Lab 2 – PolyMorpher Prototype Specification

Nathaniel DeArce

Old Dominion University

CS411W

Professor Thomas Kennedy

19 March 2018

Version 2

Table of Contents

1 Introduction 3
1.1 Purpose 3
1.2 Scope 6
1.3 Definitions, Acronyms, and Abbreviations 8
1.4 References 11
2 General Description 15
2.1 Prototype Architecture Description 15
2.2 Prototype Functional Description 17
2.3 External Interfaces 17

List of Figures

Figure 1 Current Process Flow 4
Figure 2 Solution Process Flow 5
Figure 3 Major Functional Components 16
Figure 4 API Book Flowchart 18
Figure 5 Compiler Flowchart 19

List of Tables

Table 1 Real-World Product vs Prototype Features 6
Table 2 Key to Table 1 7

Lab 2 – PolyMorpher Prototype Specification

1 Introduction

Programming as a technical skill teaches students to think logically and methodically to develop their problem-solving skills. Introductory computer science (CS) classes can be difficult for students that are new to this way of thinking. PolyMorpher will be a tool for struggling students to learn object-oriented programming (OOP) concepts and problem-solving skills.

1.1 Purpose

PolyMorpher is an educational video game designed to provide players the opportunity to learn object-oriented programming concepts with minimal prior programming experience. OOP will be used to help players understand how to connect their code to the real world.

Programming is intimidating for the uninitiated. As a result, first time Old Dominion University (ODU) programming students drop out or switch majors. Existing tools fail to teach Object-Oriented Programming (OOP) concepts and problem-solving skills. PolyMorpher was born to remedy the student progression dilemma for ODU Computer Science students. Figure 1 and Figure 2 show the current process flow and the proposed solution process flow respectively.

LAB 2 - POLYMORPHER

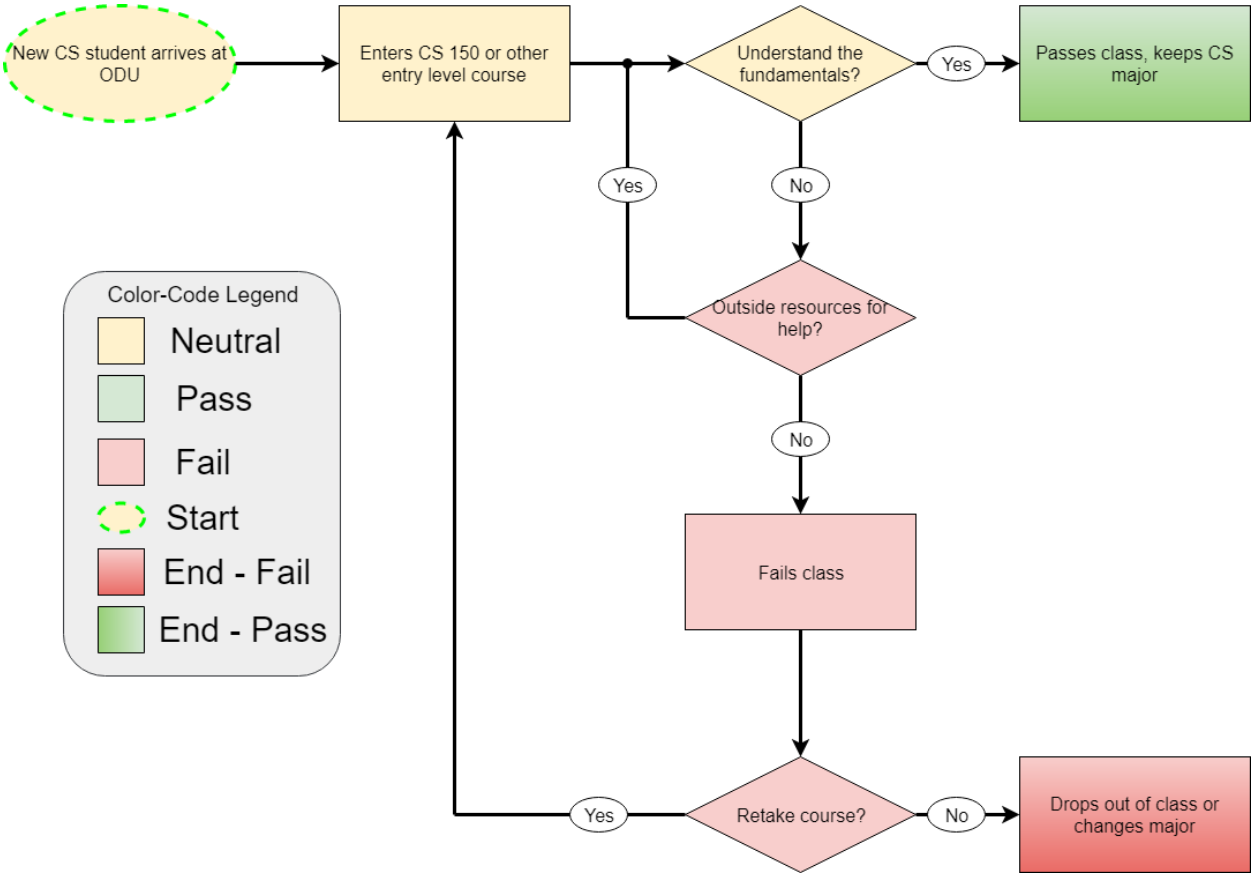

Figure 1 Current Process Flow

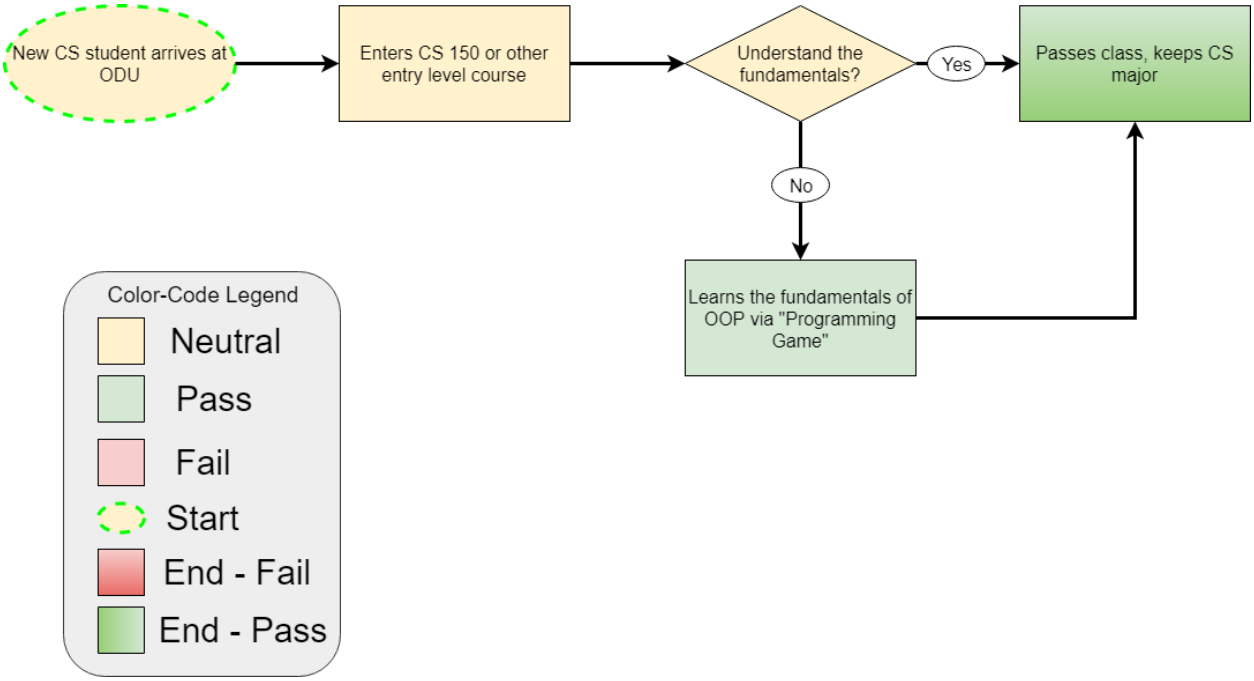

Figure 2 Solution Process Flow

PolyMorpher will target students, teachers, and anyone that wants to have a resource with which to learn object-oriented programming as end users, and Old Dominion University as the customer.

During gameplay, the player will be able to interact with and modify the code of objects. For example, the player may need to move a box that will not budge. The player may then change the box's behavior by modifying its source code to have it respond to the player's actions. This will be the core mechanic of the game. PolyMorpher will support multiple object-oriented programming languages such as C++, C#, and Java. Code written by the player will be compiled in a sandbox environment and the game will update according to the changes.

1.2 Scope

The PolyMorpher prototype will showcase how effectively the real-world product will be able to teach OOP concepts to its players. Not all of the real-world product's functionality will be required to demonstrate PolyMorpher's potential. Table 1 highlights the differences between the prototype and the real-world product. Table 2 is the associated key.

Table 1 Real-World Product vs Prototype Features

Element	Description	Real-World Product	Prototype
Abstraction	The game will provide a section that covers Abstraction		
Developed Story	The game will feature a narrative to drive progression		
Encapsulation	The game will provide a section that covers Encapsulation		
Inheritance	The game will provide a section that covers Inheritance		
Polymorphism	The game will provide a section that covers Polymorphism		
Portable Compiler	A compiler will be provided that can compile player-edited code		
Single Player	The game will feature gameplay for only a single player		
Tutorial Level	Provides players a way to familiarize themselves with game controls and mechanics		
Multiple Languages	The game will support multiple object-oriented programming languages		
Player-Made Content	Players will be able to create their own puzzles or content		
Sandbox Level	Players will have access to all tools they would receive through normal progression		
Multiplayer	Gameplay features aspects that require the cooperation of multiple players		
Web Application	Web-based version of the game running in browser		

Table 2 Key to Table 1

Fully Functional
Partially Functional
Eliminated

The PolyMorpher prototype will have a fully functional C# portable compiler that can compile code written by the player during gameplay. C# is the language used by Unity scripts, so this simplifies development of the prototype to focus on the educational aspects of PolyMorpher. The prototype will be single-player only for reasons discussed in Lab 1 Section 4.2.3. A tutorial will be provided that teaches game mechanics, such as controls and how to modify game objects, and preliminary knowledge of C#, such as control structures and data types.

1.3 Definitions, Acronyms, and Abbreviations

API: Application Program Interface

API Book: An list of functions that the play can reference and use in their code

API Index: The menu where the user can decide rather they see the API of suggested functions for their level or all functions suggested to be used in the entire game

Assets: Game assets include everything that can go into a game, including 3D models, sprites, sound effects, music, code snippets and modules, and even complete projects that can be used by a game engine.

Coding Interface: The in-game IDE where the user writes their scripts

Computer: a programmable electronic device designed to accept data, perform prescribed mathematical and logical operations at high speed, and display the results of these operations

Computer Programming: a process that leads from an original formulation of a computing problem to executable computer programs

Computer Science (CS): the science that deals with the theory and methods of processing information in digital computers, the design of computer hardware and software, and the applications of computers

Design: an outline, sketch, or plan, as of the form and structure of a work of art, an edifice, or a machine to be executed or constructed

Git: version control system for tracking changes in computer files and coordinating work on those files among multiple people

GitLab: web-based git repository manager the includes wiki and issue tracking

Gradle: an open-source build automation system that was designed for multi-project builds

GUI: Graphical User Interface

LAB 2 - POLYMORPHER

JavaScript: a programming language commonly used in web development where the the code is processed by the client's browser

Management Simulator: a way to simulate the management of a game in an organized fashion

MySQL: an open source multi-user database management system

Non-Technical Game: user-friendly gameplay able to be utilized by non-technical users

Non-Technical User: user who lacks formal education or knowledge in computer science, computer programming, object-oriented programming, or problem solving skills

Object-Oriented Programming (OOP): A schematic paradigm for computer programming in which the linear concepts of procedures and tasks are replaced by the concepts of objects and messages

ODU: Abbreviation for Old Dominion University

Platform: an integrated set of packaged and custom applications tied together with middleware

PolyMorpher: a programming game that focuses strictly on teaching OOP and problem solving skills

Problem Solving: the process of finding solutions to difficult or complex issues

Programming Game: a video game which incorporates elements of computer programming into the game, which enables the player to direct otherwise autonomous units within the game to follow commands in a domain-specific programming language

Regression Testing: a type of application testing that determines if modifications to the application have altered the application negatively

Software Development Kit (SDK): a set of software development tools that allows the creation of applications for a certain software package

LAB 2 - POLYMORPHER

Student Involvement: the amount of physical energy students exert and the amount of psychological energy they put into their college experience

Student Progression Dilemma: the problem of CS majors at ODU not advancing through the CS course schedule in order to graduate with a CS degree

TUI: Tangible User Interface

Ubuntu: open-source Linux operating system

Unity: a popular game development platform

User-Friendly: easy to comprehend by non-technical users

Virtual Machines: emulations of computer systems that provide functionalities of physical computers

Web Application: a client-server computer program in which the client (including the user interface and client-side logic) runs in a web browser

Wiki: a website on which users collaboratively modify content and structure directly from the web browser

1.4 References

12 Free Games to Learn Programming. (2016, April 25). In Mybridge. Retrieved from https://medium.mybridge.co/12-free-resources-learn-to-code-while-playing-games-f7333043de11

Batten, C. (Narrator). (2017). CS410 Dungeon Escape Demo (Short Version) [Online video]. Online: YouTube. Retrieved from https://www.youtube.com/watch?v=ynhdd1IKgps

Batten, C. (Narrator). (2017). CS410 Project Dungeon Demo [Online video]. Online: YouTube. Retrieved from https://www.youtube.com/watch?v=ynhdd1IKgps

Batten, C. (2017, November 21). CS410 Tech Demo 2 (Download Source Code). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Batten, C. (2017, November 29). VersionControlFlow. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%221IQj6SYJqC6YLAK_qMRVIQkHiUmr9laBu%22%5D,%22action%22:%22open%22,%22userId%22:%221086920031335905830 47%22%7D#G1IQj6SYJqC6YLAK_qMRVIQkHiUmr9laBu

Batten, C. (2017, October 26). CS410 Dungeon Escape Demo (Download Source Code). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Batten, C. (2017, October 26). CS410 Dungeon Escape Demo (Play Now). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Edraw. (2017, May 12). Standard Flowchart Symbols and Their Usage. In Edraw Visualization Solutions. Retrieved from https://www.edrawsoft.com/flowchart-symbols.php

Everitt, C. (2017, September 6). Current Process Flow. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B-5KdQEdqLUPd

nBFUnp2V05uMEE%22%5D,%22action%22:%22open%22,%22userId%22:%22108692003133590583047%22%7D#G0B-5KdQEdqLUPdnBFUnp2V05uMEE

Everitt, C., & Dang, D. (2017, September 24). currentProcessFlow. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B3Bc95zBWXg9TFZ6X0FMU1NTdEk%22%5D,%22action%22:%22open%22,%22userId%22:%22108692003133590583047%22%7D#G0B3Bc95zBWXg9TFZ6X0FMU1NTdEk

Everitt, C., Santos, K. & DeArce, N. (2017, November 27). Work Breakdown Structure (WBS). In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B-5KdQEdqLUPWnNoSHhIUGg2OTQ%22%5D,%22action%22:%22open%22,%22userId%22:%22108692003133590583047%22%7D#G0B-5KdQEdqLUPWnNoSHhIUGg2OTQ

Everitt, C., Santos, K. & DeArce, N. (2017, October 13). ProcessFlowDiagram_silver. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B_xBnZ1ge4PlZTVjV3h6Y2pGSWc%22%5D,%22action%22:%22open%22,%22userId%22:%22108692003133590583047%22%7D#G0B_xBnZ1ge4PlZTVjV3h6Y2pGSWc

Few, S. (2008, February 5). Practical Rules for Using Color in Charts. In Perceptual Edge. Retrieved from http://www.perceptualedge.com/articles/visual_business_intelligence/

Rules_for_using_color.pdf

Kennedy, T. (2017, September 6). kennedyData. In Google Drive. Retrieved from https://drive.google.com/drive/u/1/folders/0B_xCQd8Vk2BnSU1hNnJwSXB1NEE

O'Neill, M. (2017, March 6). Computer Science Before College. In Computer Science Online. Retrieved from https://www.computerscienceonline.org/cs-programs-before-college/

Riley, P. (2017, September 14). Using Games to Introduce Programming to Students [PowerPoint slides]. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Santos, K., Riley, P. & Dang, D.(2017. December 7) Risk matrix and description tables in Design Presentation. Retrieved from https://docs.google.com/presentation/d/1oY9lkSAHvg2OIR kljYJNZWCqVTbiw45STKglsJUQjJI/edit#slide=id.g283e74317a_0_177

Stokes, J. (Narrator). (2017). CS410 Programming Game Pitch [Online video]. Online: YouTube. Retrieved from https://www.youtube.com/watch?v=QBvgzFgZaOQ&feature=youtu.be

Stokes, J. (2017, October 9). CS410 Programming Game Pitch (Download Source Code). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Team Silver. (2017, December 13). Prototype PowerPoint Presentation. In *PolyMorpher*. Retrieved from https://docs.google.com/presentation/d/e/2PACX-1vSidnjCKAu VEtKshHkyO7A-OfW3qWIKRkxcp0em412WwL1ig6SFmnqrMUyHr8-FMvzvaRjmcKYiCytq/pub?start=false&loop=false&delayms=3000&slide=id.g25ab9a9d23_0_1542

Team Silver. (2017, November 21). Design PowerPoint Presentation. In *PolyMorpher*. Retrieved from https://docs.google.com/presentation/d/e/2PACX-1vSllslBDmSvRfMI9nbrp0R

mRaPRsHNz7YWDfKNiF5sg15cp7ycQ774MuMgm4G4qhR6hohTiUQrrjRdo/pub?start=false&loop=false&delayms=3000&slide=id.g25ab9a9d23_0_1542

Team Silver. (2017, October 25). Feasibility PowerPoint Presentation. In *PolyMorpher*. Retrieved from https://docs.google.com/presentation/d/e/2PACX-1vReG6Sodx-gVFro1ByYMOYHSyiSRiU5HW-Su-PyMVGO8F4CQ7pY49tB_pJecVApruksoGaP_00ozhmR/pub?start=false&loop=false&delayms=3000&slide=id.g25ab9a9d23_0_1542

Team Silver. (2018, February 26). Lab 1. In *docs.google.com*.

"The Benefits of Video Games." abcnews (2011, December 26). Retrieved October 19, 2017, from http://abcnews.go.com/blogs/technology/2011/12/the-benefits-of-video-games/Good-Morning-America

Unity Technologies. (2017, August 10). Company Facts. In Unity. Retrieved from https://unity3d.com/public-relations

Unity. (2016, July 6). Unity - Scripting API. In Unity. Retrieved December 21, 2017, from https://docs.unity3d.com/530/Documentation/ScriptReference/index.html

Unity. (2017, October 11). Asset Store. In Unity. Retrieved December 21, 2017, from https://www.assetstore.unity3d.com/en/

1.5 Overview

In this document, a description of the PolyMorpher prototype will be provided. It will discuss the functionality, capabilities, and features of PolyMorpher. The functional requirements are provided in a separate document

2 General Description

The PolyMorpher prototype will provide a demonstration of the educational capabilities of the real-world product. The prototype will provide levels focused on different OOP concepts such as abstraction, encapsulation, polymorphism, and inheritance. The prototype will demonstrate how these concepts can be taught through gameplay.

2.1 Prototype Architecture Description

The PolyMorpher prototype will be separated into three main components: the PolyMorpher application, Unity file structure, and the PolyMorpher website.

The application will be an executable available for download from the PolyMorpher website. The executable will be provided as software compatible with Windows, Mac, and Linux.

The Unity file structure refers to the mechanism that applies player-edited code to Unity game objects. The “StreamingAssets” directory holds files that are subject to change when the game is running. The directory will hold the scripts for objects that the player can edit during gameplay.

The PolyMorpher website acts as the central hub for information regarding PolyMorpher. The executable will be available for download from this website. A play guide with instructions and helpful hints will also be available on the website. Figure 3 shows how these components will interact with each other.

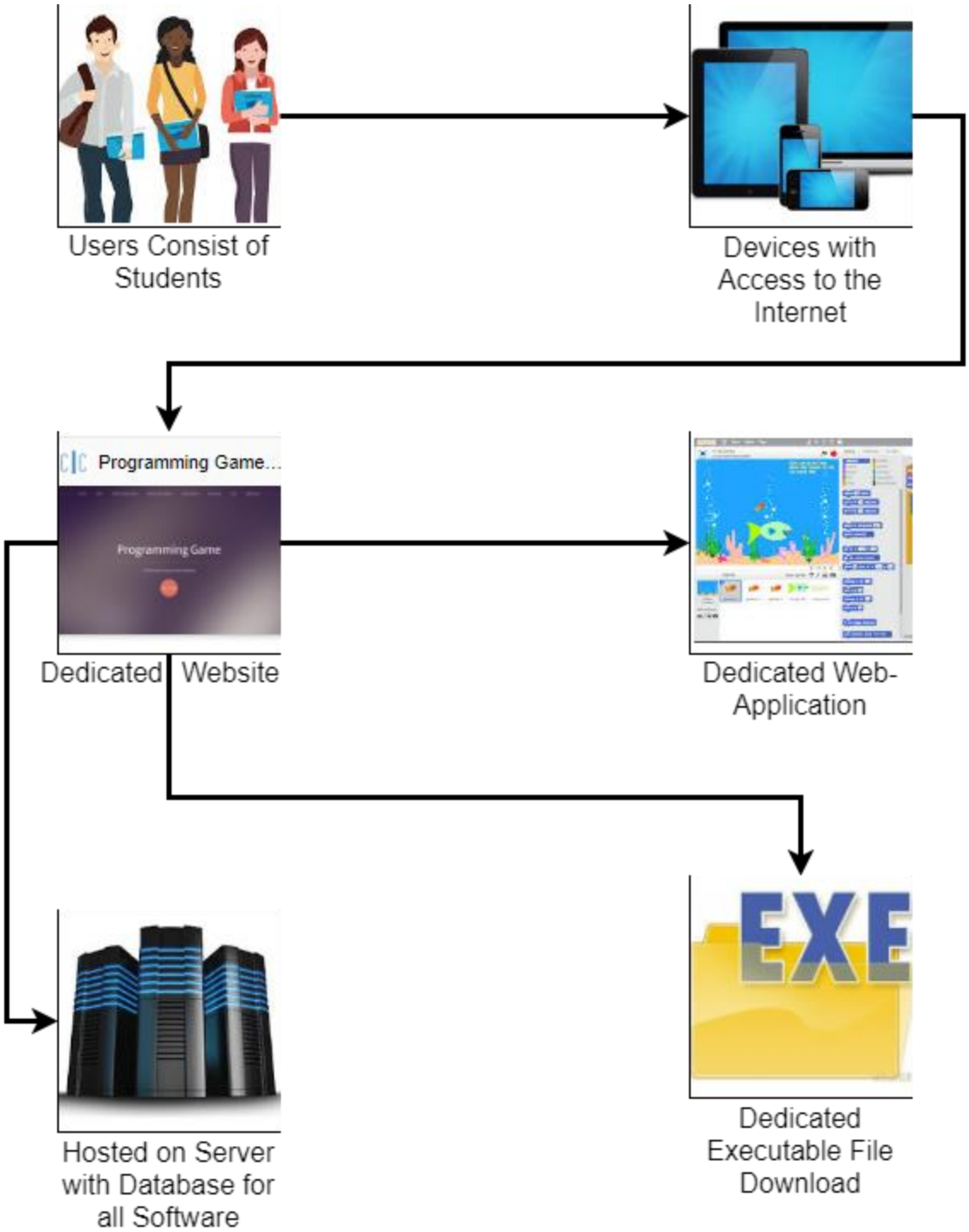

Figure 3 Major Functional Components

2.2 Prototype Functional Description

Gameplay in PolyMorpher will consist of solving puzzles to advance through levels. The levels are divided into different object-oriented programming concepts. These concepts are polymorphism, abstraction, encapsulation, and inheritance. Two more levels that serve as tutorials for game mechanics and C# will also be developed.

2.3 External Interfaces

PolyMorpher five types of external interfaces. These include hardware, software, user, API Book, and Compiler interfaces.

2.3.1 Hardware Interface

PolyMorpher will be played on a user's PC. The user's PC must have as a minimum a 4th generation Intel i3 processor. PolyMorpher will be implemented as a 2D game to reduce the draw on system resources.

2.3.2 Software Interface

PolyMorpher will be a desktop application. PolyMorpher will require a Windows, Mac, or Linux operating system to run the game. Unity has the ability to build applications for cross-platform use, so PolyMorpher will be cross-platform. PolyMorpher will be using a modified version of Mono's C# compiler to allow for runtime compilation of player-edited code.

2.3.3 User Interface

The game will be displayed on a computer monitor. PolyMorpher will be controlled with a keyboard and mouse. The keyboard will be used for movement and editing code. The mouse will be used to interact with "morphable" objects.

2.3.4 API Book Interface

PolyMorpher will provide an API Book to its players. This book will contain function skeletons for the player to use as hints in puzzles. The function skeletons will be entered into the code editing interface when clicked. The player will then be able to modify the function to their needs. Figure 4 shows the flow chart for the API Book.

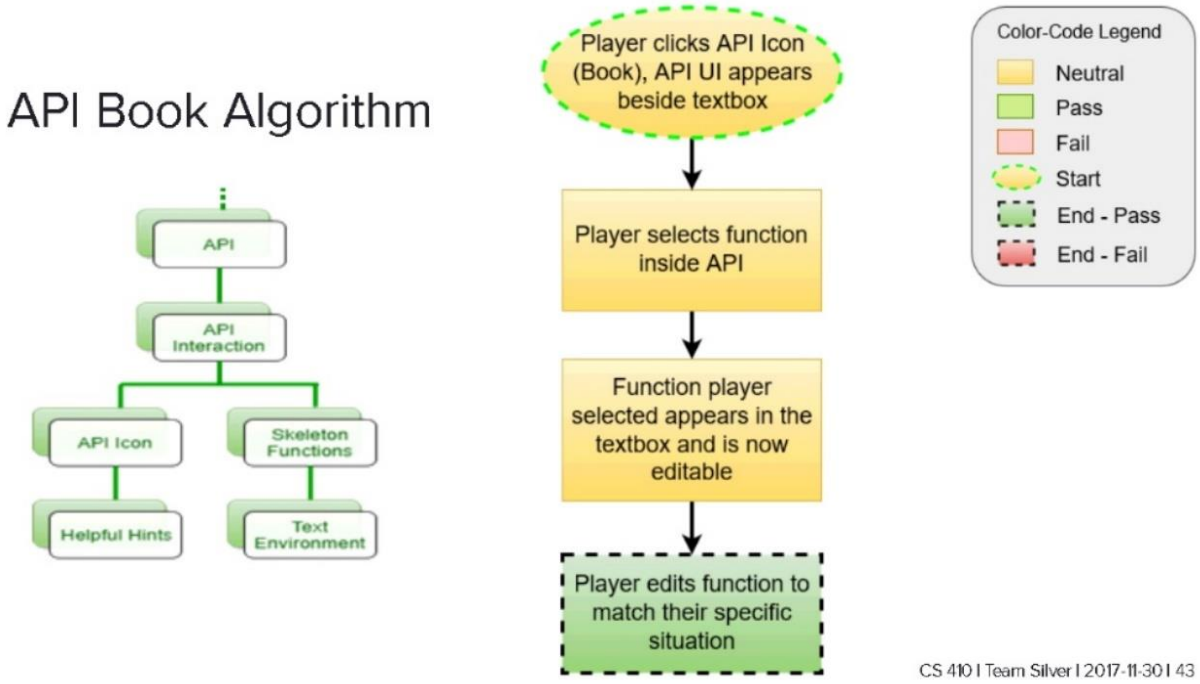

Figure 4 API Book Flowchart

2.3.5 Compiler Interface

The Compiler Interface is the ability for the player to modify the code of game objects. This is the main mechanic of PolyMorpher. The code from the API Book can be used as a template in this interface. Figure 5 shows the flowchart for the Compiler.

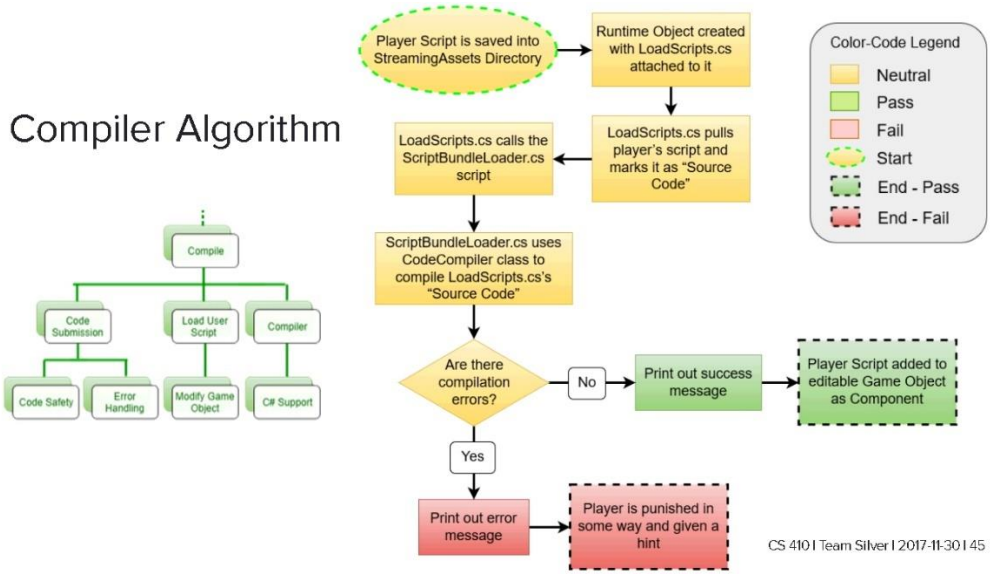

Figure 5 Compiler Flowchart