Project 1 Write Up

Team N7: Taylor Tamblin, Zack Breitenstein, Cody Curry

Description

This paper will outline the various parts that make up Project 1. This paragraph features brief descriptions of the contents of this paper.

GUI Storyboards will display the screenshots of our Graphical User Interface (GUI) through its many transitions. User Narratives will present potential users and the way they might use the ASML program. Formal Use Cases will outline the various use cases of the program. UML Class Diagrams will present the UML class diagrams that make up our project. Design Patterns Used will explain which software design patterns were used in the implementation of the program. Design Considerations will discuss various design decisions made during the development of this application. Issues will present screenshots of our team's Github repository issue tracking section.

User Narratives

Instructor Lamarche – Instructor Lamarche is going to grade this project. He will assess the GUI for completeness, test the GUI's functionality, and read the source code.

Testers (Cody, Zack, Taylor) – The testers will be the members from team N7. Testers will test the functionality of the GUI and the movement of the launcher. Testers care about ensuring the program works correctly and is free of bugs.

Formal Use Cases

ID	UC-1
Title:	Start Application
Description:	The application loads correctly when run
Primary Actor:	Instructor
Preconditions:	Executable file ran
Postconditions:	Program loads on computer and launcher is in idle mode
Main Success Scenario:	1. Executable is ran from user computer 2. Program starts up and GUI displays 3. Launcher is in idle mode
Extensions:	1. Program does not load correctly or crashes
Frequency of Use:	Using application
Owner:	Team N7
Priority:	P4 - Normal
Risk:	MITIGATION_06

ID	UC-2
Title:	Fire Missiles
Description:	Fire button pressed in the GUI and a missile is fired.
Primary Actor:	Instructor
Preconditions:	Software is in idle mode
Postconditions:	Missile is fired
Main Success Scenario:	1. Fire button on GUI is clicked 2. Launcher shoots a dart 3. Launcher goes back into idle mode
Extensions:	
Frequency of Use:	Testing
Owner:	Team N7
Priority:	P5 - Low
Risk:	MITIGATION_06

ID	UC-3
Title:	Launcher left movement
Description:	Clicking the left arrow button in the GUI causes the launcher to move left.
Primary Actor:	Instructor
Preconditions:	Software is in idle mode
Postconditions:	Launcher moves left
Main Success Scenario:	1. Left directional arrow on the GUI is clicked 2. The missile launcher turns left 3. The missile launcher stops moving (returns to idle mode)
Extensions:	1. Directional arrow is held down rather than single clicked 2. The launcher will continue movement while arrow is held down 3. The user lets off of the left mouse click on the arrow or launcher reaches maximal turning point 4. The launcher stops moving (returns to idle mode)
Frequency of Use:	Testing
Owner:	Team N7
Priority:	P2 - Low
Risk:	MITIGATION_11

ID	UC-4
Title:	Launcher up movement
Description:	Clicking the up arrow button in the GUI causes the launcher to move up.
Primary Actor:	Instructor
Preconditions:	Software is in idle mode
Postconditions:	Launcher moves up
Main Success Scenario:	1. Up directional arrow on the GUI is clicked 2. The missile launcher turns up 3. The missile launcher stops moving (returns to idle mode)
Extensions:	1. Directional arrow is held down rather than single clicked 2. The launcher will continue movement while arrow is held down 3. The user lets off of the left mouse click on the arrow or launcher reaches maximal turning point

	4. The launcher stops moving (returns to idle mode)
Frequency of Use:	Testing
Owner:	Team N7
Priority:	P2 – Low
Risk:	MITIGATION_11

ID	UC-5
Title:	Launcher right movement
Description:	Clicking the right arrow button in the GUI causes the launcher to move right.
Primary Actor:	Instructor
Preconditions:	Software is in idle mode
Postconditions:	Launcher moves right
Main Success Scenario:	1. Right directional arrow on the GUI is clicked 2. The missile launcher turns right 3. The missile launcher stops moving (returns to idle mode)
Extensions:	1. Directional arrow is held down rather than single clicked 2. The launcher will continue movement while arrow is held down 3. The user lets off of the left mouse click on the arrow or launcher reaches maximal turning point 4. The launcher stops moving (returns to idle mode)
Frequency of Use:	Testing
Owner:	Team N7
Priority:	P2 – Low
Risk:	MITIGATION_11

ID	UC-6
Title:	Launcher down movement
Description:	Clicking the down arrow button in the GUI causes the launcher to move down.
Primary Actor:	Instructor
Preconditions:	Software is in idle mode
Postconditions:	Launcher moves down
Main Success Scenario:	1. Down directional arrow on the GUI is clicked 2. The missile launcher turns down 3. The missile launcher stops moving (returns to idle mode)
Extensions:	1. Directional arrow is held down rather than single clicked 2. The launcher will continue movement while arrow is held down 3. The user lets off of the left mouse click on the arrow or launcher reaches maximal turning point 4. The launcher stops moving (returns to idle mode)
Frequency of Use:	Testing
Owner:	Team N7
Priority:	P2 – Low
Risk:	MITIGATION_11

ID	UC-7
Title:	Idle Mode

Description:	The launcher sits idle awaiting input from user
Primary Actor:	Instructor
Preconditions:	Application executable was ran or launcher is in search and destroy mode.
Postconditions:	Launcher enters idle mode and awaits further user input
Main Success Scenario:	1. The launcher is stationary and not currently firing a missile 2. The launcher is ready to accept further user input
Extensions:	
Frequency of Use:	Using application
Owner:	Team N7
Priority:	P4 - Normal
Risk:	MITIGATION_06

ID	UC-7
Title:	File Reader
Description:	File reader will read in a target description from .xml or .ini file
Primary Actor:	Instructor
Preconditions:	User selects .ini or .xml file to load
Postconditions:	File information loads into designated list box in GUI
Main Success Scenario:	1. User selects file to load into GUI from file explorer 2. File information is loaded into the list box as well as target list in program
Extensions:	1. Invalid file is selected (bad extension or bad file structure) 2. File is rejected 3. Nothing is loaded into the list box
Frequency of Use:	Using application
Owner:	Team N7
Priority:	P4 - Low
Risk:	MITIGATION_06

UML Class Diagrams

This section is broken up into Class Diagrams and Use Case Diagrams

Design Patterns Used

Adapter - We used the adapter pattern to adapt the IMissileLauncher interface with the Launcher object from missileLauncher.cs. This allows the Launcher to use the IMissileLauncher interface through the adapter.

Pros: Can override an adaptee's behavior as required, provides a way for interfaces to be used with something it wasn't made for (like a plug adapter for different outlets).

Cons: Changes to the interface will require changes to the adapter

Factory – We used the factory pattern for our file reader. The reason we used a factory pattern is because we need to be able to read several different files (.xml and .ini) and the factory will create a file reader based on the file type the program needs to read from.

Pros: Allows creation of objects of a class without knowing which class is needed

Cons: Extension of object may require special initialization,

Singleton – We used the singleton pattern for our target manager. The target manager has a list inside of it for managing target objects. The singleton ensures that we only instantiate one target manager, resulting in only one list of targets. If an ini reader or xml reader reads in new targets, the same instance of the target manager will be returned.

Pros: Can control how and when the singleton is accessed.

Cons: If multiple threads are being used, safe access to objects should be considered

Design Considerations

When working on the file reader, we found it best to use a factory design pattern because our project can accept multiple file types. A single way to create a reader was deemed best.

When working with the target class, we found that we only needed one target manager, which warranted the use of a singleton design pattern.

To load in .ini and .xml files we chose to use a configuration button. This configuration button opens up a separate window that opens a file dialog box which allows the user to select the files.

We chose to have a drop down for the webcam to be used with the program in case there are multiple webcams or video devices available.

Issues

Github issues attached. Link to our Github repository: https://github.com/N7Repository