*Florida International University*

*School of Computing and Information Sciences*

Software Engineering Focus

Feature Document

User Story ID #758 Generic Empty Device Implementation

**Name: Bernardo Pla**

**Team Member(s):** Pachev Joseph, Hamilton Chevez, Daniel Rivero, Daniel Khawand

**Project:** WEBVR 1.0

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

**Mentor(s)**: Francisco Ortega

**Instructor**: Masoud Sadjadi

**User Story Name: #758 Generic Empty Device Implementation**

* As a Developer, I want to create a class in rust to handle generic or “unsupported” devices, so that the library can support as many devices as possible.

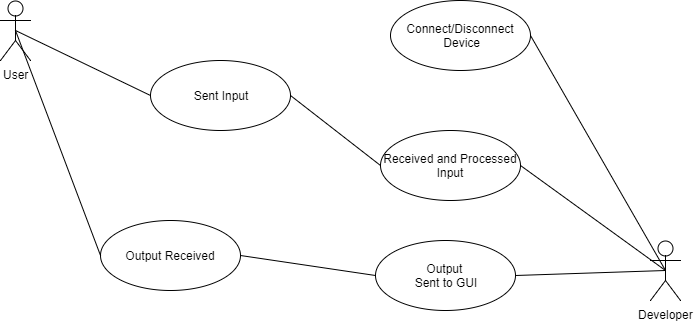
Acceptance Criteria

* Empty Device fulfills the 7-tuple device requirement
* Demo is accepted by the Product owner.

**Use Case**

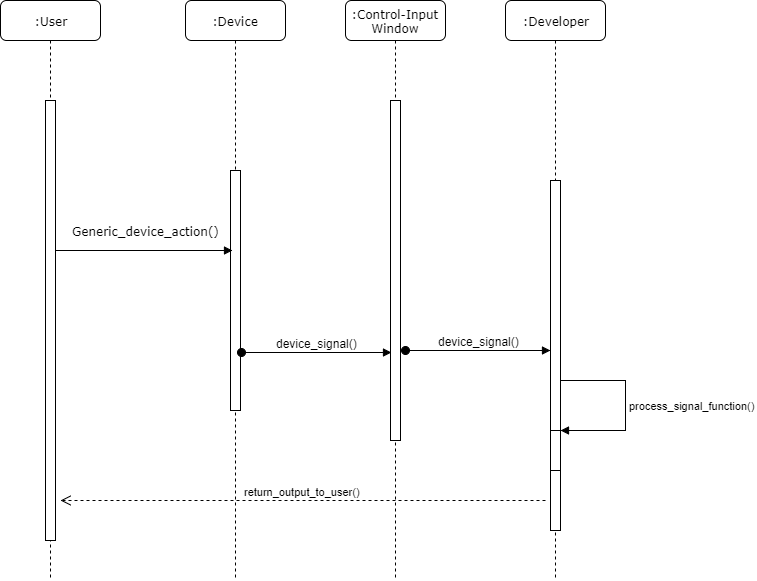
* Name: Generic Empty Device
* Actor: Device Controller, User, Developer
* Preconditions:
  + WebVr-input window is running
  + Device connected is not supported by the current rust library (ie, device is not considered a mouse, gamepad, VR-Headset, etc).
* Description:
  + Developer connects generic device
  + Developer adds generic device
  + Developer determines inputs necessary for processing
  + User provides actions on generic device
  + Output is sent to developer for processing.

**Use Case Diagram**

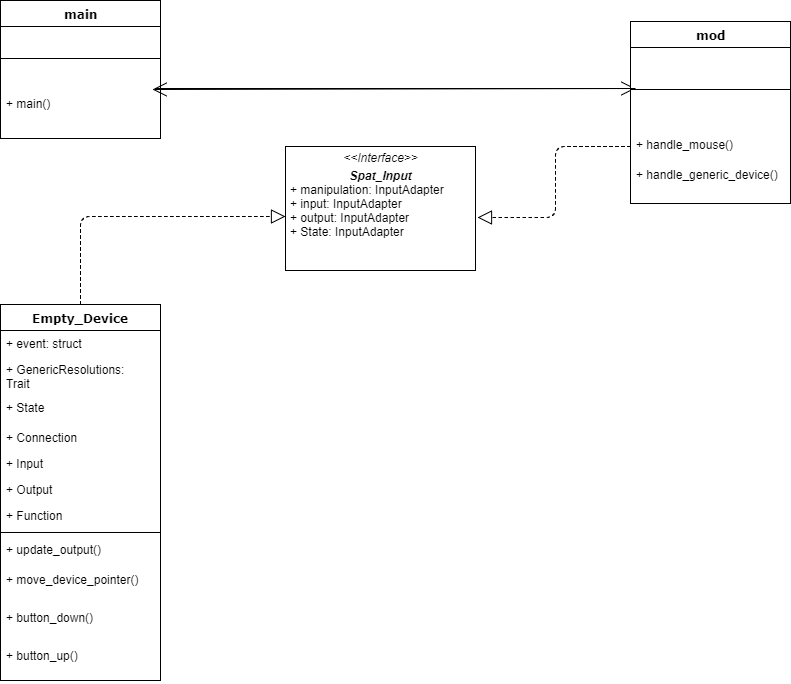


**Fig-1:** Use Case Diagram for User Story #758. Interaction between user and developer for signal sending and receiving

**Sequence Diagram**

  
**Fig-2:** Sequence Diagram for User Story #758. Events take place between user, device, input window, and developer.

**Class Diagram**



**Fig-3:** Class Diagram for User Story #758. Diagram focuses on four classes. They are main, spat\_input, mod, and empty\_device. The spat\_input (spatial input) class is inherited by empty\_device and mod

**Unit Test**

* Test case ID: Unit002 - Record generic input device
* Description/Summary of Test: The demo window is launched. The device recorded is an unrecognized device as it is not part of the rust device developer library. The user can interact with the device. The developer can process the generic inputs
* Pre-condition: Main rust class is running the demo window. The device connected to the machine is not supported
* Expected Results: Terminal reads “Button down” when an interaction is placed on the empty device. When released, terminal should read “button up”
* Actual Result: When any button was pressed, the terminal displayed “Button Down.” When any button was released, the terminal displayed “Button Up”
* Status (Fail/Pass): Pass

**Integration Test**

**Visual User Guide** <like one or two screenshots of the feature. For the hardware project, a photo of device is required>