The GStreamer Multimedia Architecture

Steve Baker
steve@stevebaker.org

What is GStreamer

- A library for building multimedia applications
- Allows complex graphs to be built from simple elements
- Supports any data-type for which elements exist
- Plugin support allows media specific extensions
- Main core dependency is Glib
- Runs on Linux (x86 & PPC), Solaris and FreeBSD

What is GStreamer

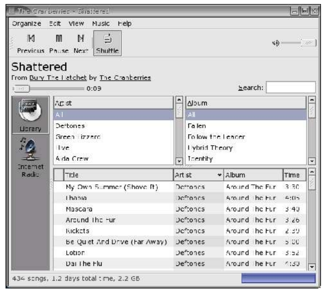
- Uses 3rd-party libs where possible
  - Code re-use is good
- GStreamer is not a sound server
  - But you could build one with it
- Associated with GNOME but is graphics-toolkit agnostic
  - KDE may adopt GStreamer

Why create GStreamer

- Linux (and other *nixes) have bad media handling
- Many hacked single-purpose apps and libs exist
- Other platforms have multimedia frameworks
  - DirectShow
  - QuickTime
  - BeMedia
  - RealMedia SDK

Applications using GStreamer

- Rhythmbox
- Totem, Gst-player
- Gnome-Sound-Recorder, gst-mixer
- Sound-juicer, Marlin
- Video-whale
- Gnonlin, VDV
- Your App Here...


The Cranberries - Shattered

Organize Edit View Music Help

Previous Pause Next Shuffle

Shattered

From Bury The Hatchet by The Cranberries

0:09 Search:

Library

Internet Radio

Artist	Album
All	All
Deftones	Fallen
Green Lizzard	Follow the Leader
Live	Hybrid Theory
Aida Crew	Identity

Title	Artist	Album	Time
My Own Summer (Shove It)	Deftones	Around The Fur	3:30
Lhabia	Deftones	Around The Fur	4:05
Mascara	Deftones	Around The Fur	3:40
Around The Fur	Deftones	Around The Fur	3:26
Rickets	Deftones	Around The Fur	2:39
Be Quiet And Drive (Far Away)	Deftones	Around The Fur	5:00
Lotion	Deftones	Around The Fur	3:52
Dai The Flu	Deftones	Around The Fur	4:30

434 songs, 1.2 days total time, 2.2 GB

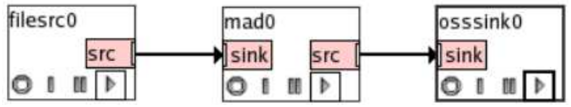
Intellectual property issues

- Core library is licensed as LGPL
- Apps are becoming LGPL to avoid GPL patent issues
- Plugins allow distributions to ship core GStreamer but leave out patent-encumbered libraries
  - For example mp3, Sorenson, etc
- Third party vendors could distribute binary-only GStreamer plugins (but it hasn't happened yet)

Language Bindings

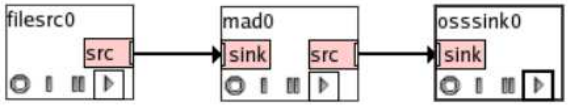
- Core library is written in C
- Python, Perl, Ruby, Guile
- Java, C#, C++
- QT style C++ (for KDE)
- Not all bindings are as mature as others

Tools and utilities



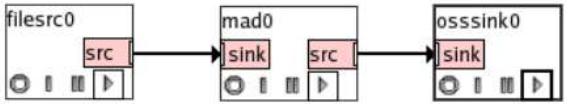
- Gst-editor – graphical pipeline building
- Gst-launch – command-line pipeline building
  - filesrc location=”britney.mp3” ! mad ! osssink
- Gst-register – updates registry of plugin features
- Gst-inspect – gives info about installed plugins

Pipelines and elements



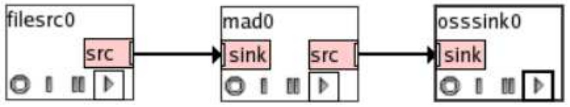
- Elements are joined together to form complete pipelines
- Each element performs a single function
- Data and events flow from left to right

Pads and capabilities



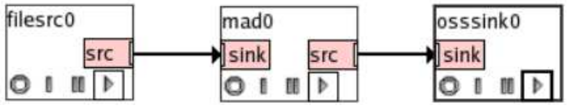
- Elements have pads which can be connected
- Data and events flow through pads
- Before data can flow pads must negotiate mutual capabilities
- Some pads only appear on-demand

Buffers and memory usage



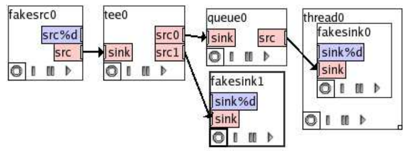
- Data travels in objects called buffers
- Data might not always be writeable
- Allocating and copying memory is avoided where possible
- Optimisations are possible without all elements being aware

Chain, get and loop elements



- “Chain” elements are the simplest and fastest – one buffer in, one buffer out
- “Loop” elements are required for more than one source or sink, or if number of buffers going in differs from those coming out
- “Get” elements only have source pads

Core elements

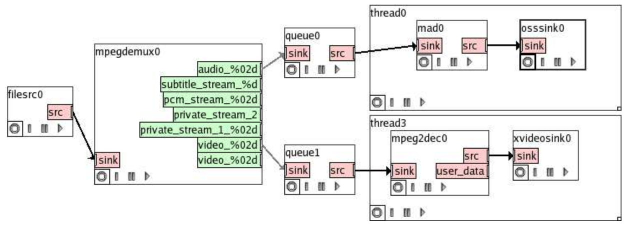


- Some elements contain other elements (Bins)
- The "thread" element is a bin which runs its children in a separate thread
- The "queue" element is placed between threads to decouple the data flow
- The "tee" element splits one input into more then one output

Input, output, codec and filter plugins

- Inputs – file, GnomeVFS, UDP, DVD
- Outputs – file, sound card, X display, SDF, network
- Codecs – mp3, vorbis, ffmpeg
- Filters – LADSPA, effectTv
- Plus many others

Clocks and synchronisation



- For A/V playback, always sync to the sound card
- Sound card clocks are precise but not accurate
- Any real-time elements synchronise to the "default" clock

Schedulers and cothreads

- Schedulers decide when each element needs to process data
- Cothreads are user-space threads which are simpler to use and faster than real threads
- Original scheduler created a cothread for each element
- Current default scheduler doesn't need cothreads, so it runs on more platforms but has higher latency

Autoplugging with spider

- Each media type has its own type-finding function
- Spider finds the media type of a stream, then finds an element to decode it
- Elements are added until a complete pipeline can be built

New features in development

- Rewritten subsystems for content negotiation, metadata, capabilities
- Element interfaces allow full interaction between apps and elements
- Current interfaces include
  - Audio mixer (used by gst-mixer in gnome-media)
  - Video overlay
  - Property probing (to find hardware device names)
  - Interactivity for DVD menus and Flash

An ogg player in python

```
import sys
from gstreamer import *

def main():

    bin = Pipeline('pipeline')

    filesrc = gst_element_factory_make ('filesrc', 'disk_source');
    filesrc.set_property('location', sys.argv[1])

    decoder = gst_element_factory_make ('vorbisfile', 'parse');

    osssink = gst_element_factory_make ('osssink', 'play_audio')

    #  add objects to the main pipeline
    for e in (filesrc, decoder, osssink):
        bin.add(e)

    # link the elements
    previous = None
    for e in (filesrc, decoder, osssink):
        if previous:
            previous.link(e)
        previous = e

    # start playing
    bin.set_state(STATE_PLAYING);

    while bin.iterate(): pass
```

More Information

- http://gstreamer.net/
- #gstreamer on irc.freenode.net
- Any questions?