*Florida International University*

*School of Computing and Information Sciences*

Software Engineering Focus

Feature Document

User Story ID #711

**Name:** Francisco Lozada

**Team Member(s):** Lukas Borges, Filip Klepsa, Nicolette Celli, Francisco Lozada, Cristian Cabrera

**Project:** AR-VR-VE for Computer Science (Circular Gesture Recognition API)

**Product Owner(s)**: Francisco Ortega

**Mentor(s)**: Francisco Ortega

**Instructor**: Francisco Ortega, Masoud Sadjadi

**User Story Name:** Translate the Point Class to C++

* Description: **As a** developer **I would like** to translate the Point class from the MTGRLibrary to C++ **so that** it can be used within the PointMap class for the API

**Acceptance Criteria**

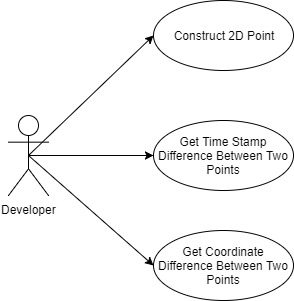
* Must have C++ API best design practices implemented
* Must have the required attributes and member functions that provide the exact translation of the class written in C# to C++ code

**Use Case**

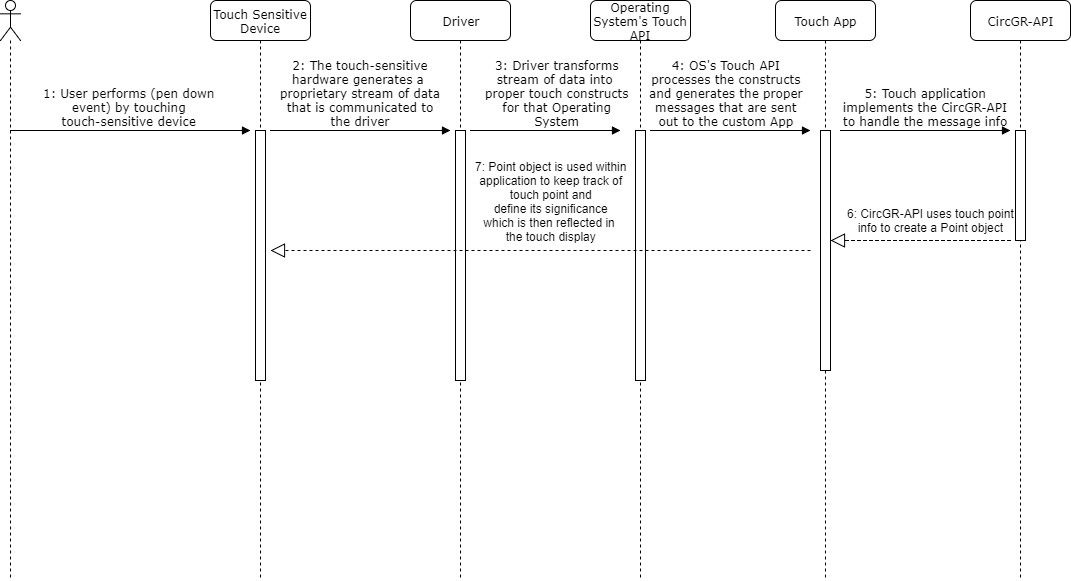
* Name: Construct 2D Point
* Actor: Developer
* Preconditions: Have the CircGR-API library (Point.cpp file in this case) and #include “Point.h” header within the code
* Description:
  + Developer uses the Point class to store a touch point’s properties
    - * System instantiates a Point object whenever it is called upon and provided the right parameters: X, Y, StrokeID, and Timestamp

* Name: Get TimeStamp Difference Between Two Points
* Actor: Developer
* Preconditions: Have the CircGR-API library (Point.cpp file in this case) and #include “Point.h” header within the code
* Description:
  + Developer instantiates two Point objects within the code
    - * System instantiates a Point object whenever it is called upon and provided the right parameters: X, Y, StrokeID, and Timestamp
  + Developer uses the TimeDifference function by passing it two Point objects
    - System calculates the difference between the timestamps of both point objects. In other words, it calculates the amount of time that passed from when one point was created to when the next was instantiated
* Name: Get Coordinate Difference Between Two Points
* Actor: Developer
* Preconditions: Have the CircGR-API library (Point.cpp file in this case) and #include “Point.h” header within the code
* Description:
  + Developer instantiates two Point objects within the code
    - * System instantiates a Point object whenever it is called upon and provided the right parameters: X, Y, StrokeID, and Timestamp
  + Developer uses the minus sign operator on the two Point objects
    - System calculates the difference between both points’ X and Y coordinates respectively. This, in essence, determines the spatial distance between the two points.

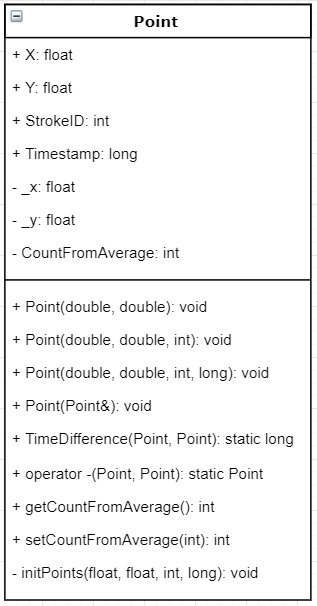
**Use Case Diagram**



**Sequence Diagram**



**Class Diagram**



**Unit Test**

* Test case ID: Construct\_Point
* Description/Summary of Test: Test whether a Point object that stores relevant touch point info is properly created
* Pre-condition: Have the CircGR-API library (Point.cpp file in this case) and #include “Point.h” header within the code
* Expected Results: Point object which contains X and Y coordinates, Stroke ID, and Timestamp information is created
* Actual Result: same as expected results
* Status (Fail/Pass): Pass
* Test case ID: Get\_Difference\_in\_Timestamp
* Description/Summary of Test: Tests whether the difference in creation time between two Point objects based on their Timestamp attribute is properly calculated
* Pre-condition: Have the CircGR-API library (Point.cpp file in this case) and #include “Point.h” header within the code
* Expected Results: Difference in creation time between two points is returned
* Actual Result: same as expected results
* Status (Fail/Pass): Pass
* Test case ID: Get\_Difference\_in\_Coordinates
* Description/Summary of Test: Tests whether the spatial difference between two Point objects based on their respective X and Y coordinates is properly calculated
* Pre-condition: Have the CircGR-API library (Point.cpp file in this case) and #include “Point.h” header within the code
* Expected Results: Spatial separation between two points is returned
* Actual Result: same as expected results
* Status (Fail/Pass): Pass

**Integration Test**

Since the Point class is the lowest level class in terms of dependencies, that is, it is the class with no other class dependencies there was no integration testing. Unit testing for this class sufficed. It isn’t until I translated the PointMap class for user story #172 that I needed to do integration testing since the PointMap class has a composition relationship with the Point class created in this user story.

**Visual User Guide**

