Project 3 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.

Graphical User Interface (GUI) Storyboards will display the screenshots of our 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 Brian Lamarche – Instructor Lamarche will be the grader of the Automated Sentry GUI project. Lamarche wants a project that will build and function as specified, and utilizes appropriate design patterns. He will assess the GUI for completeness, test the GUI's functionality, test the launcher controls on the GUI, and read the source code.

Tester Zackary– Zackary Breitenstein will be the primary tester of the project. Zackary will test the functionality of the GUI and the movement of the launcher. The tester's aim is to ensure the program works correctly and is free of bugs. The tester cares mostly about code that builds, doesn't crash, and controls the launcher as expected.

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, either by running a command line or double left clicking the file from a GUI environment. 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:	Zackary
Priority:	P4 - Normal
Risk:	MITIGATION_06

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

ID	UC-3
Title:	Launcher left movement
Description:	Left 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 in the GUI is left clicked 2. The missile launcher turns left by a hardcoded number of degrees (small number) 3. The missile launcher stops moving (returns to idle mode)
Extensions:	1. Directional arrow is held down with the left mouse click rather than single clicked 2. The launcher will continue movement at a constant speed 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:	Zackary
Priority:	P2 - Low
Risk:	MITIGATION_11

ID	UC-4
Title:	Launcher up movement
Description:	Left 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 in the GUI is left clicked 2. The missile launcher turns up by a hardcoded number of degrees (small number) 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 at a constant speed 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:	Zackary
Priority:	P2 – Low
Risk:	MITIGATION_11

ID	UC-5
Title:	Launcher right movement
Description:	Left 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 in the GUI is left clicked 2. The missile launcher turns right by a hardcoded number of degrees (small number) 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:	Zackary
Priority:	P2 – Low
Risk:	MITIGATION_11

ID	UC-6
Title:	Launcher down movement
Description:	Left 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 in the GUI is left clicked 2. The missile launcher turns down by a hardcoded number of degrees (small number) 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 at a constant speed 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:	Zackary
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:	Zackary
Priority:	P4 – Normal
Risk:	MITIGATION_06

ID	UC-8
Title:	Select File Button
Description:	The Select File Button is used to find an .xml or .ini file for loading targets into the program.
Primary Actor:	Instructor
Preconditions:	The Select File button on the GUI is left clicked
Postconditions:	A file path is displayed in the TextBoxFilePath text box located to the right of the Select File Button.
Main Success Scenario:	1. The Select File button on the GUI is left clicked 2. Windows Explorer is brought up, providing the user a UI to select the desired .ini/.xml file. 3. The user selects the desired file 4. The file path is loaded into the TextBoxFilePath text box located to the right of the Select File Button
Extensions:	1. No file is selected using the Windows Explorer 2. Nothing is displayed in the TextBoxFilePath text box.
Frequency of Use:	Using application
Owner:	Zackary
Priority:	P4 – Low
Risk:	MITIGATION_06

ID	UC-9
Title:	Load Target Info Button
Description:	A window is brought up allowing the user to select and load a .ini or .xml file into

	the program
Primary Actor:	Instructor
Preconditions:	While the TextBoxFilePath text box is populated with a .ini or .xml file path, the Load Target Info Button on the GUI is left clicked
Postconditions:	File information loads into the targetInformation list box in the GUI
Main Success Scenario:	1. The TextBoxFilePath text box contains a .ini or .xml file path 2. The user left clicks the Load Target Info Button. 3. Target information is loaded and parsed from the specified file. 4. A target object is created for each target in the file and loaded into a Target Manager. 5. The target information is displayed into the targetInformation list box.
Extensions:	1. Invalid file is loaded from the Windows Explorer (bad extension or bad file structure) 2. The Load Target Info Button is left clicked. 3. File is rejected and nothing is loaded into the targetInformation list box 1. No file path is loaded into the TextBoxFilePath text box. 2. The Load Target Info Button is left clicked. 3. No information is loaded into the targetInformation list box or target list.
Frequency of Use:	Using application
Owner:	Zackary
Priority:	P4 - Low
Risk:	MITIGATION_06

ID	UC-10
Title:	Start Button
Description:	The launcher will enter search and destroy mode
Primary Actor:	Instructor
Preconditions:	The start button in the GUI is left clicked
Postconditions:	
Main Success Scenario:	1. The start button on the GUI is left clicked 2. The missile launcher resets to a neutral position and the GUI controls are locked from user 3. A search and destroy controller is created for the launcher (if not preexisting) 4. A timer on the GUI is started 5. The controller uses destroy mode to filter the list of targets based on the mode selected by the user. The filtered list will consist of either only friendly targets, only enemy targets, or all targets. 6. The missile launcher aims and fires at each target in the list 7. Once the list of targets has been run through completely, the missile launcher resets 8. The timer on the GUI is stopped provided three minutes has not been reached 9. Search and destroy mode is exited and the program awaits further input from the user
Extensions:	1. Timer reaches three minutes while search and destroy is running 2. Search and destroy and timer both stop and turret resets 1. User left clicks the Stop button during search and destroy 2. Search and destroy and timer stop and turret resets.

	1. Search and destroy is started with no targets to destroy under specified mode 2. Turret resets and search and destroy mode is exited
Frequency of Use:	Using application
Owner:	Zackary
Priority:	P4 - Low
Risk:	MITIGATION_06

ID	UC-11
Title:	Stop Button
Description:	While search and destroy mode is running, search and destroy mode is ended
Primary Actor:	Instructor
Preconditions:	The program is in search and destroy mode
Postconditions:	The program is no longer in search and destroy mode and awaits further user input
Main Success Scenario:	1. The Stop button on the GUI is left clicked 2. Search and destroy mode is exited and the GUI timer stops
Extensions:	1. The program is in idle mode 2. The Stop button is left clicked 3. No action is taken and the program awaits further user input
Frequency of Use:	Using application
Owner:	Zackary
Priority:	P4 - Low
Risk:	MITIGATION_06

ID	UC-12
Title:	Start Video Button
Description:	Start
Primary Actor:	Instructor
Preconditions:	The GUI is loaded without the video feed displaying
Postconditions:	The video feed is streaming on the GUI
Main Success Scenario:	1. The Start Video button on the GUI is left clicked 2. The video feed streams onto the upper left side of the GUI
Extensions:	1. The program has the video displaying on the GUI 2. The Start Video button is left clicked 3. No action is taken and the program resumes operation
Frequency of Use:	Using application
Owner:	Zackary
Priority:	P4 - Low
Risk:	MITIGATION_06

ID	UC-13
Title:	Stop Video Button
Description:	The video feed on the GUI stops streaming
Primary Actor:	Instructor
Preconditions:	The video feed is currently streaming on the GUI
Postconditions:	The video feed stops streaming and freezes on the last frame captured

Main Success Scenario:	1. The video feed is currently streaming on the GUI 2. The Stop Video button on the GUI is left clicked 2. The video feed stops streaming onto the GUI and freezes on the last frame captured
Extensions:	1. The program is currently not streaming video onto the GUI 2. The Stop Video button is left clicked 3. No action is taken and the program awaits further user input
Frequency of Use:	Using application
Owner:	Zackary
Priority:	P4 - Low
Risk:	MITIGATION_06

ID	UC-14
Title:	Mode Selection Combo Box
Description:	The user selects the desired search and destroy mode using the comboBoxSelectedMode combo box on the right side of the GUI
Primary Actor:	Instructor
Preconditions:	The program is in idle mode
Postconditions:	The desired mode is selected by the user
Main Success Scenario:	1. The program is in idle mode with the combo box set with the default option of "Destroy Only Enemies". 2. The user left clicks the comboBoxSelectedMode combo box and three choices are available in the drop down box: Destroy Only Enemies, Destroy Only Friends, and Destroy All Targets 3. The user left clicks the desired option 4. The program is ready to destroy targets in search and destroy mode based on the desired selection
Extensions:	1. The program is in idle mode 2. The user left clicks the start button to enter search and destroy mode without choosing a mode option from the comboBoxSelectedMode combo box 3. The program starts search and destroy mode with a default mode of "Destroy Only Enemies".
Frequency of Use:	Using application
Owner:	Zackary
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: Adds a level of complexity to code that can potentially make debugging more difficult.

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

Observer – We used an observer pattern in generating a logfile of targets. Our program's target manager notifies our GUI of how many targets were read in from a .ini or .xml file, and calls a function in our GUI which calls our logfile class. The logfile class then generates a text based logfile based on the list of targets that was passed into it.

Pros: Good pattern for sending data to multiple objects

Cons: System performance can degrade with a large number of observers

Strategy – We used the strategy pattern in determining the correct logic to use for the missile launcher based on what mode the user selects for the turret from the comboBoxSelectedMode combo box. We use the strategy pattern with our SearchAndDestroyFactory class. The SearchAndDestroyFactory class then creates a DestroyMode class depending on which string was passed in from the comboBoxSelectedMode combo box. Destroy Mode is the abstract class from which the three concrete DestroyMode classes inherit from. The concrete DestroyMode class filters the list of targets based on the mode selected by the user for use with the missile launcher.

Pros: Can determine the different logic implementation needed for a certain problem without knowing what the context will be.

Cons: The strategy pattern can't be handed all of the data or memory could run out

Mediator – We used a mediator pattern with our SnDController. The SnDController mediator receives information from our TargetManager which is then used in directing the missile launcher where to point and fire.

Pros: Modularizes objects that do not communicate to each other

Cons: The Mediator encapsulates protocols, making it likely to become more complex than any individual colleague. This can make the mediator difficult to maintain.

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 on the GUI. This configuration button opens up a separate window which allows the user to select the files via Windows Explorer and load them into the program through an open file button.

We realized that for our GUI and missile launcher to work in tandem, threading was necessary. We threaded both our GUI and our missile launcher so the GUI could be manipulated during autonomous operation of the missile launcher. However, this also posed a few problems of its own.

Most of the buttons on the GUI ended up interfering with normal launcher operation while in search and destroy mode. In order to protect the user and data within the launcher, such as current launcher coordinates, we disabled all buttons on the GUI after entering search and destroy mode except for the stop button, which will stop search and destroy mode.

Issues

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