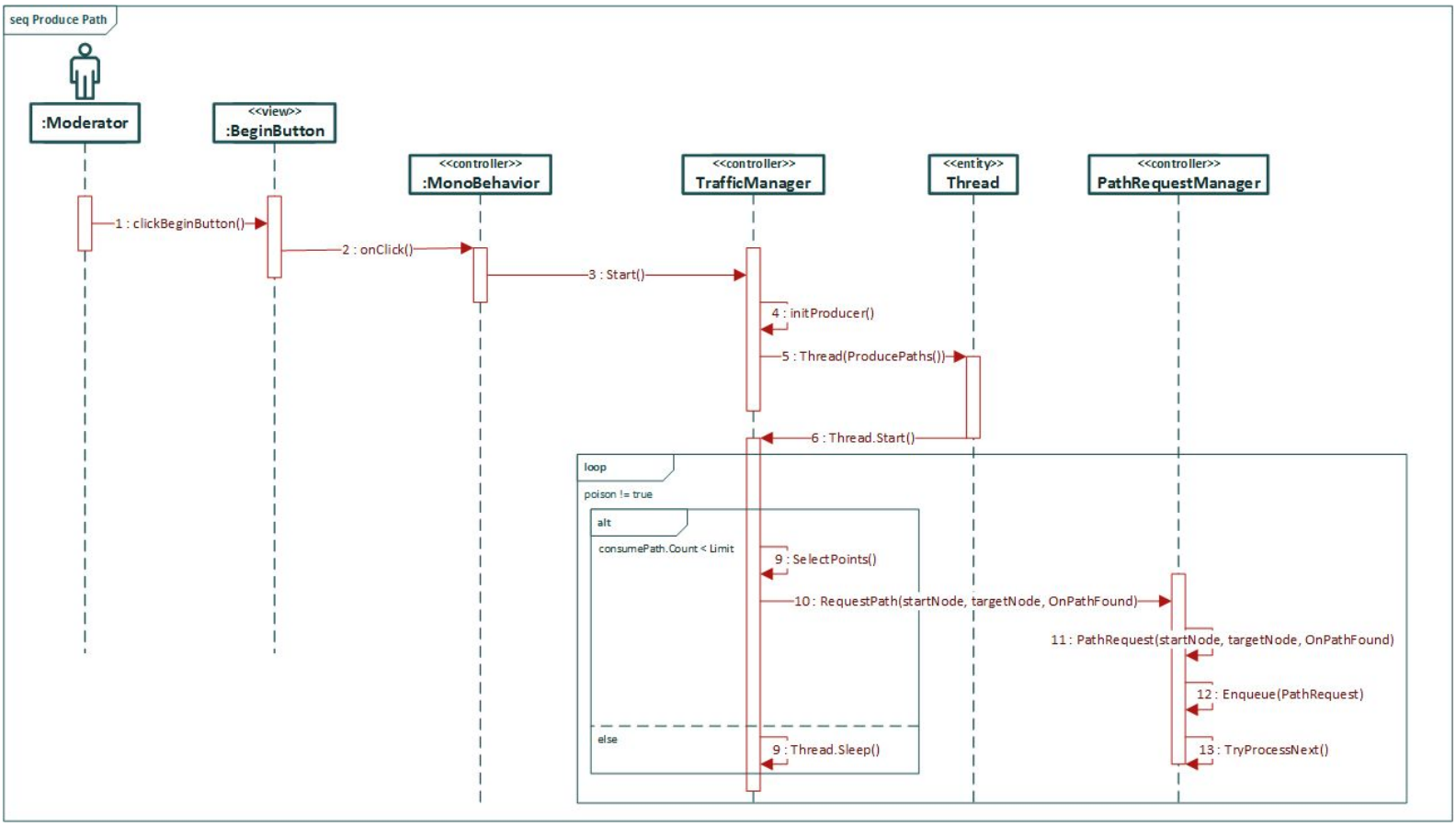


Figure # A.17 - Produce Path

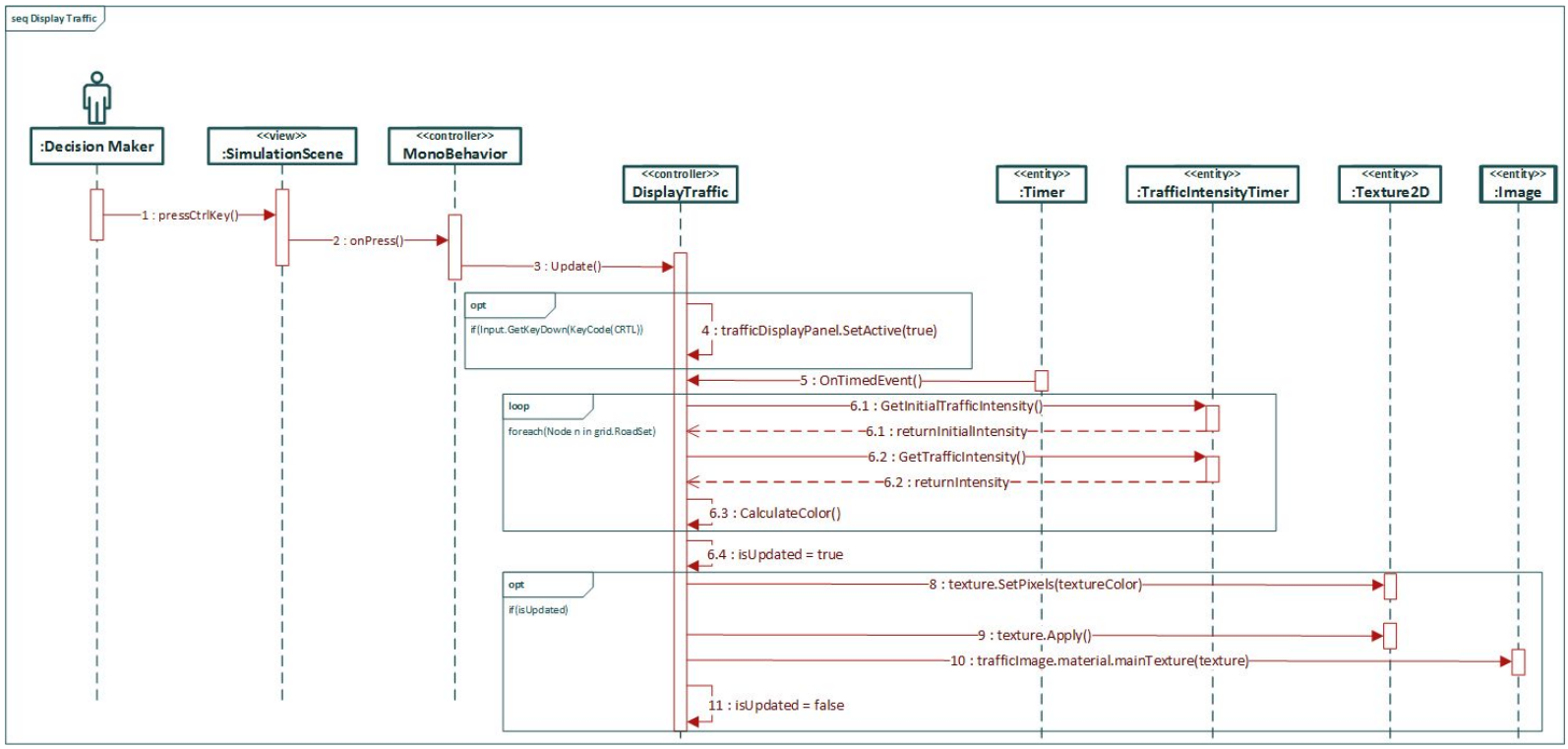


Figure # A.18 - Display Traffic

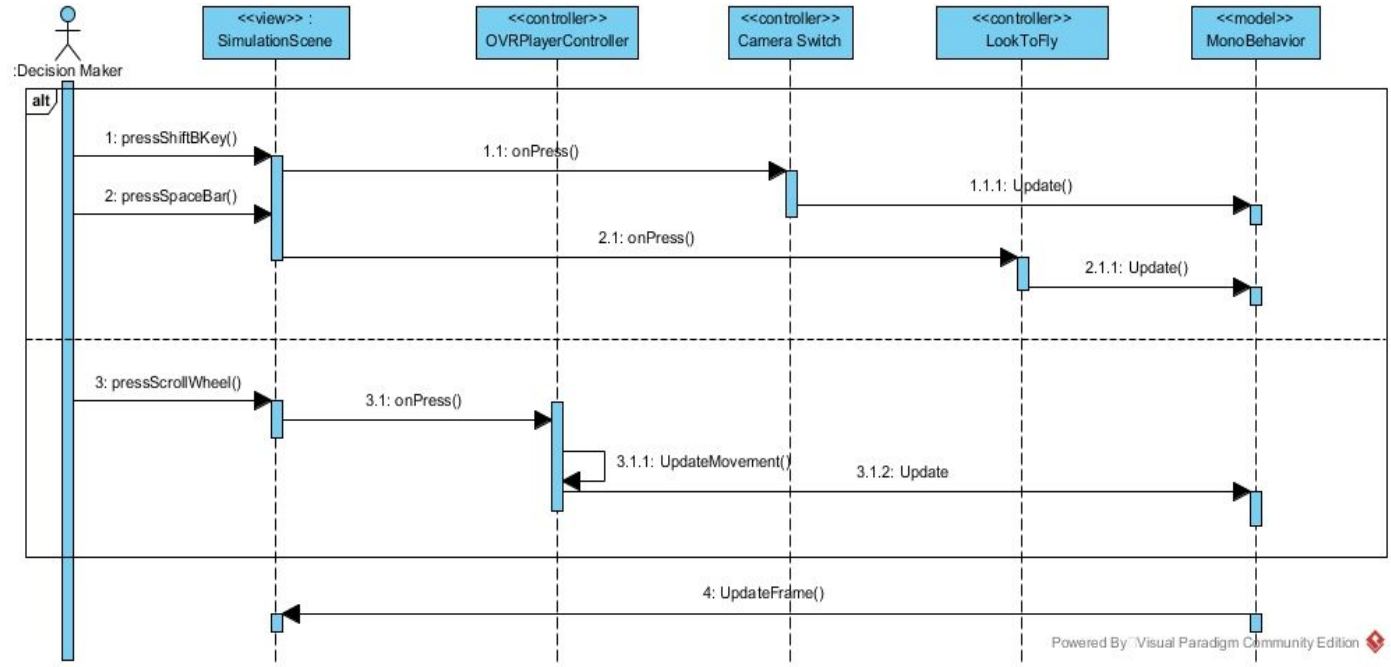


Figure # A.19 - Create View Perspectives

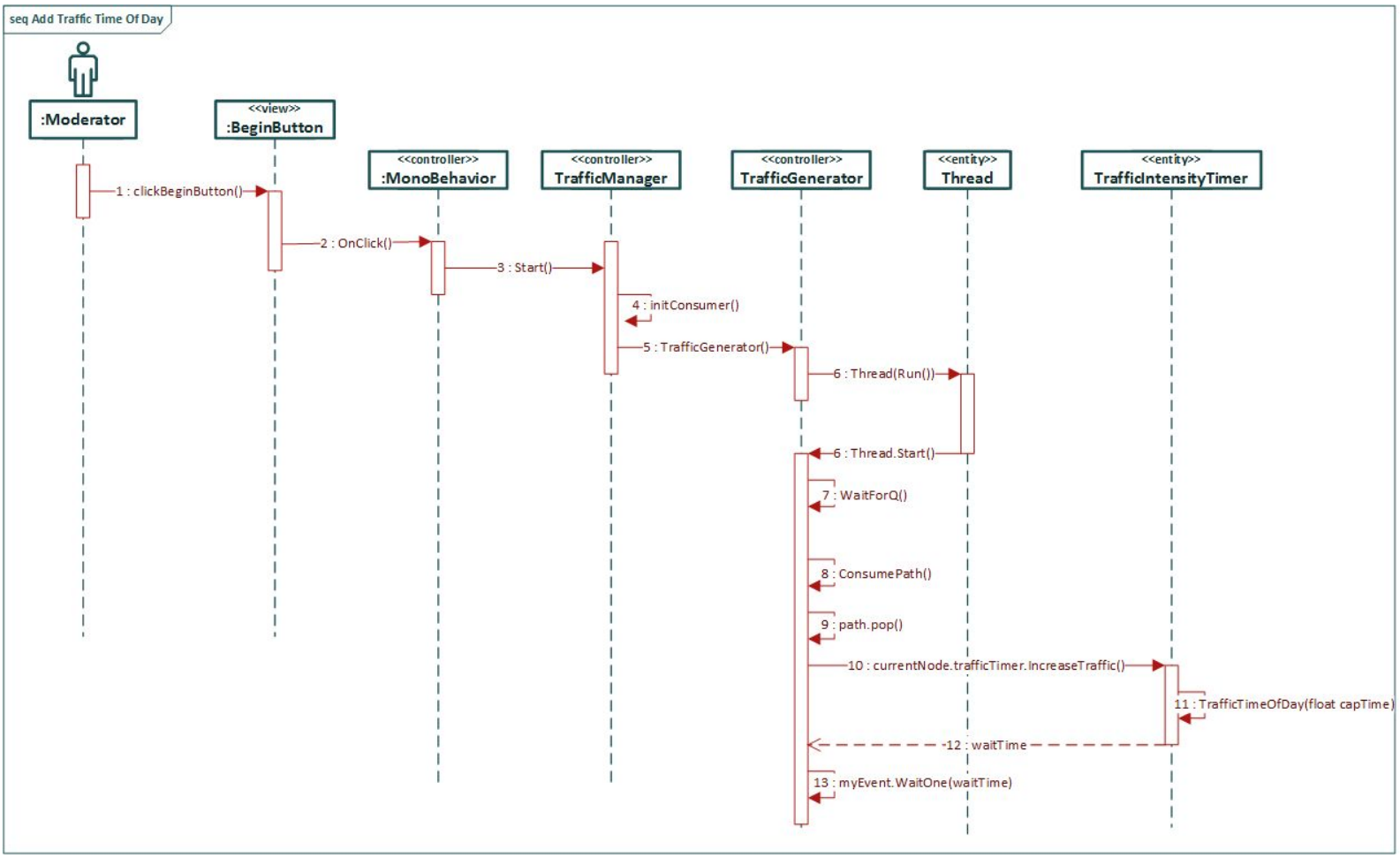


Figure # A.20 - Add Traffic Time of Day

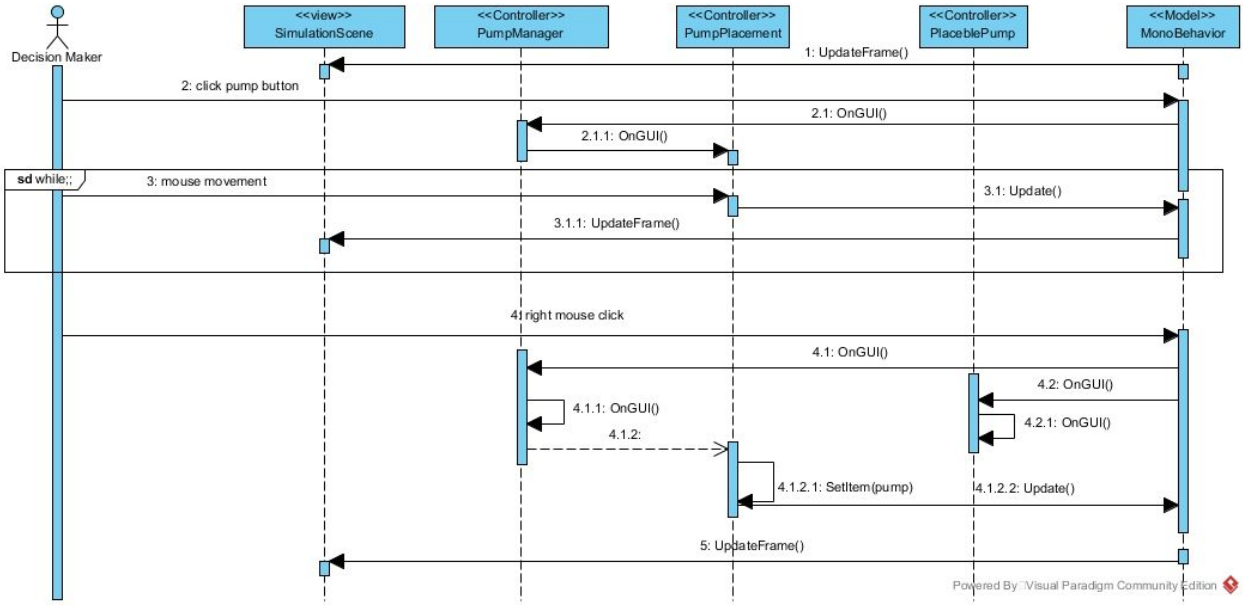


Figure # A.21 - Create Budget-Pump Transactions

Appendix B - User Interface Design

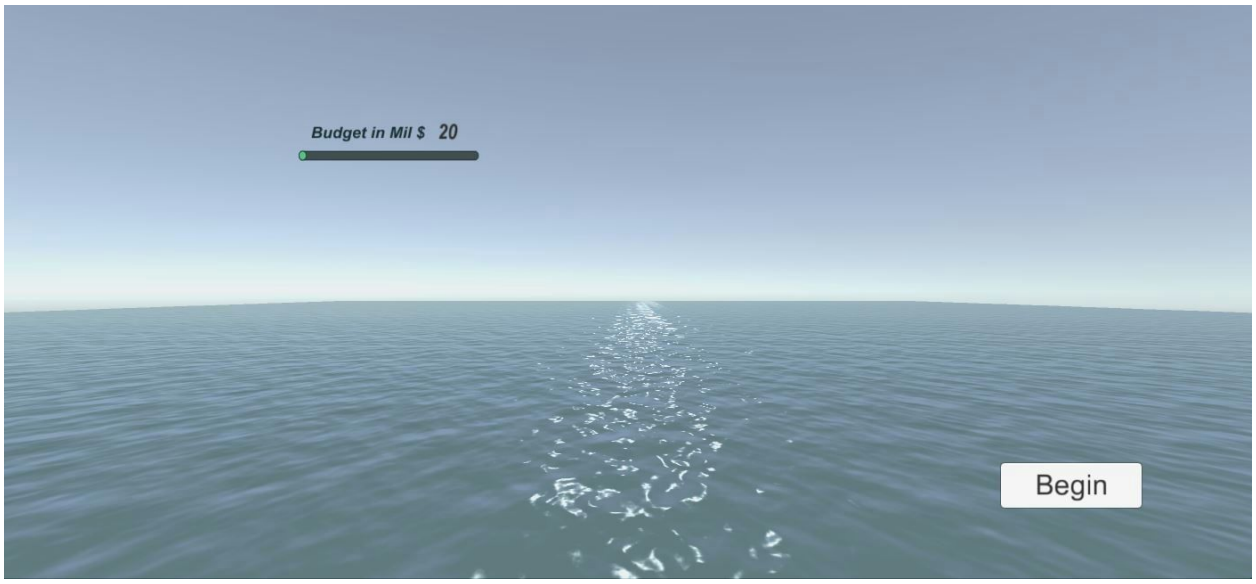


Figure # B.1 - Start Menu

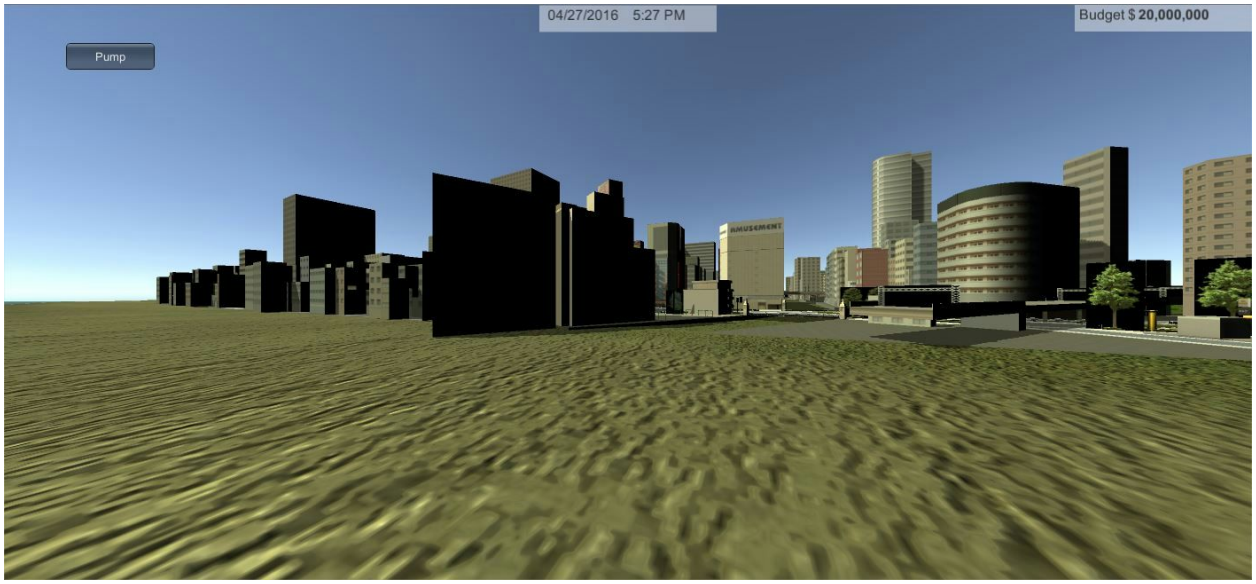


Figure # B.2 - Simulation Scene

Figure # B.3 - Traffic Visual

Figure # B.4 - Top Perspective (“Shift+B”)

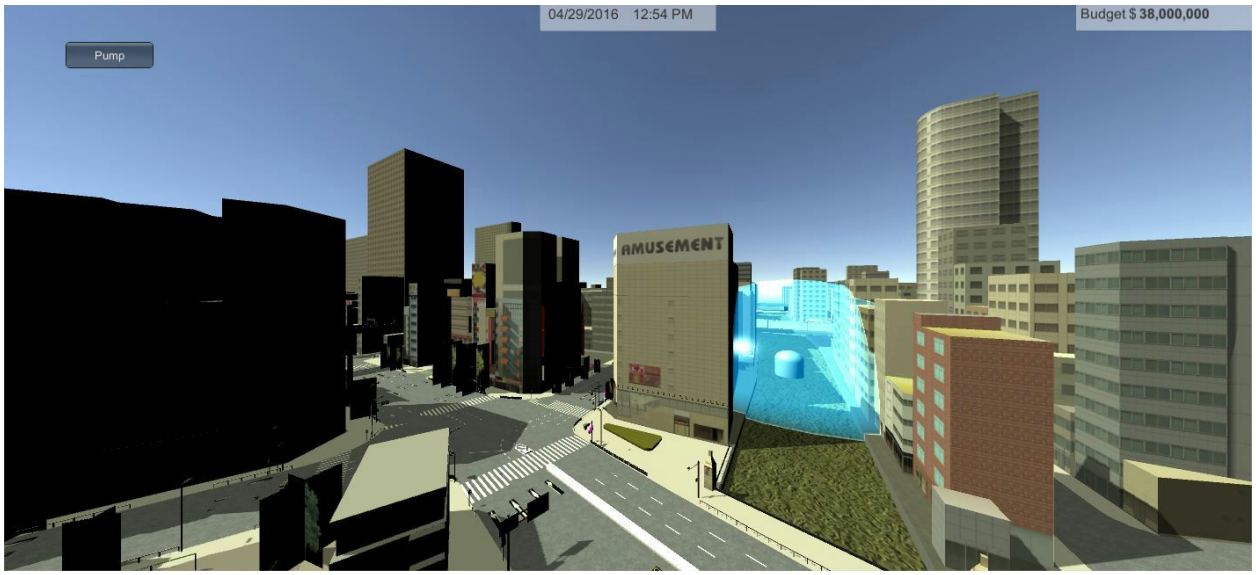


Figure # B.5 - Placing a Pump into Scene

Appendix C - Sprint Review Reports

Sprint 1 Report

Date: 1/29/2016
Attendees: Olena Tkachenko, Renan Santana, Ali Mostafavi, and Peeraya Inyim
Discussed Topics:
Setup Unity 5.3 and research the Unity manual to understand the pipeline. Implemented the Start menu scene and the budget slider. Both were completed by the end of the sprint. The budget slider had some minor problems, because it was a text and not a primitive data type such as a float. The Start menu is the first scene in the project that will house many functions just like a view.

Sprint 2 Report

Date: 2/12/2016
Attendees: Olena Tkachenko, Renan Santana, Ali Mostafavi, and Peeraya Inyim
Discussed Topics:
Discussed the dimensions / scale and the particular area of which the Miami model will be used. A height map from terrain.party was used to accurately depict Miami as a model. The next scene was implemented and called the SimulationScene. This scene had many conflicts, meaning how to break components in the scene into actual user stories. Thus the Begin Button had to incorporate many components into the scene but had no functionality to them. The navigation of the player was implemented. This report had many assets that were imported in the project. No other issues were encountered in this sprint.

Sprint 3 Report

Date: 2/26/2016
Attendees: Olena Tkachenko, Renan Santana, Ali Mostafavi, and Peeraya Inyim
Discussed Topics:
Created the ability to use the Oculus Rift as a navigation piece. The Display Budget and Display Game Time had to be moved to Sprint 3 due to documentation issues. The implementation of Display Budget and Display Game Time were completed before the sprint.

Sprint 4 Report

Date: 3/11/2016
Attendees: Olena Tkachenko, Renan Santana, Ali Mostafavi, and Peeraya Inyim
Discussed Topics:
Updated the documentation of the two user stories mentioned in the previous sprint report. Created the functionality of the sea level to be able to rise and fall with the accordance of the Game Time. Created the ability to place a water pump removal system. There are still issues with the appearance of the pump and the apparent pump-to-player-collider issue. The pump should also snap to the ground; instead it goes through the ground. The pump also masks the seawater behind the object, it should only hide the seawater that is in the radius of the pump.

Sprint 5 Report

Date: 4/1/2016
Attendees: Olena Tkachenko, Renan Santana, Ali Mostafavi, and Juan Sotomayor Paez
Discussed Topics:
Divided the city into a grid, which contains nodes that represents a road and/or a city sector. Created a script that adds weight to the road node to symbolize the traffic intensity the particular node can have. Created script that utilizes the grid to produce a path, so that the traffic can be simulated. Created a consumer of the path, which modifies the road node traffic intensity. Setup the server and created the connection to the database. Along with the model that the database will have. Created the script that will gather the data that will be stored on the database.

Sprint 6 Report

Date: 4/15/2016
Attendees: Olena Tkachenko, Renan Santana, Ali Mostafavi, and Juan Sotomayor Paez
Discussed Topics:
Added the ability to view the traffic as a HUD. This feature was accomplished by using a single image and updating the texture of the image periodically. Added the ability to have different perspectives in the game. This feature includes the ability to view the city from the top and look to fly (the use of a scroll wheel). There were some issues that were persisted from sprint 5, they include the lack of the ability to build the project due to the use of the editor library as a function to display a pop up, the MongoDB connector had the wrong settings to export as a build.

Sprint 7 Report

Date: 4/29/2016
Attendees: Olena Tkachenko, Renan Santana, Ali Mostafavi, and Juan Sotomayor Paez

Discussed Topics:
Added new functionality to the traffic, by having the traffic naturally increase and decrease according to the game time. Added transaction process when placing a pump into the scene. There were some issues with budget. The first was the lack of cohesion from the persistent data that was used to house the budget from the first scene. The second was the lack of formatting when displaying the budget to the HUD. There are issues with getting into contact with the Oculus lab to test our project. The lab contained an old version of an Oculus that wasn't compatible with our source code in Unity.

Appendix D - Sprint Retrospective Reports

Sprint 1 Retrospective

Date: 01 29, 2016
Attendees: Olena Tkachenko, Renan Santana
Discussed Topics:
It went well that Unity is easy to pick-up and Unity asset store provides functionality from the community. Not so well learning to use Unity Integration Testing Suite and learning to fully document code integration with Unity. It was hard to start planning structured building blocks of Urban Decision Theater when the application itself was then a very fluid concept-not clearly defined.

Sprint 2 Retrospective

Date: 02 12, 2016
Attendees: Olena Tkachenko, Renan Santana
Discussed Topics:
We need to try to have everyone attend the sprint demo meetings. The user stories were approved and a model for the city was implemented. Need to define clear understanding of the game's capabilities and ask detailed questions of the product owners vision of the UDT application.

Sprint 3 Retrospective

Date: 02 26, 2016
Attendees: Olena Tkachenko, Renan Santana
Discussed Topics:
Product owner was pleased with Oculus additions, FPC, Tides, Game Time, and Budget features. Need to improve water transparency in our city, find time to test Oculus (lab is usually not open).

Sprint 4 Retrospective

Date: 03 11, 2016
Attendees: Olena Tkachenko, Renan Santana
Discussed Topics:
Product owner was pleased with first implementation of Pump and City Sectors. Need to brainstorm, prioritize and divide user stories for Urban Theatre. Team members were on time with completions of their user stories.

Sprint 5 Retrospective

Date: 04 01, 2016
Attendees: Olena Tkachenko, Renan Santana
Discussed Topics:
User Stories 763, 768, 778, and 782 implementation was what the product owner wanted. Divide each user story timewise for better time management. Brainstorm ideas for product improvement.

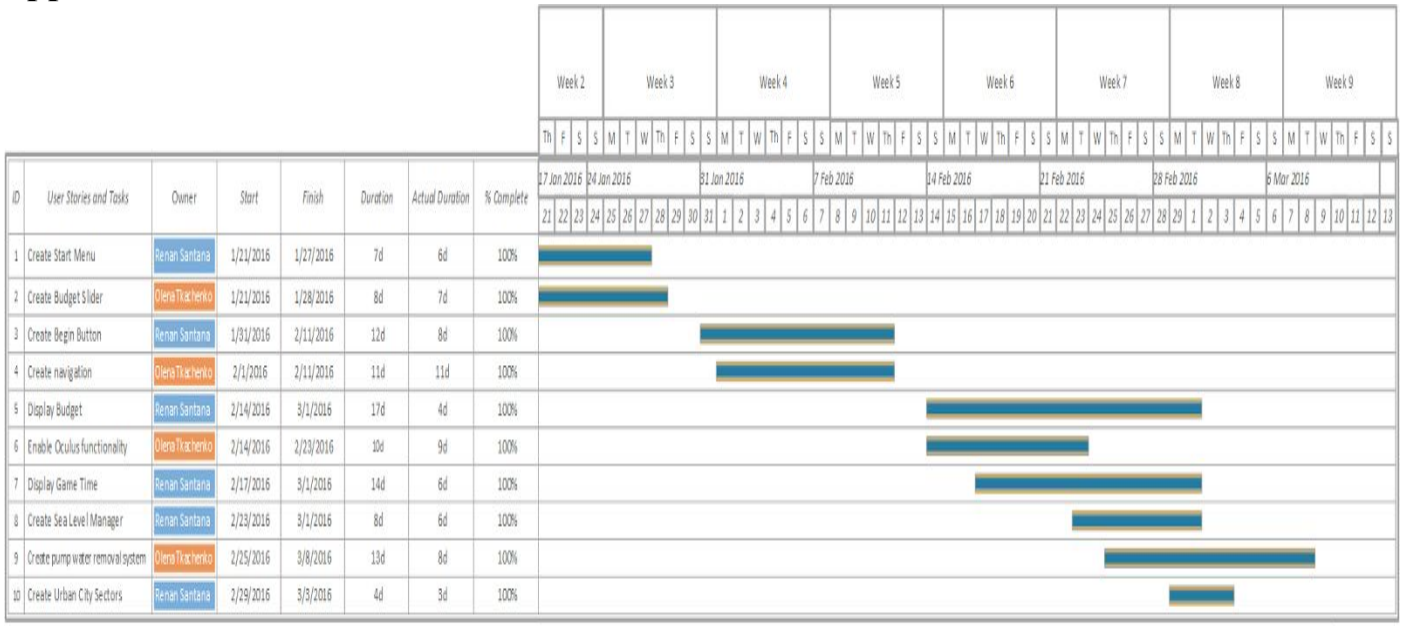
Sprint 6 Retrospective

Date: 04 15, 2016
Attendees: Olena Tkachenko, Renan Santana
Discussed Topics:
It was good that Completed the user stories within the sprint time. We need more sprints to finalize the project, incorporate all the components. Cannot resolve the build errors of the project. Need to make a demo sample of our project for presentation.

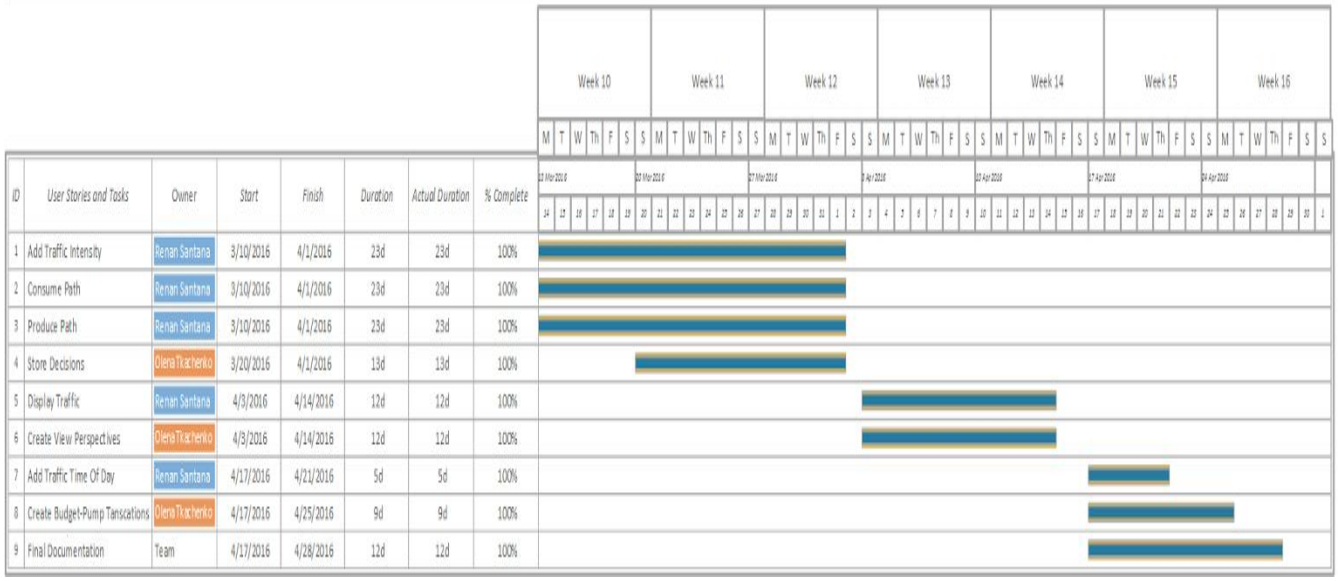
Sprint 7 Retrospective

Date: 04 29, 2016
Attendees: Olena Tkachenko, Renan Santana
Discussed Topics:
It was good that: we resolved build bugs to create an executable of our game; have approval of final user stories 799 and 803; have great collaboration on the Final Project Documentation from both team members. We need to collaborate on a good demo video of project.

Appendix E -Gantt Chart



ID	User Stories and Tasks	Owner	Start	Finish	Duration	Actual Duration	% Complete
1	Create Start Menu	Renan Santana	1/21/2016	1/27/2016	7d	6d	100%
2	Create Budget Slider	Olena Tkachenko	1/21/2016	1/28/2016	8d	7d	100%
3	Create Begin Button	Renan Santana	1/31/2016	2/11/2016	12d	8d	100%
4	Create navigation	Olena Tkachenko	2/1/2016	2/11/2016	11d	11d	100%
5	Display Budget	Renan Santana	2/14/2016	3/1/2016	17d	4d	100%
6	Enable Oculus functionality	Olena Tkachenko	2/14/2016	2/23/2016	10d	9d	100%
7	Display Game Time	Renan Santana	2/17/2016	3/1/2016	14d	6d	100%
8	Create Sea Level Manager	Renan Santana	2/23/2016	3/1/2016	8d	6d	100%
9	Create pump water removal system	Olena Tkachenko	2/25/2016	3/8/2016	13d	8d	100%
10	Create Urban City Sectors	Renan Santana	2/29/2016	3/3/2016	4d	3d	100%



ID	User Stories and Tasks	Owner	Start	Finish	Duration	Actual Duration	% Complete
1	Add Traffic Intensity	Renan Santana	3/10/2016	4/1/2016	23d	23d	100%
2	Consume Path	Renan Santana	3/10/2016	4/1/2016	23d	23d	100%
3	Produce Path	Renan Santana	3/10/2016	4/1/2016	23d	23d	100%
4	Store Decisions	Olena Tkachenko	3/20/2016	4/1/2016	13d	13d	100%
5	Display Traffic	Renan Santana	4/3/2016	4/14/2016	12d	12d	100%
6	Create View Perspectives	Olena Tkachenko	4/3/2016	4/14/2016	12d	12d	100%
7	Add Traffic Time Of Day	Renan Santana	4/17/2016	4/21/2016	5d	5d	100%
8	Create Budget-Pump Tanscations	Olena Tkachenko	4/17/2016	4/25/2016	9d	9d	100%
9	Final Documentation	Team	4/17/2016	4/28/2016	12d	12d	100%

Figure # E.1 - Gantt Chart

REFERENCES

From the Unity Asset Store

- AQUAS-Lite : Publisher - Dogmatic. (used to model sea water)
    - https://www.assetstore.unity3d.com/en/#!/content/53519
- Unity Test Tools : Publisher - Unity Technologies (used as a tool to do unit tests and integration tests)
    - https://www.assetstore.unity3d.com/en/#!/content/13802
- Japanese Otaku City : Publisher - Zerin Co. (used as the urban city)
    - https://www.assetstore.unity3d.com/en/#!/content/20359

Download and usage of Oculus Plugins from: https://developer.oculus.com/downloads/
Oculus Utilities for Unity 5:
https://developer.oculus.com/downloads/game-engines/1.3.2/Oculus_Utilities_for_Unity_5/