ShowControl: Sound Effects Player overview

Sound Effects Player is a component of ShowControl, which is a set of programs intended to provide computer assistance to a stage performance. Sound Effects Player plays the sound effects for the performance.

Who Uses it?

There are three people who use a sound effects player: the sound effects designer, who creates the sounds, the sound effects operator, who plays them at the correct time in the performance, and the sound rehearsal operator, who plays sounds during rehearsal. In community theatre these three roles are often performed by the same person. In addition, if the sound effects are very simple, the stage manager can trigger them remotely. In this description I will often refer to the sound rehearsal operator and the sound sound effects operator together as simply the sound effects operators, since they have very similar needs.

What must it do?

The sound effects player will

1. be able to make the sound of a ringing telephone,
2. be able to sequence a play as complex as The Passion of Dracula, and as simple as Oklahoma,
3. Be able to be operated by a sound effects operator with little previous experience,
4. Be able to be operated from a remote station, and
5. Require only a modest computer.

In addition, one thing that has always annoyed me in a human interface is that the meaning of a button can change just as I am about to push it, with the result that it does something other than what I intended. This design avoids that.

Design Overview

This design has a flexible internal sequencer, which is necessary for telephone rings and The Passion of Dracula. The sequencer is optional, so that a sound effects operator has a simpler interface if he is doing a production with only a few sound effects that do not overlap. For the simplest case, sounds can be triggered by the stage manager's computer using MIDI over Ethernet or OSC.

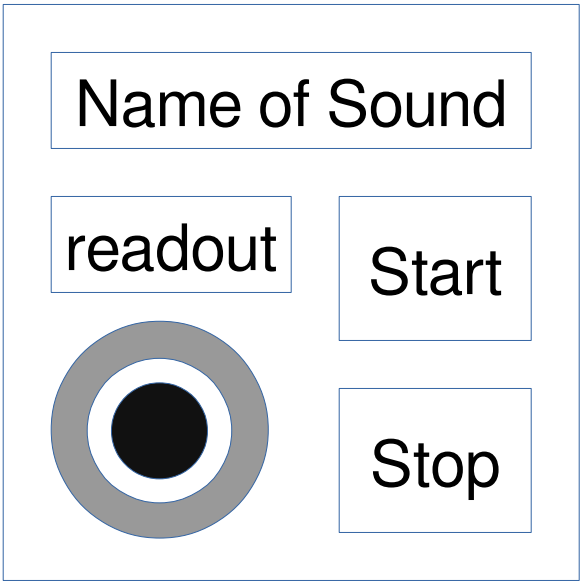
For each sound effect, the sound effects designer prepares a waveform for each independent sound

channel, the amplitude envelope, and some additional information. Many community theaters have a monophonic sound system, and so sound designs targeting such a theater will have only a single waveform for each sound effect. I prefer to do sound effects in stereo, so I always prepare two waveforms. I did one show using four independent sound channels. This design supports up to eight channels, since that is the maximum number of channels supported by the gstreamer WAV file parser.

In addition, for all but the simplest shows the sound designer prepares a collection of sequence items for the internal sequencer. I will describe the internal sequencer after describing the gstreamer plug-in on which it is based, but first I will describe the cluster.

The Cluster

From the point of view of the sound effects operators and the sound designer, the major item they work with is a group of controls called the cluster. Whenever there is a sound that is playing, or an opportunity to play a sound, it is linked to a cluster. A cluster looks something like this:



Drawing 1: cluster

A cluster is a 3-inch square, with text across the top, two buttons labeled Start and Stop on the right, and a soft dial with a readout above it on the left. The sound designer specifies the text displayed across the top. The dial controls two variables: the operator's volume and the operator's pan. The sound effects operators choose between them by pressing the dial as though it were a button. The text above the dial shows either the current volume as a number or the current pan as R number, L number or Centered. The sound effects operators control the starting and stopping of the sound using the Start and Stop buttons.

The Gstreamer plug-ins

The sound effects player includes two custom gstreamer plug-ins which are designed to meet its needs: a looper and an envelope filter. The looper allows a section of a waveform to be played repeatedly. The envelope filter implements an attack, decay, sustain and release (ADSR) envelope.

The amplitude envelope design is based on electronic musical instruments. It has four stages: attack, decay, sustain and release. It is described by six numbers:

- the attack duration time,
- the attack level,
- the decay duration time,
- the sustain level,
- the release start time and
- the release duration time.

Note that the release start time can be modified by an external action or reaching the end of the WAV file while the sound effect is playing, so the release start time specified by the sound designer is its initial value. In addition, the release duration time can be infinite.

Internally,

- times are measured in nanoseconds,
- volume levels are 0.0 for silence and 1.0 for full scale, and
- pan values are 0 for center, -1.0 for full left and +1.0 for full right.

These parameters are all specified by the sound effects designer when creating a sample. When displayed, times are shown in seconds with four decimal places, and levels as 0 to 100%. When the release duration time is infinite it is shown as ∞.

An example of the ADSR envelope is the wind heard when opening a door. The attack is the initial inrush of air. The sound then decays to the sustain level, where it remains until the door is closed, when the release fades the wind out.

In addition to the envelope information we have the following additional parameters specified by the sound designer, some of which are used elsewhere in the gstreamer pipeline, or in the application:

- the loop from time,
- the loop to time,

- the loop limit value,
- the starting position,
- the maximum duration time,
- the designer's volume level,
- the designer's pan value: Center, Left 1 to 100 or Right 1 to 100,
- the name of the sound effect, and
- the sound effect's MIDI program number, MIDI note number, OSC name and function key.

The maximum duration time is used to limit the amount of sound the looper holds in its local buffer. This can be used to sample some sound from a source of infinite duration, or, with starting postiion, to capture a portion of the sound from a WAV file also used for other sounds.

Other variables mentioned below are initialized when the sound effect is triggered. The algorithm for computing the amplitude of each channel of a sound effect at each time step is described here in several stages, for clarity of exposition. The algorithm is performed independently for each channel in its source, and passes the same number of channels to the next stage.

When a sound effect is triggered, the following local variables are set:

- current time and the loop counter are set to 0,
- the current position is set to the starting position,
- The operator's volume level is set to 100% (internally 1.0),
- The operator's pan value is set to Center (internally 0.0),
- the releasing condition is set to false,
- the end of input condition is set to false,
- the pausing condition is set to false.
- The looping condition is set to true if the loop from time is not equal to the loop to time; otherwise it is set to false.

Looper Element

The looper element sinks data from a WAV file parser. It accepts audio/x-raw sound samples in U8 (8-bit unsigned), S8 (8-bit signed), S16LE (16-bit signed), S32 (32-bit signed), F32LE (32-bit floating point) or F64LE (64-bit floating point) format. It has these parameters: loop-to, loop-from, loop-limit,

max-duration, start position and autostart. During preroll it copies all of the upstream data, up to max-duration, into its memory. It then sends silence downstream until it receives a Start message, whereupon it begins to pass its buffered data downstream, beginning at the start position. As it passes its data downstream it looks for the buffer position hitting loop-from, in which case it loops back to loop-to, unless the loop-counter is exhausted. The time stamps it sends downstream show the unrolled loop: that is, the time stamps always increase. When it receives a Release message it stops looping. When it hits the end of its data it sends a completion message downstream and returns to sending silence. It uses a custom completion message instead of End of Stream (EOS) because it doesn't want to disturb the pipeline—another Start message will cause it to send its data again.

The looper also processes pause and continue messages. These set and clear the pausing condition. When the pausing condition is true, the looper outputs silence and does not advance in its buffer, but continues to count time.

To shut down the gstreamer pipeline cleanly, the looper element accepts a Shutdown message, which causes it to send End of Stream (EOS) and stop sending data.

The output of the looper is constrained to have the same data type as its input. The autostart parameter is used for testing: it causes the looper to start sending its data as soon as it has filled its buffer, rather than waiting for a Start message. To avoid long startup times, the looper can read directly from its WAV file into its buffer.

Envelope Filter

Envelope processing starts with the attack stage upon receipt of the Start message. Once envelope processing is started, at each time step, the envelope filter computes the volume as follows:

1. If the current time is less than the attack time, set the volume to

   $$(attack\ level * (\frac{current\ time}{attack\ time}))\quad .$$

2. The attack is complete when current time reaches attack time, so if current time is greater than or equal to attack time but less than attack time + decay time, instead set the volume to

   $$(\frac{(attack\ time + decay\ time - current\ time)}{decay\ time}) * attack\ level\quad +$$

   $$(1 - \frac{(attack\ time + decay\ time - current\ time)}{decay\ time}) * sustain\ level\quad .$$

3. The decay is complete when the current time reaches attack time + decay time, so if current time is greater than or equal to attack time + decay time, instead set the volume to $sustain\ level$.

4. If the current time reaches release start time and the releasing condition is not true,
    a) set the releasing condition to true,
    b) remember the last volume value, which is probably sustain level, and
    c) send a message to the application telling it that the sound has started the release stage of its amplitude envelope.
5. If the releasing condition is true, we are in the release stage of the envelope, so do the following:
    a) If the release duration time is infinite and we have received an EOS or Complete message from upstream, the processing of this sound effect is complete. The volume is 0.0.
    b) If the release duration time is not infinite and the current time is greater than or equal to release start time + release duration time, the processing of this sound effect is complete. The volume is 0.0.
    c) If the release duration time is infinite, use the volume setting we remembered when release started.
    d) Otherwise compute the volume setting as the volume setting we remembered when release started multiplied by $1-\frac{current\,time-release\,start\,time}{release\,duration\,time}$ unless release duration time is 0, in which case set the volume to 0.0.

After computing the volume based on the envelope as described above, the volume is further multiplied by $designer's\,volume\,level$. If the sound was triggered by a Note On MIDI event, it is additionally multiplied by $\frac{(note\,velocity-1.0)}{126.0}$. If the sound was triggered by pressing a button on the front panel, the note velocity is set to 127.0. If the sound was triggered by the internal sequencer, the value of note velocity may be specified in the sequence item; if it is not, and there is no Note On in its sequencer history, the velocity is set to 127.0.

After envelope processing is complete the sound is presented on the bin's source pad as audio/x-raw, format F64LE or F32LE. The format of the source pad will match the format negotiated for the sink pad.

When the envelope has finished, the application is sent a completed message and silence is sent until the sound effect is restarted.

Other Elements

The operator's volume level, the operator's pan value and the pausing condition are controlled by the sound effects operator during the performance or rehearsal. The application signals changes in volume and pan to volume and pan elements which follow the envelope element in the gstreamer pipeline. The echo and equalization elements are handled similarly, but they do not have dedicated controls in the cluster.

The sound effects operator can also use a function key as a convenient way to trigger the sound.

Note Off

If the sound was triggered by a Note On MIDI event, a corresponding Note Off initiates the release of the sound by sending a Release message to the looper and envelope elements. The release process

1. sets the release condition to true,
2. sets the release start time to the current time,
3. remembers the most recent volume value, and
4. terminates any looping,

If the sound was triggered by pressing the start button on a cluster, an adjacent stop button initiates the release process. If the sound was triggered by the internal sequencer, there will be an item later in the sequence which will initiate the release process, or the sound can be stopped by the sound effects operator. If the sound was triggered by a MIDI Sound Control message, a subsequent message will initiate the release process. Note that unless a sound loops indefinitely and has release start time of 0, it will eventually complete even if it is never told to initiate the release process, in which case triggering the release process is optional.

The sound designer can specify actions to take when the release starts and when the sound terminates.

Defaults

There are default values for the parameters of a sound:

- attack duration time defaults to 0,
- attack level defaults to 1.0,
- decay duration time defaults to 0,
- sustain level defaults to 1.0,
- release start time defaults to 0,

- release duration time defaults to 0,
- loop from time defaults to 0,
- loop to time defaults to 0,
- loop limit value defaults to 0,
- the starting position defaults to 0,
- the max duration defaults to unlimited,
- the sound designer's volume level defaults to 1.0,
- the sound designer's pan value defaults to Center, and
- the MIDI program number, MIDI note number, OSC name and function key default to unspecified.

The result of using these defaults is that a sound effect will simply play once with no volume adjustments when triggered. To make a sound effect loop indefinitely, set the loop from time to the duration of the WAV file. It will stop when the release process is initiated. To terminate a looping sound effect automatically after a particular length of time, set the release start time to the length of time the sound is supposed to last.

When the sound designer creates a sound effect, he will create the WAV file with full volume, and use the designer's volume level to attenuate it so that it plays correctly if the sound effects operator does not adjust the operator's volume level. If a sound effect must be played with different volumes at different times, the sound designer will set the designer's volume level for the loudest instance, and use the volume parameter of the Start Sound sequence item, or the velocity of the Note On MIDI message from an external sequencer, to attenuate other instances.

The starting position parameter is intended for rehearsals. The director may wish to play the last 15 seconds of the previous song as a lead-in to the next song. The rehearsal sound operator can get this effect by editing the previous song's parameters to start 15 seconds before its end, when it will transition to the next song. In some cases it might also be necessary to change the release start time to 14 seconds. Another common situation is wishing to play a dance number starting at a specified time.

If the release duration time is greater than zero, there is an interaction between the envelope and the looper which must be considered. On release, the looper will stop looping, and this can happen any time between the start and end of the loop. In case it happens at the end, the sound designer should provide enough sound after the end of the loop to cover the release duration time.

Internal Sequencer

The internal sequencer adds additional complexity that may not be justified, so using it is optional. If the internal sequencer is not being used, the sound effects are matched to clusters based on their MIDI program numbers and note numbers, if they are specified, or in the order they appear in the project file, if they are not. Otherwise, the internal sequencer controls how sound effects are presented to the sound effects operator.

The internal sequencer holds a list of sequence items. A sequence item does one of these things:

1. Start Sound: begin playing a sound effect. The data in the item contains
    a) the name of the sound effect,
    b) the tag for this sound effect,
    c) a flag called Use External Velocity, which is 1 if the velocity of an external MIDI Note On message it to be used
    d) The program number, bank number and cluster number in which to display this sound,
    e) the name of the sequence item to execute when this sound effect completes,
    f) the name of the sequence item to execute when this sound effect is terminated,
    g) The name of the sequence item to execute when this sound effect starts,
    h) The name of the sequence item to execute when the sound enters the release segment of its amplitude envelope, unless the sound was released by an external signal,
    i) the importance of this sound for the sound effects operator,
    j) The Q_number, which is a list of numbers separated by periods used by MIDI Show Control, and
    k) text to display to the operator in the cluster.
2. Stop Sound: cease playing a sound effect. The data in the item contains
    a) the tag of the sequence item that started the sound effect, and
    b) the sequence item to execute next.
3. Wait: pause this fork of the sequencer for a specified length of time. The data in the item contains
    a) the time to wait,
    b) the sequence item to execute after the specified time has passed,

c) the tag, which can be used by a Cancel_wait sequence item to terminate this Wait before its time has elapsed, and not execute the sequence item specified to be executed after the time has elapsed,

d) text to display to the operator while we are waiting, and

e) the sequence item to execute next.

4. Offer Sound: define what one of the sound clusters does. The data in the item contains

a) The name of the sequence item to execute when the operator pushes the start button in the cluster,

b) the tag for this offering,

c) the program number (0-127), bank number (0-7) and cluster number (0-15) of the cluster,

d) the MIDI program number and note number from an external sequencer which causes the named sequencer item to be executed,

e) the OSC name which causes the named sequence item to be executed,

f) The Q_number, which is a list of numbers separated by periods used by MIDI Show Control to execute the named sequence item using the Go command,

g) The macro number, a value from 0 to 127 which is used by MIDI Show Control to execute the named sequence item using the Fire command,

h) The function key which causes the named sequence item to execute,

i) text to display to the operator in the cluster, and

j) the sequence item to execute next.

5. Cease Offering Sound: remove the definition of what a cluster does. The data in this item contains

a) The tag of the Offer Sound sequence item which offered the sound, and

b) the sequence item to execute next.

6. Operator Wait: wait for the sound effects operator to press the “play this sound effect” key. The operator is shown the text for this item. In addition to the text there is a lamp which lights when the internal sequencer is waiting for the operator. The data in this item contains

a) the sequence item to execute when the operator presses his key

b) text to display to the operator,

c) The function key which causes the sequencer to position to this sequence item and proceed as though the operator had also pressed the “play this sound effect” key,

d) The Q_number, which is a list of numbers separated by periods used by MIDI Show Control to execute this sequence item using the Go command, and

e) The macro number, a value from 0 to 127 which is used by MIDI Show Control to execute this sequence item using the Fire command,

f) The tag, which can be used to terminate the Operator Wait before the operator presses the Play button without executing the sequence item to execute when the operator presses the Play button, and

g) The sequence item to execute next.

7. Cancel Wait: terminate the specified Wait and Operator Wait sequence items.

   a) The tag of the Wait and Operator Wait sequence items to terminate, and

   b) The sequence item to execute next.

8. Start Sequence: there must be exactly one of these in a sequence. The data in this item contains

   a) The sequence item to execute next

The sound effects designer constructs the collection of sequencer items. The Operator Wait items not specified as hidden are presented to the sound effects operator sorted by display text. When the sequence is loaded, or in response to the Reset button or MIDI Show Control command Reset, the Start Sequence sequencer item is executed. It should set up whatever sound effects are to be offered to the operator before the performance starts, and then either terminate or execute an Operator Wait.

If the show goes well, the sound effects operator just has to press a key when it is time to play the next sound effect, and/or stop the last one. To deal with the actors jumping around in the script, the operator needs a dial which will rapidly scroll forward or backward. The designer should include the page number in the text shown to the operator, so he can see where he is while scrolling.

A sequence item's name can be changed, and doing that automatically changes all references to it elsewhere in the sequence.

To use the velocity of the Note On MIDI message from an external sequencer, use the Offer Sound sequence item to make a Start Sound sequence item with Use External Velocity equal to 1 available to the external sequencer.

If there is a need for sounds which differ only in volume or pan, create different sounds using the same WAV file.

If a Start or Offer Sound sequence item specifies a cluster on which a sound is in its release, that sound is detached from the cluster and the new one takes its place. It is an error to specify a cluster which holds a sound that has not started its release, or one on which there is a sound being offered (but see below for an exception to this rule). If no cluster is specified, the cluster corresponding to the sound effect's MIDI program number and note number is used, if it is specified and available; otherwise the lowest available cluster is used. However, if the Start Sound is executed because of the release, completion or termination of a previous Start Sound, this sound will use the same cluster as the previous sound.

Note that the Start Sound sequence item can do a three-way fork, by executing a sequence item when the sound effect starts, one when it releases, and a third when it completes. (If a sound is terminated there is only a two-way fork: one when the sound effect starts and the other when it terminates.) The Wait sequence item provides a two-way fork, by executing a sequence item when the waiting starts and another when it ends. Forking allows multiple sound effects to be played at the same time, and the Wait sequence item allows for a timed delay between them. If a sound needs to stop based on on-stage action, the sequencer can wait for the operator's signal to initiate the release of a sound effect using the Stop Sound sequence item. The sequence as a whole ends when all forks have ended and no sounds are being played or offered by the internal sequencer.

The Offer Sound sequence item also does a two-way fork. It is used when the order of sounds to be played cannot be determined in advance. When a sound is offered on a cluster, the sequencer will fork when the Start button is pressed, a MIDI message is received or an OSC message is received. The volume displayed in the cluster is initially 100%, which is the external representation of the initial operator volume level, 1.0. If the operator modifies the value it is carried over to subsequent sound effects in the same cluster until the Cease Offering Sound sequence item for that cluster. This is also true for the operator pan value, which starts at Center. The operator volume level can range from 0% to 400%.

An exception to the rule that a cluster occupied by an Offer Sound cannot be used by a Start Sound applies to sounds started by triggering an offered sound. Such sounds, if they do not specify a cluster, or if they specify the same cluster as the Offer Sound, will use that cluster, hiding the offering of the sound. If more than one sound plays at the same time (not counting release), other sounds must use different clusters. The offered sound becomes active again when no sound is playing on its cluster. This exception permits continuity of the operator volume and pan adjustment.

The importance number in the Start Sound sequence item controls the display of information about the current sound. If this number is 0 the sound is never considered “current”; this would be used for long-running background sounds that the sound effects operator does not need to supervise. A value of 1, which is the default, causes the sound to be regarded as the current sound when it is playing. In the general controls section of the sound effects player there is a display of the amount of time remaining

in the current sound, which can be switched to display the amount of time the current sound has been playing. If there is more than one sound playing, the one with the largest importance number, greater than 0, is considered current. If more than one playing sound has the largest non-zero importance number, one of them is chosen arbitrarily to be the current sound.

Whenever there are two or more forks in execution, there is the problem of serializing access to the sound effects operator's Play button. If a fork executes an Operator Wait, other forks executing Operator Wait will wait for the operator to press his Play This Sound Effect button to complete the Operator Wait command on the first fork. Forks using Operator Wait will get access to the operator in the order they requested access. The Wait sequence item is lower priority: its display is pre-empted by a subsequent Operator Wait. When there is no Operator Wait in progress, the Wait that will end soonest has its text on display. However, the Wait sequence item is not delayed by its inability to display its message—it completes on schedule even if it was unable to display its message at all.

The Cancel Wait sequence item is useful if you want to let the sound effects operator control the sequencing, but also let an external sequencer have control. The sequence can wait for the operator, but if an external command comes in it can terminate the wait and perform the command.

Telephone ring example

To make a telephone ring, we need two sound effects and seven items in the internal sequencer. When the sound effects operator initiates the ring the sequencer plays the ringing sound effect, which can include a loop to conserve sample space. It has release start time set to 5 milliseconds less than the desired ring time, and release duration time set to 10 milliseconds. When it starts its release, whether by being stopped or by reaching the release point in the sound, the internal sequencer plays the ringout sound effect. The ringout sound effect starts with 10 milliseconds of attack and no decay, providing a click-free transition from ringing to ringout, and has an infinite release duration time. If the ringout completes without being stopped the internal sequencer will again play the ringing sound effect, and continue alternating between ringing and ringout until the sound effects operator signals that the ring is to stop.

If a stop command arrives during the ringing, the sound stops immediately and the internal sequencer plays the ringout once, then stops. If a stop command arrives during ringout the sound ends after the ringout. Thus, the telephone will ring until it is stopped, and when it is, the sound transitions in 10 milliseconds to the ringout unless the ringout is already in progress.

Here is the sequence, in detail:

1. Operator Wait, text “telephone rings”, next is item 2.
2. Start sound ring, cluster 0, tag “telephone_ring”, text “ring”, on start go to item 3, on release

start go to item 5, or on termination go to item 7.

3. Operator Wait, text “stop telephone ring”, next is item 4.
4. Stop Sound tag “telephone_ring”.
5. Start sound ringout, cluster 0, tag “telephone_ring”, text “ring”, on completion go to item 6, on termination go to item 8, which is the Operator Wait sequence item for the next sound effect.
6. Start sound ring, cluster 0, tag “telephone_ring”, text “ring”, on release start go to item 5, or on termination go to item 7.
7. Start sound ringout, cluster 0, tag “telephone_ring”, text “ring”, on completion go to item 8, which is the Operator Wait sequence item for the next sound effect.

The above sequence forks at item 2, with one branch proceeding to items 3 and 4, and the other looping between items 5 and 6 until it is terminated by the operator proceeding from item 3, in which case it may execute item 7 before it gets to item 8. Note that item 4 stops either the ring or the ringout, whichever is playing, because both are tagged “telephone_ring”.

If it is necessary to do other sound effects while letting the telephone ring in the background, change item 2 to go to item 8 when the sound starts, omit items 3 and 4, remove the references to item 8 on items 5 and 6, and instruct the sound effects operator to press the Stop button on cluster 0 (marked “ring”) when the telephone ringing is to stop. The sound effects designer must avoid using cluster 0 for other sound effects until the telephone ringing is definitely over.

If the telephone is to ring several times, the sound effects designer might decide to devote a cluster to it for a period of time. In that case he would specify an Offer Sound sequence item which plays the ring, plays the ringout, and loops as described above. He would instruct the sound effects operator to press the Start button on the designated cluster to start the telephone ringing, and press Stop to make it stop. After all the telephone rings are done, the sound effects designer can use the Cease Offering Sound sequence item to free up the cluster for other uses.

External Sequencer

An external sequencer can send Note On MIDI messages to trigger sound effects. On receipt of such a message, if the internal sequencer is running, the offered sound with matching MIDI program number and note number is started, and runs until it ends. The player only accepts MIDI messages from one channel—the channel number is set by the operator. After sending Note On, the external sequencer can stop the sound by sending Note Off for the same program number and note number. The sound effects player also accepts All Notes Off and All Sound Off, to handle emergencies. The Note Off velocity is not used.

Note Off finds all sound effects being played by an offered sound with the specified program number and note number which have not yet started release. It initiates the release process on all of those sound effects. If there is no such sound effect, Note Off does nothing. All Notes Off finds all sound effects which have not yet started release and initiates release on them. If there are no such sound effects, All Notes Off does nothing. All Sound Off mutes the master output.

The external sequencer can also use OSC messages to start and stop sounds, and generally control the operation of the sound effects player.

If the internal sequencer is not running, the MIDI and OSC information for each sound effect specified by the sound designer is used to match incoming MIDI Note On and OSC messages to sound effects.

MIDI Show Control

The sound effects player also supports MIDI Show Control messages from an external sequencer. It regards itself as a sound device of type 13, which means “EPROM player”. The device ID and group ID are set by the sound effects operator. The sound effects player does not implement time code, cue lists or cue paths. The internal sequencer must be running for MIDI Show Control to work.

Q_number

Some of the sequence items have a data item called Q_number to support MIDI Show Control. This text field contains a list of numbers separated by periods, for example, 36.7.832. For some purposes sequence items containing Q_numbers must be sorted in cue order. In this sort, the numbers are considered separate keys, with priority decreasing from left to right. Thus, the Q_number values of 1.1, 1.5, 1.10 , 1.100 and 2 would sort in that order.

The first number in the Q_number is referred to as the Parent of the cue.

Commands

The sound effects player supports the following MIDI Show Control commands:

1. Go (01_{16}): If no Q_number is specified, continue sequencer processing from the current Operator Wait as though the sound effects operator had pressed his Play This Sound Effect key. If a Q_number is specified, start the Offered Sound with the specified Q_number. If there is no sound being offered with this Q_number, search the sequence items for an Operator Wait with this Q_number, position the sequencer to it, and proceed as though the operator had pressed his Play This Sound Effect key. Q_list and Q_path are ignored.

1. Resume (03_{16}): If no Q_number is specified, resumes all stopped sounds. If a Q_number is specified, resumes only stopped sounds which were running due to a Start Sound with that Q_number, leaving all others stopped.

2. Timed_go (04_{16}): treated as Go.
3. Stop (02_{16}): If no Q_number is specified, pauses all sounds. If a Q_number is specified, pauses only those sounds running due to a Start Sound with that Q_number, leaving all others running. Q_list and Q_path are ignored. When a sound is paused, the looper element stops counting time, stops advancing its position, and outputs only silence on all channels. The echo element in the sound effect's gstreamer bin is not affected, so a sound may continue to echo after it is stopped.
4. Load (05_{16}): positions the sequencer to the Operator Wait sequence item with the specified Q_number.
5. Set (06_{16}): Adjust the operator's volume level of a sound. This can refer to any offered sound, or any sound that is playing. The reference to a sound is based on the cluster in which it is offered or playing. This cluster number is specified by the sound designer, either in the definition of a sound or in the sequence item that plays or offers the sound. The body of the Set message contains 12 bytes, each 00_{16} to $7F_{16}$, interpreted as ASCII. The first byte specifies whether we are controlling the volume of a specific sound or the master volume control. A value of “1” means we are controlling a specific sound, in which case the second, third and fourth bytes contain the program number (“000” to “127”) the fifth the bank number (“0” to “7”) and the sixth and seventh the cluster number (“00” to “15”). The eighth through twelfth bytes contain the volume, formatted as a digit, a period, and three digits. The internal volume level is a number between 0.000 and 4.000, but it is displayed to the operator as a value between 0% and 400%.

   If the first byte is “2”, we are adjusting the master volume control. Bytes two through seven must be ASCII spaces, and bytes eight through twelve specify the setting of the master volume control, as described above.
6. Fire (07_{16}): Start the Offered Sound with the specified macro number. If there is no sound being offered with this macro number, search the sequence items for an Operator Wait with this macro number, position to it, and proceed as though the operator had pressed his Play This Sound Effect key.
7. All_off (08_{16}): mutes all sounds. This takes place just after the sound meter, and does not affect processing; it is equivalent to shutting down the speakers.
8. Restore (09_{16}): unmutes all sounds, reversing a prior All_off.
9. Reset ($0A_{16}$): Cancel all running forks, terminate all sounds, clear all clusters, reinitialize all internal state, position the sequencer to the Start Sequence item and start the sequencer. The sequencer will proceed until it encounters an Operator Wait sequence item. If there is no Start

Sequence item the sequencer is not started.

10. Go_off ($0B_{16}$): If a Q_number is specified, initiate the release of all sounds not already releasing that were started by a Start Sound with that Q_number. If a Q_number is not specified, initiate the release of all sounds that are not already releasing.
11. Go/Jam_clock (10_{16}): treated as Go.
12. Standby_+ (11_{16}): If the sequencer is in an Operator Wait sequence item with a Q_number, position the sequencer to the Operator Wait sequence item with the next Q_number, when the Q_numbers are sorted in cue order. If there is no such sequence item, or if the sequencer is not at an Operator Wait sequence item with a Q_number, do nothing.
13. Standby_- (12_{16}): If the sequencer is in an Operator Wait sequence item with a Q_number, position the sequencer to the Operator Wait sequence item with the previous Q_number, when the Q_numbers are sorted in cue order. If there is no such sequence item, or if the sequencer is not at an Operator Wait sequence item with a Q_number, do nothing.
14. Sequence_+ (13_{16}): If the sequencer is in an Operator Wait sequence item with a Q_number, position the sequencer to the first Operator Wait sequence item with a Q_number containing the next Parent cue, when sorting by cue number. For example, if the Q_number of the current Operator Wait sequence item was 29.3.24.98.7, and the the Q_number values of subsequent (when sorted in cue number order) Operator Wait sequence items were 29.3.25, 29.4, 29.7, 29.9.876, 36.7, 36.7.832, 36.8, 37 and 37.1, then position the sequencer to the Operator Wait sequence item with Q_number 36.7. If there is no such sequence item, or if the sequencer is not at an Operator Wait sequence item with a Q_number, do nothing.
15. Sequence_- (14_{16}): If the sequencer is in an Operator Wait sequence item with a Q_number, position the sequencer to the first Operator Wait sequence item with a Q_number containing the previous Parent cue, when sorting by cue number. For example, if the Q_number of the current Operator Wait sequence item was 37.4.72.18.5 and the Q_number values of previous (when sorted in cue number order) Operator Wait sequence items were 29.3.25, 29.4, 29.7, 29.9.876, 36.7, 36.7.832, 36.8, 37 and 37.1, then position the sequencer to the Operator Wait sequence item with Q_number 36.7. If there is no such sequence item, or if the sequencer is not at an Operator Wait sequence item with a Q_number, do nothing.

Gstreamer pipeline

The sound effects player is a gstreamer application and uses its custom elements, described above, along with several standard plugins. To avoid requiring a high-speed computer, the complete pipeline is constructed when a project is loaded or a sound effect modified; it is not changed while sound is

playing. The pipeline structure looks like this:

The application creates the following gstreamer bin for each sound effect defined in the project:

1. A filesrc element which identifies the WAV file containing the waveforms.
2. A wavparse element, which converts the WAV file into up to eight channels of raw audio.
3. A looper element, which unrolls loops and handles some aspects of release.
4. An envelope element, which handles attack, decay, sustain and release. It also has a volume parameter which handles the designer volume control and MIDI Note On velocity.
5. An echo element, which has delay, intensity and feedback controls. These are specified by the sound effects designer, and can be edited. They default to no echo.
6. An equalization element, which has gain values for the 29 Hz, 59 Hz, 119 Hz, 237 Hz, 474 Hz, 947 Hz, 1889 Hz, 3770 Hz, 7523 Hz and 15011 Hz bands. These are specified by the sound effects designer, and can be edited. They default to 0.
7. If the WAV file contains one or two channels, two audiopanorama elements, which handle panning. The first of these elements is set to the designer's pan value. The control input to the second element is the operator pan value, represented as 0.0 for center, -1.0 for full left and +1.0 for full right. The output from these two elements is always two channels. If the WAV file contains more than two channels, the sound designer marks it as not panning, and these elements are not placed in the bin, and the output of the previous element is connected directly to the input of the next element. The sound effects designer can also omit the pan on the sound effect if it is a mono sound that is to be fed to a specific speaker.
8. A volume element, which handles the operator volume control
9. A deinterleave element, which separates each channel of its input into a different output stream.

The output of these bins is a single-channel gstreamer source for each channel of each sound effect. Each of these bins produces only silence on all its channels until its sound is started. The gstreamer pipeline contains all of the above bins plus one additional bin, which contains:

1. Eight adder elements, one for each output of the sound effects player. The connection between the sources of the sound effects bins and the sinks of each adder is specified by the sound effects designer, and can be edited. The information is retained in the project file. The default is that output channel 1 gets channel 1 from each sound effect bin, output channel 2 gets all the channel 2s, etc.
2. An Interleave element, which combines the outputs of the eight adder elements into a single stream with eight channels.

3. A Volume element, which implements the master volume control.
4. A Level element, which extracts level information from the stream for display.
5. Another Volume element, which implements the master mute control.
6. A sink element, which sends the stream to an output device. For the software implementation it defaults to autoaudiosink, which finds a good output device. For the fancy console it is fixed to the device that drives the eight XLR connectors. An alternative in the software implementation is to use splits and queues to write a copy of the output to a WAV file, for debugging.

Tools

The sound effects player has some tools for the sound effects designer and sound rehearsal operator.

Create a Cue from an m3u File

An m3u file contains a list of sound files. This tool provides a convenient way to import them into the sound effects player as a single cue. It does the following:

1. Creates a sound effect for each sound file specified in the m3u file. All of the parameters take their defaults.
2. Creates a Start Sound sequence item for each sound effect. Each sequence item, except the last, plays the next when it completes or terminates. The last plays the first, forming a loop. The text of the Start Sound sequence item is the name of the sound file.
3. Creates an Operator Wait sequence item which will play the first Start Sound sequence item when the operator pushes the Play button. The text displayed to the operator is the name of the m3u file.
4. If there is no Start Sequence sequence item, creates one and makes the above Operator Wait sequence item its successor.
5. Positions the sequencer to the Operator Wait sequence item created above.

After running the tool the operator can play the list of songs by pressing the Play button. The Stop button on the cluster that is playing the songs causes the player to skip to the next song. The Reset button stops the player.

The operator can edit the Operator Wait sequence item to be an Offer Sound sequence item, specifying the cluster that is to play the songs.

Note that the m3u file can contain names of directories in addition to names of sound files. In that case the directories are scanned for sound files.

Create a Play List from a Folder of Sound Files

This tool provides a convenient way to turn a directory of sound files into a list of cues, any of which can be played on demand. It does the following:

1. Creates a sound effect for each sound file in the directory. All of the parameters take their defaults.
2. Creates an Operator Wait sequence item and a Start Sound sequence item for each sound effect. The text of the Operator Wait sequence items is the sequence number of the sound file followed by its name.
3. Links each Operator Wait sequence item to its Start Sound sequence item as its next item to execute. The Start Sound sequence item has the next Operator Wait sequence item as the item to execute when the sound completes or is terminated. If there is no next Operator Wait sequence item, it links to the first.
4. If there is no Start Sequence sequence item, creates one and makes the first Operator Wait sequence item its successor.

After running the tool, the operator can scroll among the Operator Wait sequence items for the song he wants, and press his Play button to play it. If the operator ends the song early using the Stop button on the song's cluster, or if the song completes, the next song on the play list appears and the sound effects player waits for the operator to play the next song by pushing the “Play” button or select a song to play from the displayed list.

Sound Effects Player Design Summary

When a sound designer constructs sound effects, he creates the waveform for each channel using a tool like Audacity. He also specifies the envelope numbers and other information that control how the sound effect is played. Each sound effect is described in an XML file containing the envelope and other information, plus the name of the file containing the waveforms. A waveform file can be used by more than one sound effect. Sounds can be routed to up to 8 independent output channels.

Unless the show is very simple, the sound effects designer will also construct a collection of sequence items. They will walk the sound effects operator through the show, giving him buttons to play the appropriate sounds at the appropriate times. The sequence is also contained in an XML file.

Implementations

I imagine two implementations of the sound effects player. One is a software-only implementation based on gstreamer 1.0. The other is a self-contained fancy console. I am going to write about the

fancy console first, then consider how to approximate it using standard computer parts.

Fancy Console

The largest section of the fancy console contains 16 clusters, arranged in a 4 by 4 grid. Each cluster consists of two buttons marked Start and Stop, each back-lit, a soft dial with click stops, a small display near the dial, and a one-line display at the top of the cluster. When the operator pushes a Start button, the appropriate sound effect is started, and the Start button is lit. When the sound effect finishes playing, the lamp goes out. Pressing the Start button while the lamp is lit has no effect.

If the operator presses the Stop button while the Start button's lamp is lit, the Stop button's lamp is lit and the sample begins its release. During release both buttons are lit, and when the release is complete the Stop button goes out along with the Start button. Pressing the Stop button while its lamp is lit has no effect.

The lamps and buttons work the same way if the start and stop come from the internal or an external sequencer instead of the operator pressing the Start and Stop buttons. When the internal sequencer is running it controls the text at the top of each cluster. Otherwise the text is the name of the sound effect.

When a sound effect is playing, the small display in the cluster shows the operator volume. It starts at 100%, which is the external representation of 1.0, but the sound effects operator can adjust the volume by turning the soft dial. By pushing the dial as though it were a button, it is switched to control the pan. The display shows Center, L followed by a number if the pan is to the left, or R followed by a number if the pan is to the right. If the WAV file being played is stereo, this functions as a balance control. The sound designer will make the pan function unavailable if the WAV file has more than two channels.

Above the 16 clusters are the general controls.

1. A bank dial to specify which of the eight banks of samples the 16 clusters refer to.
2. Two program dials, one with 16 positions and the other with 8, to specify the program number.
3. A sequencer section with
    1. a small text display which shows the current Operator Wait item, plus a few items before and after the current item in alphabetical order,
    2. a dial to rapidly scroll forward and backward in the sequence; pushing it repositions the sequencer at the designated item,
    3. a large Play This Sound Effect button which lets the sequencer proceed to its next item, and
    4. a small lamp that lights when the sequencer is waiting for the operator.

4. A Pause button, which causes the clock to stop advancing in the gstreamer looper element and forces it to output only silence. The button lights when pressed, and pressing it again turns off the light and turns off the pausing condition in the gstreamer looper element.
5. The master volume control, a soft dial with an adjacent 2½-digit display. It defaults to 1.0, which is displayed as 100%. All eight channels of output are multiplied by the master volume control value before output.
6. A mute button. The mute button is back-lit and functions like the pause button. When the sound effects player is muted the looper runs normally but the outputs are silenced.
7. A single-channel stack of LEDs which functions as a sound meter. The sound meter can be set to any channel of output, and it is before the mute button in the signal chain. Depending on how much room there is on the console, we might make this eight stacks of LEDs and no switching control, or two stacks and a four-position switch.
8. The Reset button stops all processing and initializes the device. If there is a sequence defined, it is started at its Start Sequence item.
9. A display of the amount of time remaining in the current sound. It can be switched to show the amount of time the current sound has been playing. When the internal sequencer is running, it specifies which sound is current. Otherwise, the last sound triggered by the operator is the current sound.
10. The Shutdown button stops all processing, waits for all file writing to complete, and turns off power.

When the sequencer plays a sound effect, it designates one of the clusters as the operator control for the sound effect, and that cluster's Start button lights. If the sound effects operator wishes to stop a sound effect he presses the cluster's Stop button. The sequencer can be placed in edit mode, in which it does not perform sequence items but instead allows them to be created, deleted, modified and renamed. Information is shown on the display and can be manipulated using the dials and buttons normally used to run the cues. Pressing the large Play This Sound Effect button plays the current sequence item and all following items just as it would during a performance until all forks are at an Operator Wait item, but when all the sounds are complete the current item returns to the item that is being edited.

The sound effects can also be edited. In sound effects editing mode, the text at the top of each cluster shows the name of a sound effect. Pressing a Start button chooses a sound effect for editing. The operator uses the sequencer controls to scroll through the envelope and other information provided by the sound designer. Each of the parameters can be changed, but there is no provision for editing the waveform files. Pressing the large Play This Sound Effect button plays the sound effect currently being edited. This can be alternated with adjusting the value dial to hear how changing the current parameter

affects the current sound effect. When a sound effect is playing its Start button is lit. Pressing the corresponding Stop button causes the sound effect to release. Sound effects can be created and deleted.

The console has an Ethernet port so it can accept commands from an external sequencer. It uses RTP-MIDI (RFC 6295) as implemented by Apple (AppleMIDI) and by Mackie on their DL-series mixing desks. It also supports the Behringer X32 producer, both MIDI and OSC. It supports both IPv4 and IPv6. It also supports Zeroconf (RFC 3927).

The console has a USB interface which allows a thumb drive to be connected, so sound files prepared externally can be loaded by the tools, or using the sound effects editor.

Sound output is provided by eight XLR connectors. Digital output is also a possibility, using the Ethernet connector. Storage of WAV files and the other information is provided by an SD card slot, protected by a latching cover The files on the SD card can be edited in a computer, which is also the source of the WAV files. The files on the SD card have the same format and organization as the software implementation of the sound effects player, so the software implementation can be used to develop sound effects for the fancy console.

If the software internal to the console needs to be updated, new software can be loaded from the SD card or from the USB interface.

To prevent an inexperienced sound effects operator from inadvertently damaging the files, the latching cover for the SD card can include a key lock, which prevents the SD card from being removed without the key, and also signals to the console that the SD card is not to be written. In addition, the SD card can be write locked using a switch on the card. If more protection is considered necessary, there can be a slide switch inside the SD card cover which prevents placing the console in an editing mode.

There is a Setup button which provides access to options which do not need to be changed during a performance. These include MIDI channel, MIDI Show Control device ID, MIDI Show Control group ID, MIDI Show Control UDP port number and the routing of each channel of each sound effect to the eight outputs. Also in the setup group is a button to format the SD card and write an empty project on it. A project can be loaded from a thumb drive on the USB interface, in which case the SD card is formatted and receives the loaded project. A backup copy of the current project can be written to the thumb drive.

Software simulation

Having indulged my fantasies by imagining a fancy sound effects console, I will now be more practical and imagine how it might be implemented using a standard computer. The program will be written as a gstreamer 1.0 application, so it can use the gstreamer plugins described above to play sounds.

The program has four modes: Run Sequencer, Run Non-sequencer, Edit Sound Effects and Edit

Sequence. In Run Sequencer mode the screen will show a scrolling list of Operator Wait sequence items. When the sequencer is waiting, the operator presses the space bar to play all the sound effects until the next Operator Wait. The operator can also use the up and down arrows, and the Page Up and Page Down keys, to scroll through the Operator Wait sequence items, to deal with a jump in the performance or choose songs to play from a play list.

In Run Non-sequencer mode the operator will be shown a scrolling list of sound effects. He can use the up and down arrow keys, and the Page Up and Page Down keys, to scroll among them. When he presses the space bar the current sound effect plays, and when it starts to release the next sound effect becomes the current sound effect. If he presses the Backspace key, this forces the current sound effect to start its release immediately, the same as receiving a Note Off from an external sequencer.

In Edit Sound Effects mode the scrolling list of sound effects shares the screen with a layout of all the parameters of the current sound effect. The values can be adjusted by using the usual GUI facilities, which includes typing new values into each parameter's cell. Pressing shift space plays the current sound effect but does not change which sound effect is current. Pressing shift backspace does a Note Off if there is a sound effect playing. There are provisions for creating, deleting and renaming sound effects, but WAV files are prepared by an external program. Shift combined with up arrow, down arrow, page up and page down navigate through the list of sound effects.

In Edit Sequence mode the scrolling list of sequence items (not just Operator Wait sequence items) shares the screen with a layout of all of the parameters of the current sequence item. The values can be adjusted by using the usual GUI facilities, which includes typing new values into each parameter's cell. Pressing shift space plays the current sequence item and all following items until an Operator Wait. Pressing shift backspace does a Note Off if there is a sound effect playing. There is provision for creating, deleting, and renaming sequence items. Shift combined with up arrow, down arrow, page up and page down navigate through the list of sequence items. The items are ordered by their names, and changing the name of a sequence item changes its position in the list, and all references to it. Because every sequence item names its successor(s), changing a sequence item's name does not change any path through the sequencer.

In all four modes a section of the screen shows a scrollable list of clusters, which are complex boxes containing a volume or pan display, a start button, a stop button, and a text label over the top. In Run Non-sequencer mode and Edit Sound Effects mode each sound effect is assigned a cluster based on its MIDI program number and note number, if these are specified; otherwise the cluster is assigned based on the sound effect's order in the project file. A sound effect can be started by clicking on its Start button, and stopped by clicking on its Stop button. When the sound effect is playing, its Start button is emphasized. When it is releasing, its Stop button is also emphasized. Clicking anywhere in the cluster selects it. The keypad plus and minus keys adjust the selected sound effect's operator volume. Shift combined with the keypad plus and minus keys adjust the pan. In Edit Sound Effects mode the

selected sound effect has its parameters displayed in the control section of the display, and these parameters can be edited.

In Run Sequencer mode and Edit Sequence mode, the contents of this region is controlled by the sequencer, which can place clusters where it wishes when it offers or plays sound effects. If the operator selects a cluster it remains selected until it leaves the screen or the operator selects another cluster. Adjusting the operator volume for a cluster carries over to every sound effect played in that cluster until the cluster leaves the screen.

When the program starts it can Open a project file containing global information about the sound effects for a play, including the file name for each sound effect. Each of these sound effect description files contains the path to the WAV file that contains the waveforms, and the values for the other parameters set by the sound designer for each sound effect. The project file may also name the file which contains all the information for the internal sequencer. All of these files can be written using the Edit modes.

The project file is an XML file. For convenience of editing, the XML file can refer to other XML files to contain various parts of the project information. When the sound effects player reads the project file it ignores information intended for other components of show_control.

A function key can be tied to a sequence item or sound effect. Pressing the function key works like receiving the MIDI Show Control command Go.

The screen also includes a pause button, a mute button, a master volume control, sound level meter, a Reset button and a time remaining display. See the description of the fancy console for what these controls and displays do.

The application accepts commands from an external sequencer. It uses RTP-MIDI (RFC 6295) as implemented by Apple (AppleMIDI) and by Mackie on their DL-series mixing desks. It also supports the Behringer X32 producer, both MIDI and OSC. It supports both IPv4 and IPv6 using the facilities of its operating system.

Output can be directed to the computer's native sound output connectors, or to a USB sound module, such as the M-Audio Fast Track Ultra 8R, which has eight balanced ¼-inch phono connectors for output. Digital output is also possible, using the computer's Ethernet interface.

To prevent an inexperienced sound effects operator from inadvertently damaging the files, they can be protected from writing by the sound effect operator's account using operating system facilities.

There is an Edit Preferences function which provides access to options which do not need to be changed during a performance. These include MIDI channel, MIDI Show Control device ID, MIDI Show Control group ID, MIDI Show Control UDP port number and the ALSA device that outputs the

sound. There is also provision for editing the routing between each channel of each sound effect and the eight outputs.

If a project is being loaded and a file referenced in it cannot be found, a requester asks the user to locate the file. If he is successful in locating the file, the parts of the old and new file paths are remembered. If a subsequent file cannot be found, an attempt is made to find the file by substituting the new part of the file path for the old. If the file is found, the user is not notified. This provides a convenient way to reorganize the files of a project, which might be done when moving a project from one computer to another. The new locations of the files are saved when the project is saved.

City Street example

Imagine a scene that takes place on a city street. There are low-level environmental sounds, which are heard throughout the scene. In addition, there are louder sounds which come at certain points in the dialogue, such as a police siren or a bus stopping to load and unload passengers. The volume of the background sounds will likely need to be adjusted during the scene, but precisely when and by how much cannot be determined in advance.

We will have a sound effect which holds 15 minutes of low-level background sounds, looped. This sound effect has a 3-second attack to full volume, 0 decay, full volume sustain, and 3 second release duration time, which is triggered at the end of the scene. The WAV file is 15 minutes and three seconds long, the loop from time is set to 15 minutes, and the first three seconds of the file are duplicated at the end.

Each of the other sounds has its own sound effect. The sequence would be constructed like this:

1. Operator Wait, text “scene starts”, next is item 2.
2. Start background sound effect, cluster 1, text “background”, tag “street_noises”, when it starts go to item 3.
3. Operator Wait, text “horn honks”, next is item 4.
4. Start horn sound, cluster 2, text “horn honking”, when it completes or terminates go to item 5.
5. Operator Wait, text “siren”, next is item 6.
6. Start siren sound, cluster 2, text “siren”, when it completes or terminates go to item 7.
7. Operator Wait, text “scene ends”, next is item 8.
8. Stop Sound tag “street_noises”, next is item 9, which is the Operator Wait for the start of the next scene.

If there also needs to be a ringing telephone, it can be included based on the Telephone example.

Because the background sound appears to the operator in a fixed place on his screen, he can click on it and thereafter adjust its volume using the numeric keypad, independent of using the space bar to start other sound effects.

The Passion of Dracula example

The most complex sound effects I have done were for The Passion of Dracula. It has music, wind, hounds, rats and heartbeat background sounds, which overlap each other and the many foreground sounds, such as thunder, various explosions and wolf howls. I ran the show without a sequencer, using a Zoom R16 and a Roland SP-404SX. One effect the director asked for that I could not do was slow down Dracula's heartbeat near the end of the show. The design of this sound effects player has been motivated by a wish to be able to play the sound effects from this show using a sequencer.

I am not going to work out the whole sequence for the show here—there are about 120 sound cues. Instead I shall just sketch out how to construct it.

The music is always on cluster 0. At the top of the show it is an offered sound. The operator turns the operator volume down to 0.0 and starts it. When the sound effects operator wishes to bring it up, he selects it and uses the mouse to raise its volume, and lower it later. On the fancy console he just turns its volume dial for cluster 0. The music sound effect is looped and at full volume, with the sound designer's volume set so that raising the operator's volume to 100% plays the music at the loudest level needed by the show.

The wind is an offered sound in cluster 1. The sound effects operator starts it when the doors open, and stops it when they close. The close time cannot be predicted because an actor may forget to close the doors on cue, and close them later when he remembers. The amplitude envelope for this sound effect has an attack for the initial inrush of air, a decay to a reasonable sustain level if the door is left open, and a release which models closing the door.

The hounds, rats and heartbeat never sound together, so they are played in cluster 2 under control of the sequencer. The sequencer also stops them when they are done. The other sound effects do not overlap each other, so they are played by the sequencer and presented in cluster 3.

The heartbeat is a complex sound, like the ringing of a telephone, alternating from a beat to the time between beats. The first time the sequencer signals that the sound is to stop, it transitions to a slower rhythm. The second time it stops abruptly.

The pre-show music is a series of songs. They run in a loop until the sound effects operator signals that the show is about to start, when they fade out over three seconds. This is done like the telephone example, but with a sequence item for each song in the loop, and no ringout. Alternatively, they can transition to a pre-show safety announcement.

Doing Standard Things

There are certain things a sound effects player is expected to do. In this section I will explain how this sound effects player does those things.

Basic Controls

1. Play a sound. In non-sequencer mode, all the sounds are visible, each with a Start and Stop button, and a volume control. By using the internal sequencer very complex sounds can be made using the same interface. MIDI Note On, OSC and MIDI Show Control can be used to play sounds remotely.
2. Stop playing a sound. As described above, each sound has a Stop button. MIDI Note Off, OSC and MIDI Show Control can be used to stop sounds remotely.
3. Stop All. The MIDI command All Sound Off can be used to stop all sounds remotely. There are also the Pause and Mute buttons which can be operated remotely using MIDI Show Control.
4. Fade. As described above, each sound has a volume control. In addition, he volume of any sound can be controlled remotely using the MIDI Show Control command Set.
5. Fade All. There is a master volume control, which can be operated remotely using MIDI Show Control. In addition, the MIDI command All Notes Off can be used to fade all sounds remotely.
6. Advance. The operator can skip to the next Operator Wait sequence item using the sequencer controls. This can be done remotely using the MIDI Show Control command Standby_+.
7. Rewind. There is a local Reset button. This can be done remotely using the MIDI Show Control command Reset.

Edit or Create a Cue

1. Select the sound file. This is done with the File Name parameter in Edit Sequence mode.
2. Set levels: the designer volume is a parameter of each sound effect.
3. Set EQ: There is an equalizer for each sound effect. The parameters are set by the sound designer, but can be modified by the sound operators. They are saved with each sound.
4. Set Echo/Reverb: There is an echo processor for each sound effect. Like the equalizer the parameters are set by the sound designer, but can be modified by the sound operators, and are saved with each sound.
5. Set fade in/out, cross fade (levels and time). Fade in is done using the attack duration time and level parameters. Fade out is done with release start time and release duration time. Cross

fading is done by starting the new sound when the old one begins its release.

6. Play a sound on operator input or receipt of a command from a remote sequencer: Any sound presented to the operator can also be triggered using MIDI Note On, OSC and MIDI Show Control.
7. Fade an active cue remotely: On receipt of a Note Off, a sound can fade out or transition to one with a lower volume using the Start Sound sequencer item which forks to a new sound on termination of this one. Also, the MIDI Show Control command Set can be used to change the volume of any playing sound.
8. Pause a playing sound: The operator has a Pause button. This can be done remotely using the MIDI Show Control command Pause.
9. Background/Independent: Background or independent cues can be played by forking the sequencer, using Start Sound and specifying sequence items to be executed when the sound starts, when it enters the release segment of its ADSR envelope, and when it completes or is terminated. These cues can be acted upon externally since they have MIDI note numbers, and the Stop All and Fade All commands control them.

Summary of Significant Changes

November 11, 2014

1. Added the section Doing Standard Things in response to Mac's review of the November 10 edition. Adding that pointed out the need for some additional controls and remote commands. That led me to MIDI Show Control.
2. Added support of MIDI Show Control in addition to the usual Note On and Note Off commands.

November 14, 2014

Mac showed me Cueplayer, a very good sound player. To achieve feature parity with it, I added the following features:

1. Allow sound to be re-channeled into the eight outputs
2. Replace the 3-band equalizer with a 10-band equalizer.
3. Allow the sound rehearsal operator to start a sound later than its normal start point.
4. Display the time left in the current sound
5. Let a function key start a sound

6. Add panning of monophonic and stereo sound effects.
7. Add tools to create play lists from imported sound files or m3u files.
8. Provide a convenient way to relocate sound files.

Also, I acknowledged the role of the rehearsal sound operator as a third user of the sound effects player.

November 15, 2014

Make the goal of not requiring a high-speed CPU explicit, and use this goal to justify the fixed gstreamer pipeline.

December 21, 2014

Based on what I have learned about gstreamer, update the volume definition to be a floating-point number between 0.0 and 4.0, and update the time base to nanoseconds.

January 4, 2014

It is actually the adder, not the funnel, that provides the mixing of many sounds into one. Also some updating of the gstreamer information based on prototyping.

May 3, 2015

Update some of the gstreamer information based on getting the envelope plugin nearly working. There are now just two custom plugins: looper and envelope.

June 14, 2015

Do some more updating of the gstreamer information based on getting both plugins working. The plugins are well-commented, so there is some overlap between the information presented here and the information in the source code.

June 23, 2015

Add tagging of playing and offered sounds. This simplifies the stop sound and stop offering sound sequence items.

June 27, 2015

Did some minor wordsmithing.

June 28, 2015

Some small changes to the sequencer.

October 20, 2015

Don't let the sequencer modify the volume or pan. Use multiple sounds instead, sharing the WAV file.

October 12, 2016

Update the title and running header to better describe this document.

October 20, 2016

1. Improve the handling of release start in the sequencer,
2. add better control of panning,
3. improve the display of the current and nearby sequence items, and
4. add max duration.

October 23, 2016

Minor edits for clarification.

November 9, 2016

More minor edits for clarification.

May 29, 2017

Rename the Stop sequence item Stop Sound.

July 10, 2017

Add the Cancel Wait sequence item, to increase the capabilities of an external sequencer.