David Cordero Placer

Índice:

1.- Introducción

1.1 ¿Qué es ecasound?

Ecasound es un paquete de software desarrollado para multiprocesamiento de audio. Puede ser usado para tareas simples similares a la reproducción de audio, grabación y conversión de formatos, pero también para el procesamiento de efectos multipista, mezclando, grabando y reciclando señales.
Ecasound soporta un amplio rango de entradas de audio, salidas y algoritmos de efectos. Efectos y objetos de audio pueden combinarse de varias formas, y sus parámetros pueden ser controlados por el operador de objetos parecido a los osciladores y MIDI-CCs. Incluye una consola muy versátil a modo de interfaz de usuario en el paquete.
A todo esto hay que añadirle que es una herramienta de procesamiento de audio en tiempo real, aun que esto no se cumpla estrictamente, ya que para ello necesitaríamos correr nuestro ecasound bajo un sistema en tiempo real.

1.2 Historia

Las primeras versiones de ecasound funcionan bajo IBM Os/2. Al portar ecasound para GNU/Linux se reescribio mucho código y durante este proceso se le añadieron nuevas funcionalidades como la capacidad de trabajar con multipista. Mucha gente es la que se ha decantado por esta opcion de procesamiento de audio para todos sus proyectos de música.

2.- Conceptos de Ecasound

2.1 Objeto audio.

Los objetos de audio son usados para transferir sonido desde y al ecasound. Normalmente los objetos de audio son ficheros (wav, mp3 o ogg) o dispositivos de entrada/salida. Hay algunos tipos de objetos especiales de audio para transferencias entre aplicaciones.

2.2 Chain (cadena)

Chain es la abtraccion central del flujo de señal. En muchos casos los chains son similares a los cables de audio, como tener una entrada y una salida la cual puedes conectar a un productor y un consumidor como puede ser una guitarra y un amplificador.
Pero hay algunas diferencias. Primero que es posible conectar un operador de chain (normalmente efectos) a un chain. Esto es algo parecido a reemplazar un cable por dos, y poner una caja de efectos entre ellos, pero con chains es mucho más fácil. Una segunda importante diferencia es q los chain pueden transportar multiples canales de audio. Es posible conectar amplificadores de sonido mono, estéreo ó de 24 canales (incluso más) a un solo chain. También los operadores de chain pueden manejar estos flujos multicanal.
Hay que añadirle a los operadores de chain, que estos también tienen funciones de "mute" y "bypass"

2.3 Operadores chain y controladores.

Los operadores chain son usados para procesado y analisis de muestras de datos. Pueden ser divididos en puentes, convertidores, analizadores de señal y tradicionales efectos de reverberaciones, retardos y filtros.
Esto es posible al conectar un controlador especial de objetos chain. Pueden ser usados para controlar los parámetros de operador de chain.
Los ejemplos típicos de uso son varios osciladores y controladores continuos MIDI.
Ambos tipos de objetos son conectados a chains. El término chain object (objetos cadena) se refiere al todos los objetos que pueden ser conectados a las cadenas (chain)

2.4 Configurador de chain

El configurador de cadenas (chainsetup) es el objeto de datos central. Todos los otros objetos (entradas, salidas, chains) se conectan a un configurador de chain. La mayoria de los configuradotes de chain pueden existir al mismo tiempo (durante una sesión), pero solo uno de ellos puede ser usado. En la documentación de Ecasound, el término conectado se usa para describir el configurador de cadena que está en uso.
Otro importante concepto de configurador de chain es el de un configurador de chain seleccionado. Todas las operaciones de edición se hacen mientras el configurador esté seleccionado. Es posible tener un configurador conectado (mientras procesa audio), mientras editamos otro, el configurador que este seleccionado para editar.
Cargando y guardando configuradotes de chain, es el principal mecanismo para almacenar y restaurar el estado de la información. Cuando guardamos ficheros, se usa el formato de ficheros .ecs.

La sintaxis usa la misma notación que el interface de la consola de Ecasound. Esto hace fácil poder editar un fichero de un configurador guardado, manualmente o usando utilidades externas.

2.5 Posición actual

La información a cerca del estado actual solo se almacena para los objetos de audio y configuradotes de chain. Cuando se cambia la posición de un configurador, todos los objetos de sonido se ven afectados. Por el otro lado, las posiciones de todos los objetos de sonido se pueden cambiar independientemente.

2.6 Ecasound Control Interface – ECI

El ECI de ecasound es una API para desarrolladores de aplicaciones que quieran usar las ventajas de las librerias de ecasound en sus propias aplicaciones.

2.7 Ecasound Interactive Mode – EIAM

La mayoría de las funcionalidades de Ecasound están situadas en la librería central (libecasound). Una cosa que esta librería nos da, es un interprete simple, que podemos usar para controlar ecasound. Esta manera de usar Ecasound es el que se conoce como EIAM. La mayoria de los frontend comunes para EIAM es el modo consola del programa Ecasound. Para entrar en el modo interactivo se usa el parámetro –c

2.8 Opciones de sintaxis. EOS

Una forma muy buena de usar el modo consola de ecasound, es la opción de la sintaxis por línea de comandos. Se pueden hacer muchas cosas alucinantes desde la linea de comandos. Pero esto no se acaba con la linea de comandos de ecasound. De hecho, interpretando estas opciones que están en la librería principal de ecasound esta muy atado al modo interactivo. Como resultado la sintaxis que se usa es la misma que se usa en varias partes de la librería libecasound.

3.- Usando Ecasound

3.1 Por donde empezar

No hay un único camino correcto para empezar a usar ecasound. Usarlo puede ser tan simple como pegar un componente para que haga tareas que no son manejadas por otras aplicaciones que se están usando o porque ecasound hace estas tareas mucho mas fáciles (o mejor ☺)
Pero ecasound también puede usarse como el centro del estudio de sonido, procesando efectos, grabando multipistas y mezclando.
Es difícil describir toda la flexibilidad que ecasound nos da en pocas frases, porque el comienzo sólo viene descrito en el manual de nuevos usuarios de la documentación, el resto es cosa de nuestra imaginación.
El manual de introdución no es una perfecta introducción pero nos da una vista global del potencial de la aplicación mostrandonos el uso de muchas tareas sencillas y comunes para hacer.

3.2 Reglas de edición de configuradotes de chain.

Aquí están algunas de las reglas que ayudan a escribir configuradotes correctos. Cuando editamos ficheros de configuradotes, con frontend gráficos o simplemente usando la linea de comandos, las reglas a aplicar son las siguientes:

- Cada chain tiene exactamente una entrada y una salida.
- Todas las entradas y salidas deben estar conectadas a algún chain.
- Para cada entrada/salida, hay una y solamente una definición (Ej. “–i:file.wav”)
- Todas las rutinas desde y a los chain, están basadas sobre la selección de chains y luego especificando una entrada o una salida (Ej. “-a:1,2 –i:file.ext”)
- Todas las copias y mezclas de sonido es hecha channel-wise. Si conectamos una entrada de 4 canales y una salida de 2 a la cadena, esta, tendrá 4 canales de sudio, pero solo los 2 primeros de la entrada serán escritos en la salida.

3.3 Operadores de cadena y controladores

La mejor manera de aprender el uso de estos operadores es con el manpage-1 de ecasound, que tiene toda la información propia de los objetos chain.

3.4 Configuración

Las preferencias de ecasound se guardan en un archivo llamado /.ecasound/ecasoundrc. Por defecto los ficheros de efectos y preconfiguraciones de los osciladores estan en /share/ecasound/

4 Interfáz de usuario y aplicaciones.

4.1 Ecasound

"ecasound" es la aplicación primaria de interfaz con el usuario.

4.2 Ecasignalview

Ecasignalview es una utilidad para monitorizar la amplitud de la señal y estadisticas de los picos. Con ella podemos ajustar los niveles de señal para la grabación.

4.3 Uso básico

El escenario básico de uso es la grabación de sonido desde la tarjeta de sonido, visualizándolo con medidores de amplitud y escribiendo la salida a /dev/null

```
# OSS-drivers (or properly installed ALSA OSS-emulation)
ecasignalview /dev/dsp null
# native ALSA-mode, recording from the 'default' device
ecasignalview alsa,default null
```

Es posible resetear el pico máximo y los contadores de muestras con clipping enviando una señal de SIGHUP al proceso. (Ej. "killall –v –HUP ecasignalview")
Podemos usar alsamixer (ALSA), aumix (OSS) u otras aplicaciones para activar o desactivar las entradas de la tarjeta de sonido. Otra opcion es usar ecasignalview para monitorizarlas. En este caso el comando correcto será:

```
# OSS input and output
ecasignalview /dev/dsp /dev/dsp
# corresponding ALSA command
ecasignalview alsa,default alsa,default
```

Las opciones de ecasignalview permiten un ajuste fino para la monitorización.

```
# increased refresh rate 20Hz
ecasignalview -r:50 /dev/dsp null
# larger buffersize (1024 samples)
ecasignalview -b:1024 /dev/dsp null
# recording in mode 32bit/10channels/96000Hz with
# interleaved channels
ecasignalview -f:s32,10,96000,i /dev/dsp null
```

5 Ventajas avanzadas

5.1 Audio loop devices.

Usando solamente una conexión chain normal no es posible encaminar audio desde ecasound a otra cadena. La forma de hacer esto es mediante dispositivos de bucle (loop devices)

5.2 Ejemplos de uso.

Mezclando

Un uso normal donde nosotros encaminamos sonido desde chain "1" y "2" al chain "3" el cual está conectado a la salida de la tarjeta de sonido.

```
# note, the second loop parameter is the loop id-number;
# it is used to associate loop inputs with correct loop outputs
ecasound -a:1 -i:some.mp3 -o:loop,1
-a:2 -i:another.mp3 -o:loop,1
-a:3 -i:loop,1 -o /dev/dsp -ea:200
```

Las dos entradas son encaminadoas a chain "3", donde a –ea:200 es aplicado a la señal. Esto hace que tenga un inconveniente, el dispositivo de buble añade latencia (-b:x -> latencia de x frames)

Conversiones de formato

Convertimos de wav a cdr.

```
ecasound -i:somefile.wav -o:somefile.cdr
ecasound -i somefile.wav -o somefile.cdr

ecasound -c -i somefile.wav -o somefile.cdr
```

Parecido al anterior pero este comienza en modo interactivo.

Conversiones de formato resampleando

```
ecasound -f:16,2,96000 -i resample,auto,foo44100.wav -o bar96k.wav
ecasound -f:16,2,44100 -i resample,auto,bar96k.wav -o foo44100.wav
ecasound -f:16,2,44100 -i resample-hq,48k,foo48k.wav -o bar.wav
ecasound -f:16,2,44100 -i resample,96k,third.raw -o foo44100.wav
```

En el último ejemplo resampleamos de 48000Hz a 44100Hz, usando el "resimple-hq" (de alta calidad). Este modo tiene que ser compilado en Ecasound por ser un modo que consume mas recursos de cpu, pero da mejores resultados.

Salidas en tiempo real

```
ecasound somefile.wav
```

ecasound -i somefile.wav
ecasound -i:somefile.wav
ecasound -i somefile.wav -o /dev/dsp
ecasound -i somefile.mp3 -o alsahw,0,0
ecasound -i somefile.mp3 -o alsaplugin,0,0
ecasound -i somefile.mp3 -o alsa,soundcard_name

Los drivers ALSA y OSS tienen diferentes parámetros y tenemos que tenerlo en cuenta si usamos /dev/dsp

mpg123 -s sometune.mp3 | ecasound -i:stdin -o alsahw,0,0
Envía la salida de mpg123 a la salida estandar (opción –s) y lee de la entrada estandar con escaound (opción -i:stdin) Si queremos usar soporte nativo ALSA con programas OSS, esta es una manera de hacerlo facil. También puede ser usado para añadir efectos a los flujos de datos.

Entradas en tiempo real (grabando desde la tarjeta de sonido)

ecasound -i:/dev/dsp0 -o somefile.wav
ecasound -i:/dev/dsp0 -o somefile.wav -c
ecasound -i alsahw,1,0 -o somefile.wav

Procesamiento de efectos

Ecasound es una herramienta muy versátil para procesado de efectos.

ecasound -i somefile.mp3 -o /dev/dsp -ea:120
ecasound -a:default -i somefile.mp3 -o /dev/dsp -ea:120
Este ejemplo hace que la entrada de un mp3 tenga un efecto de amplificación y lo escuchemos por la salida OSS amplificada en un 120%.

ecasound -i somefile.mp3 -o /dev/dsp -etr:40,0,55 -ea:120
Parecido al anterior ejemplo, pero ahora con un efecto de reverberación con un retardo de 40ms, suround desactivado y mezcla del 55%, añadido todo al chain (cadena) antes del efecto de amplificado.

ecasound -a:1,2 -i somefile.mp3 -a:all -o /dev/dsp \
-a:1 -etr:40,0,55 -ea:120 \
-a:2 -efl:400
Veámos un ejemplo de procesado paralelo. Dos chain son creadas y el fichero de salida es el mismo asignado a ambas. Esto permite añadir efectos a varios chain y mezclarlos por la misma salida. Se pueden crear tantos chain como queramos.

Usando controladores de fuentes con efectos

ecasound -i somefile.wav -o /dev/dsp -ef3:800,1.5,0.9 -kos:1,400,4200,0.2,0 -kos:2,0.1,1.5,0.15,0

ecasound -i somefile.wav -o /dev/dsp -ef3:800,1.5,0.9 -km:1,400,4200,74,1 -km:2,0.1,1.5,71,1
El primer ejemplo usa 2 osciladores seno ('-kos:parameter,range_low,range_high,speed_in_Hz,initial_phase') para controlar una filtro resonante de paso bajo. La frecuencia de corte varia entre 400 y 4200Hz mientras la resonancia varia entre 0.1 y 1.5. La fase inicial es 0 (veces pi). El segundo ejemplo usa controladores continuos MIDI ('-km:parameter,range_low,range_high,controller_number,midi-channel') como controladores de la fuente. Los rangos son los mismos que en el primer ejemplo. El controlador usa 74 de cutoff y 71 de resonancia.
Es también posible controlar controladores con otros controles usando la opción "-kx". Normalmente cuando se añade un controlados, estamos controlado el ultimo operador especificado de chain. "-kx" cambia esto. Veamos este ejemplo:

ecasound -i file.wav -o /dev/dsp -ea:100 -kos:1,0,100,0.5,0 -kx -kos:4,0.1,5,0.5,0

Mezclando

Veámos unos ejemplos comunes de mezclado.

ecasound -c \
-a:1 -i drums.wav \
-a:2 -i synth-background.wav \
-a:3 -i bass-guitar_take-2.ewf \
-a:4 -i brass-house-lead.wav \
-a:all -o /dev/dsp
Primeramente, tenemos seleccionado el modo-interactivo con "-c". Luego añadimos 4 entradas. Todas las 4 chain son asignadas a una única salida, la cual esta vez es la tarjeta de sonido (/dev/dsp).

5.3 Ficheros Wave en ecasound – el formato ewf

Los ficheros wave ecasound (.ewf) son un simple formato envoltorio para controlar otros objetos de sonido. Los ficheros ewf son usados para ficheros de offsetting o time-shifting (para instanciar un clip de sonido en el medio de una mezcla multipista mas grande, minimizando el espacio de disco usado durante la grabación multipista y looping.

5.3 Formato de ficheros

Los ficheros ewf son almacenados en formato ASCII. La sintaxis se basa en parejas de "key=valor". La misma sintaxis se usa en los ficheros de recursos de Ecasound. Se reconocen las siguientes palabras clave en ficheros ewf:

- source – nombre del objeto de sonido [read,write]
- offset – objeto de sonido insertado en un offset (segundos) [read,write]
- start-position – comienzo del offset dentro del objeto de audio (segundos) [read]
- lengh – como de largo son los datos del objeto de sonido usado (segundos) [read]
- looping – si existe datos de muestra de bucle (true or false) [read]

5.4 Ejemplos de uso de ewf

Echemos un vistazo a un ejemplo sencillo de ewf:

-- test.ewf --
source = test.wav
offset = 5.0
start-position = 2.0
length = 3.0
looping = true

Ahora que ocurre cuando usemos “ecasound –i test.ewf –o /dev/dsp”. Al tener el offset 5.0, los primeros 5 segundos serán silenciados, después de esto ecasound empezara a leer datos desde test.wav. Pero como la posición de comienzo no es 0, saltara 2 segundos. Después de 8 segundos (“offset+lengh”), ecasound repetirá el bucle hasta que el usuario aborte la ejecución.

5.5 Efectos preconfigurados

5.5.1 General

Ecasound tiene un potente sistema de efectos preconfigurados que permiten crear nuevos efectos combinando efectos básicos y controladores.
Los efectos preconfigurados pueden almacenarse en ficheros separados o de forma global en la base de datos. Cualquier camino para preconfigurar formatos es el mismo.

preset_name = effects controllers | ... | effects controllers

Efectos y controladores son especificados usando sintaxis EOS, la misma sintaxis que es usada por línea de comandos (“-ea:100”, “-kl:1,0,100,5”, etc). El carácter de tubería “|” se usa para separar chains paralelos.
Parecido a los scripts de shell, el carácter “/” se usa para hacer definiciones en múltiples líneas.

5.5.2 Ejemplos de uso de preconfiguraciones

Los efectos preconfigurados de ecasound son un pequeño motor de ecasound que hace q parezcan efectos nativos. Aquí veremos un ejemplo de un efecto preconfigurado multi-chain:

# let’s put the low freqs into one chain and high freqs in another
bassbooster = -efl:2000 -ea:200 | -efh:2000 -ea:50
# note, the ’|’ sign separates parallel chains

Y una vez definido podemos usarlo asi:

ecasound -a:1 -i:some.mp3 -pf:bassbooster.ecp
-a:2 -i:another.mp3 -pf:bassbooster.ecp
-a:1,2 -o:/dev/dsp

Cuando separamos ficheros usando (la opción "-pf:name"), Ecasound siempre carga primero las preconfiguraciones encontradas. Si el fichero contiene más preconfiguraciones (parejas adicionales de "key=value"), son ignoradas.
Una alternativa para definir la preconfiguraciones es poner la definición en una lista global (normalmente situada en /usr/local/share/ecasound/effect_presets) Una vez añadido la linea definiendo "bassbooster", se puede usar asi:

ecasound -a:1 -i:some.mp3 -pn:bassbooster
-a:2 -i:another.mp3 -pn:bassbooster
-a:1,2 -o:/dev/dsp

5.5.3 Parámetros de preconfiguraciones

Parámetros de operadores pertenecientes a preconfiguraciones se puedes usar asi:
f_res_lowpass = -ef3:%1,1.5,0.7

En el ejemplo, el filtro de paso bajo cortado, es mostrado como un parámetro de la preconfiguración "f_res_lowpass". Los siguientes dos comandos dan un resultado identico:
ecasound -i:foo.mp3 -o:/dev/dsp -pn:f_res_lowpass,800
ecasound -i:foo.mp3 -o:/dev/dsp -ef3:800,1.5,0.7

5.5.4 Descriptores de parámetros

Los parámetros de las preconfiguraciones de Ecasound pueden describirse usando el siguiente juego de descriptores:

-pd:name_of_preset = preset description
-ppn:par1,...,parN = parameter names (public params)
-ppd:val1,...,valN = default param values
-ppl:val1,...,valN = lower bounds for param values
-ppu:val1,...,valN = upper bounds for param values
-ppt:flags1,...,flagsN = special flags for param N
('i'=integer, 'l'=logarithmic, 'o'=output, 't'=toggle)

La opción puede usarse sólo dentro de las definiciones de preconfiguraciones (en ficheros "effect_presets" o individualmente en ficheros "*.ecp"). Un ejemplo de definición de parámetros de preconfiguraciones :
f_two_filters = -efl:800 -ea:%1 | -efh:800 -ea:%2 \
-pd:Parallel_highpass_and_lowpass_filters \
-ppl:0,0 -ppu:1000,- \
-ppd:100,100 -ppn:lowgain,highgain

La preconfiguración "f_two_filters" tiene dos parámetros, los cuales se describen usando el descriptor "-pd". Los limites inferiores y superiores recomendados para los parámetros son definidos como "-ppl" y "-ppu". Por defecto estos valores son especificados por "-ppd".

5.5.5 Operadores puente

Los puentes se parecen a otros operadores de chain. Se asignan a un chain y procesan los buffers de paso del audio. Una funcion especial de los puentes es la habilidad de recoger secciones de los ficheros de audio, para crear un volumen automatico a partir de un corte de una zona del flujo de audio.

5.5.6 Ejemplo de uso

La siguiente secuencia corta la seccion [60:00 seg -> 61:00 seg] de "guitar.wav" dentro de "gate-test.wav"

```
|\$ ls -la guitar.wav
-rw-rw-r-- 1 kaiv kaiv 15790124 Sep 30 23:27 guitar.wav
|\$ ecasound -i guitar.wav -o gate-test.wav -gc:60,1
|\$ ls -la gate-test.wav
-rw-rw-r-- 1 kaiv kaiv 180268 Dec 12 22:13 gate-test.wav
```

El umbral del puente usado es similar a esto:

```
|\$ ecasound -i gate-test.wav -o gate-test-rms.wav -ge:11.2,5,1
|\$ ecasound -i gate-test.wav -o gate-test-peak.wav -ge:5,5,0
|\$ ls -la gate*wav
-rw-rw-r-- 1 kaiv kaiv 163884 Dec 12 22:18 gate-test-peak.wav
-rw-rw-r-- 1 kaiv kaiv 143404 Dec 12 22:17 gate-test-rms.wav
-rw-rw-r-- 1 kaiv kaiv 180268 Dec 12 22:13 gate-test.wav
```

En el primero de los casos el puente es abierto cuando el volumen-RMS sobrepasa el "11,2%" el umbral y cerrado cuando el volumen-RMS baja de "5%" (volumen de pico)

6. LAPDSA plugins

Ecasound soporta LADSPA-effect plugins (Linux Audio Developers´s Simple Plugin API) Para que ecasound soporte LADSPA plugins necesitar haber sido compilado con la LADSPA activado para que soporte los plugins. Una vez instalado los plugins de LADSPA podemos probarlo de la siguiente manera:

Echo "ladspa-register" | ecasound –c

6.1 Servidor de sonido JACK

JACK en un servidor de sonido de baja latencia, escrito principalmente para GNU/Linux. Puede conectar un numero diferente de aplicaciones a un dispositivo de audio y asi permitir compartir el sonido entre ellas. Los clientes pueden lanzar sus propios procesos como aplicaciones normales o lanzarlos dentro del servidor JACK.
JACK es diferente de otros servidores de sonido, Esta desarrollado para tener sincornismo de ejecución entre los clientes y baja latencia.

6.1.1 Entradas y salidas básicas

Veámos como escuchar un fichero usando Ecasound con Jack:

Ecasound –i foo.wav –o jack_alsa

Esto creara un puerto de salida separado para cada canal de foo.wav, y automáticamente los conectará Ecasound estos puertos a la salida de los puertos ALSA PCM del servidor JACK. Las conexiones son las siguientes:

ecasound:out_1 --> alsa_pcm:playback_1
ecasound:out_2 --> alsa_pcm:playback_2

Para grabar un fichero haremos esto:

ecasound -f:32,2,44100 -i jack_alsa -o foo.wav

Aquí usamos "-f:bits,channels,rate" para configurar tantos canales como queramos de la entrada de la tarjeta de sonido usando JACK. Como JACK siempre usa muestras de 32bits y la misma frecuencia de muestreo debe ser marcada la primera vez que es usada por el servidor JACK, el contador de canales es el único parámetro configurable. Podemos usas otro "-f:x,y,z" antes de "-o foo.wav" si queremos escribir el fichero en un formato diferente.

6.1.2 Creación de puertos más avanzados

Ecasound tambien ofrece la siguiente alternativa para crear puertos de entrada y salida:

ecasound -i foo.wav -o jack
ecasound -i foo.wav -o jack_auto,remote_client
ecasound -i foo.wav -o jack_generic,local_portprefix
ecasound -i jack -o foo.wav
ecasound -i jack_auto,remote_client -o foo.wav
ecasound -i jack_generic,local_portprefix -o foo.wav

"jack" y "jack_auto" ambos crean un juego de puertos de entrada y de salida. En el primero de los casos, Ecasound no hace una conexión implícita, pero en el último de los casos los puertos creados son conectados automáticamente a los puertos marcados por el cliente especificado.

6.1.3 Control de transporte

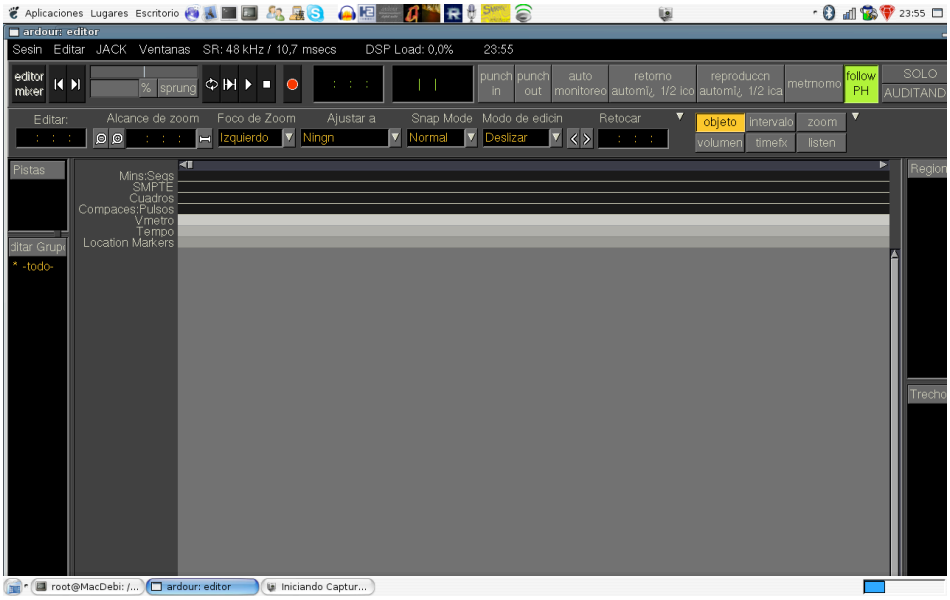
Las funciones de control son "start", "stop", "seek", "etc", comúnmente usadas en la mayoria de las aplicaciones de sonido. El transporte de JACK permite controlar todos las aplicaciones conectadas a un servidor JACK desde una simple aplicación. Ecasound puede soportar esta funcionalidad en cuatro modos diferentes ("notransport", "send", "recv" y "sendrecv")
Por defecto Ecasound enviará y recivirá eventos de transporte (posición y estado) a otros clientes JACK (mode "sendrecv"):

ecasound –c –i null –o jack

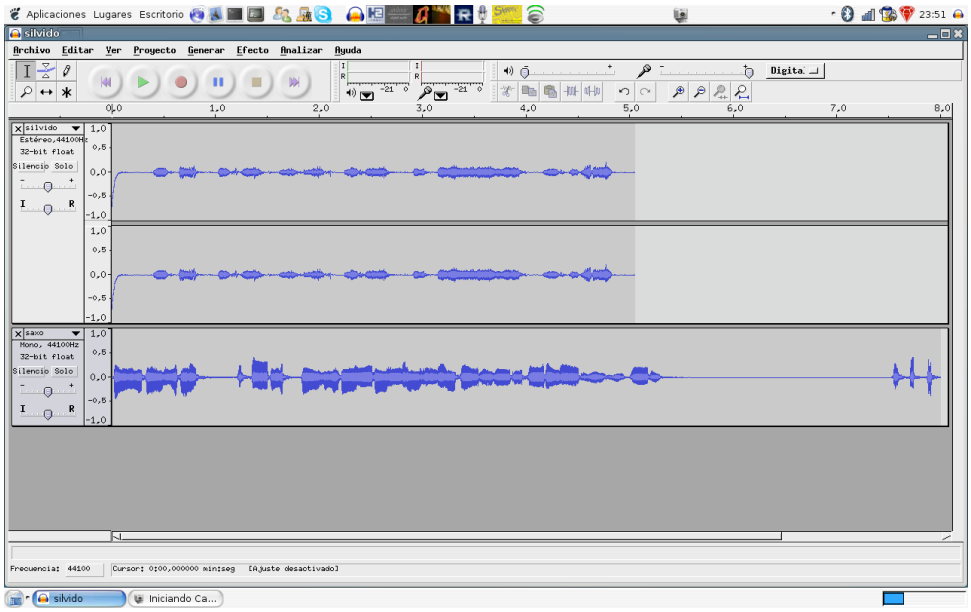
Para usar un control de transporte en ecasound, tenemos que tener al menos una entrada o salida, anunciada en un puerto de JACK.

7. Frontends Gráficos

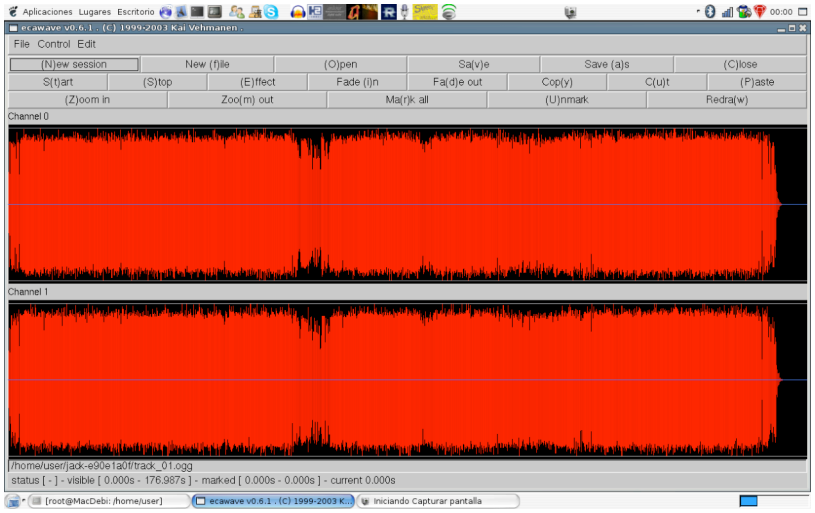
Podemos encontrar una gran variedad de frontend gráficos para ecasound que nos facilitaran las operaciones de edición de audio. Aquí veremos unos screenshots de los principales y destacaremos Audacity como uno de los más completos.



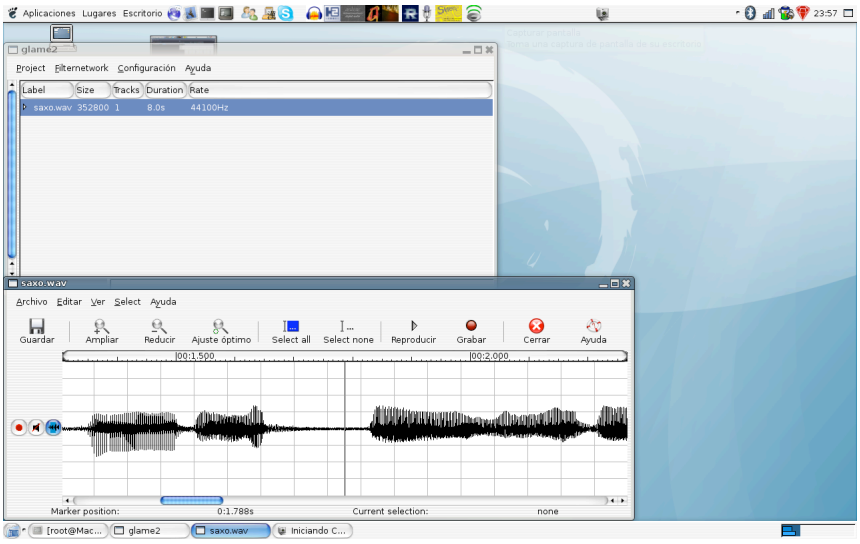
Ardour



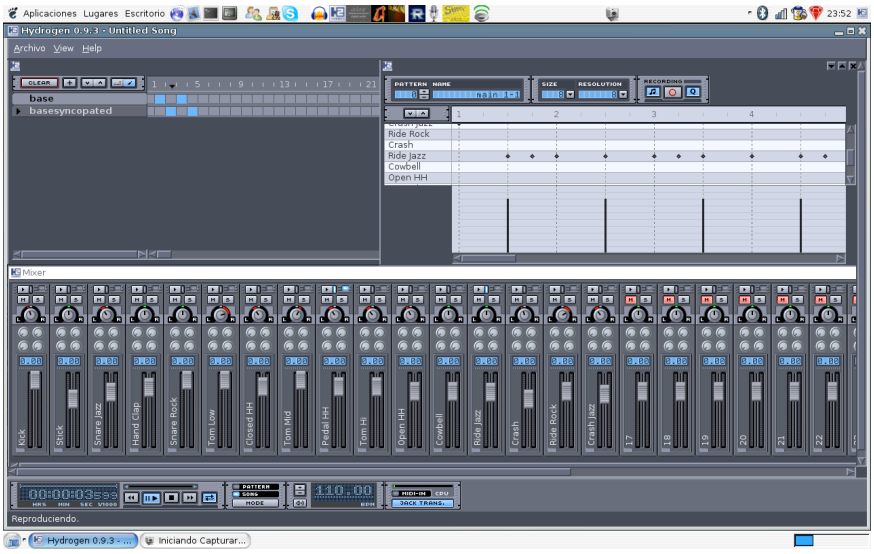
Audacity



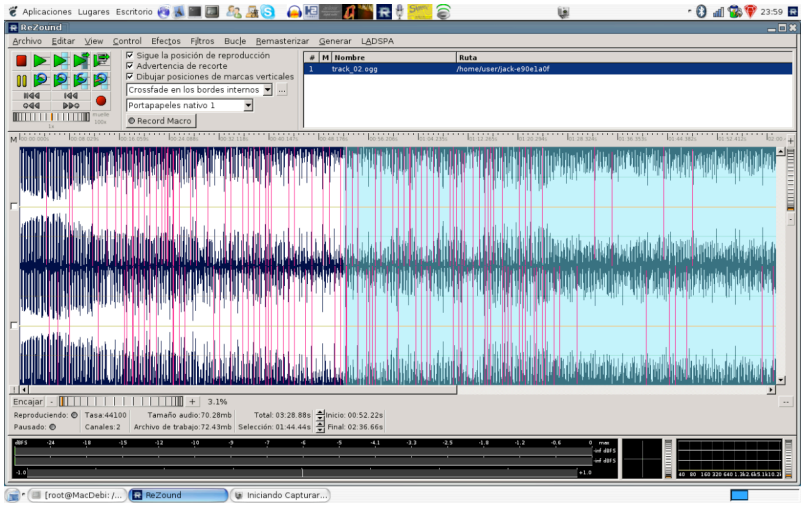
Ecawave



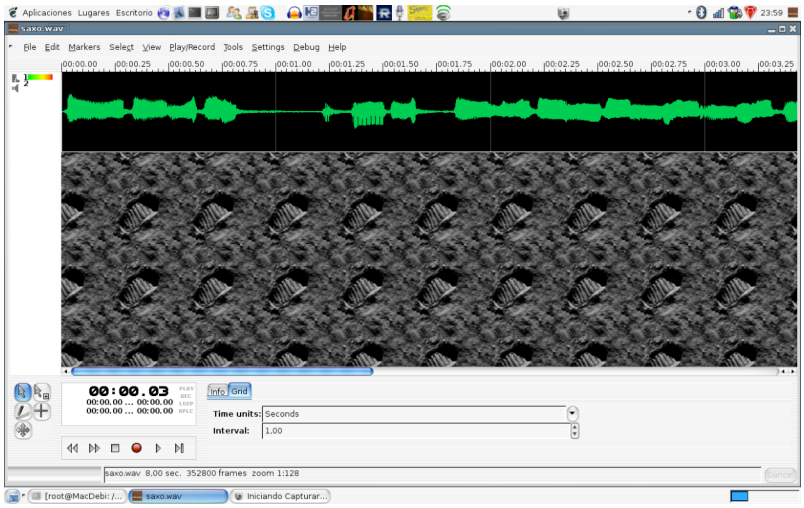
GLame



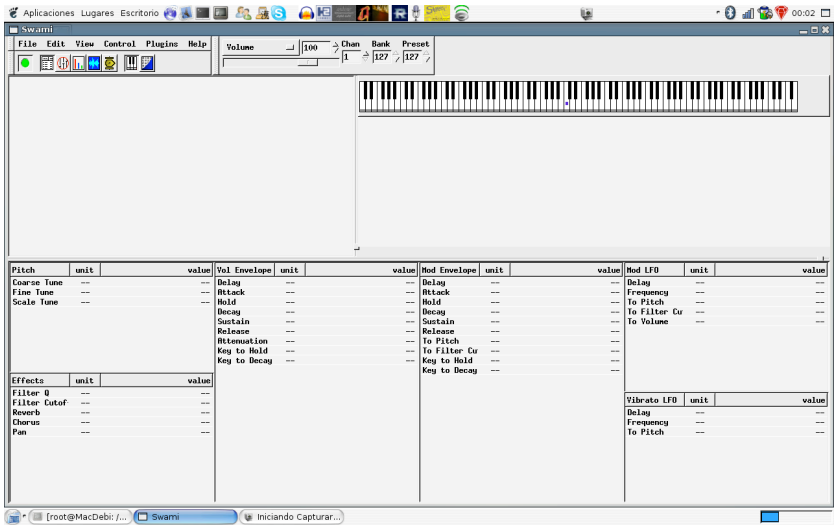
Hydrogen



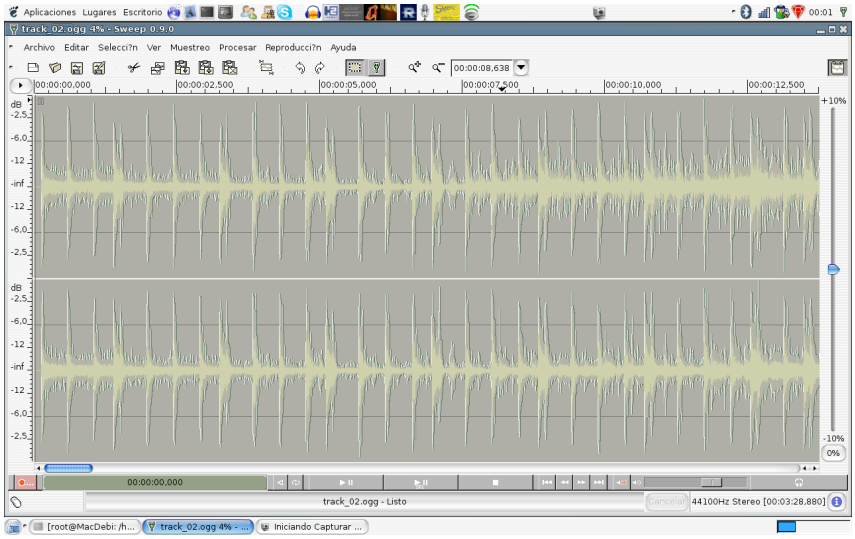
Rezound



GNUSound



Swami


Aplicaciones Lugares Escritorio
track_02.ogg 4% - Sweep 0.9.0
Archivo Editar Selecci?n Ver Muestreo Procesar Reproducci?n Ayuda
00:00:08,638
00:00:00.000
00:00:02.500
00:00:05.000
00:00:07.500
00:00:10.000
00:00:12.500
+10%
-10%
0%
00:00:00.000
track_02.ogg - Listo
Cancelar
44100Hz Stereo [00:03:28.880]
[root@MacDebi: /h...
track_02.ogg 4% - ...
Iniciando Capturar ...

Sweep

8. Referéncias:

http://che.wikidot.com/howto:linuxgrabarcintas
http://www.eca.cx/ecasound/Documentation/examples.html
http://www.wakkanet.fi/~kaiv/ecawave/
http://cantor.ee.ucla.edu/~jsab/ecasound.html