Liquidsoap, la planetaria multimediale

a.k.a Come non imparare le flag di ffmpeg e vivere felici

Storia

- Iniziato da **David Barelde** e **Samuel Mimram** come progetto scolastico a Lione nel 2004, chiaramente la prima versione non funziona.
- Inizialmente era uno script in perl che interagiva con **Ices**.
- Successivamente si aggiungono Ramain Beauxis ed il progetto si spota da Lione a Parigi. Da **savonet** nasce **liquidsoap**.

La fase OCaml

- **Savonet**: *SAm and daVid Ocaml NETwork*
- Da maledetti studenti universitari hanno voluto fare la roba funzionale.
- A peggiorare la faccenda **OCaml** in quel periodo era il linguaggio più di moda in Francia (w l'autarchia dei linguaggi).
- Così nasce la loro prima web radio: *geekradio*.

Liquidsoap, il linguaggio

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
At line 5, char 8-49:
Error 7: Invalid value:
That source is fallible
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
security = single("~/radio/sounds/default.mp3")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
radio = fallback(track_sensitive = false, [radio, security])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
security = single("~/radio/sounds/default.mp3")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
radio = fallback(track_sensitive = false, [radio, security])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
security = single("~/radio/sounds/default.mp3")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
radio = fallback(track_sensitive = false, [radio, security])
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
At line 7, char 8-49:
Error 7: Invalid value:
That source is fallible
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
security = single("~/radio/sounds/default.mp3")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
radio = fallback(track_sensitive = false, [radio, security])
radio = mksafe(radio)
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
security = single("~/radio/sounds/default.mp3")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
radio = switch(
        track_sensitive = true,
        [ ({ 4w and 13h-15h }, radio) ])
radio = fallback(track_sensitive = false, [radio, security])
radio = mksafe(radio)
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

```
myplaylist = playlist("~/radio/music.m3u")
jingles = playlist("~/radio/jingles.m3u")
security = single("~/radio/sounds/default.mp3")
radio = myplaylist
radio = random(weights = [1, 4],[jingles, radio])
radio = switch(
        track_sensitive = true,
        [ ({ 4w and 13h-15h }, radio) ])
radio = fallback(track_sensitive = false, [radio, security])
radio = crossfade(fade_out=3., fade_in=3., duration=5., radio)
radio = mksafe(radio)
output.icecast(%mp3,
    host = "localhost", port = 8000,
    password = "hackme", mount = "basic-radio",
    radio)
```

Liquidsoap, cose belle

Supporto alla galassia multimediale

- **Codec interni**: opus, vorbis, mpeg-3, mpeg-4, theora
- **Codec esterni**: encoder e decoder
- **I/O**: Alsa, pulseaudio, jack, v4l2
- **Output**: file, HTTP, icecast, HLS, SRT
- **Input**: Harbor (icecast) input ffmpeg, gstreamer
- **Input protocol**: Youtube (via RTMP & ffmpeg), beets, youtube-dl, mpd, process, s3, say (text-to-speech), synth, text-to-wave, etc...

Manipolazione audio/video

- **Audio**: normalization, amplify, replaygain, cross-fading, rilevamento rumore bianco, mono2stereo, stereo2mono, etc...
- **Audio/Video**: Ladspa, dssi, lilv & ffmpeg filtri ffmpeg

Il linguaggio di script

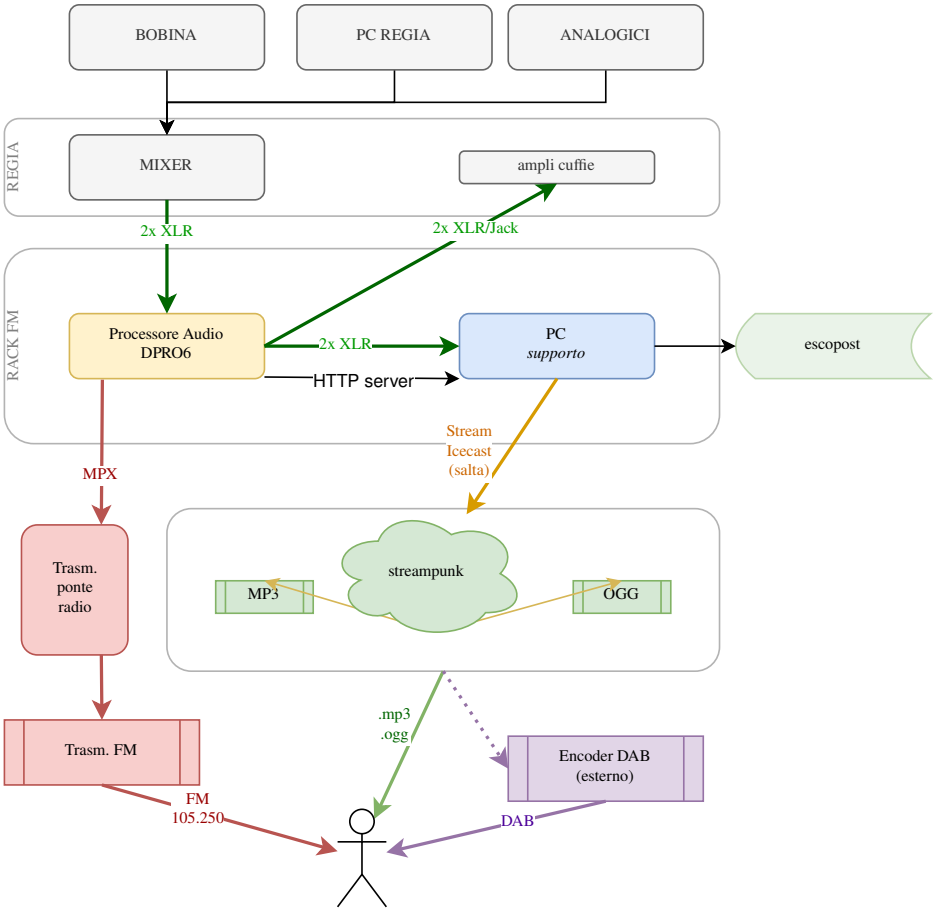
- threading
- configurazioni dinamiche
- input api (telnet, http)
- self-documented
- fortemente tipato
- clock integrati

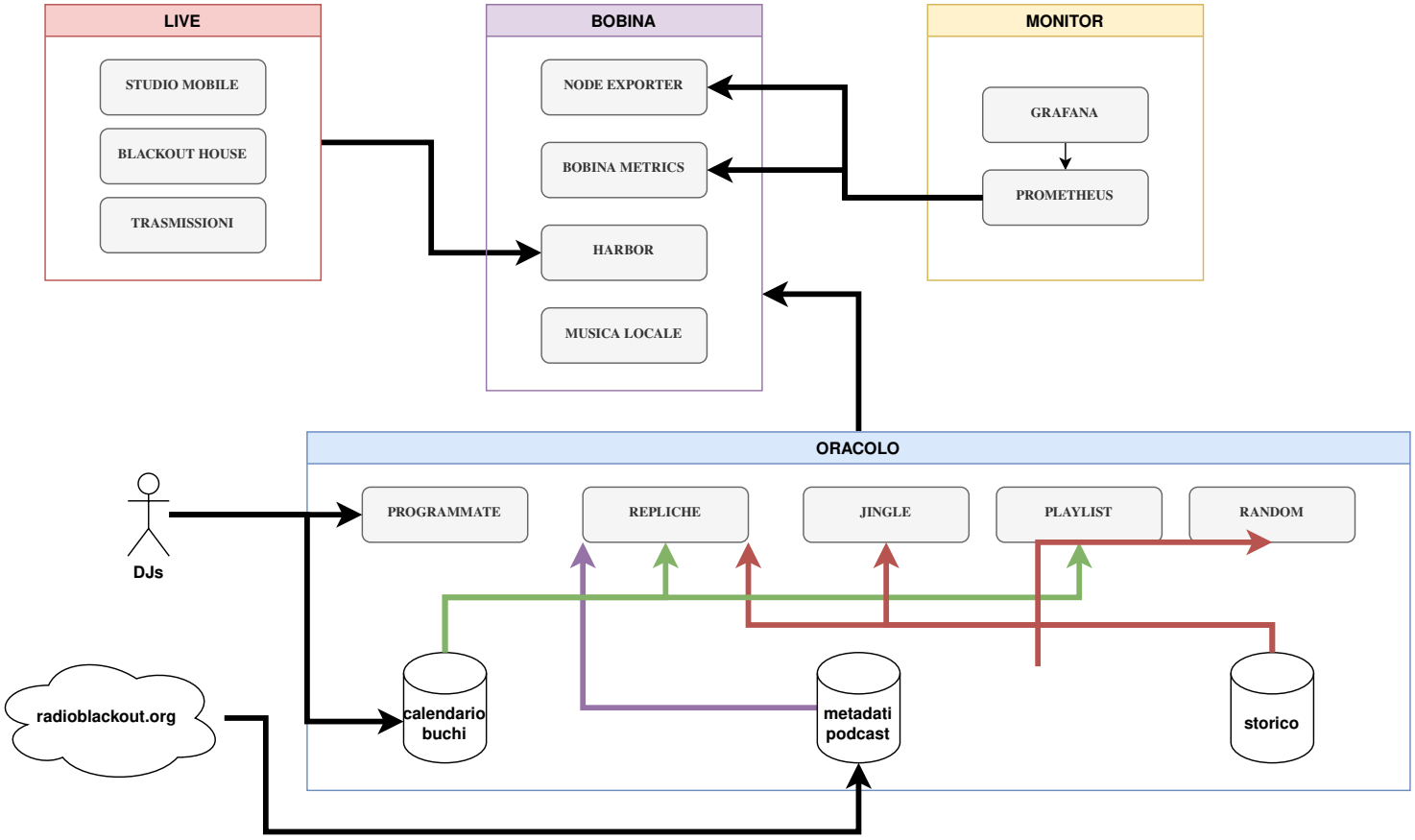
Liquidosoap, cose brutte

OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml
OCaml OCaml OCaml OCaml OCaml OCaml OCaml

Bobina

Fatta con ❤ dal black tech team di Radio Blackout 105.250FM


BOBINA
PC REGIA
ANALOGICI
REGIA
MIXER
ampli cuffie
2x XLR
2x XLR/Jack
RACK FM
Processore Audio
DPRO6
2x XLR
PC
supporto
escopost
HTTP server
Stream
Icecast
(salta)
MPX
Trasm.
ponte
radio
streampunk
MP3
OGG
.mp3
.ogg
Trasm. FM
Encoder DAB
(esterno)
FM
105.250
DAB


LIVE
STUDIO MOBILE
BLACKOUT HOUSE
TRASMISSIONI
BOBINA
NODE EXPORTER
BOBINA METRICS
HARBOR
MUSICA LOCALE
MONITOR
GRAFANA
PROMETHEUS
ORACOLO
DJs
PROGRAMMATE
REPLICHE
JINGLE
PLAYLIST
RANDOM
radioblackout.org
calendario buchi
metadati podcast
storico

I flussi

L'elenco è in ordine decrescente di priorità, per il tipo di flussi che gestiamo il passaggio da un flusso ad un altro dovrebbe essere istantaneo e non legato alla fine di una traccia dato che molti flussi potrebbero non finire mai (tracksensitive=false).

Live

- **mobile**: occhio ai buffer e deve essere esposto all'interweb
- **blackout house**: rete interna, pochi problemi
- **trasmissione** (es radio notav): come mobile, ma ogni trasmissione ha degli accessi diversi e limitata nell'orario. Se provi a trasmettere fuori dal tuo orario arriva un http unauthorized. Sarebbe bello dare alle trasmissioni un endpoint di debug/gioco.

Trasmissioni

- **programmata**: un file audio che deve essere riprodotto per intero ad un determinato orario, viene esplicitamente inserita.
- **differita**: un file audio che deve essere riprodotto per intero, viene scelta casualmente da un ventaglio di papabili.

Archivio

- **playlist**: una playlist di pezzi selezionati, deve essere riprodotta per intero
- **jingle**: andrebbero pesati in base alle date, attualmente è rarissimo sentirne uno
- **musica**: singoli pezzi selezionati a casaccio (attualmente bobina_rotazione.liq)

Come lo facciamo?

- teniamo la logica dentro liquidsoap?
- interagiamo con liquidsoap dicendogli esegui questo o altro?
- riusciremo ad integrare le differite in modo sensato?
- ma c'è un modo di fare delle selezioni musicali meglio di rotazione?
- liquidsoap è potente, ma un po' fastidioso da scrivere.

Proviamoci con liquidsoap

Liquidsoap ha la possibilità di essere "comandato" da fuori con alcune interfacce (telnet, http, variabili interattive, etc), il problema è che mentre è estremamente comodo gestire i flussi (es: se quell'input è spento, usa altro) ed in generale tutto quello che è manipolazione multimediale non si può dire lo stesso di scrivere la parte logica (es: programma questa trasmissione).

Fatte queste premesse teniamo liquidsoap per fare le seguenti cose:

- gestisci flussi con priorità
- raccogli flussi esterni con eventuali permessi
- logga (la nostra unica fonte di verità)
- deve essere in grado di funzionare anche quando gli altri componenti muoiono o sono irraggiungibili. Nella pratica si traduce nell'avere sempre un fallback sensato (non vale rumore bianco).

Cosa delega?

- logica: se non ho live... cosa riproduco? me lo dice l'oracolo!
- interfaccia: santo cielo javascript...

L'oracolo DJ

Risponde alla domanda: cosa riproduco? Cosa usa per rispondere? (in ordine di priorità)

- C'è una trasmissione programmata?
- Siamo in un buco del palinsesto? Cerco una differita che stia nel tempo e nel tipo richiesto
- Ho delle playlist musicali che mi coprono il tempo fino alla prossima trasmissione?
- Scelgo un pezzo a caso chiedendolo a beets

Quali informazioni servono?

- palinsesto
- calendario buchi
- trasmissioni programmate
- playlist

Deve avere una serie di interfacce che potranno essere ricavate dal nuovo sito o da robe dedicate come l'automatico:

- programmate: l'automatico
- playlist: creare playlist a partire dall'archivio
- musica: mpd, beet: è necessario un server/archivio indicizzato, basta cartelle condivise o meglio... teniamole in sola lettura e facciamo caricare la musica in altro modo.
- calendario buchi: ehm giovedì non trasmetto dalle 13 alle 15, ci starebbe un contenuto informativo!

E ora, mani in pasta

Cose rimaste

- rilevamento silenzio
- output metadati

Progetti carini

- **AzuraCast**
- **AuRa**: Automated Radio
- **AutoRadio**: tante grazie a streampunk.cc
- **Radio france**: eh già

Presentazioni

- Functional audio and video stream generation with Liquidsoap
- Streaming at Radio France
- Easing automation and improving your sound with Liquidsoap and FFmpeg
- Liquidsoap current and future features

Questa presentazione

0xacab.org/rbo/liquidsoap-slides