*Florida International University*

*School of Computing and Information Sciences*

CIS 4911 - Senior Capstone Project

Software Engineering Focus

Feature Document

User Story #669

**Team Member:**

Andrew Mitchell

**Product Owner(s)**:

Francisco R. Ortega

**Mentor(s)**:

Francisco R. Ortega

...

**Instructor**: Masoud Sadjadi

**User Story –** Implement Chrono for Device Handler

* As a User I would like a smooth and lag-less experience even when the program checks for device connection so I can enjoy the drawing experience.

Acceptance Criteria

* Implement a Chrono timer on our device connection.
* Do testing to find out how long a delay on the Chrono is best for performance vs user noticing a device is not connected right away.

**Use Case**

Use Case

User launches application. Chrono check should be performed on every update.

Details:

Actor: User

Pre-conditions:

Program must be running.

Description:

Use case begins when the user launches the application. After the application is launched it should be checking for device connection rapidly, but not impact performance.

Post-conditions:

The program should be able to tell if a device is unplugged. It should not impact drawing performance by slowing it down and making lines ‘edgier’.

Decision Support:

Frequency: Always. The feature will always be running during the programs lifetime.

Criticality: Very High. Device connection detection is very important so the user understands what devices are connected and working.

Risk: Low. Adding a chrono timer is an easy task. The product owner just wants sufficient testing to ensure that the program runs smoothly as well as detects devices frequently and so there isn’t a very noticeable delay for the user.

Reliability: Highly.

Mean time to Failure – Almost Never. Device Detection should always occur within the allotted time frame. The device should always disconnect if the USB device is removed. The device will not always connect however if the user has already done ‘override’ mode functions.

Availability – Always running in the background.

Performance:

Should have very lower performance impact.

Supportability:

Entire Program and all devices.

Supported by LibUSB – version 1.0

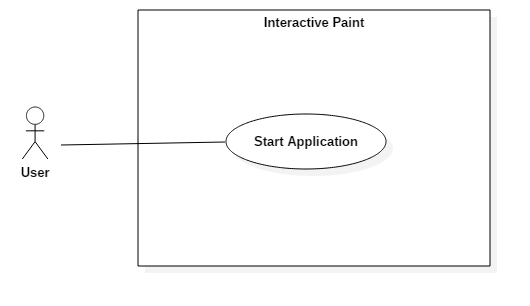
Modification History:

Owner: Andrew Mitchell

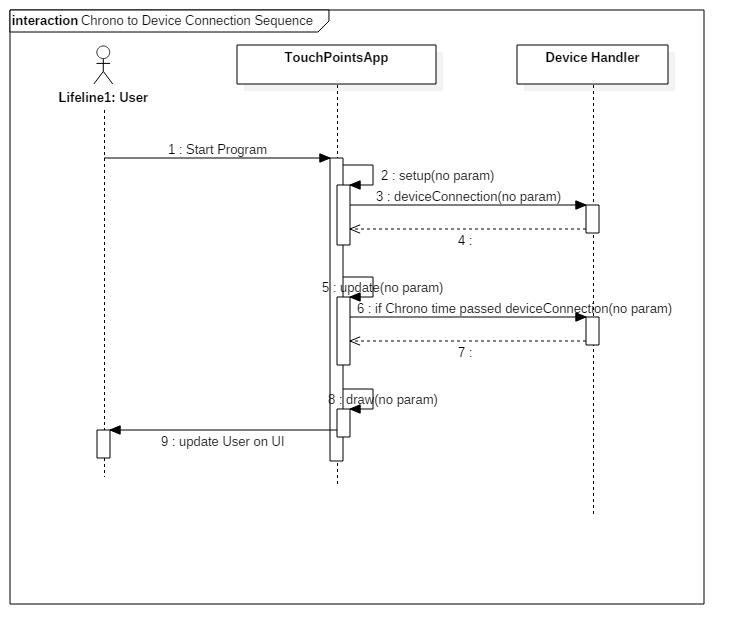
Initiation date: 4/4/2016

Date last modified: 4/4/2016

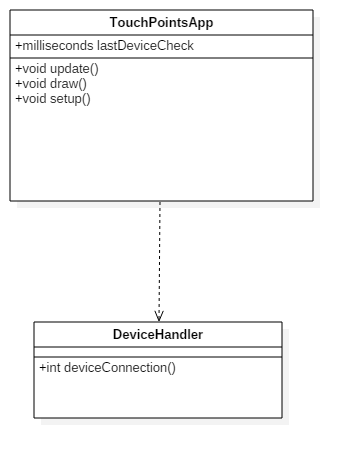
**Use Case Diagram**



**Sequence Diagram**



**Class Diagram**



**Unit Test**

Sunny Day:

Test Case: Leap Single Line Draw

Test Purpose:

Ensure the program runs smoothly while drawing a single line with leap inputs, as well as testing device disconnect and connect.

Test Setup

1. Start Program
2. Open Device Modes menu
3. Plug in Leap Motion wait ‘x’ seconds
4. Unplug Leap Motion wait ‘x’ seconds
5. Plug in leap motion
6. Enable Leap Draw
7. Attempt to draw a single line with leap motion.

Test Output:

Devices statuses were properly updated.

Line was drawn smoothly

Expected Output:

Plugging in the leap motion should toggle on the ‘Leap Motion Device’

Unplugging it should turn off the device

Plugging it back in should turn off the device.

Drawing should be fairly smooth and fast (But ‘Jumpy’ because leap motion input is not as precise).

Test Case: Multitouch Single Line Draw

Test Purpose:

Ensure the program runs smoothly while drawing a single line with multitouch inputs, as well as testing device disconnect and connect.

Test Setup

1. Start Program
2. Open Device Modes menu
3. Plug in Multitouch wait ‘x’ seconds
4. Unplug Multitouch wait ‘x’ seconds
5. Plug in Multitouch
6. Attempt to draw a single line with multitouch.

Test Output:

Devices statuses were properly updated.

Line was drawn smoothly.

Expected Output:

Plugging in the Multitouch should toggle on the ‘Multitouch’

Unplugging it should turn off the device

Plugging it back in should turn on the device (green box).

Drawing should be smooth and fast.

Rainy Day Tests:

Test Case: Rapid Circle Drawing Leap

Test Purpose:

Make sure circles are still drawn very smooth (Smoothly grow larger and smaller) and do not seem to make large leaps in radius changes using leap motion.

Test Setup:

1. Make sure leap device is connected and ‘leap draw’ is enabled.
2. Change to circle mode
3. Change to filled shapes
4. Move 5 fingers into the draw zone
5. Move around the hand
6. Leave the draw zone

Test Output:

Drew circles. No huge jumps in radius between frames. Growth and shrinking of circles were smooth.

Expected Output:

Circles should expand and retract smoothly without huge jumps in radius between each re-draw.

Test Case: Rapid Circle Drawing Multitouch

Test Purpose:

Make sure circles are still drawn very smooth (Smoothly grow larger and smaller) and do not seem to make large leaps in radius changes using multitouch.

Test Setup:

1. Make sure multitouch is connected.
2. Change to circle mode
3. Change to filled shapes
4. Place 5 fingers on the touch screen.
5. Move around your fingers on the screen.
6. Release your fingers.

Test Output:

Drew circles. No huge jumps in radius between frames. Growth and shrinking of circles were smooth.

Expected Output:

Circles should expand and retract smoothly without huge jumps in radius between each re-draw.

**Integration Test**

Currently improved the performance of our entire program.

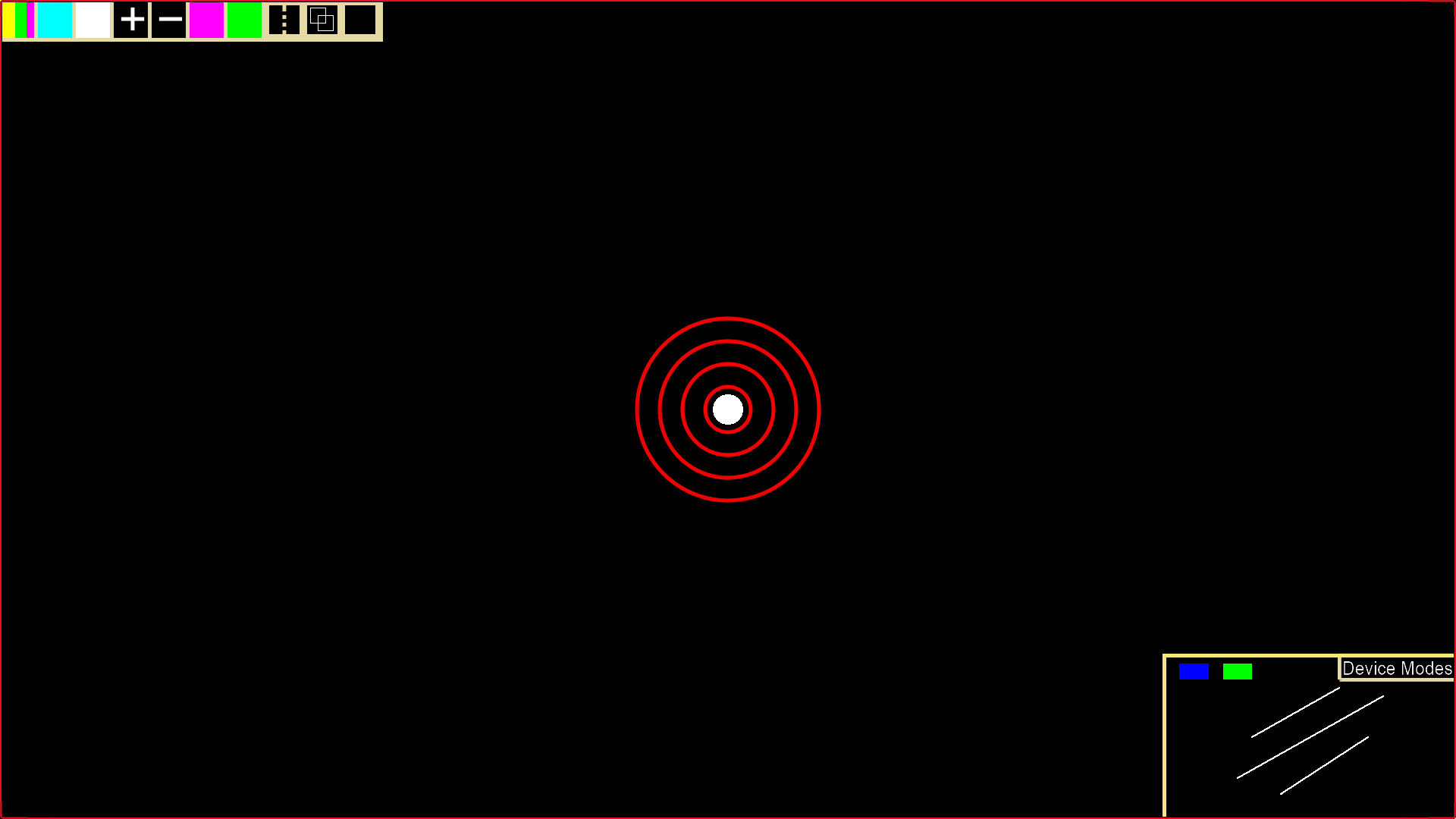
Actively increased the frame rate of the idle program from 14~ FPS to 45~ FPS.

Made the use of the leap motion as well as Real Sense device easier because the improved framerate.

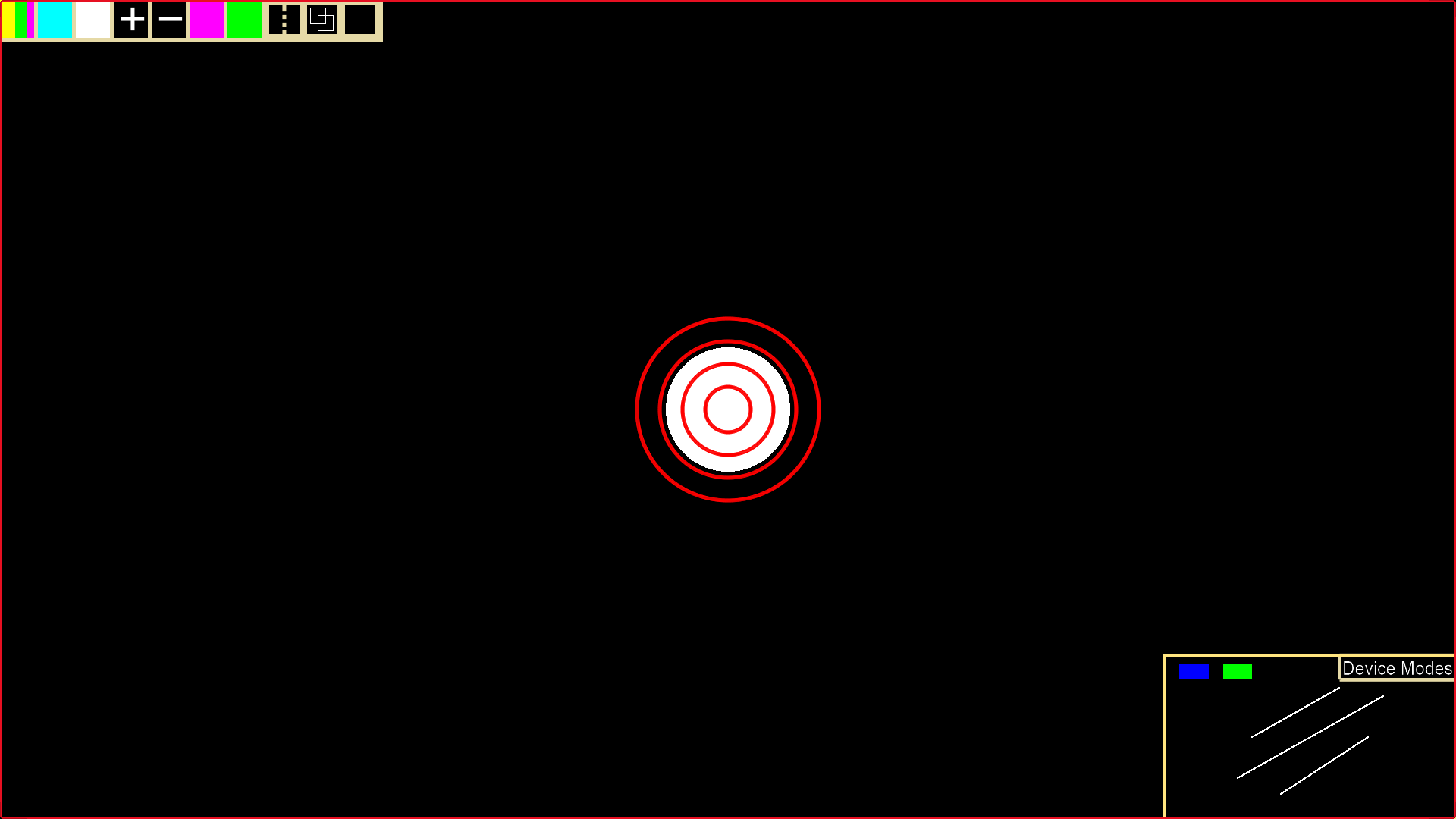
**User Guide**

I wrote a small script which showed our frames per second using circles as visual indicator. The smallest red circle being 15 FPS, then 30, then 45, then 60 FPS. The white circle indicates our Frames that occurred in the last second.

Before adding the chrono hold to our device connection we were getting under 15 FPS with the program just running.



After implementing a Chrono hold (checking only every 1 second) we improved our FPS to almost 45 (Consistently above 40~) when the program is idle.



**Glossary**

FPS – Frames per second – how many frames the program displays every second.

Chrono – a C++ Library that helps you interact with the system clock