Hacker News Radio

@jeffsigmon – 6/22/2011

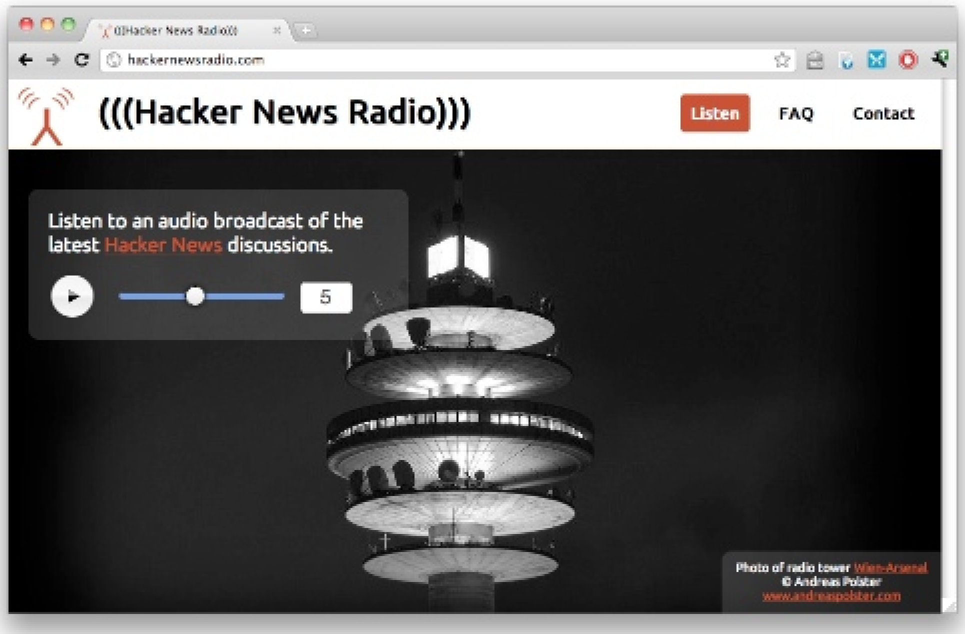
Overview

What is it?

A model for publishing audio streams from textual sources on the web.

More Specifically...

- A processor that scrapes news.ycombinator.com, converts text representation to synthesized speech and publishes audio streams.
- A website providing an embedded audio player and links to the published streams.



Listen to an audio broadcast of the latest Hacker News discussions.

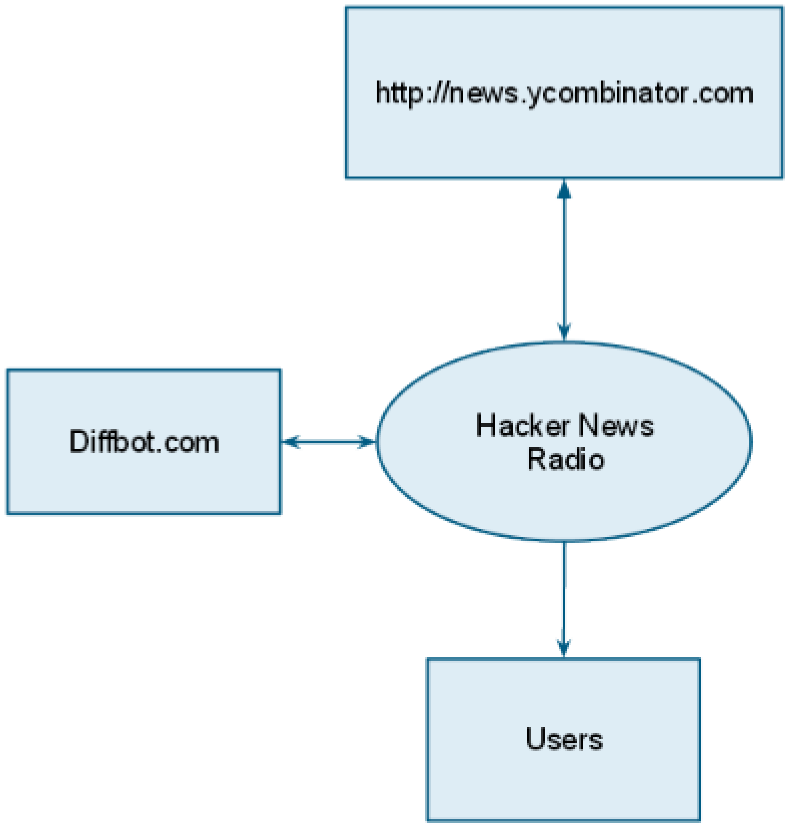
Photo of radio tower Wien-Arsenal
© Andreas Polster
www.andreaspolster.com

Demo

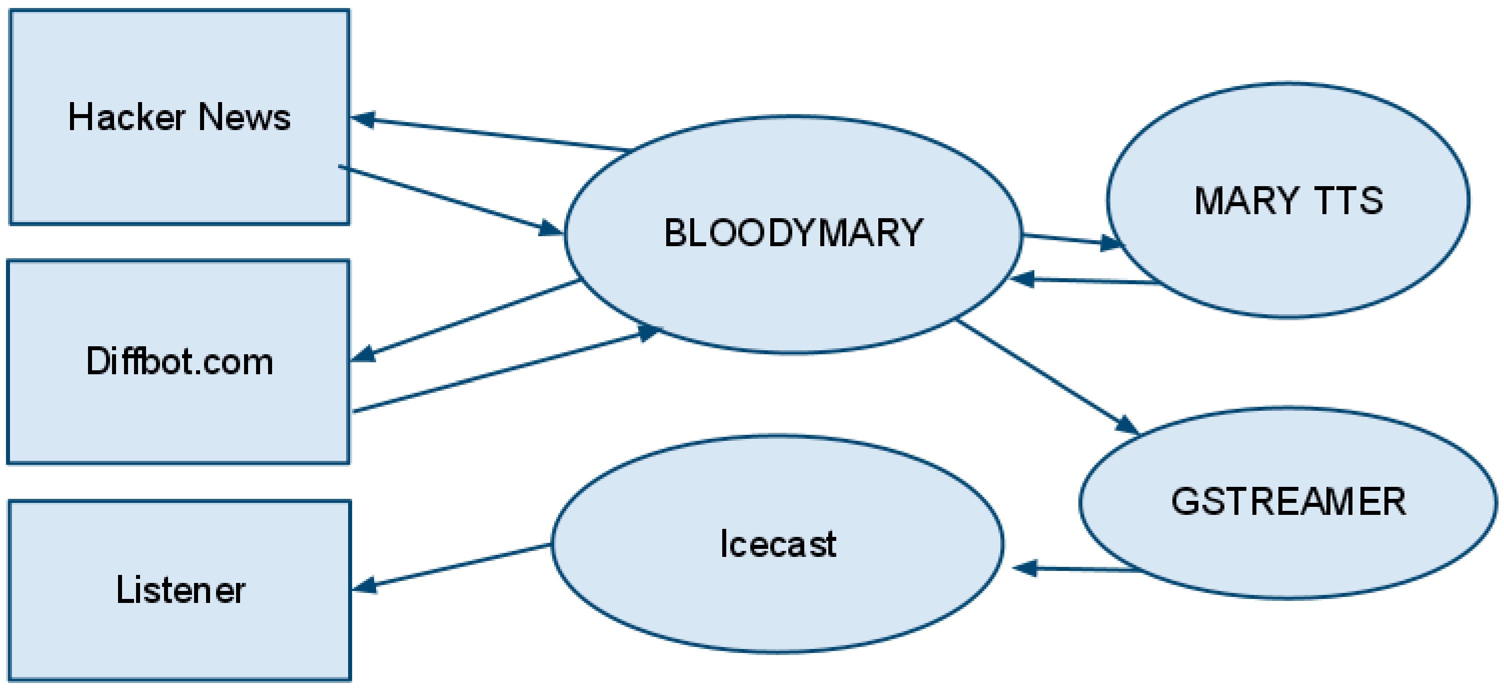
What software/services are used?

- Clojure is the glue binds it all
- MARY TTS is an open-source, multilingual Text-to-Speech Synthesis platform written in Java.
- gstreamer is an open source multimedia framework.
- gstreamer-java is a Java interface to the gstreamer framework.
- Icecast is free server software for streaming multimedia.
- Enlive is selector-based templating and transformation system for Clojure.
- HTML5 BOILERPLATE is the professional badass's base HTML/CSS/JS template for a fast, robust and future-proof site.
- pynch is a clojure library for scraping hacker news.
- Diffbot Article API will extract and clean article text from news article web pages.
- diffbot-clj Clojure client for Diffbot.
- linode.com is a linux vps hosting provider.
- tunein.com gives you access to over 50,000 of the world's radio stations so you can find and listen to unlimited music, sports and talk radio.

Context


http://news.ycombinator.com
Diffbot.com
Hacker News Radio
Users

Data Flow Level 1



Parsing Hacker News

Pynch

A clojure library to scrape/parse submission from hacker news.

Get the most recent submission

```clojure
(-> "http://news.ycombinator.com" java.net.URI. get-subs first)

;;;output

{:points 120,
 :title "F.B.I. Seizes Web Servers, Knocking Sites Offline ",
 :sub-time #<DateTime 2011-06-22T00:08:00.000Z>,
 :sub-url "http://bits.blogs.nytimes.com/2011/06/21/f-b-i-seizes-web-servers-knocking-sites
 :user "tshtf",
 :com-url "item?id=2680922",
 :com-count 66}
```

Get submission details

```
(-> "http://news.ycombinator.com/item?id=2681410"
            java.net.URI. get-sub-details)

;;;output

({:title "Brendan Eich: New JavaScript Engine Module Owner",
  :time #<DateTime 2011-06-22T03:17:00.000Z>,
  :points 20,
  :user "pufuwozu",
  :notes "\n",
  :com-url "item?id=2681410",
  :com-count 2,
  :comments
  ({:user "keyle",
    :time #<DateTime 2011-06-22T03:17:00.000Z>,
    :cmnt-url "item?id=2681497",
    :cmnt-text
    ("Is that likely to have a significant impact on the way we write Javascript today?")}
   {:user "wmf",
    :time #<DateTime 2011-06-22T03:24:00.000Z>,
    :cmnt-url "item?id=2681560",
    :cmnt-text
    ("No. JavaScript is (now) important enough that no one person has that much influence
```

Enlive

Enlive is an extraction and transformation library for HTML and XML documents written in Clojure. It uses CSS-like selectors.

Usual Enlive applications include templating and screenscraping.

Here is a very simple example to retreive list of submission titles on hacker news

```clojure
(map #(enlv/text %)
  (enlv/select
     (-> "http://news.ycombinator.com" java.net.URI. enlv/html-resource)
  [:td.title :a]))

  ("Revolutionary \"Light Field\" camera tech - shoot-first, focus-later"
   "F.B.I. Seizes Web Servers, Knocking Sites Offline "
   "Please, make yourself uncomfortable"
   "Pinboard.in service limited - FBI raided hosting company and pulled equpiment"
   "Brendan Eich: New JavaScript Engine Module Owner"
   "Comparing Indian states with countries"
   "Impressions of Android from a dev's perspective"
   "Firefox 5 is now officially released"
   "5 Months of Customer Service Hell with HTC"
   "Sony officially 50% of all GitHub's DMCA notices ")
```

Select list of titles and submitters

```clojure
(enlv/let-select
  (-> "http://news.ycombinator.com" java.net.URI. enlv/html-resource)
  [titles [:td.title :a]
   users [[:a (enlv/attr-starts :href "user")]]]
    (map #(hash-map :title (enlv/text %1) (enlv/text %2) titles users)))

;;; OUTPUT

(({:title "Revolutionary \"Light Field\" camera tech - shoot-first, focus-later",
   :user "hugorodgerbrown"}
  {:title "F.B.I. Seizes Web Servers, Knocking Sites Offline ",
   :user "tshtf"}
  {:title "Please, make yourself uncomfortable",
   :user "buf"}
  {:title "Pinboard.in service limited - FBI raided hosting company and pulled equipment",
   :user "rograndom"}
  {:title "Brendan Eich: New JavaScript Engine Module Owner",
   :user "pufuwozu"}
  {:title "Comparing Indian states with countries",
   :user "gopi"}))
```

Diffbot

Diffbot Article API

The Article API takes in as input any news story page. Statistical machine learning algorithms are run over all of the visual elements on the page to extract out the article text and associated metadata, such as its images, videos, and tags. If the article spans multiple pages, Diffbot will follow the next pages to get the whole article. There is also experimental support for extracting reader comments.

Using diffbot-clj

```
(analyze
  "http://brendaneich.com/2011/06/new-javascript-engine-module-owner/"
  *token*)

;;; OUTPUT

{:icon "http://brendaneich.com/favicon.ico",
 :author "Bill Joy",
 :text "As you may know, I wrote JavaScript in ten days.... (SHORTENDED BY JEFF FOR THIS PR
 :title "New JavaScript Engine Module Owner",
 :date "21 June 2011",
 :url "http://brendaneich.com/2011/06/new-javascript-engine-module-owner/",
 :xpath "/HTML[1]/BODY[1]/DIV[1]/DIV[2]/DIV[1]/DIV[1]/DIV[3]"}
```

Speech Synthesis

Mary TTS

MARY is an open-source, multilingual Text-to-Speech Synthesis platform written in Java. It was originally developed as a collaborative project of DFKI's Language Technology lab and the Institute of Phonetics at Saarland University and is now being maintained by DFKI.

As of version 4.3, MARY TTS supports German, British and American English, Telugu, Turkish, and Russian; more languages are in preparation. MARY TTS comes with toolkits for quickly adding support for new languages and for building unit selection and HMM-based synthesis voices.

Key Components

- Multi-threaded server
- Java Client
- Built in voices

Input

Plain Text

Sable

SABLE is an XML markup language used to annotate texts for speech synthesis. It defines tags which control the way words, numbers, and sentences are reproduced by a computer. SABLE was developed as an informal joint project between Sun Microsystems, AT&T, Bell Labs and Edinburgh University (the initial letters of each make the word "SABLE") as an initiative to combine the three previous speech synthesis markup languages SSML, STML, and JSML.

Speech Synthesis Markup Language (SSML)

Speech Synthesis Markup Language (SSML) is an XML-based markup language for speech synthesis applications. It is a recommendation of the W3C's voice browser working group. SSML is often embedded in VoiceXML scripts to drive interactive telephony systems. However, it also may be used alone, such as for creating audio books. For desktop applications, other markup languages are popular, including Apple's embedded speech commands, and Microsoft's SAPI Text to speech (TTS) markup, also an XML language.

Mary XML

MaryXML is an internal, relatively low-level markup which reflects the modelling capabilities of this particular TtS system.

Example SSML

```xml
<?xml version="1.0"?>
<speak xmlns="http://www.w3.org/2001/10/synthesis"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   version="1.0">
   <metadata>
     <dc:title xml:lang="en">Telephone Menu: Level 1</dc:title>
   </metadata>
   <p>
     <s xml:lang="en-US">
       <voice name="David" gender="male" age="25">
        For English, press <emphasis>one</emphasis>.
       </voice>
     </s>
     <s xml:lang="es-MX">
       <voice name="Miguel" gender="male" age="25">
        Para español, oprima el <emphasis>dos</emphasis>.
       </voice>
     </s>
   </p>
 </speak>
```

API Usage

```clojure
(ns bloodymary.mary
  (:import
    (marytts.client.http Address)
    (java.io ByteArrayInputStream ByteArrayOutputStream)))

(def *host* "localhost")
(def *port* 59125)
(def *input-type* "TEXT")
(def *audio-type* "AU")
(def *locale* "en_GB")
(def *voice-name* "dfki-spike-hsmm")
(def *style* "")
(def *effects* "")

(defn get-audio [s & options]
  (let [stream (ByteArrayOutputStream.)
        opts (apply hash-map options)
        client (get-client (get opts :host *host*) (get opts :port *port*))]
    (.process
      client s
      (get opts :input-type *input-type*)
      "AUDIO"
      (get opts :locale *locale*)
      (get opts :audio-type *audio-type*)
      (get opts :voice-name *voice-name*)
      stream)
    (.toByteArray stream)))
```

Demo

gstreamer

What is it?

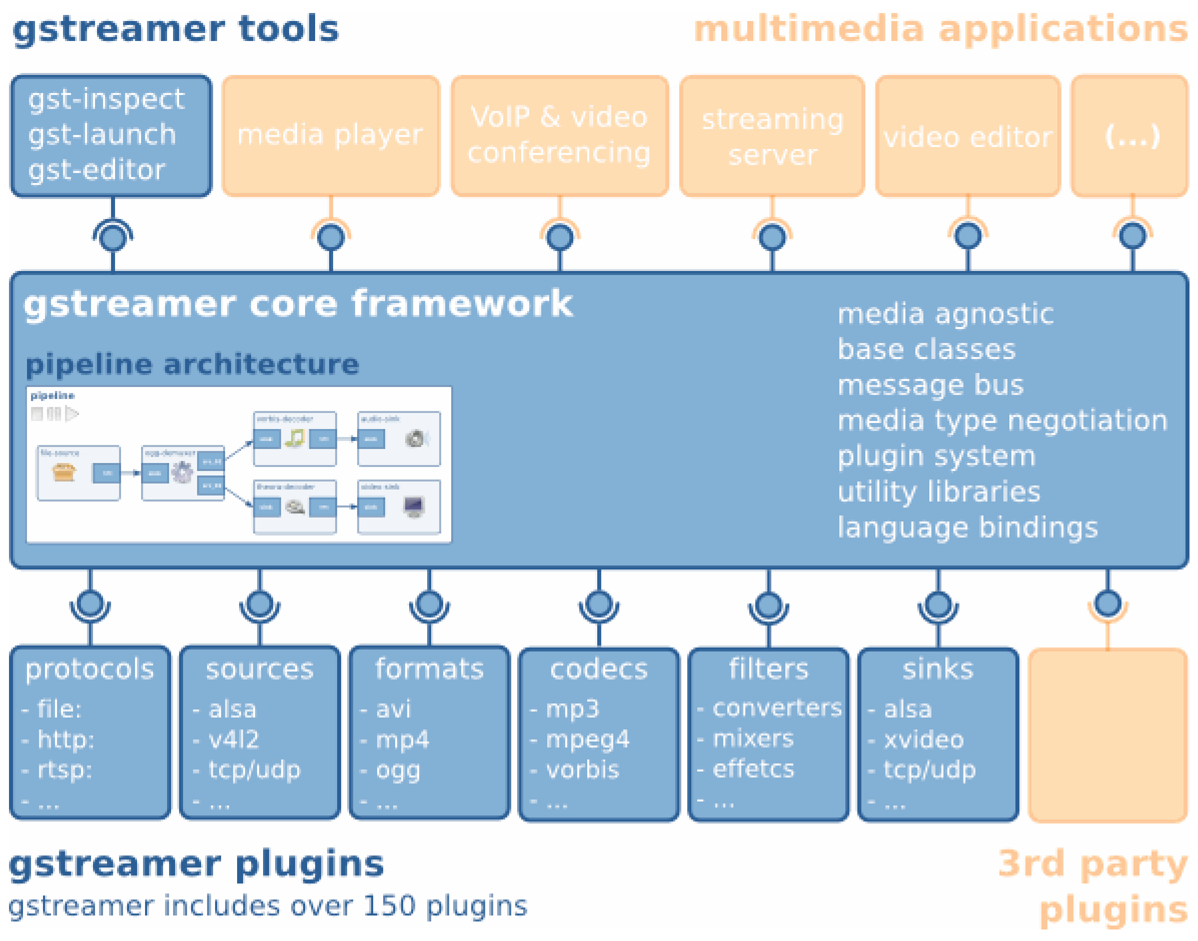
gstreamer is a framework for creating streaming media applications.

The fundamental design comes from the video pipeline at Oregon Graduate Institute, as well as some ideas from DirectShow.

Specifically

- API for multimedia applications
- Plugin architecture
- Pipeline architecture
- Mechanism for media type handling/negotiation
- Over 150 plug-ins
- A set of tools

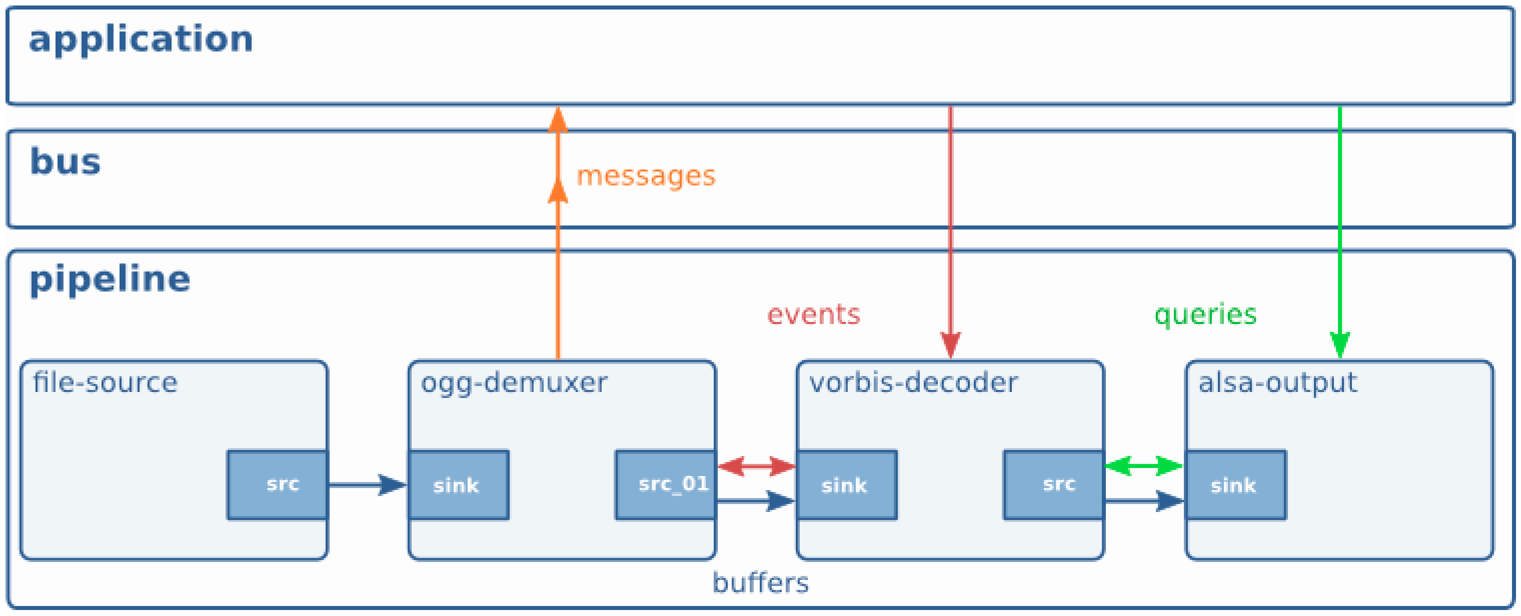
Context


gstreamer tools
multimedia applications
gst-inspect
gst-launch
gst-editor
media player
VoIP & video conferencing
streaming server
video editor
(...)
gstreamer core framework
pipeline architecture
pipeline
media agnostic
base classes
message bus
media type negotiation
plugin system
utility libraries
language bindings
protocols
- file:
- http:
- rtsp:
- ...
sources
- alsa
- v4l2
- tcp/udp
- ...
formats
- avi
- mp4
- ogg
- ...
codecs
- mp3
- mpeg4
- vorbis
- ...
filters
- converters
- mixers
- effetcs
- ...
sinks
- alsa
- xvideo
- tcp/udp
- ...
gstreamer plugins
gstreamer includes over 150 plugins
3rd party plugins

Core Concepts

- Element
- Pad
- Bin
- Pipeline
- Bus

Data Flow


application
bus
messages
pipeline
events
queries
file-source
src
ogg-demuxer
sink
src_01
vorbis-decoder
sink
src
alsa-output
sink
buffers

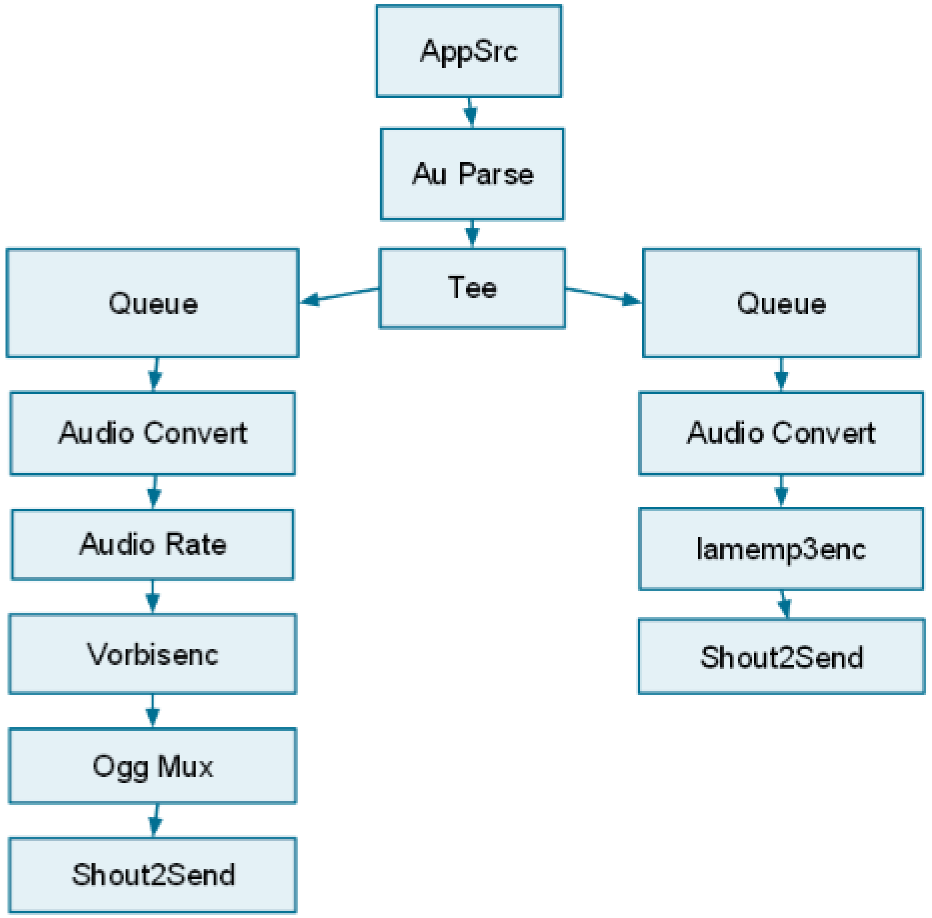
Packages

- **gstreamer** contains the core library and elements
- **gst-plugins-base** contains an essential exemplary set of elements
- **gst-plugins-good** contains a set of good-quality plug-ins under our preferred license, LGPL
- **gst-plugins-ugly** contains a set of good-quality plug-ins that might pose distribution problems
- **gst-plugins-gl** contains plug-in and helper libraries for OpenGL integration
- **gst-ffmpeg** contains FFmpeg based plugin
- **tools** contains gst-launch, gst-inspect, gst-typefind, gst-feedback

Key Elements for HNR

- appsrc
- auparse
- audioconvert
- tee
- lamemp3enc
- queue
- audioresample
- capsfilter
- shout2send
- vorbisend
- audiorate
- oggmux

HNR Pipeline



Demo

Icecast

About

Icecast is a streaming media server which currently supports Ogg Vorbis and MP3 audio streams. It can be used to create an Internet radio station or a privately running jukebox and many things in between. It is very versatile in that new formats can be added relatively easily and supports open standards for commuincation and interaction.

There are two major parts to most streaming media servers: the component providing the content (what we call source clients) and the component which is responsible for serving that content to listeners (this is the function of icecast).

Concepts

A **source client** is an external program which is responsible for sending content data to icecast. Some source clients that support icecast2 are Oddcast, ices2, ices0.3, and DarkIce.

The **slave server** in a relay configuration is the server that is pulling the data from the master server. It acts as a listening client to the master server.

The **master server** in a relay configuration is the server that has the stream that is being relayed.

A **mountpoint** is a resource on the icecast server that represents a single broadcast stream. Mountpoints are named similar to files (/mystream.ogg, /mymp3stream). When listeners connect to icecast2, they must specify the mountpoint in the request (i.e. http://192.168.1.10:8000/mystream.ogg). Additionally, source clients must specify a mountpoint when they connect as well. Statistics are kept track of by mountpoint. Mountpoints are a fundamental aspect of icecast2 and how it is organized.

Mountpoints

Each icecast server can house multiple broadcasts (or mountpoints) each containing a separate stream of content. A 'mountpoint' is a unique name on your server identifying a particular stream - it looks like a filename, such as '/stream.ogg'. A listener can only listen to a single mountpoint at a time. This means you can have a single icecast server contain either multiple broadcasts with different content, or possibly the same broadcast but with streams of different bitrates or qualities. In this case each broadcast or stream is a separate mountpoint.

- Static
- Dynamic

Configuration

- Concurrent Clients
- Sources
- Timeouts
- Authentication
- Relay
- etc

Admin interface

Through this interface the user can manipulate many server features. From it you can gather statistics, move listeners from mountpoint to mountpoint, disconnect connected sources, disconnect connected listeners, and many other activities.

Relying

Relaying is the process by which one server mirrors one or more streams from a remote server. The servers need not be of the same type (i.e. icecast can relay from Shoutcast). Relaying is used primarily for large broadcasts that need to distribute listening clients across multiple physical machines.

Type of Relays

There are two types of relays that icecast supports. The first type is when both master and slave servers are icecast2 servers. In this case, a "master-slave" relay can be setup such that all that needs to be done is configure the slave server with the connection information (serverip:port) of the master server and the slave will mirror all mountpoints on the master server. The slave will also periodically check the master server to see if any new mountpoints have attached and if so will relay those as well. The second type of relay is a "single-broadcast" relay. In this case, the slave server is configured with a serverip+port+mount and only the mountpoint specified is relayed. In order to relay a broadcast stream on a Shoutcast server, you must use the "single-broadcast" relay and specify a mountpoint of "/".

Yellow Pages

A YP (Yellow Pages) directory is a listing of broadcast streams. Icecast2 has it own YP directory located at http://dir.xiph.org. Currently icecast2 can only be listed in an icecast2-supported YP directory. This means that you cannot list your stream in the Shoutcast YP directory.

In the icecast2 configuration file are all the currently available YP directory servers. Listing your stream in a YP is a combination of settings in the icecast configuration file and also in your source client.

Listener Authentication

Listener authentication is a feature of icecast which allows you to secure a certain mountpoint such that in order to listen, a listener must pass some verification test. With this feature, a simple pay-for-play operation (eg user/pass), or some filtering based on the listener connection can be performed. This section will show you the basics of setting up and maintaining this component.

To define listener authentication, a group of tags are specified in the group relating to the mountpoint. This means that authentication can apply to listeners of source clients or relays.

Demo

Wrap-up

Questions