Defense Of Staley

Bill Mak
Logan Williams
Angelo Scattini
Alec James
Juan-Carlos Ferrel

Overview

4 Parts:

1. Introduction to Defense of Staley
    a. Game Overview
    b. Gameplay
2. Why we're Developing Defense of Staley
    a. Compared to other games
3. Design Overview
    a. Activity, State, and Sequence Diagrams
    b. Class Diagram
    c. Design Patterns
4. Current Implementation
    a. Testing & Sonarqube results
    b. Functionality Demo
    c. Analytics

Introduction

- TD game
- Based on CPE 357 as taught by Clint Staley
- Student must defend their grade against Staley's attacking assignments
  - Done by purchasing towers based on studying, sleeping, drinking coffee, etc.
- 10 round survival
  - If the student lasts 10 rounds without the grade dropping to F, they win
  - If the grade drops to F before then, they lose

Why we're developing Defense of Staley

- Another similar game: Bloons TD4
  - Easy to learn
  - Entertaining
  - On many popular flash game sites

Why we’re developing Defense of Staley

- Why our System is better
    - [intended] humor
    - Two-player mode
    - A higher difficulty
    - Gameplay that ends
    - No paid DLC
    - And…
        - Easy to learn (it’s a TD game!)
        - Entertaining
        - Can be on flash game sites (once this is implemented)
        - Also can be on your system (easy access, no net required!)

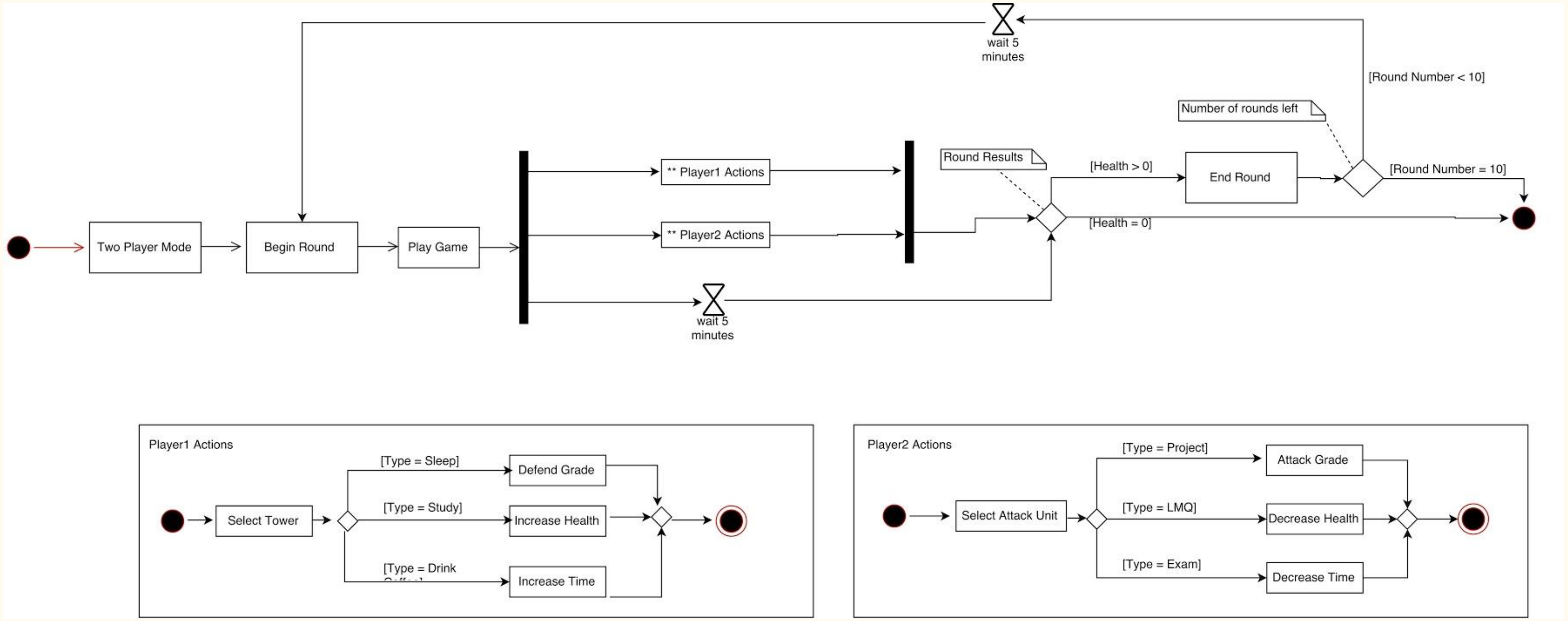
Let’s see some Diagrams!

- Why?
    - Organization
    - Designing aid - figure out how we’re going to implement items

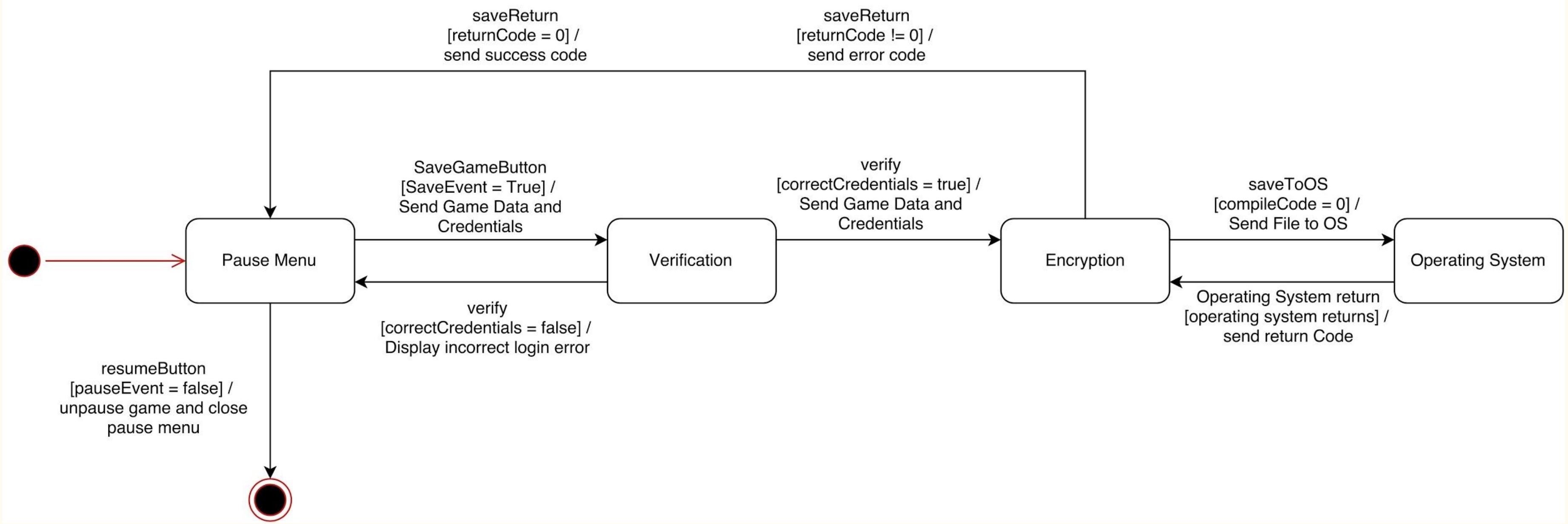
Class Diagram

Refer to reference (it’s big).

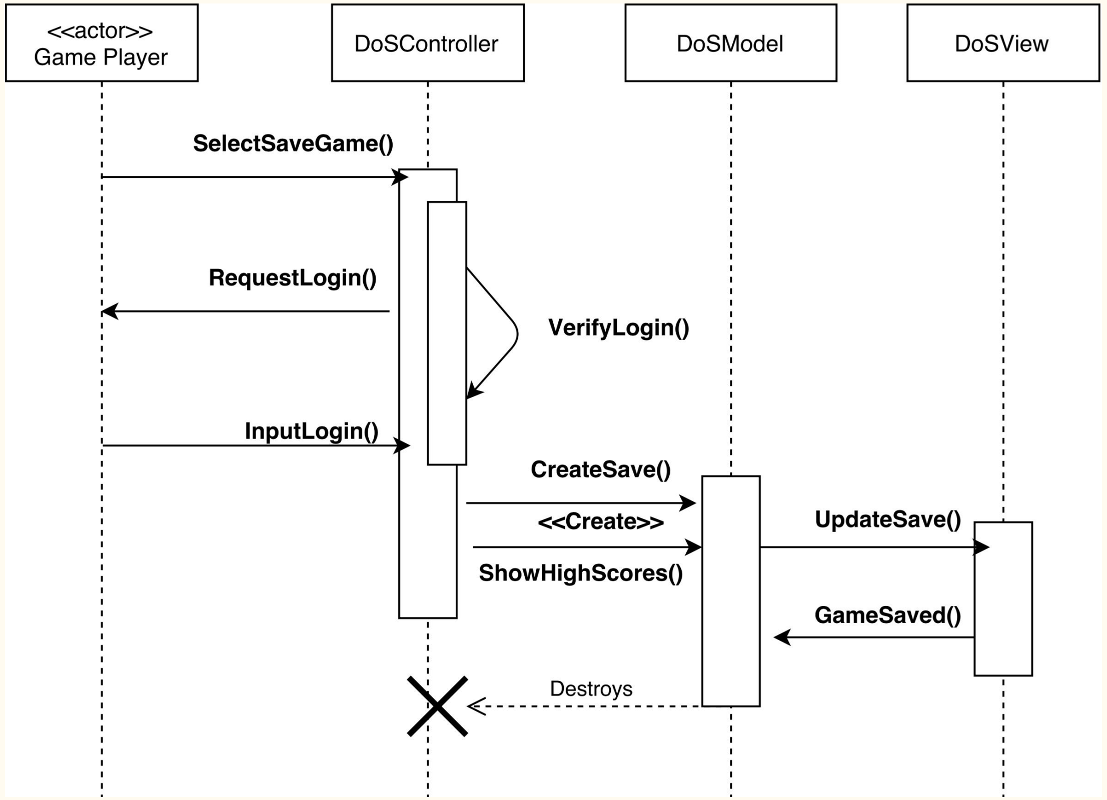
Activity Diagrams



State Diagrams


saveReturn [returnCode = 0] / send success code
saveReturn [returnCode != 0] / send error code
SaveGameButton [SaveEvent = True] / Send Game Data and Credentials
verify [correctCredentials = true] / Send Game Data and Credentials
saveToOS [compileCode = 0] / Send File to OS
Pause Menu
Verification
Encryption
Operating System
verify [correctCredentials = false] / Display incorrect login error
Operating System return [operating system returns] / send return Code
resumeButton [pauseEvent = false] / unpause game and close pause menu

Sequence Diagrams



Design Patterns Present

- Entity Control Boundary (model, dos)
    - ECB D.P. - Game Logic is separated by a layer of classes
    - Applicable b/c we want to be able to separate the logic from the graphics, making graphics changeable
- Container (GameState, Unit)
    - Container D.P. - the class is intended to hold data
    - Applicable b/c the GameState & Unit are only needed to hold data
        - So that we don't have to manipulate all the individual game variables

Testing Demonstration

1. Unit Tests
    a. Demo
    b. Separate classes are functional
2. Integration Tests
    a. Demo
    b. Classes work with one another
3. Loop Tests
    a. Demo
    b. Loop functionality is verified
4. System Tests
    a. See document
    b. (Most) Requirements have been successfully met

Sonarqube Results

1. Demo
2. Why Sonarqube?
    a. Minimize technical debt (ours still has some smoothing out)
    b. In the future, it’s easy for other programmers to implement additional features
        i. 2 player (one player controls Staley)
        ii. TD-style gameplay (at the moment, we have a defense-style gameplay)
        iii. Saving and loading (at the moment, coded but not used)

But Enough about Testing....

Let’s actually see this thing in action!

Analytics Performed

- Show our Analytics

Any Questions?