Lab 2 – PolyMorpher Product Specification Outline

Team Silver

Tyler Johnson

CS411W

Professor Thomas J. Kennedy

February 25, 2018

Version 1

Table of Contents

List of Figures

1 Introduction

Programming is intimidating for the uninitiated. As a result, first time ODU programming students drop out or switch majors. Existing tools fail to teach Object-Oriented Programming (OOP) concepts and problem-solving skills. The reality is that programming is a skill that can be learned, like any other skill such as drawing, or cooking. The main problem is a lack of understanding of the programming fundamentals. As a result of not knowing the fundamentals, many students find themselves lost, frustrated, and end up dropping their core CS class or switch majors. The current process flow is shown in Figure 1. After having Polymorpher introduced in process, as shown in Figure 2.

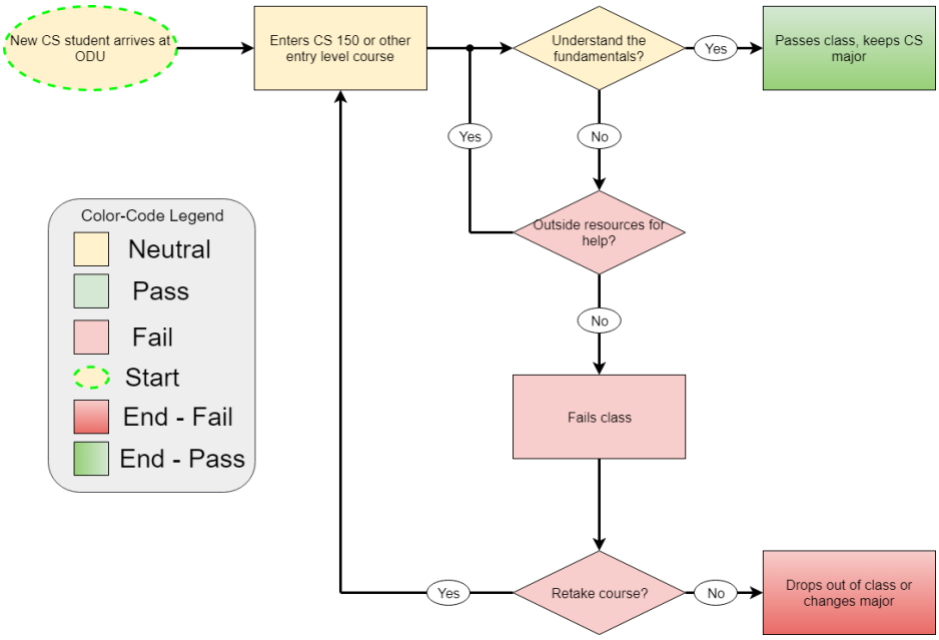

Figure 1 - ODU's Current Process Flow

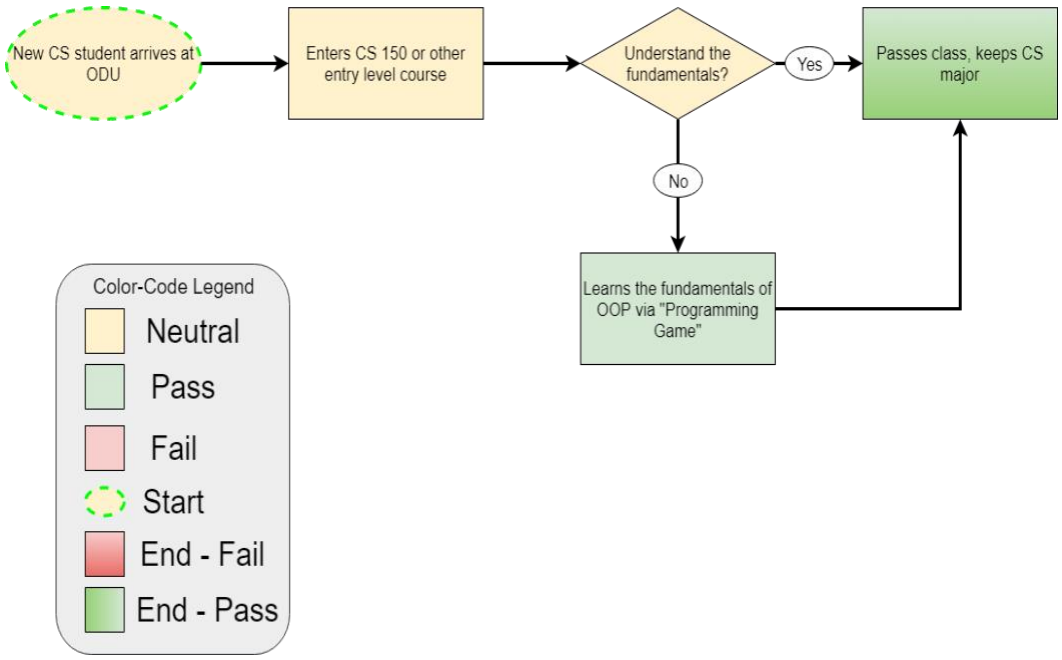

Figure 2 - Solution Process Flow

1.1 Purpose

PolyMorpher is a programming game that can be used as a teaching aid for computer science students. It will help players learn Object-Oriented Programing (OOP) concepts and program solving by using a management simulator and a Tangible User Interface (TUI). Polymorpher will: teach OOP concepts, teach problem solving, strive to teach multiple languages, be developed for multiple platforms, and potentially have multiplayer capability. As shown in Figure 1, the current process for new CS students being introduced to Computer Science is more complex and can more often lead to students dropping their intended course or abandoning the major as whole. Figure 2 shows how this process can be improved with Polymorpher. Polymorpher will increase the retention rate of students and the possibility of passing.

Polymorpher will be initially aimed at ODU and other educational institutes that teach players computer programming, Object-Oriented Programming concepts, and problem-solving skills. The game's goal is to better prepare students for the rigors of their respective computer science program. Object-Oriented Programming is the main focus of Polymorpher since it is a fundamental part of the computer science curriculum. This will be done by teaching them the fundamentals of programming in a fun and engaging way. Polymorpher will also eventually target secondary educational institutions such as middle schools and high schools. This would allow high school and middle school teachers to introduce computer science to their students in fun and engaging way. Polymorpher can also be used by anyone who is interested in learning programming skills. It is especially useful for those that want to learn Object-Oriented Programming.

1.2 Scope

Polymorpher's main objective is to become a teaching aid for university professors at ODU and a learning tool for CS students. Polymorpher will do this by presenting students with challenges that force them to implement different concepts of OOP. This will provide the student with a more engaging way of learning OOP rather than reading through a textbook or working through short exercises. The goal of the Polymorpher prototype is to provide a baseline for the product. It will include features fundamental to the Real-World Product. It will be playable but may not be as engaging as the full-fledge game. The Polymorpher prototype will teach players Object-Oriented Programming through different levels. Each of Polymorpher's levels will present the player with different challenges to teach them the concept they need to learn and to "win" the level.

1.3 Definitions, Acronyms, and Abbreviations

API: Application Program Interface - A tool for assisting developers in creating applications

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 - A graphical display for users of electronics to interact with the content displayed

JavaScript: a programming language commonly used in web development where 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

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: Abbreviation for 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

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:%22108692003133590583047%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-5KdQEdqLUPdnBFUnp2V05uMEE%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%220B5KdQEdqLUPWnNoSHhIUGg2OTQ%22%5D,%22action%22:%22open%22,%22userId%22:%221086920031335905830 47%22%D#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:%221086920031335905830 47%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/1oY9lkSAHvg2OIRkljYJNZWCqVTbiw45STKglsJUQjJI/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

This product specification will provide a general overview of the prototype of Polymorpher. It will emphasize the hardware and software configuration, external interfaces, capabilities and features. The information that will be provided in the remaining sections of this document include a detailed description of the software, hardware and external interfaces of the Polymorpher prototype.

2 General Description

The goal of the Polymorpher prototype is to provide a baseline for the product. It will include features fundamental to the RWP. It will be playable but may not be as engaging as the full-fledge game. It will still teach OOP concepts and problem-solving skills and provide players an opportunity to experience the mechanics of the game. The Polymorpher prototype will teach the most important concepts of OOP such as: encapsulation, abstraction, polymorphism and inheritance.

2.1 Prototype Architecture Description

The Polymorpher prototype architecture and the fully developed product will be exactly the same. The architecture of the prototype will feature two main components: the Polymorpher application and the Polymoprher website. The PolyMorpher website will as a method of distribution for the prototype. The website will also include a game guide that will have game rules, basic gameplay instructions and helpful tips. The most important part of the prototype is the Polymorpher application. The Polymoprher application is an executable file that will be downloaded from the website and run directly from the user's computer. The interaction between the users, the website and the application are shown below in Figure 3.

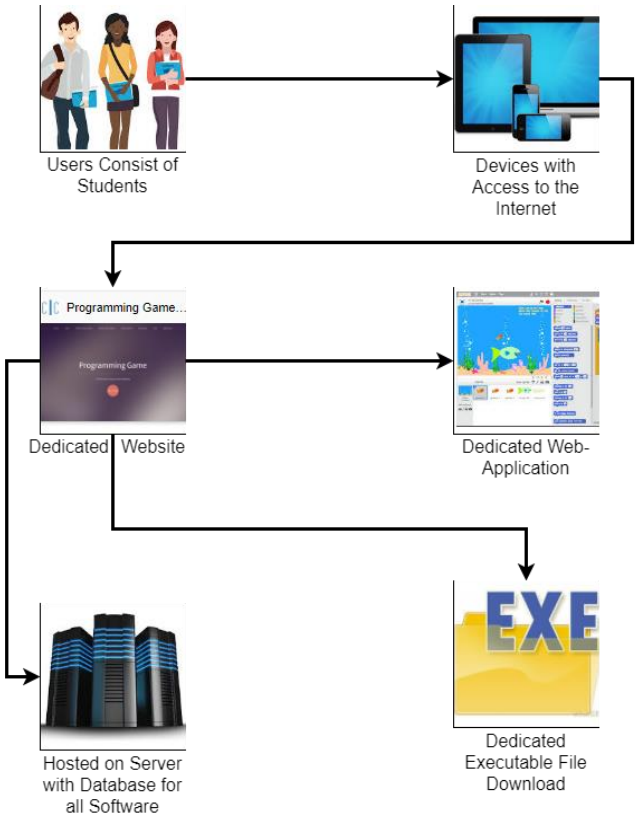

Figure 3 - Deployment Flow

2.2 Prototype Functional Description

The Polymorpher prototype will have many of the features that the real-world product version would have. The prototype will: teach OOP, be single player, include game assets, have a developed story, include a downloadable executable file, include a portable compiler, and a tutorial section. The Polymorpher prototype will only include what is needed or is essential to create a minimum viable product.

2.3 External Interfaces

The Polymorpher prototype will consist of five different interfaces: Hardware Interface, Software Interface, User Interface, API Book Interface, and Complier Interface. The Polymopher prototype will also require that the user meet specific hardware specifications. This will ensure that the Polymorpher prototype will run on the user's computer smoothly.

2.3.1 Hardware Interfaces

Polymorpher will be an executable that will be available to download from the Polymorpher website. The system requirements for the game will include: fourth generation i3 Intel or AMD equivalent processor, at least 4GB of RAM or more and at least 500MB of available Hard Disk space. A mouse and keyboard will also be needed to play Polymoprher.

2.3.2 Software Interfaces

Since Unity was used build the game, this should allow Polymorpher to run on most operating systems. Polymorpher will run on the following: Windows, Linux, and Mac OSX. DirectX may need to be installed as well prior to running Polymorpher.

2.3.3 User Interface

- Keyboard: A keyboard will be needed to move the main character of Polymoprher. The arrow keys of the keyboard will mainly be used for movement. The keyboard would also be used when the player uses the API Book to look up examples and hints. The keyboard will also be used to write C# code to in a text editor to change objects.
- Mouse: A mouse will be needed to select objects during gameplay that the player wants to change. The mouse will also be needed to interact with the API book and the general user interface of the game.
- Computer Monitor: A computer monitor will be needed to view and play the game once it has been launched.

2.3.4 Compiler Interface

The Compiler Interface is used by the player to change in-game objects. The player will use their mouse to select the object they intend to change. Once they click on the object, a text box will appear that is filled with C# code. The player will then have three options: compile, reset and cancel. The compile button will run the code that is within the textbox. The reset button will reset the code and the cancel button will close the text box. In Polymorpher, the code presented to the user will often be incorrect and will need to be corrected by the user in order to pass the level. The player may not always be given starting code, so the compiler text box may be empty. The Compiler Interface and the API Book are dependent upon each other, and critical components to Polymorpher's overall design. The flow of the Compiler Algorithm is shown in Figure 4.

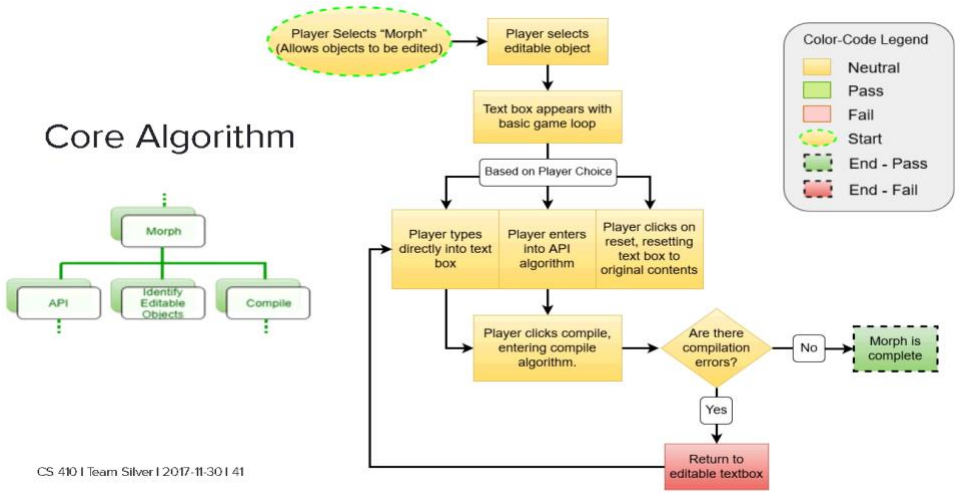

Figure 3 - Core Algorithm Flow

2.3.5 API Book Interface

When the player clicks on the API Book Icon, the API Book interface launches. As shown in Figure 5, the API Book Algorithm details the usage of the API Book within PolyMorpher. The API Book Interface is used by the player to find examples and helpful hints. It serves as a core resource for the player to use throughout the game rather than having to go to outside help.

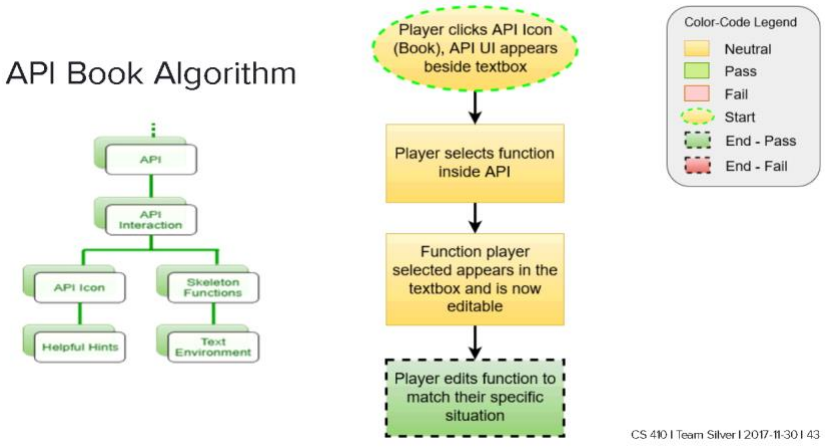

Figure 4 - API Book Algorithm Flow