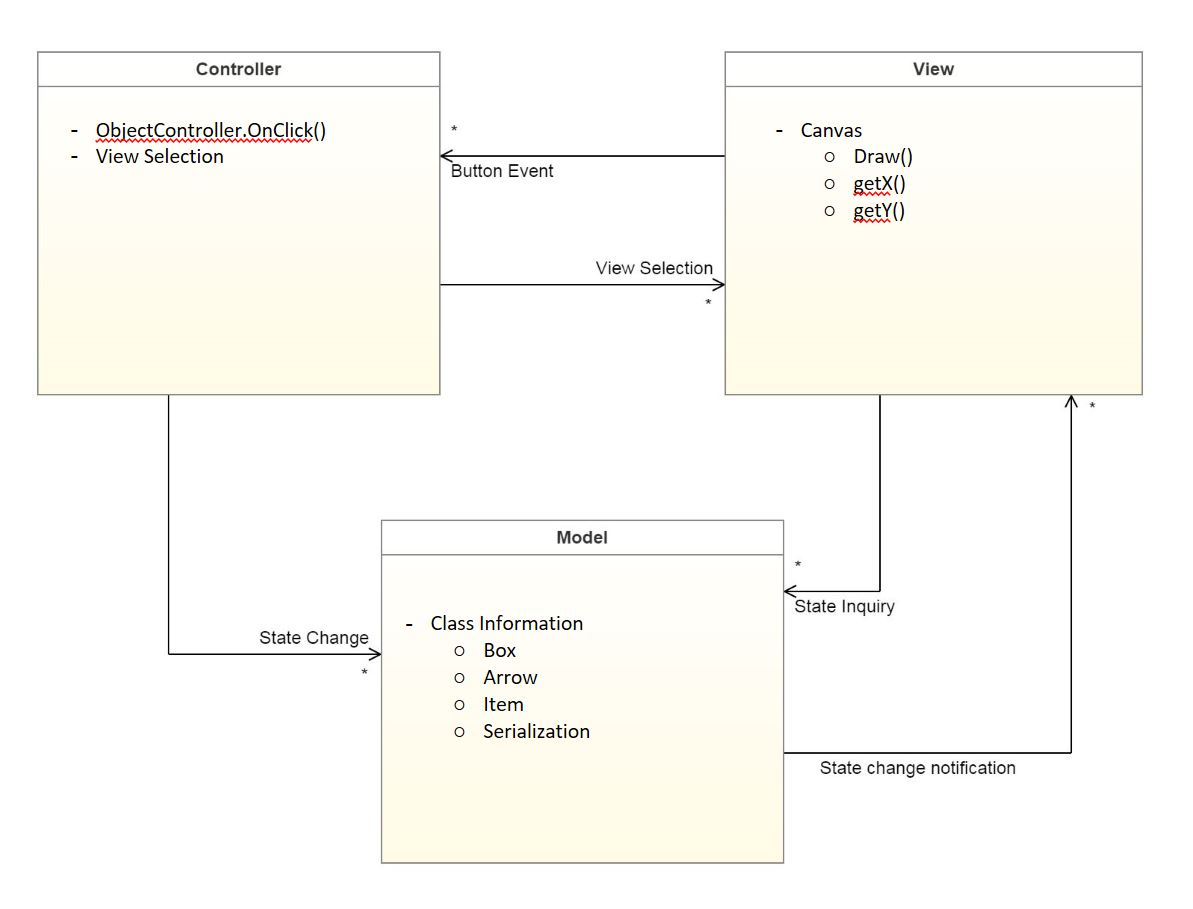
|  |
| --- |
| Pick-A-Path |
| Design Document |
| Olivia Langley, Lucia Ristea, Logan Murphy, James Eprenbeck, Pranaya Kalidindi |

**Context**

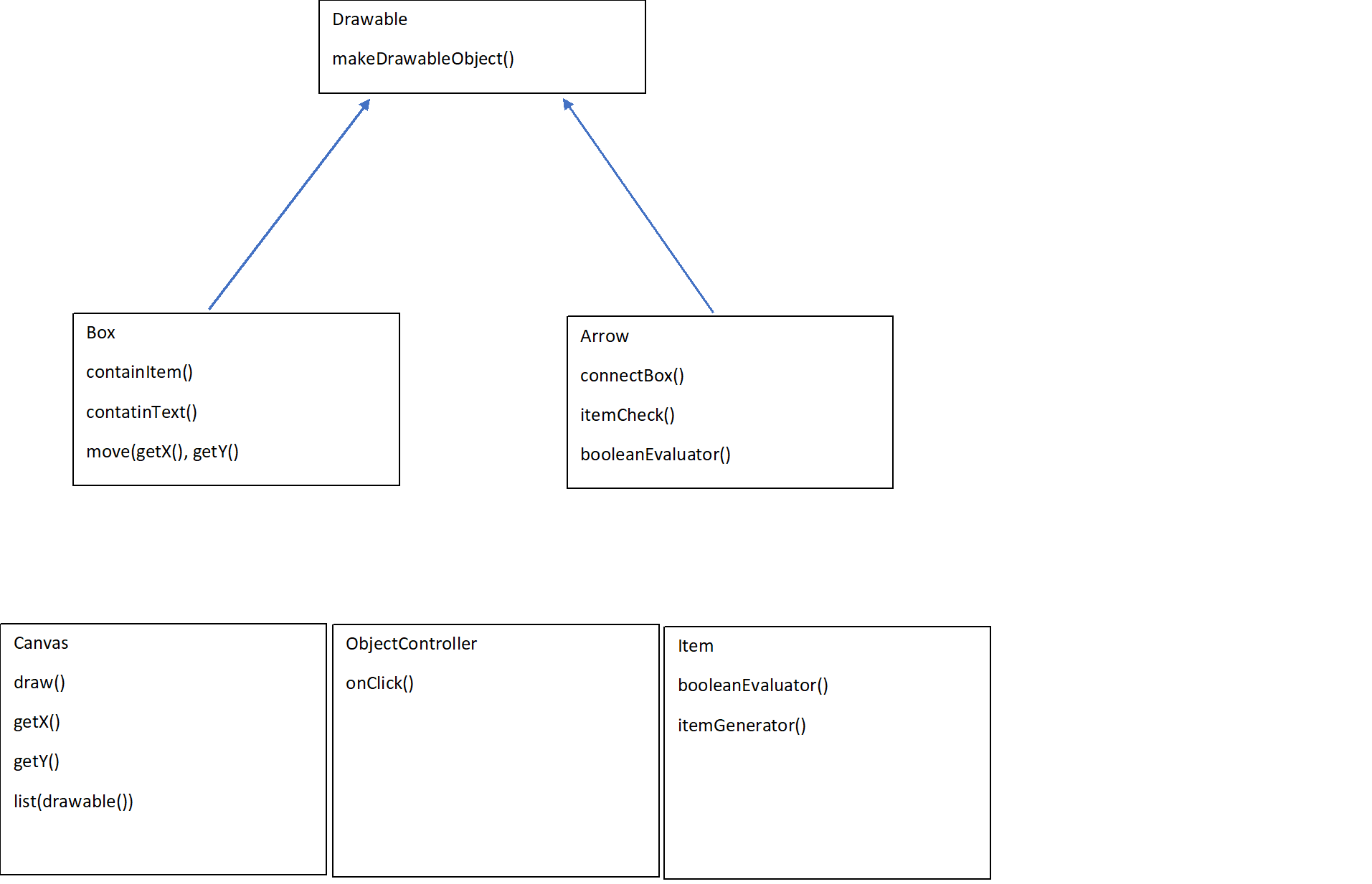
There are a lot of game development tools in the market today that help users create games in a code-editor environment. Pick-A-Path is a text-based game development tool that provides an interface to develop games without needing any prior knowledge of programming. Pick-A-Path is a platform that can be used by beginners or professionals to create simple text-based games in an interface that is easy to use. The program contains two modes: Editor Mode and Player Mode. The Editor Mode allows the user to build a new game or edit an existing one, while the Player Mode allows the user to access the games that are already built using this software. The games that can be built using this software usually start with a situation in a story. There are options provided to the user regarding the path of the storyline they want to commit to. Choosing options in this manner will progress the game further until the end.

Pick-A-Path is a Java-based program and is compatible with any platform that supports Java. To create the game, the user will be provided with unlimited boxes and arrows. The user should fill the box with the scenario and provide options for the player to choose from. According to the option chosen, another box would be created which will contain more options that are generated by the user. An arrow is used to connect to these two boxes so that the player can move from the previous scenario into the next scenario. In this manner, the user can create a game that contains thousands of boxes, and access it later in the player mode to play it.

**System Architecture**

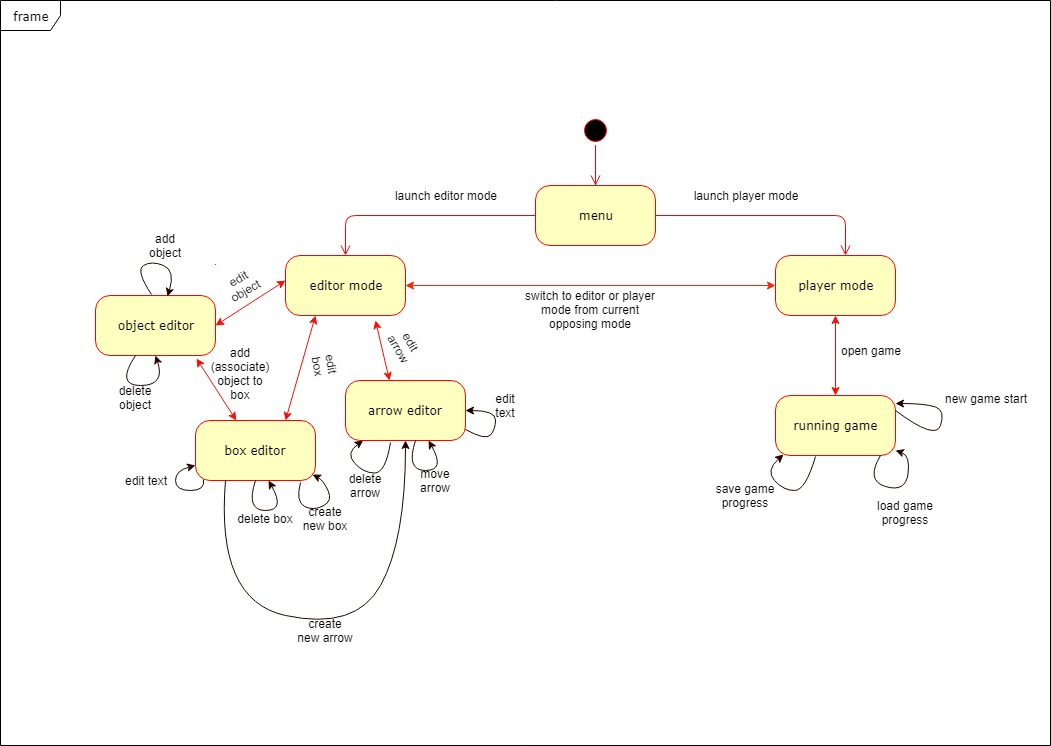


**Class Diagram**

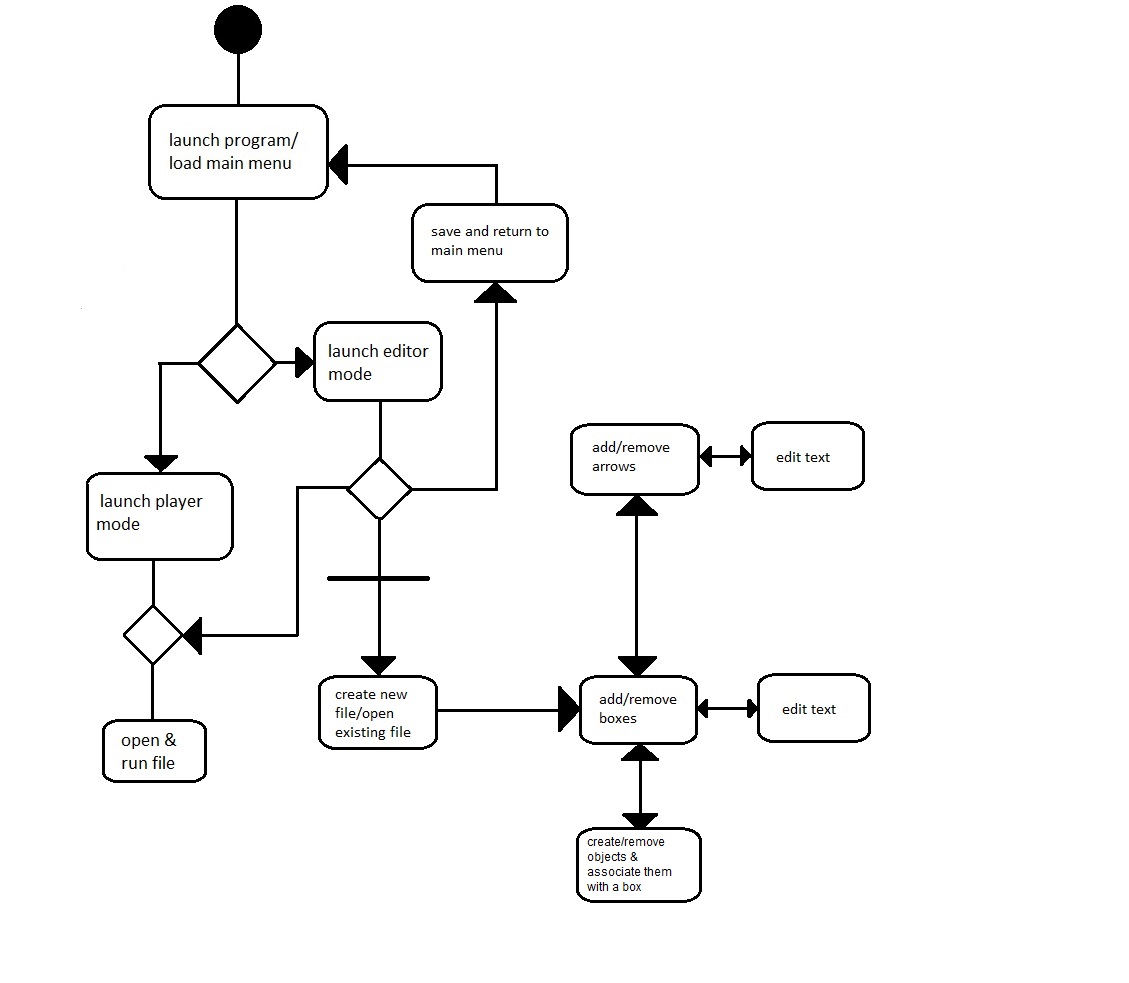


**Design Models**

State Diagram:



Activity Diagram:



**Timeline**

**September 14th, 2018 (Fri):**

- Discuss and finalize the architecture pattern for the project

- Initialize building the GUI for the project.

**September 21st,2018 (Fri):**

- Create the architectural pattern diagram and the state diagram of the software.

- Continue building the GUI.

- Discuss the classes that need to be involved to create the program.

- Complete the Context and the Timeline for Project 2

**September 28th,2018 (Fri):**

- Complete the left-over details of the architectural and the state design diagrams.

- Discuss the classes included in the program and how to translate them into a class diagram.

- Build the GUI until the boxes can be moved around, for project 2 prototype.

**October 3rd, 2018 (Wed):**

- Complete the class diagram

- Complete the prototype as much as possible.

**October 5th, 2018 (Fri):**

- Complete putting the entire document together.

- Finalize the prototype and make it ready for submission.

- Straighten out any final details of the project.

- Submit the Project 2.

**October 12th, 2018(Fri):**

- Work on implementing the classes of the software

- Continue the functionality of boxes and arrows in the software.

- Start trying to figure out how to make the boxes and arrows interact.

- All of us submit the Assignment 2 to blackboard, individually

**October 19th, 2018(Fri):**

- Boxes and Arrows need to be functional.

- Continue implementing the classes in the software.

- Work on the GUI design details.

**October 26th, 2018(Fri):**

- Complete the functionality of the entire GUI of the software.

- Make the program run without glitches

**October 30th, 2018(Wed):**

- Make sure that all the final details of the GUI are accounted for.

- Have the final version of the GUI running successfully.

- Make sure all the classes are accounted for.

**November 2ndth, 2018(Fri):**

- Final Checking of Project 3

- Submit Project 3

**Prototype**

Our prototype is located on GitHub under our repository:

COMP3100-fall2018-2/pickapath/Prototype.java

Context: 2/5

The context of your system is not explained, and you don’t provide a diagram.

System Architecture: 13/15

MVC is a good choice for your system architecture, but you should explain more.

Class Diagram: 18/20

Your class diagram was helpful. Where is the view class that holds everything? What uses a design pattern?

Design Models: 11/15

Your state diagram gives useful information about how your program will transition through different states. However, your activity diagram duplicates the same information. Furthermore, that information doesn’t make sense in an activity diagram because it’s not as if your program will be a set of decisions.

Prototype: 29/30

Your prototype is a good start. It’s attractive and looks very much like the final product will likely look. The box creation feature is nice, but the ability to drag boxes around or put text in them would have been useful.

Timeline: 4/5

The goals in your timeline should have been more specific. “Complete the functionality of the entire GUI of the software” was a goal. Isn’t that the entire program?

Spelling, Grammar, and Style: 8/10

Your prototype had reasonable comments. More text explaining your program would have been useful in your design document. It’s a couple of paragraphs followed by diagrams.

Total: 85