Lab 1 - PolyMorpher Product Description

Kevin Santos

Old Dominion University

CS411W

Professor Thomas Kennedy

29 January 2018

Version 1

Table of Contents

1. Introduction **5**
1.1 Team Members 5
1.2 Problem Statement 6
1.3 Problem Characteristics 6
Figure 1: Current Process Flow Diagram 1 7
1.4 Solution Characteristics 7

2 Product Description **9**
2.1 Goal and Objectives 9
2.2.1 Game concept (Basic idea of the game) 10
2.2.2 Focus on OOP (why is it different than the competition?) 10
2.3 Major Functional Components (Hardware/Software) 11

3 Identification of Case Study **12**
3.1 End-User: Students 13
3.1.1 ODU 13
3.1.2 Other colleges 13
3.1.3 Those generally interested in programming 14
3.2 Customers 14
3.3 Why make this for students 15

4 Product Prototype Description **16**
4.2 Prototype Features and Capabilities 17
4.2.5 Algorithms 19
4.2.5.1 Core Algorithm 19
4.2.5.2 API Book Algorithm 20
4.2.5.3 Compiler Algorithms 21
4.3 Prototype Development Challenges 23
4.3.1 Design continuity 23
4.3.2 Difficulty debugging 24
4.3.3 Playtesting 24
4.3.4 Maintaining player engagement 24
4.3.5 Teaching enough 24
4.4 Risk Matrix/Risk Mitigation 25
4.4.1 Technical risks and mitigations 26

4.4.1 Customer risks and mitigations 27

5 Development Pipeline **28**

5.1 Unity Software 29

5.2 SourceTree Software 29

5.3 Version Control 30

5.4 Development Model 31

5.5 Work Management 32

Glossary **34**

References **36**

List of Figures

Figure 1: Current Process Flow Diagram 1 **7**

Figure 2: Solution Flow diagram 9

Figure 3: Major Functional Component Diagram 12

Figure 4: Combined Matrix, Bar Graph and Pie Chart 16

Figure 5: Dataflow Diagram for the Core Algorithm 20

Figure 6: Dataflow Diagram for the API Book Algorithm 21

Figure 7: Dataflow Diagram for the Compiler Algorithm 23

Figure 8: Dataflow Diagram for Version Control 31

Figure 9: Dataflow Diagram for Agile Development Model 32

List of Tables

Table 1: Competition Matrix 8

Table 2: Features of the Completed Product against the Prototype 18

Table 3: Risk/Mitigation Matrix of Customer & Technical Risks for PolyMorpher 26

Lab 1 - PolyMorpher Product Description

1. Introduction

Programming is a particular technical skill that requires one to retain it but also to regurgitate it in the form of practice. This skill can be very intimidating for those that are not committed to maintain perseverance for learning it. There us a different thinking perspective when it comes to learning programming because it requires a logical angle that laymans in this fields are not used to thinking. Traditional forms of teaching Object Oriented Programming and problem-solving skills at Old Dominion University and possibly other colleges/universities are not producing the desired results. Thus, PolyMorpher is a game that is designed to teach the player complex OOP concepts and problems solving skills. In order for PolyMorpher to succeed at solving the issues within this problem, it will have to improve on the deficiencies and mistakes traditional academic learning has.

1.1 Team Members

A group of Computer Science majors from ODU students proposed and are currently developing a solution to this particular to this problem under the guidance of Thomas J. Kennedy, who is their mentor. The name of the team is Team Silver and its member are Casey Batten, Peter Riley, Matthew Tuckson, Kevin Santos, Colten Everitt, Daniel Dang, Nathaniel DeGrace, and Tyler Johnson. Most of these students are in their final semester pursuing their undergraduate degree in CS.

1.2 Problem Statement

As mentioned before, programming can be a rigorous and intimidating skill for many people to learn. ODU programming students that are particularly CS majors, dropout or switch

majors because of the rigor in terms of learning this particular skill especially dealing with Object Oriented Programming(OOP) and problem-solving skills. In addition, the traditional academic learning techniques and tools that are used to teach OOP have been proven to not work effectively. The evidence for this is because of the high dropout rates and switch out of majors the CS major.

1.3 Problem Characteristics

Team Silver's mentor, Mr. Kennedy provided very astonishing data from the ODU's CS department enrollment records that shows a decreasing trend of CS students moving successfully forward to advance level courses. The statical numbers will be shown in later parts of this lab report. However, the volume of students enrolled in advance level CS courses at ODU has decreased significantly. Figure 1 below shows the current data flow diagram of the possible paths CS students take towards their CS degree. It illustrates both conditions if the student successfully understands the fundamentals of OOP and proceeds to the next CS course. Otherwise, it shows that if they don't understand such content, then the possible scenarios would be they drop out or switch to another major.

[This space intentionally left blank]

.

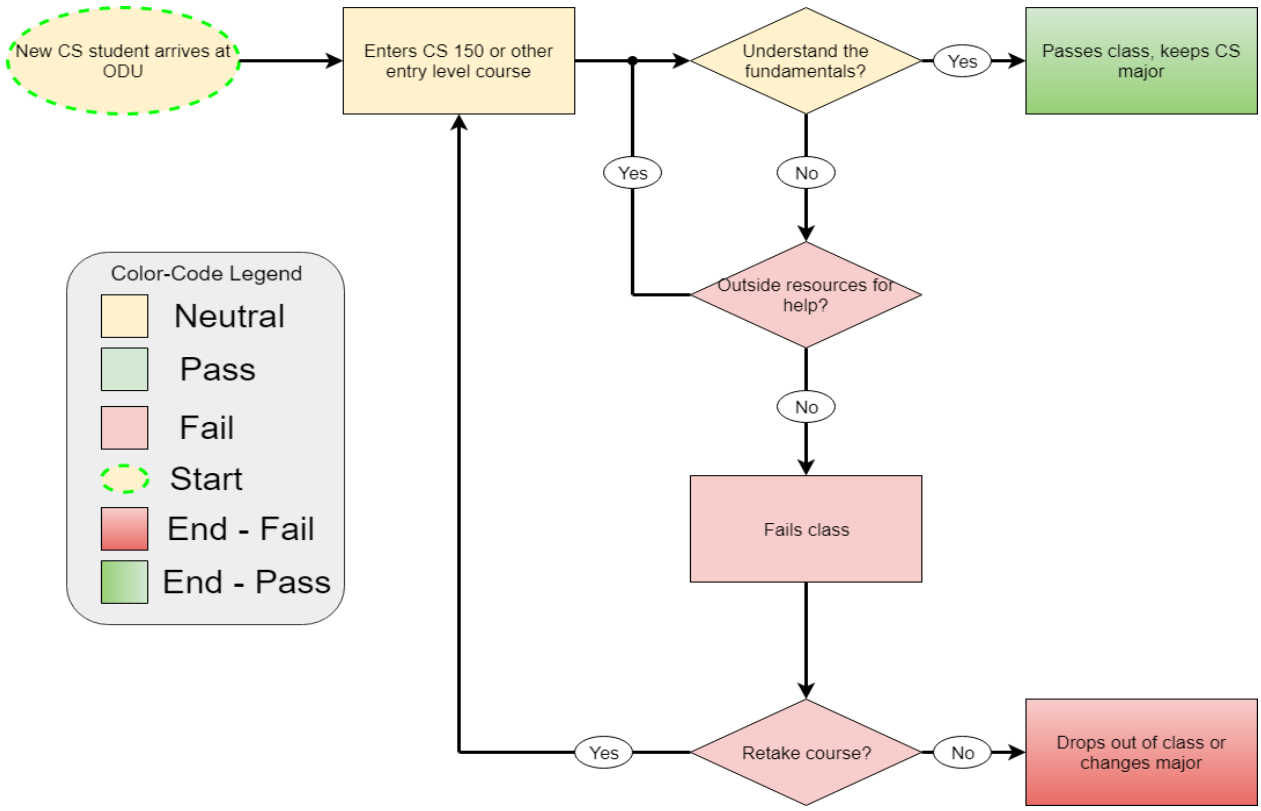


Figure 1: Current Process Flow Diagram 1

1.4 Solution Characteristics

PolyMorpher is a software game that is intended to teach the player OOP concepts in an engaging and interactive way. In addition, this game will make use of Tangible User Interface (TUI) and a management simulator add teach both of previous concepts mentioned before. Polymorpher will improve on using and teaching OOP concepts that it is not strictly addressed by its competitors. Table 1 below shows the competition matrix that makes PolyMorpher stand out from its competitors in the aspects of teaching and using OOP concepts. In addition, Figure 2 that is below Table 1 which is the Solution Flow diagram shows what would be the ideal scenario if a student at ODU understands the fundamentals of OOP and continues as a CS major

progressing through the next required courses. This is one of the main accomplishments Team Silver states that PolyMorpher is intended to do.

Game	Experience	Uses OOP	Teaches OOP	# Languages	Multiplayer
PolyMorpher	Low-Mid	Yes	Yes	1	No
Code Combat	Low	Yes	No	5	No
Screeps	Mid-High	Yes	No	1	Yes
CheckIO	Low-High	Yes	No	1	Yes
Code Monkey	Low	No	No	1	No
Elevator Saga	Mid-High	Yes	No	1	No
Codewars	Mid-High	Yes	Yes	6	Yes
Codingame	Low-High	Yes	No	25+	Yes
Git Games	Low	No	No	1	No
CSS Diner	Low	No	No	1	No
Flexbox Defense	Low-Mid	No	No	1	No
Ruby Warrior	Low	No	No	1	No
Untrusted	Mid-High	No	No	1	No
Empire of Code	Low-Mid	Yes	No	2	Yes
Ruby Quiz	Mid-High	Yes	No	1	No

Table 1: Competition Matrix

[This space intentionally left blank]

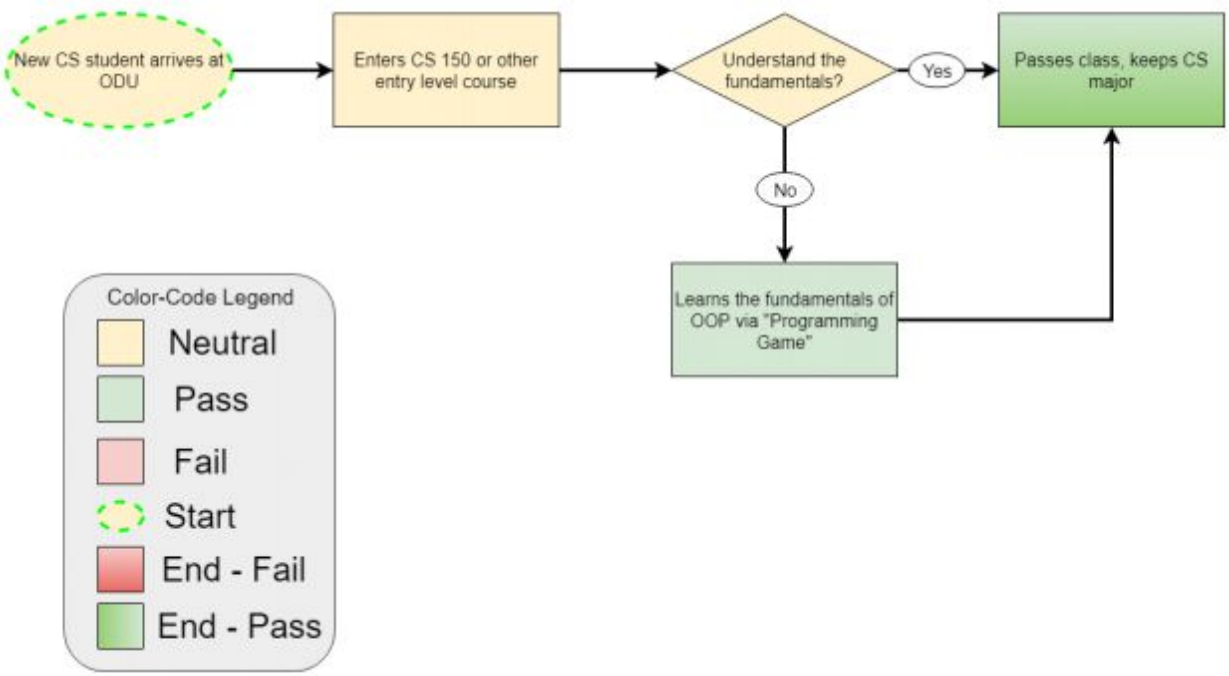


Figure 2: Solution Flow diagram

2 Product Description

The uniqueness of PolyMorpher is that it is a game intended for learning purposes that, as previously mentioned, uses and teaches OOP concepts in a way that will interactive and engaging. I addition, PolyMorpher will make use of a Tangible User Interface (TUI) and a management simulator so that the traditional computer-based gaming experience is offered to the player as they play the game in their Personal Computer (PC).

2.1 Goal and Objectives

The goals and objectives of PolyMorpher are to teach OOP and problem-solving skills in a unique where it is engaging and interactive to the player. Since games, in general, are interactive and engaging, PolyMorpher as a game, will take advantage of that to teach OOP

concepts that traditional academic learning methods don't. By using a management simulator and a TUI, PolyMorpher should adhere to accomplishing its goals and objectives to effectively teach its players the rigorous concepts in OOP and problem-solving skills.

2.2.1 Game concept (Basic idea of the game)

The basic idea of the game is that it will consist of three important concepts which are: Realist Approach, Improvement on Teaching, and a Balance Gameplay. The Realist Approach allows the game to use applicable and retractable examples to teach the player OOP. Improvement on Teaching will consist of facilitating the teaching of complex OOP concepts to the player. Lastly, the Balance Gameplay concept will allow the implementation of the gameplay to not jeopardize the experience the player gains while at the same time they are learning.

2.2.2 Focus on OOP (why is it different than the competition?)

PolyMorpher focuses on OOP heavily along with problem-solving skills and it is different from its competitors in a positive way. This is because as seen in Table 1 for the competition matrix, most of the competitors do not primarily focus on OOP. As the problem statement for Team Silver embossed previously, their research pointed out that traditional educational tactics towards teaching OOP concepts are not effective to new learners. Thus, they claim that PolyMorpher will effectively teach the player mainly the important aspects of OOP. The fact of the matter is that only 1 of their 14 competitors, according to Table 1, teaches OOP concepts so PolyMorpher’s intent is to improve and lead in regards to teaching this aspect.

[This space intentionally left blank]

2.3 Major Functional Components (Hardware/Software)

PolyMorpher focuses on OOP heavily along The major functional components that are needed to run the PolyMorpher are not sophisticated. Common operating systems like PolyMorpher will be compatible are modern versions of Windows, Linux, and MacOSX. PolyMorpher will be implemented as a 2-Dimensional game that will draw minimal computing power resources. In addition, Polymorpher will be optimized to perform with Intel's 4th generation i3 processor. The player will need an internet connection through an IPv6 protocol or earlier versions to navigation in Team Silver's website and then download the game as an executable file.

2.3.1 Need to download the .exe

PolyMorpher will be single player game that will not require internet connection to play it. At the primitive stage of designing this game, the single player and no internet connection options were conducted, per Team Silver, to provide security against hacks and other cyber malicious intents. Therefore, in order play PolyMorpher, the player will need to download an executable file in their native PC. Assuming they have basic computing skills, the player is to run the downloaded executable program and simply play the game. Figure 3 shows the major functional components as a flow diagram to illustrate the procedure that is need to download the executable file.

[This space intentionally left blank]

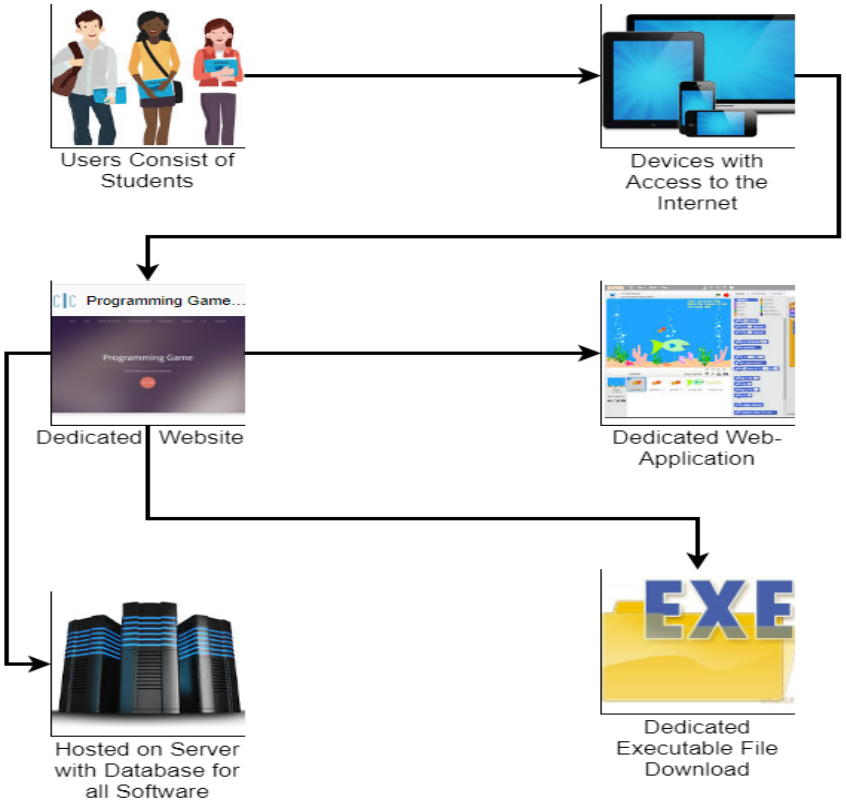


Figure 3: Major Functional Component Diagram

3 Identification of Case Study

The problem and issue that PolyMorpher intents to solve is geared towards particular customers and end users that are affected by it. However, many unexpected customers and end user that are not in CS and are in other academic fields could potentially be interested in PolyMopher because of the fact that it's a game focused on teaching. The potential end users for PolyMorpher are students that are either from ODU or other colleges/universities. Potential customers for PolyMorpher are in same category as end users but extend to people from different academic and work areas.

3.1 End-User: Students

The end users for PolyMorpher will be primarily students who want to learn programming focusing on OOP concepts. Since PolyMorpher's goal is to teach programming, Team silver has claimed that students would be the ones who would be most interested in exploiting the benefits of this game. In addition, PolyMorpher is also geared towards helping students that enjoy playing games as a hobby. Thus, PolyMorpher provides the benefit of letting the student play it and at the same time teach complex OOP concepts.

3.1.1 ODU

One of the primary end users are particularly ODU students in the CS departments. According to the research data that Team Silver gathered from their mentor, there exists a significant decrease in the volume size of enrollment in advance CS courses. Their data points out that the cause of low enrollment in advance CS courses is due to CS students either dropping out of this major or switching out of it because they don't understand the complex concepts of OOP and problem-solving skills. Thus, Team Silver states that the traditional methods do not work towards teaching students OOP and problem-solving skills effectively. Therefore, they reiterate that PolyMorpher will achieve teaching OOP concepts to CS students in a more interactive and effective way through playing a game.

3.1.2 Other colleges

Other colleges and universities, according to Team Silver, will be potential end users as well. Particularly, their CS students will benefit greatly from PolyMopher because it will address the difficulties they are possibly having with comprehending and understanding OOP concepts and problem-solving skills. In addition, PolyMorpher will not have to accommodate to particular

colleges/universities regarding their CS curriculums because they are more likely similar to the one at ODU. Thus, PolyMorpher will help CS students from other colleges/universities because the same solution characteristics that Team Silver has stated will help ODU CS students, will help them as well.

3.1.3 Those generally interested in programming

PolyMorpher will also take into account those individuals that interested in programming as potential end users. According to the data by provided by Team Silver's mentor, there are other ODU students from different majors particularly in engineering who either want or have to take CS intro courses but struggle with learning programming. OOP and problem-solving skills are traditionally taught in these introductory courses. Team Silver's data shows that students from various major backgrounds are struggling with learning these concepts. Therefore, PolyMorpher can help these types of end users because it focuses on dealing with the teaching and interactive aspect of the game instead of the individual itself.

3.2 Customers

The customers that will be considered for PolyMorpher as potential targets will consist of students and instructors not only form ODU, but also from other colleges and universities. In addition, other potential customers are individuals that are interested in learning programming either for academics or work skills. In addition to mentioning why students and individuals, in general, might be potential customers, instructors can greatly benefit from using PolyMorpher. The reason is that if they can benefit from learning OOP concepts and problem-solving skills, then they can teach it effectively to their students. In addition, it is possible that instructors that

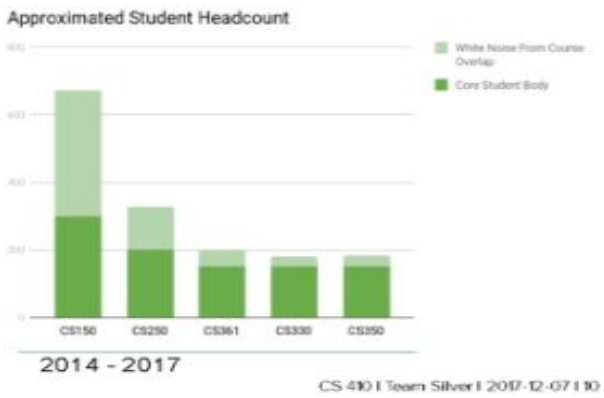
benefit from PolyMorpher can use it as part of their programming course curriculum labeling it as a new effective teaching tool.

3.3 Why make this for students

The traditional academic methods that are used to teach complex OOP and problem solving skills to CS students have been proven to not work effectively. According to the data that Team Silver gathered from Mr. Kennedy, it showed that there is a reduction in the enrollment size for advance CS level courses. This due to the fact that students that are not able to comprehend OOP concepts or problem solving skills either drop out or they switch majors if they cannot pass the CS introductory courses. Figure 4 has a combined matrix, bar graph and pie chart that shows the decrease number of students enrolled in CS courses as levels increase. This data was provided to Team Silver directly from Mr. Kennedy.

[This space intentionally left blank]

	CS 150	CS 250	CS 361	CS 330	CS 350
2013-2014	804	327	161	111	93
2014-2015	672	367	208	203	148
2015-2016	937	327	217	195	183
2016-2017	920	337	199	180	182



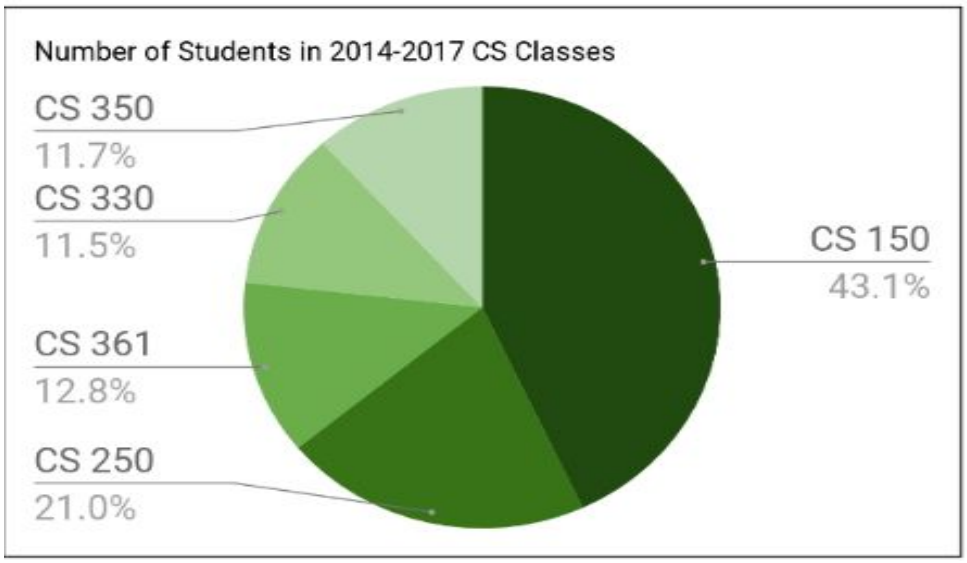


Figure 4: Combined Matrix, Bar Graph and Pie Chart

4 Product Prototype Description

The prototype deliverable will be a very exact simulation of the possible final product of PolyMorpher. This is due to the various video demonstrations that Team Silver has been providing in their CS411W presentations. One of the main purposes to create a prototype is to show the potential end users and customers a version of PolyMorpher that they can find a

potential of benefit from. The software and hardware deliverables for PolyMorpher will only require the download of an executable file from Team Silver's website. The executable file requires a reliable internet connection using IPv6 protocol or compatible versions, as previously stated. No external hardware is required for using PolyMorpher besides the player's PC. Figure 3 illustrated through a functional component diagram how a potential prototype can be delivered.

4.2 Prototype Features and Capabilities

The features and capabilities for the PolyMorpher prototype and final product will consists of 17 possible features that are ought to be delivered. However, those possible 17 features in the prototype will have possible constraints. This is because of the limitability that Team Silver will have to successfully build the prototype assuming technical issues occur in the software development process. On the other hand, the final product will be able to deliver all the possible 17 features that Team Silver decided to incorporate on developing PolyMorpher.

[This space intentionally left blank]

Elements	Description	Real World Product	Prototype
Teaches Polymorphism	Provision of a single interface to entities of different types		
Teaches Abstraction	Technique for arranging complexity of systems		
Teaches Encapsulation	Building of data with the methods that operate on that data		
Teaches Inheritance	When an object or class is based on another object or class, using the same implementation		
Single Language Taught	A single programming language will be focused on C#.		
Single Player	Focused on an experience targeted to interact with only one player		
Downloadable .EXE File	Desktop application version of the game		
Game Assets	Primary components that are used as building block to construct the more complex features and levels of the game		
Developed Story	Narrative used to drive progression or direct player throughout a more guided/linear experience		
Portable Compiler	Code compiler used to run player-made code on the fly in game		
Tutorial Section	Precursor series of levels meant to help the player adjust to the in-game toolset given to them and also prep them with knowledge of the language(s) they will be working with		
Multiple Platforms	Version support for multiple operating systems (Windows, Mac OS, Linux)		
Sandbox Level	Open level where the player has access to all tools at once and can build their own level sequences and puzzles		
Player-Made Content	Variant of Sandbox Level, potentially allows the player to share custom levels with one another		
Multiple Player	An experience geared toward multiple players interacting with a game environment together		
Web Application	Web based version of the game running in-browser		
Multiple Languages Taught	Alternative programming languages for the player to use and learn in-game		

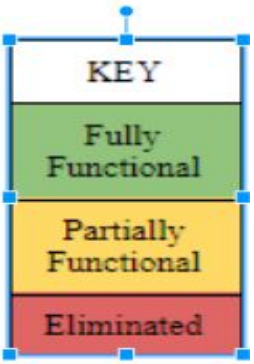


Table 2: Features of the Completed Product against the Prototype

As Table 2 shows, it will be guaranteed that PolyMorpher will be able to teach some of the most important OOP concepts that includes Abstraction and Encapsulation. In addition, the player will be able to benefit from a story progression and a tutorial section so that they can see their amelioration as they move from one level to another while they play through the game. One of the partially functional features that might be implemented on PolyMorpher is a sandbox level.

This is where the player is allowed to have access to all the tools available in the game so that they can design their game levels and puzzles. Lastly, the eliminated features of PolyMorpher for both the prototype and the final product are multiplayer, multiple languages taught, and the web application. All three of these applications requires a constant internet connection. Thus, Team Silver decided not to implement these three features because it is not guaranteed that the player will always have internet connection.

4.2.5 Algorithms

Team Silver decided to incorporate three algorithms. The first algorithm deals with the core implementation of PolyMorpher as a game. The second algorithm will focus on the API Book which will deal with the player having access to references that will help them with implementing particular snippets of code. The third algorithm deals with the compiler which will assemble and process the code the player is inputting in the game to see if it matches with the correct solution of the particular puzzle or task they solving on a level.

4.2.5.1 Core Algorithm

The Core Algorithm will consist of one component option named “Morph” that will deliver three sub-options. The procedure of the Algorithm starts with allowing the player to select the main option of “Morph” which allows objects in the game be edited. Then, the algorithm through the game will create a text box with a game loop that will give the player three option. The player will have the following available suboptions: the player types into the provided text box, the player enter into the API algorithm, or the player clicks on the reset option and the text box resets to its original setting. Once the player has selected and entered one of those three options, then the player selects the compile option which will enter into the compile

algorithm which will be mentioned next. Lastly, this algorithm with the aid of the compiler algorithm, will determine if there were any compilation errors from the input entered in three options selected before. If the there are no compilation errors, then the algorithm confirms that the "Morph" option is complete. Otherwise, the algorithm returns the player to the editable text box from the three previous sub-options. Figure 5 below illustrates the Core Algorithm in details as a dataflow diagram.

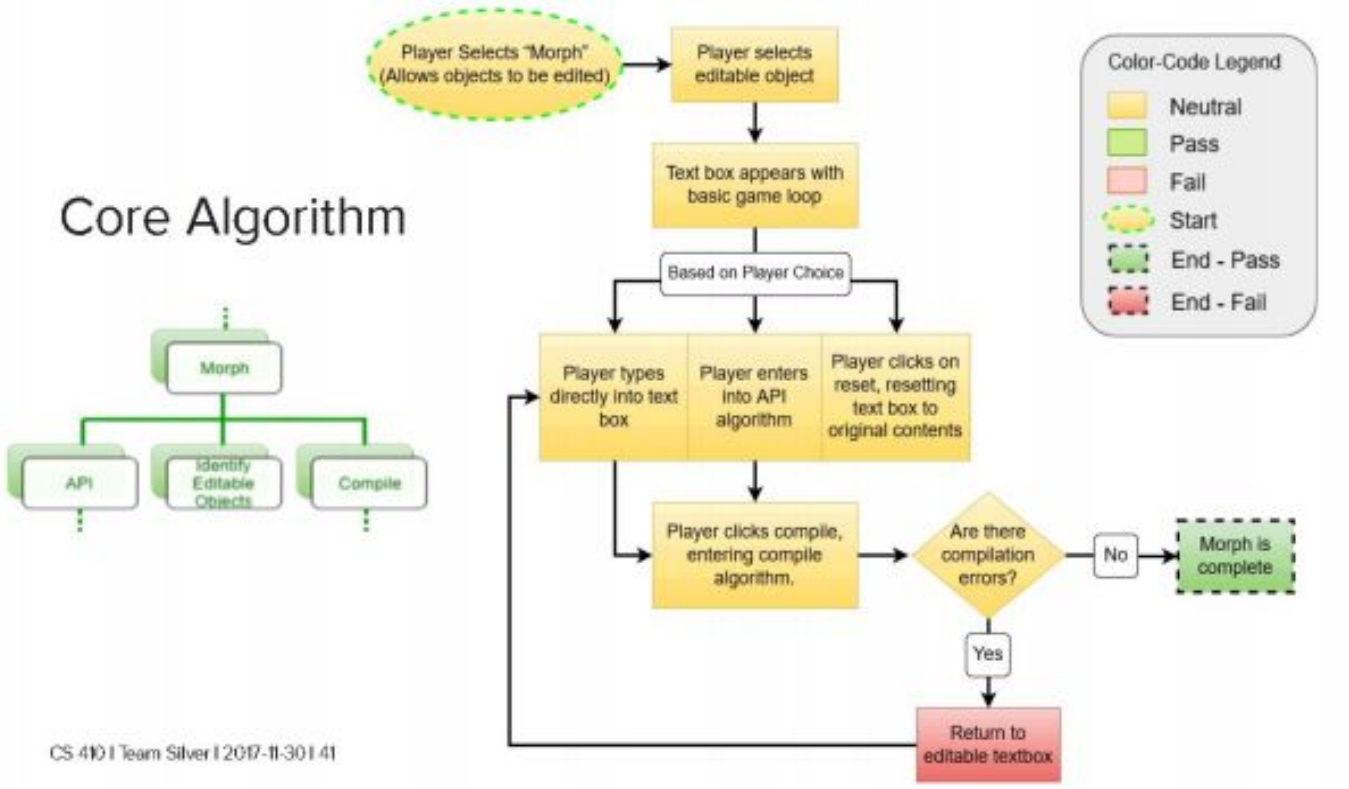


Figure 5: Dataflow Diagram for the Core Algorithm

4.2.5.2 API Book Algorithm

The API Book Algorithm will consist of a very straightforward procedure that will allow the player to basically access the API Book that PolyMorpher provides. This algorithm

administers the educational information that Team Silver had decided to implement in their design and grants that to the player. Thus, the API Book provides a variety tools to help them complete the puzzle and challenges in each level. In addition, this algorithm consists of three simple procedures. First, the player selects the API icon which will be an image of a book that appears in a textbox. Then, they select a function inside the provided API Book. Lastly, the function the player selects appears in the textbox which it can be editable and manipulated. Figure 6 shows API Book Algorithm dataflow diagram for further details.

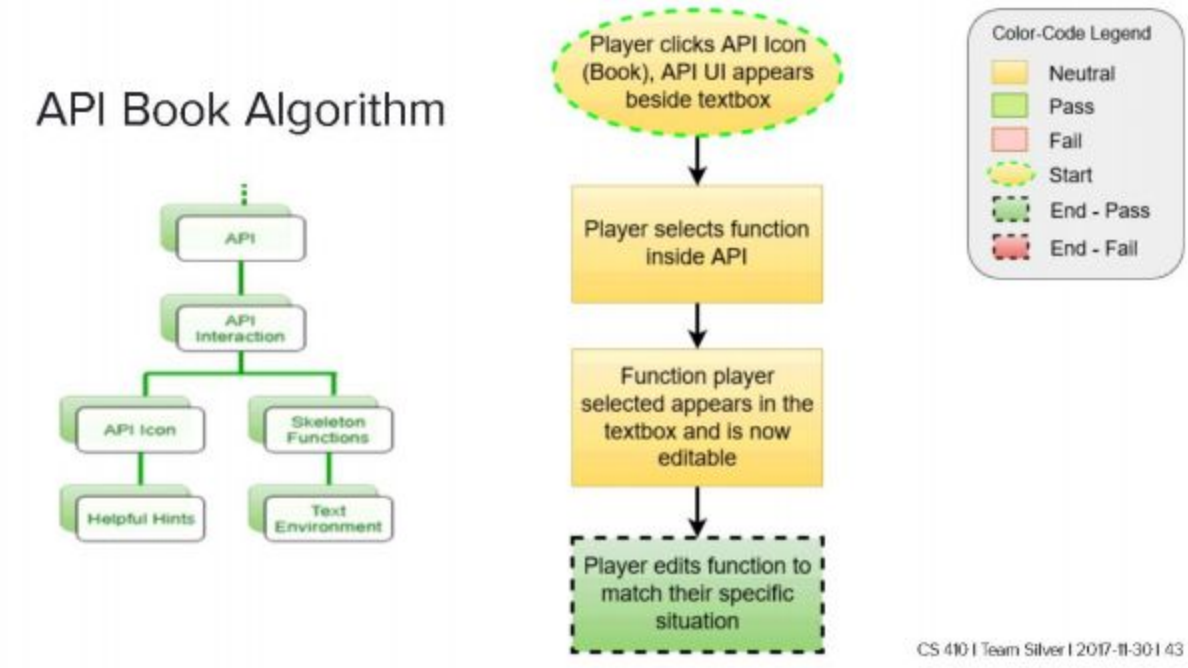


Figure 6: Dataflow Diagram for the API Book Algorithm

4.2.5.3 Compiler Algorithms

The Compiler Algorithm will act as a sort of portable assembler to process the scripts the player enters as they are involved in solving a puzzle or another particular challenge within a

level in the game. In addition, since PolyMorpher allows the player to “Morph” or manipulate objects as they progress in particular levels; the compiler has the duty to determine if the player’s script commands are acceptable within in the rules and conventions of the particular language (C#) will game will be teaching them. The Compiler algorithm procedure beings with saving the player’s script in the StreamingAssets Directory. Next, a Runtime Object is created and it functions along with premade LoadScripts.cs file that pulls the player’s script and marks it as “Source Code”. Then, the LoadScript.cs file calls another premade file called ScriptBundleLoader.cs to perform the portable compilation process for the code syntax labeled as “Source Code” in the LoadScripts.cs. ScriptBundleLoader.cs has a CodeCompiler class that performs the compilation of the player’s script that was stored in LoadScripts.cs. Lastly, ScriptBundleLoader.cs will determine if there were compilation errors in the player’s script. If such script is free from compilation errors, the PolyMorpher Graphic User Interface (GUI) will display a success message and player’s script is added to the editable Game Objects particularly as a component. Otherwise, the the GUI will display an error message and player will given a hint to correct their coding mistake. Figure 7 below illustrates the dataflow diagram for the Compile Algorithm.

[This space intentionally left blank]

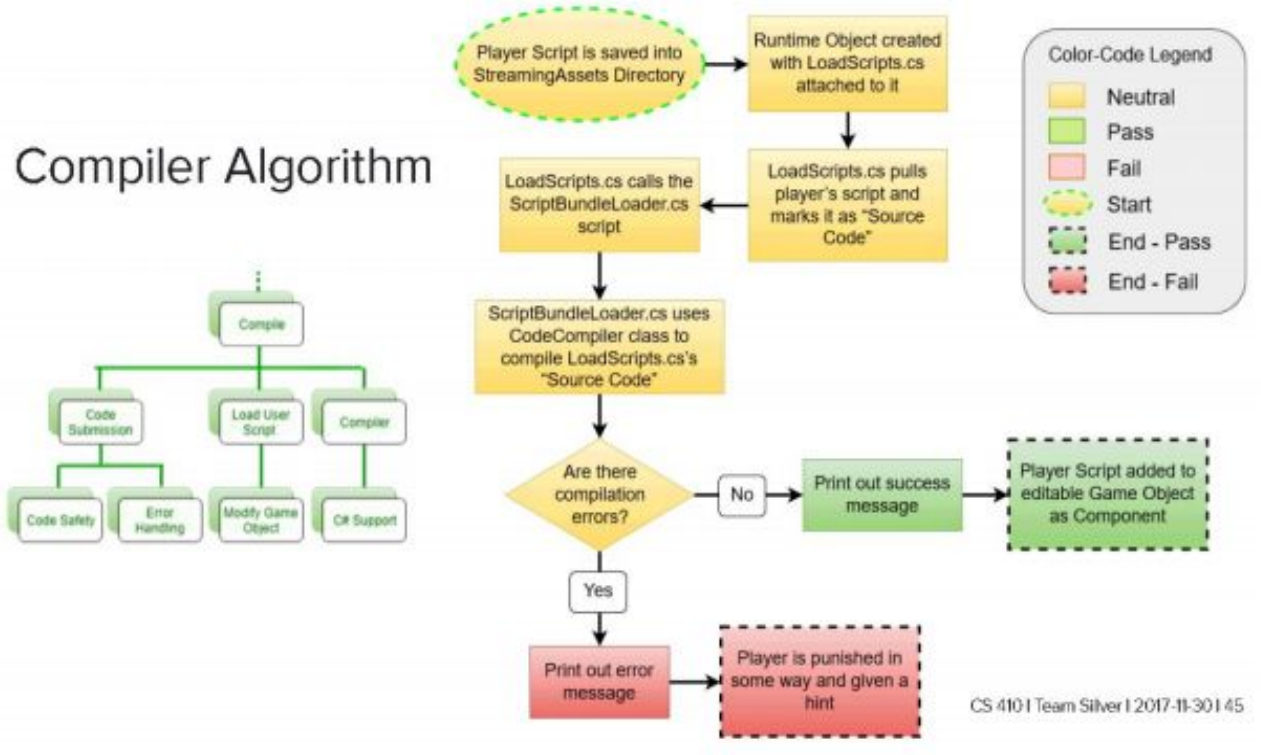


Figure 7: Dataflow Diagram for the Compiler Algorithm

4.3 Prototype Development Challenges

The development of the PolyMorpher prototype by Team Silver will encounter five challenges that were analyzed closely. The subsections below will elaborate further on each of those five prototypes challenges.

4.3.1 Design continuity

The consistency of the design of PolyMopher is imperative because that is what promotes the stability for the actual design of this game as well as the final product. Because of the time constraints for this semester and being that project will remain active until the beginning of May of 2018, Team Silver finds it arduous to keep a consistent design of PolyMorpher.

4.3.2 Difficulty debugging

A rigorous development challenge that Team Silver will encounter is performing debugging procedures on PolyMorpher. One aspect that Team Silver has to consider is debugging the numerous and vague scripts the player will enter in order to make their code compile. Thus, Team Silver has to strategically find ways to cause errors and glitches within PolyMorpher as a form to debug it and fine potential mistakes in the design. Thus, this presents itself as a major challenge for Team Silver.

4.3.3 Playtesting

Playtesting is an important aspect that Team Silver has to take into consideration and it also presents itself as a potential challenge. It requires to test if the main components of PolyMopher are working efficiently. Those main components that are required to work are if the compilation of code from the player is assembled correctly, the API Book is providing all the references that the player needs, and if the TUI is helping the player navigate the levels properly.

4.3.4 Maintaining player engagement

Keeping the player engaged and entertained is another major challenge that Team Silver is faced when designing PolyMorpher. That is main problem that Team Silver wants to solve when teaching OOP concepts and problem solving skills. Thus, making PolyMorpher as the alternative to the traditional educational methods when it comes to teach programming can be accomplished if the player is heavily engaged.

4.3.5 Teaching enough

PolyMorpher will be designed to teach the main concepts of OOP and problem solving skills. However, in programming there are a lot of other important concepts that can be

beneficial to the player. Thus, Team Silver is faced with challenge that if teaching enough is only sufficient with OOP and problem solving or should they go beyond these two concepts.

4.4 Risk Matrix/Risk Mitigation

The risk mitigation and risk matrix for PolyMorpher is designed to demonstrate the three main scenarios of impact and probability risks under two important categories. Those two important categories are Customer and Technical risks. The matrix itself as it will be illustrated below, will be divided into three colored sections. The green section is at the lower bottom part of the matrix and it illustrates a low impact and probability risk. The middle yellow part that extends from the upper left side to the lower right bottom side shows the medium impact and probability risks. Lastly, the upper left side of the matrix which is red shows high impact and probability risks. The Customer Risk for PolyMorpher are: User Gets Lost, Dissatisfied User, and Insufficient Content/Time. In addition The Technical Risks for PolyMorpher are: User Implements Bad Code, Insufficient Hardware, Critical Software Bugs, and Insufficient API Support. Table 3 illustrates the Risk/Mitigation matrix for PolyMorpher.

[This space intentionally left blank]

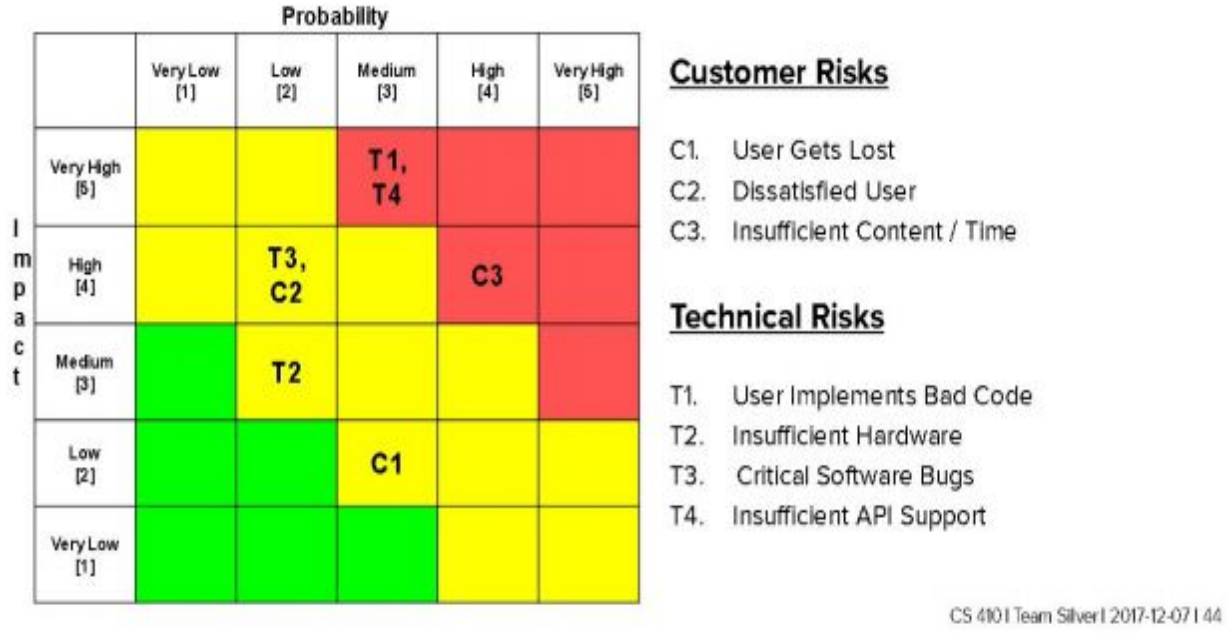


Table 3: Risk/Mitigation Matrix of Customer & Technical Risks for PolyMorpher

4.4.1 Technical risks and mitigations

The four potential Technical risks that were mentioned previously are between the median and high impact and risk probability portions of the matrix. The user implements bad code risk is considered a high level impact and medium probability. The reason behind this mitigation is if the user implements malicious code that comes from a uncredible forum, it could potentially crash PolyMorpher. In addition, the player would not learn the programming concepts taught by the API Book appropriately because they are referencing other websites. The second technical risk for PolyMorpher deals with there being insufficient hardware for the game itself. Team Silver considered this risk to be a medium impact and low probability risk. The mitigation for this technical risk is that PolyMorpher will be implemented as 2-dimensional game allowing it not to draw much power and resources from the player PC In order to use and play

PolyMorpher, just downloading an executable file from Team Silver's website is required. In addition, PolyMorpher is single player and requires no constant internet connection. It also draws minimal power and resources from the player's computer.

The third technical risk for PolyMorpher deals with critical software bugs within the game which is a high impact and low probability risk. The mitigation for this risk is to test and make sure that the game is fully optimal and functioning at the expectations that PolyMorpher is set to perform. Critical bugs and glitches that could occur in the gameplay are important aspects that Team silver will need to look at while testing and performing implementation on PolyMorpher. The last and fourth technical risk for PolyMorpher deals with insufficient API support. It has a very high impact and medium risk potential. The mitigation for this risk is that if the API that is built within PolyMorpher does not provide all the sufficient tools that are required for player to effectively learn the OOP concepts, then they cannot use the appropriate code syntax to solve puzzles and challenges as they progress in the game. In addition, if the player feels that they don't find the supporting material from the API when they are stuck in a puzzle, then that can have a very negative impact in PolyMopher as a teaching game. This could question PolyMorpher's credibility.

4.4.1 Customer risks and mitigations

The three main customer risks and mitigations are mainly within the medium section of impact and risk probability portions of the referred matrix. The first customer risk deals with risk of the user/player getting lost or stuck in solving the puzzles and challenges are in a particular level. It has a low impact and medium probability risk. The mitigation for this risk deals with the user struggling to solve puzzles and challenges while playing PolyMorpher and they get lost or

give up. The players who can be affected by this risk either have low level or no programming experience.

The customer risk that Team Silver has determined PolyMorpher could face is if the user/player is dissatisfied particularly with GUI and UI/UX design and implementation. It has a high impact and low probability risk. The mitigation for this risk is that Team Silver can not possibly accommodate for all the possible types of GUI and UI/UX designs that are available for PolyMorpher to implement. However, Team Silver will seek a way to design and implement a GUI and UI/UX design that can satisfy the majority of its users.

The third and final customer risk for PolyMorpher deals with the insufficient content and time that PolyMorpher will be able to teach to its players. This risk has both high impact and probability potential. The mitigation of this risk is that PolyMorpher will have sufficient content to teach that will make the player gain skills and confidence in programming. In addition, Team Silver also took into consideration the time constraints for this customer risk to determine if the player will have the sufficient time to get through the game puzzles and challenges as well as to learn the OOP concepts all at once.

5 Development Pipeline

The tools that will be needed to successfully develop PolyMorpher include the following: Unity SDK software, Gitlab, SourceTree, MonoDevelop IDE that comes integrated with Unity, and C# programming language along with its compiler from Microsoft and its associated third-party libraries. In addition, the software and system requirements to officially run and use PolyMorpher as a game are either any modern operating system like Windows 7 and beyond, MacOS, or various versions of Linux distributions..

5.1 Unity Software

The Unity SDK software, as Team Silver states, is a popular gaming engine that has the sufficient development built-in tools to construct PolyMorpher as game. In addition, Unity has many premade objects and environments that can be useful for developing the GUI and UI/UX design for PolyMorpher. There are two major tools that Unity contains that will be used heavily to build PolyMorpher. They are the C# programming language and the MonoDevelop IDE.

C# is the programming language that Unity uses for game development. Thus, Team Silver will use C# to build PolyMorpher and will be labeled as its source code. C# was developed by Microsoft and so Unity requires their compiler to used as well so its already integrated in their engine platform. The MonoDevelop IDE is the code development tool that comes with Unity and Team Silver will instruct its developers to use it for the coding process. Visual studios from Microsoft is the IDE for C# and can be used with unity. However, Team Silver decided to use MonoDevelop instead because its an integrated feature of Unity.

5.2 SourceTree Software

SourceTree will be one the main essential tools that will be used coordinate and manage the version control of the development for PolyMorpher. It is a git client mainly for Windows and MacOS. In addition, it uses PuTTY to connect and perform an interface with GitLab which Team Silver will use it to manage their repositories. PuTTY is a popular pageant client that SourceTree uses to connect and communicate with GitLab remotely through an internet connection.

5.3 Version Control

In order to develop PolyMorpher successfully using version control, it will require for SourceTree, Gitlab, Unity, and PuTTY to connect and communicate simultaneously. This is so that the developers from Team Silver can share their work and progress during the development phase of PolyMorpher. The developers will use Unity to perform the coding process with a copy of the project file saved in their local repository. If the developer makes changes to their work in the source code on the project file so that other developers can use it to continue their work in the main Team Silver repository, the developer must update and commit those changes in SourceTree. SourceTree will then use PuTTY to securely connect and commit such changes to the Gitlab repository in Team Silver's master branch. Once PuTTY is able to connect and commit the developer's changes from SourceTree, those changes will be seen by all the other developers in the master branch.

On the other hand, if the developer need to work on the most updated source code for PolyMorpher, they need to clone the master branch from the Gitlab repository to their local repository using SourceTree. The connection required to securely access the master branch in Gitlab will be handled by PuTTY. The master branch is where the most updated versions of the source code are that were worked on by all the developers in Team Silver. In addition, the master branch will contain the source code for the final product of PolyMorpher. The dataflow of Figure 8 illustrates a complete and detailed diagram of version control for PolyMorpher.

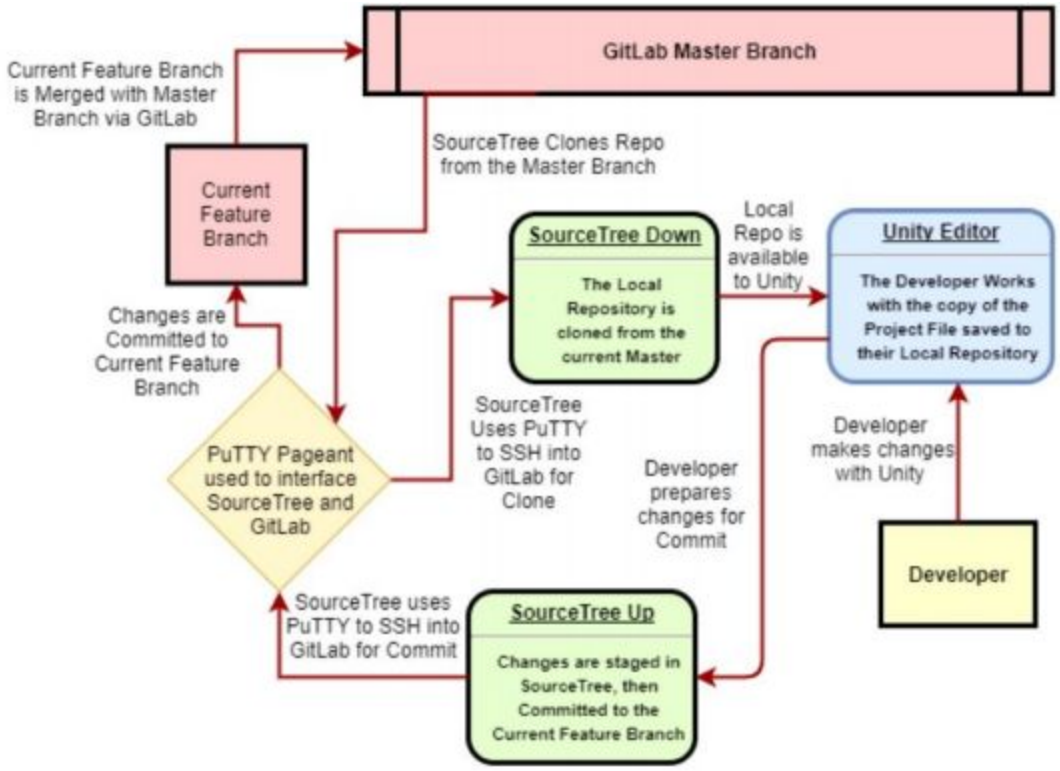


Figure 8: Dataflow Diagram for Version Control

5.4 Development Model

Agile was the development model that was chosen by Team Silver to develop PolyMorpher. This Agile Development Model consists of the developers in Team Silver to take the initiative and to develop, test, and perform a demonstration of the PolyMorpher prototype. Figure 9 illustrates the Agile Development Model cycles for detailed reference.

If Team Silver is contented with the developer's prototype, then they decide to whether they built on such prototype to make the final product of PolyMorpher. Otherwise, if the team decides the developer's prototype model does not meet their design perspective, they ask for another developer to conduct the same Agile Development process as explained before. The Kickoff

phase, as shown in Figure 8, is where the Agile Development process begins and repeats itself for each developer that presents their prototype version of PolyMorpher.

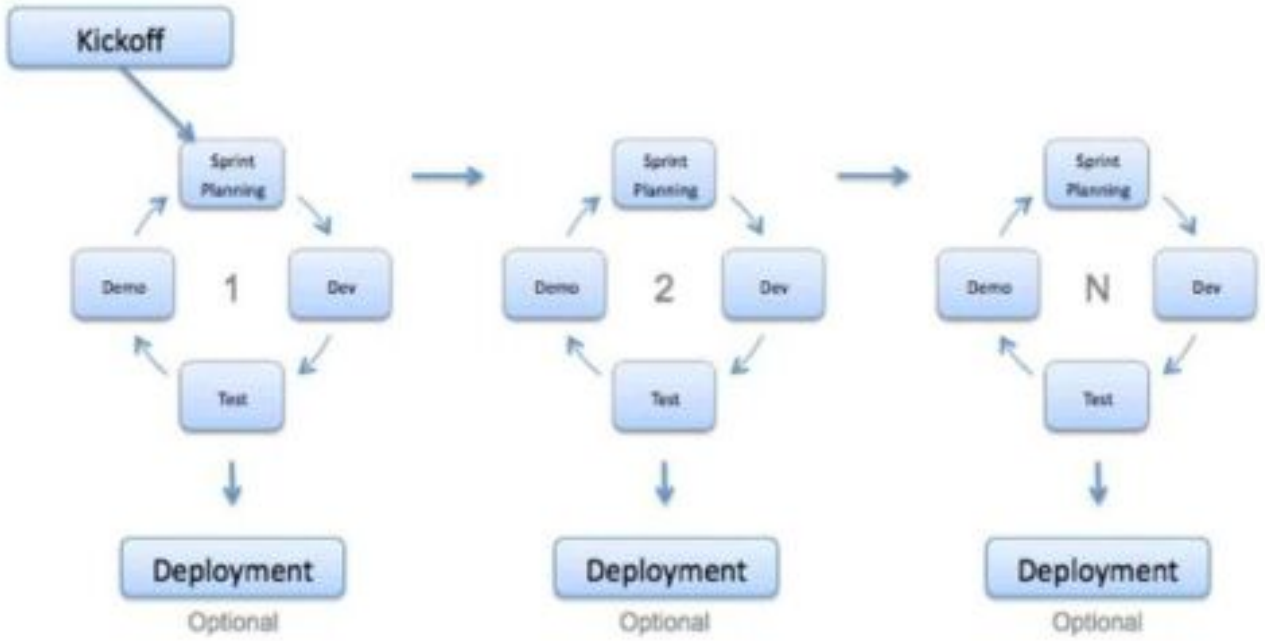


Figure 9: Dataflow Diagram for Agile Development Model

5.5 Work Management

The work management for the development of PolyMorpher will consist of Team Silver dividing their developers into three separate groups. Each group will be provided with two OOP concepts and they will be given the task to design a puzzle and or challenge for a particular level. They have to design such puzzle and challenge to effectively teach their assigned OOP concepts to the player when they play this game. Once all three groups successfully complete their tasks, then all the levels will be properly allocated in PolyMorpher. The combination of the tasks

completed by each group will constitute in the final deployment product of PolyMorpher as a game.

Glossary

API: Application Program Interface

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

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

IPv6: Modern date protocol used to connect to the internet

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

PC: Personal Computer that will be used to describe the player computer/machine

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: 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

References

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:%221086920031335905830 47%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:%221086920031335905830 47%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:%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

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

“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

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

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/

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