Release Process checklist  
 1. Planning  
 a. Identify a task to complete for the release  
 b. Assign a release number (starting at version 1.0.0)  
 c. Enter tickets for use cases in question as tasks into the GitHub repo.  
 i. Identify a definition of done for each task to validate completion.  
 d. Identify a release date as set by course schedule and attach the Due Date to the GitHub milestone as well.

1. Documentation  
   a. Update use cases with new features to be added  
   b. Update the README if the version is going to be deployed  
   c. Update UML and Source Code documentation as prep for final release
2. Implement  
   a. Add ticket (or update progress, depending on ticket task)  
   b. Revise Ticket with information as to what is happening with this task  
   c. Resolve ticket as tasks are completed
3. Testing  
   a. Verify that each individual portion of the system relating to the new task works as intended  
   b. Verify that the whole task works as a system as well, that each part of the task works in synergy with the others to solve the task in question  
   c. Verify that tutorials were updated for the new tasks  
   d. Verify that any documentation relating to the new tasks were updated as needed as well  
   e. Verify each ticket for the milestone has been resolved as intended.
4. Release  
   a. Build an installer for the software so it can move to other platforms  
   b. Beta-Test with sample user

i. Add ticket for defects in documentation and software  
ii. Determine if defects can be rectified prior to release date

c. If it passes verification, close milestone as well as all tickets that were attached to milestone.

UML  
Class diagrams (From earlier part)

Initial Sequence Diagrams names and Story Boards  
a. Initialization of program

* Program start through .exe or cl
* Connected ML calibrates
* Program waits at GUI for input

b. Selection of Missile Launcher

* Delete the older missile launcher instance
* Construct a new instance with the selected Missile Launcher type
* Calibrate new missile launcher.

c. Starting Live Video

* Delete any old instances
* Construct new camera object based on what’s plugged in
* Create a new thread to get the video
* Pass the video thread to the GUI

d. Contacting live

* Delete old info(if any) from target list and game settings
* Ask server (or wherever) for game/target info
* Update target info
* Create singleton object with the new target information

e. Starting game

* Delete any info from previous runs
* Query the game information to know the game type
* Proceed with game run as determined by the game type

f. Stopping game

* Stop the timer
* Tell the missile launcher to stop moving completely
* KEEP THE INFO, in case need to restart
* Pause the live video gathering (not sure if needed, but can’t hurt)

g. Reloading the Launcher

* Stop any activity that the ML is doing
* Keep timer going
* Update missile count
* Have button on GUI to resume

Initial Activity Diagram Name

Initializing the program