Virtual Studio Technology (VST) Programming

The scope of this project is to detail the design of a VST plugin used to produce and alter sounds.

VSTs integrate virtual instruments (VSTi) and effects (VSTfx) into Digital Audio Workstations (DAWs). For Windows OS, the VSTs are usually coded in C++ and compiled in a single “.vst3” file which is manually loaded into a DAW to be used in producing music in a digital environment.

The VST responds to user’s input, transforming MIDI input (Musical Instrument Digital Interface) into high-fidelity audio.

The program is built around three signal generators called oscillators. Each oscillator has its own properties which can be adjusted by tweaking specific controls (knobs, buttons, sliders, etc.).

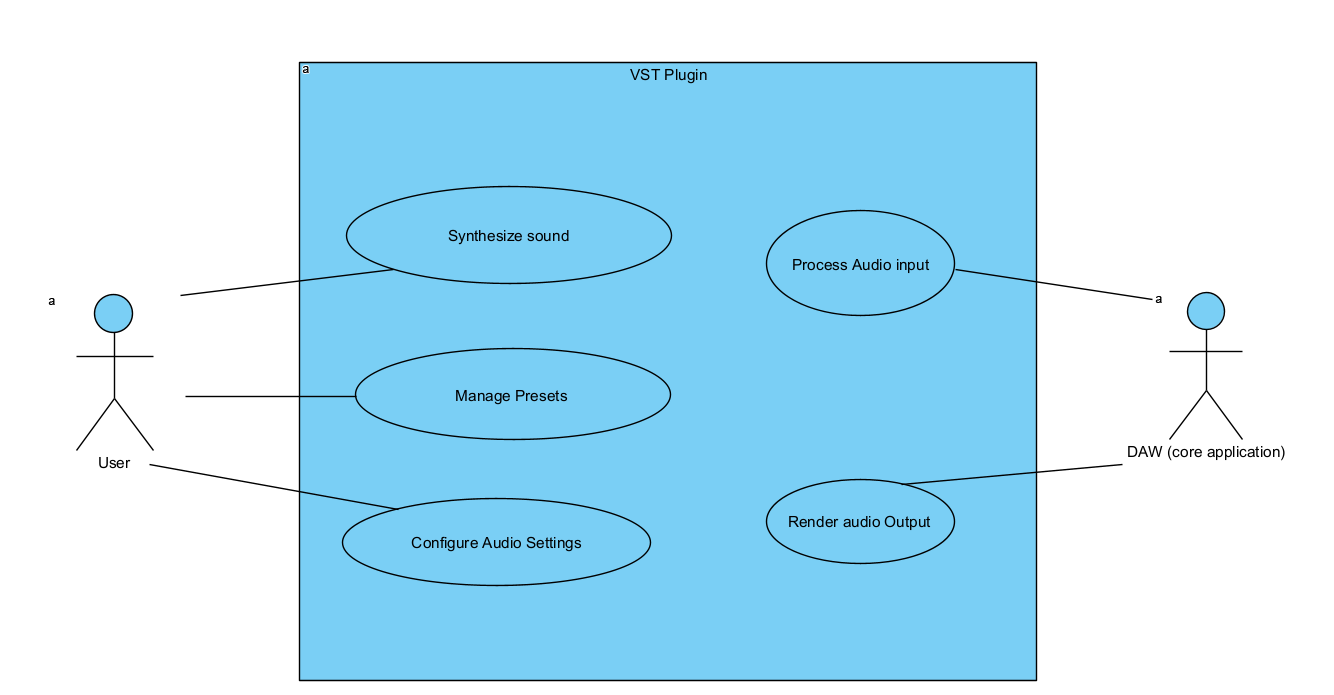
The shape of each oscillator can be set by the user to generate Sinewave, Sawtooth, Square or Triangle waveforms, and all three can be merged into a unique sound. Each oscillator can have its shape altered further, by modifying its ASDR Envelope (Attack, Decay, Sustain, Release).

As for effects, the output signal generated by the oscillator can be individually modified with simple effects such as Reverb, Delay, and EQ (equalizer).

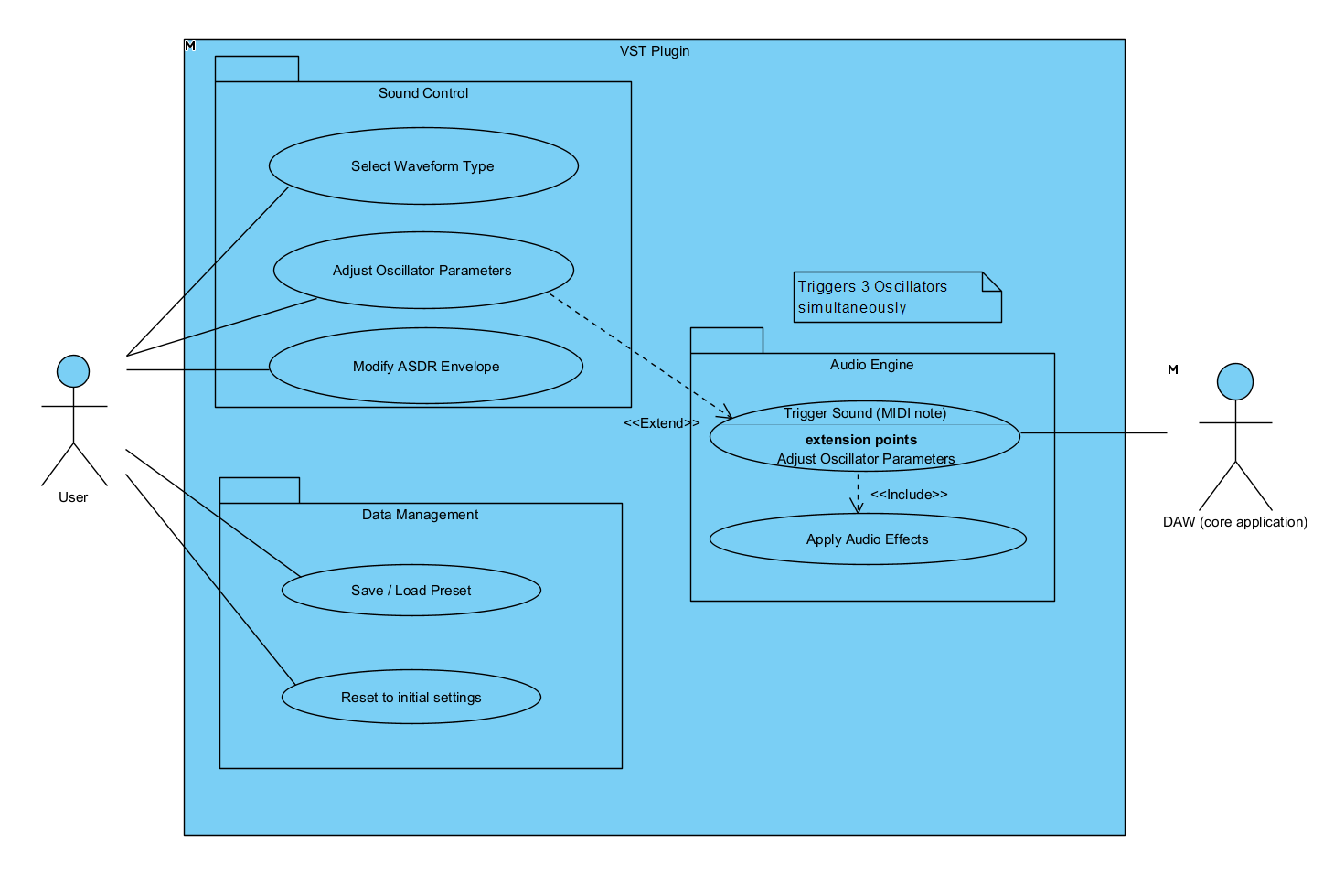
The user interface will be easy to understand and practical. It will be split into three identical parts, each with its own buttons, knobs and sliders.

The scope of developing such an instrument is to create a way to make music production a bit easier, as the current alternatives on the market may be seen as too complex, especially for beginners.

General use case diagram



Detailed use case diagram



Textual description of use cases for each diagram:

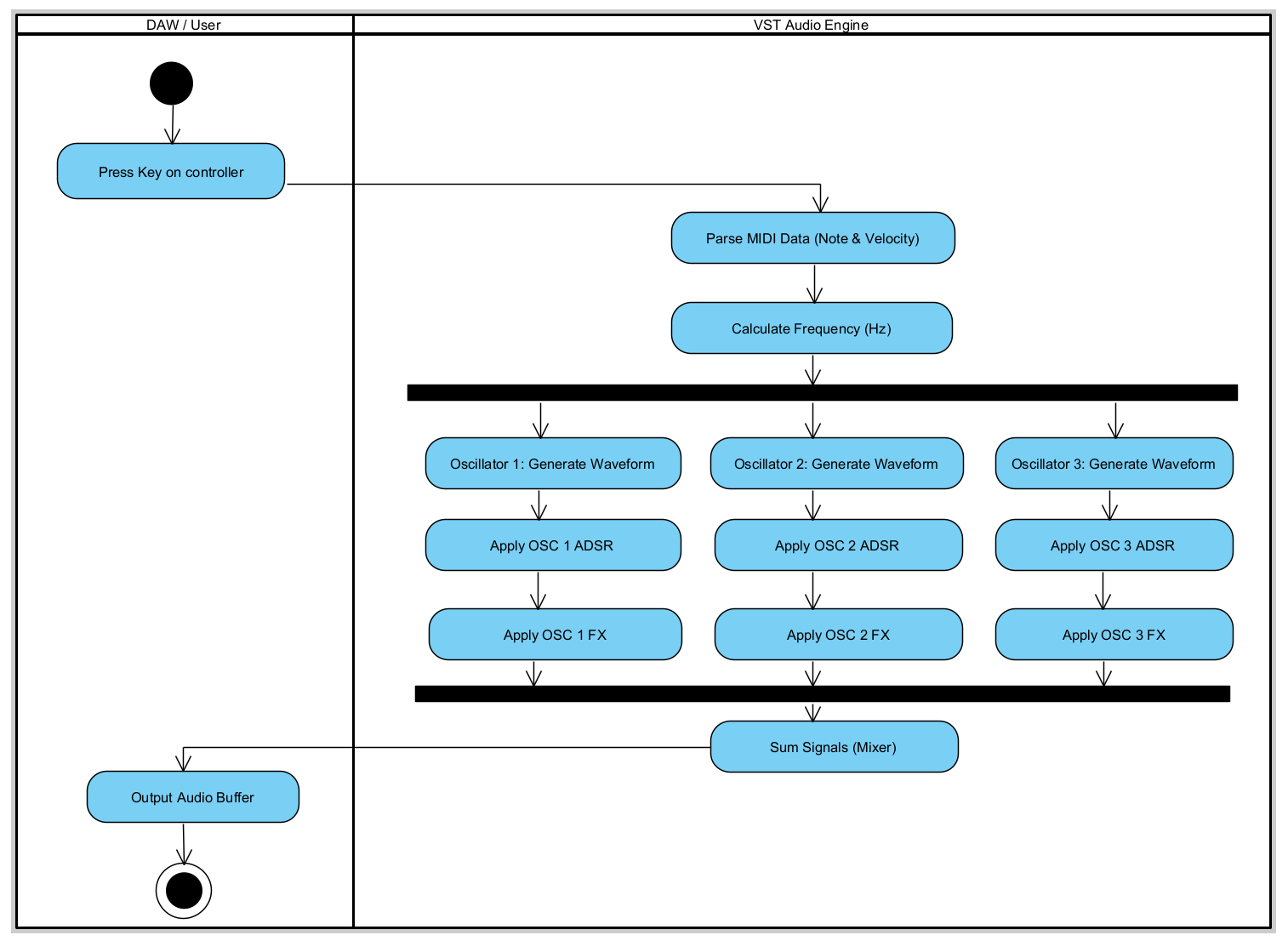
1. General use case diagram:

* Synthesize sound
* Manage Presets
* Configure Audio Settings
* Process Audio Input
* Render Audio Output

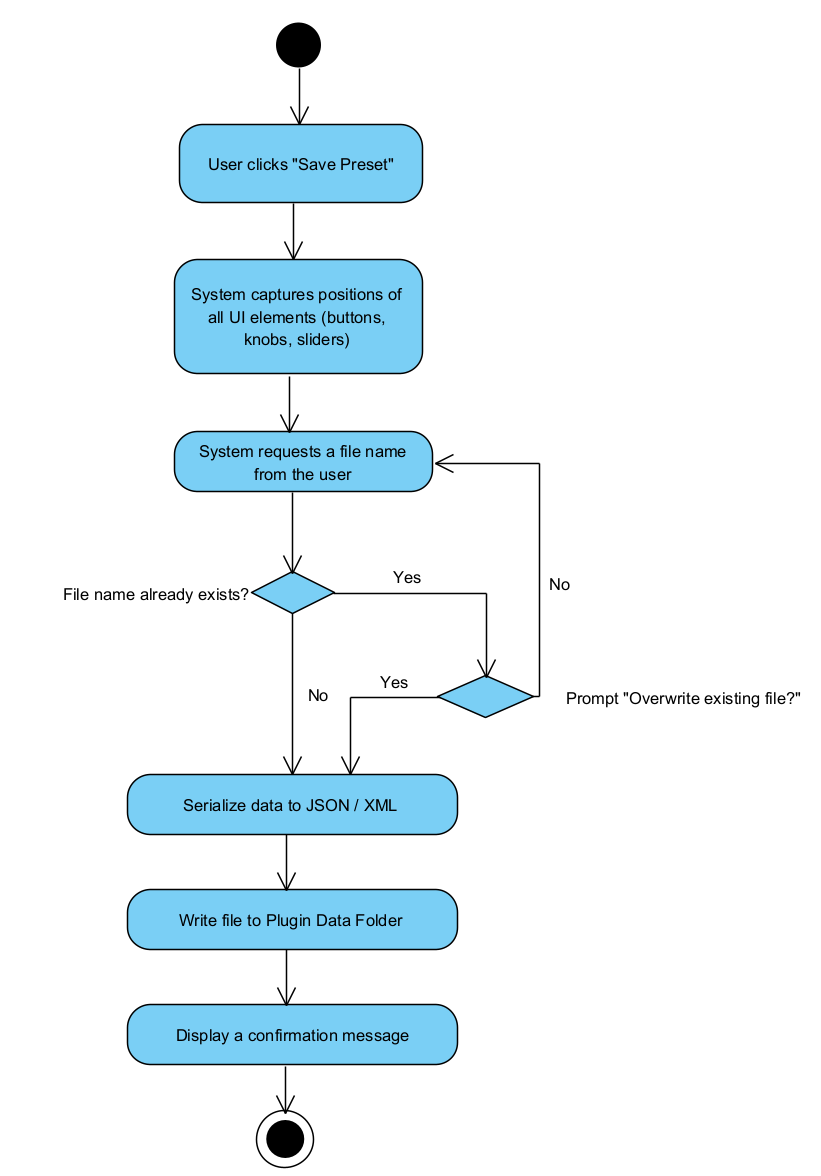
1. Detailed use case diagram:

* Sound Control
  + Select Waveform Type
  + Adjust Oscillator Parameters
  + Modify ASDR Envelope
* Data Management
  + Save/Load Preset
  + Reset to initial settings
* Audio Engine
  + Trigger Sound (MIDI Note)
  + Apply Audio Effects

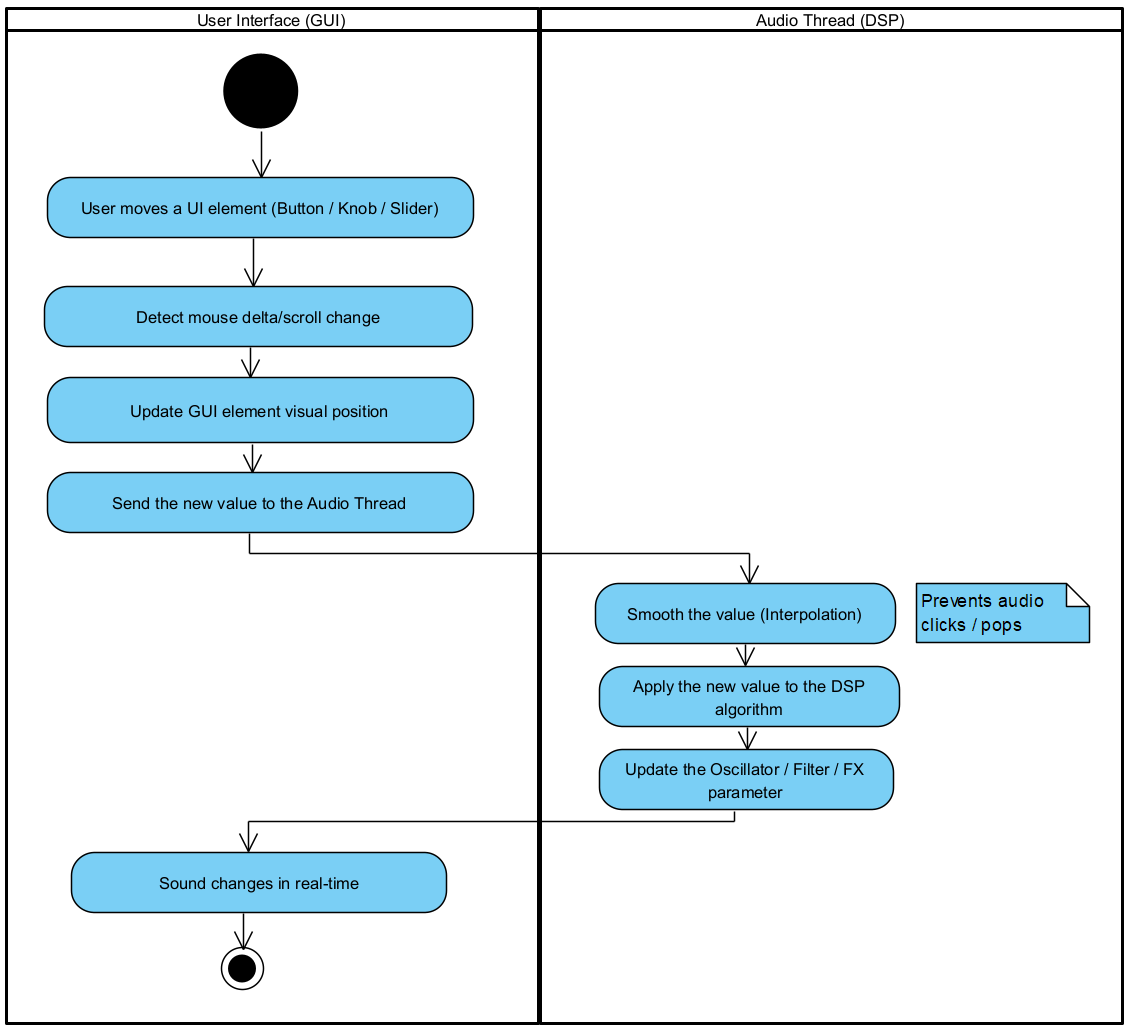
Activity Diagram 1 – The “Note On” Logic



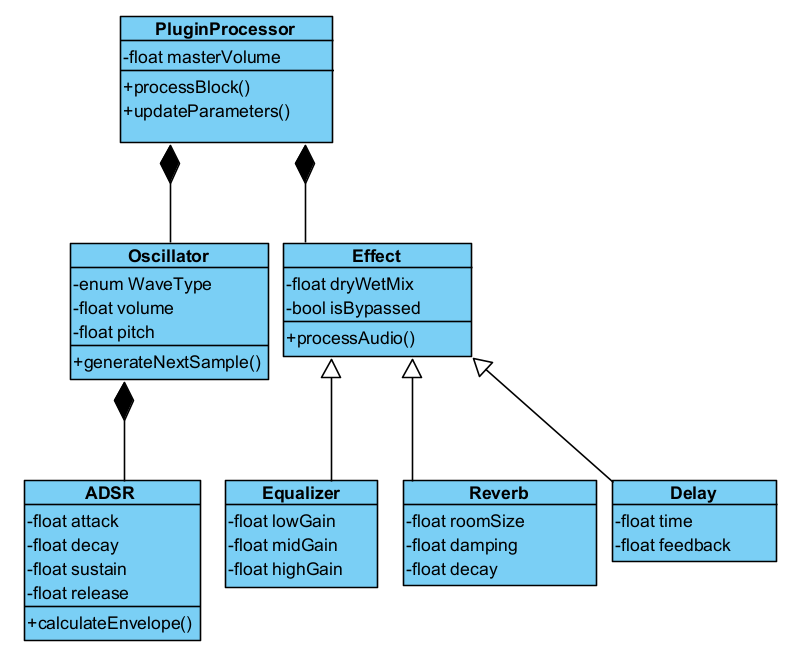
Activity Diagram 2 – Preset Saving Process



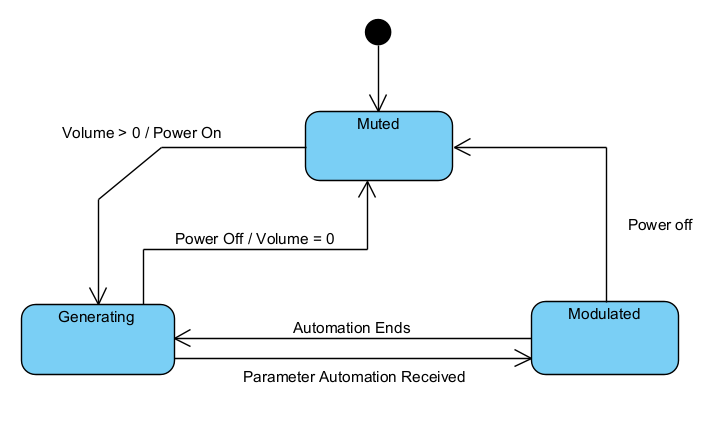
Activity Diagram 3 – Real-time Parameter Change



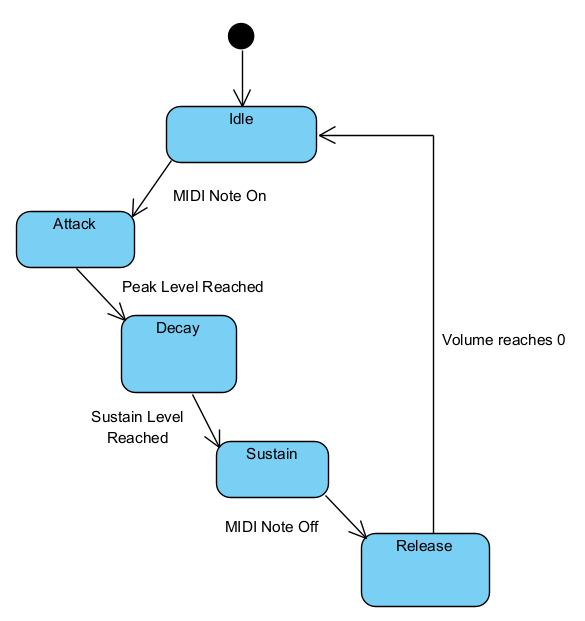
Class diagram – VST Synthesizer (Analysis)



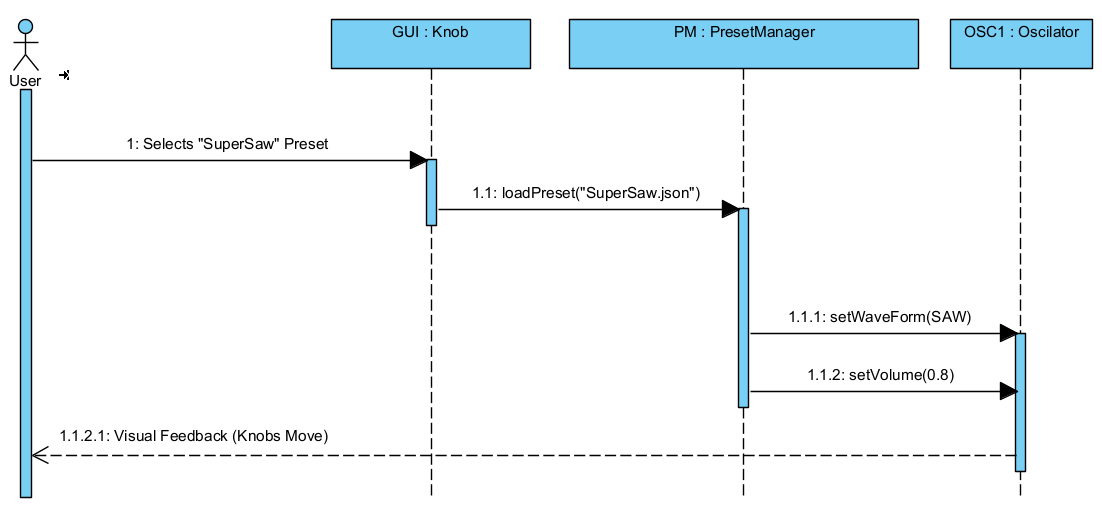
State Machine Diagram 1 – Oscillator Operational Status



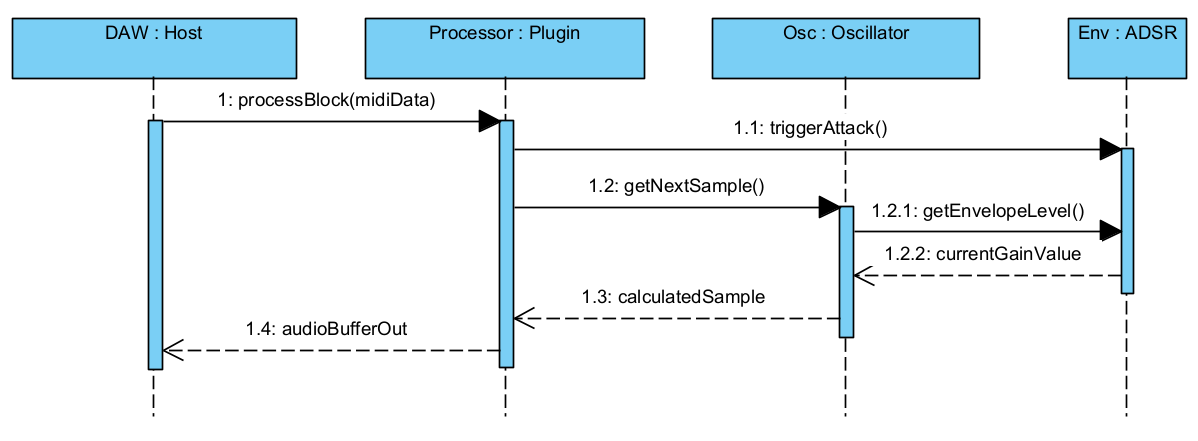
State Machine Diagram 2 - ADSR Envelope Lifecycle



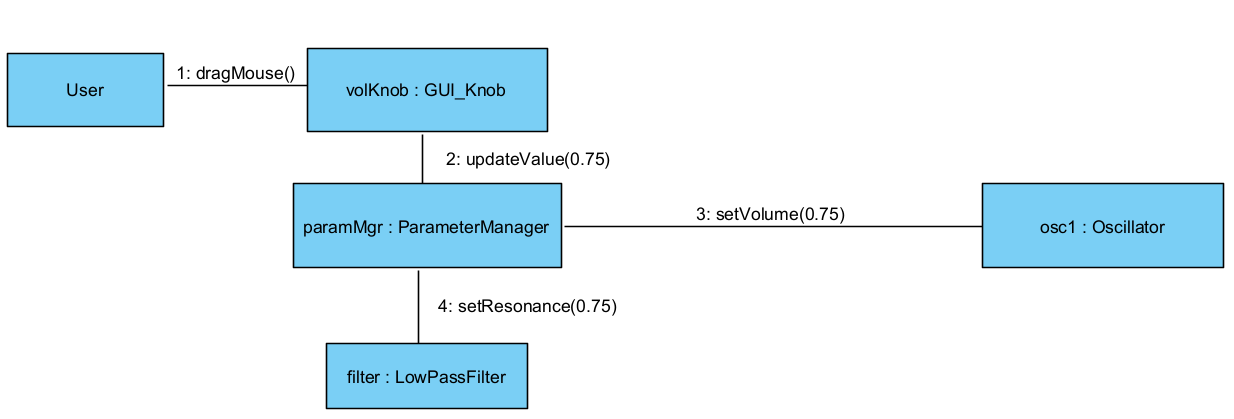
Sequence Diagram 1 – Loading a Preset



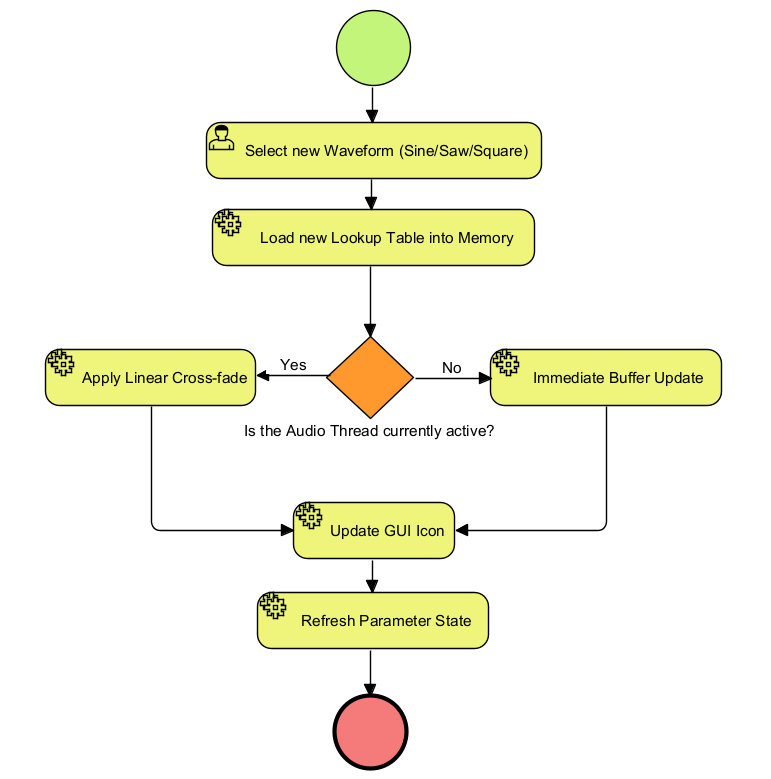
Sequence Diagram 2 – Note Trigger to Audio Output



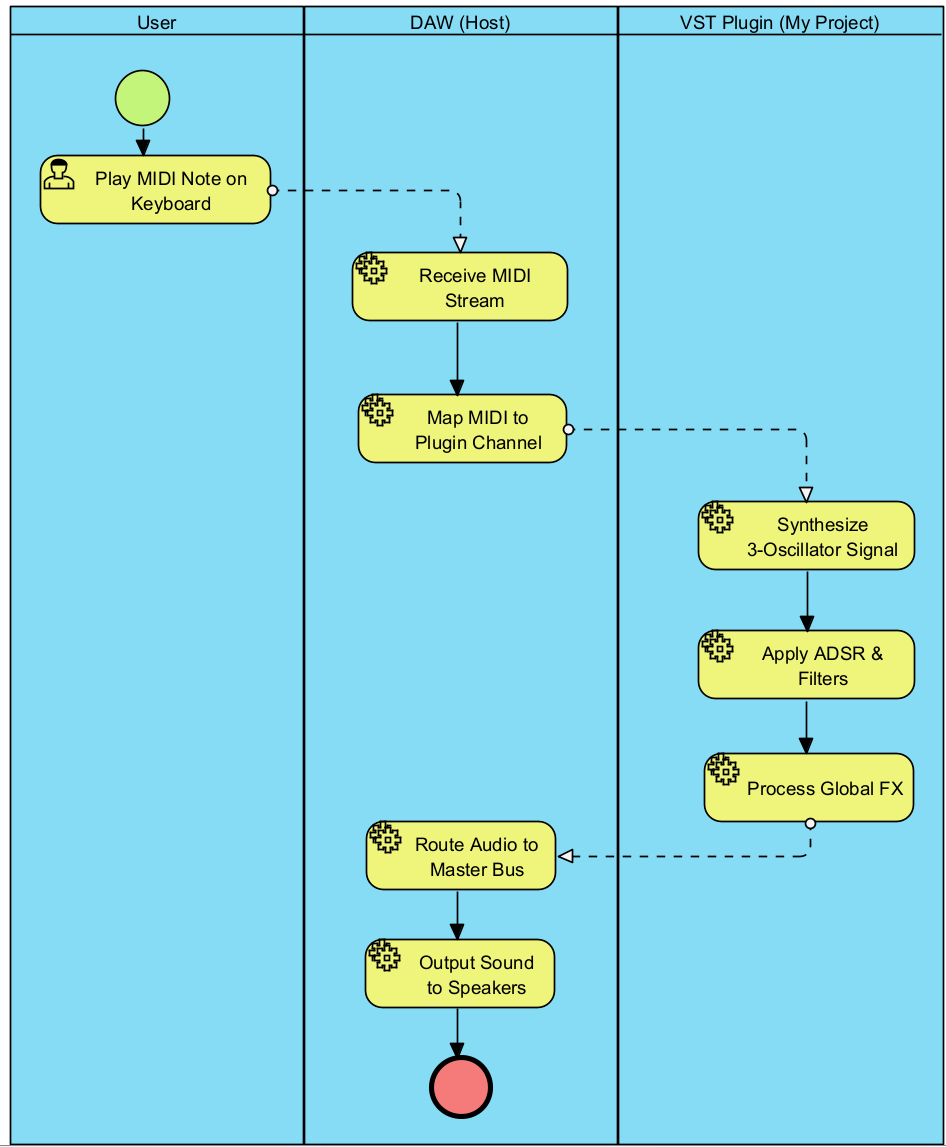
Communication diagram: Parameter Update



BPMN – Process diagram: Waveform Update Logic



BPMN – Collaboration diagram:



Detailed Class Diagram: