G-Streamer Subsystem

Overview

GStreamer is an open-source multimedia framework written in C using the GObject framework. It has a pipeline-based architecture, which allows to freely configure the data flow across a variety of plugins that handle different data formats.

Multimedia processing in GStreamer is done by connecting several **elements** into a pipeline. Each element contains one or more **pads**, which serve as connection points between them. Pads have a direction and one or more data types they can handle. A pipeline is created by connecting elements using pads. In order for the connection to succeed the pads must have opposite direction and it must be possible to establish a common data type that both of them understand. The latter is done in process of a format negotiation, which happens during pad connection.

A set of elements connected with each other can be grouped in form of a **bin** object, which is externally usable as a regular element. This allows to create larger pipeline fragments for easier management.

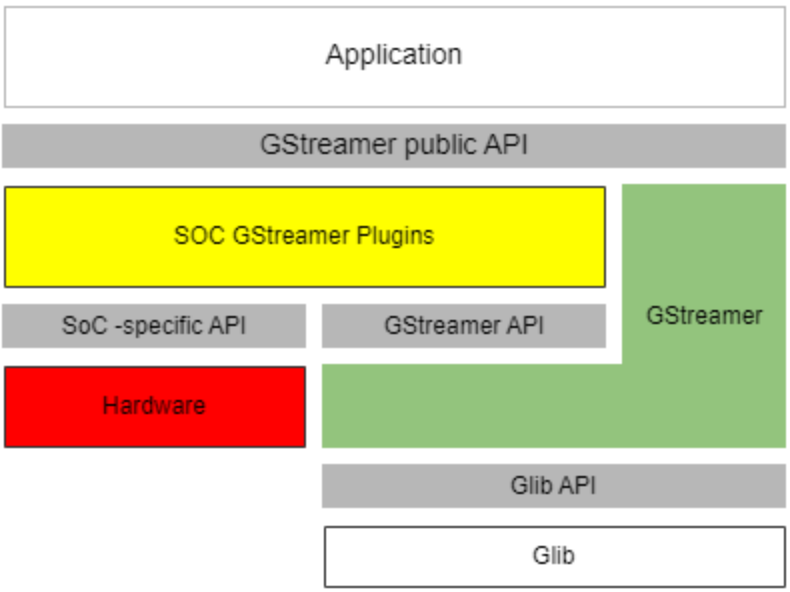
All elements and bins connected together form a **filter graph**.

GStreamer elements are implemented as shared objects are provided by plugins. The GStreamer distribution offers several basic sets of plugins:

- **gst-plugins-base** - contains high-quality, well-maintained plugins handling the basic multimedia processing tasks.
- **gst-plugins-good** - good-quality plugins that the community consider well-written and correctly working. All of these plugins also have a clean license - LGPL (or compatible).
- **gst-plugins-bad** - a set of plugins that although being mostly good but are often missing something (a good review, documentation or active maintainer) and therefore are not up to be considered "good".
- **gst-plugins-ugly** - these plugins, although having relatively good quality, may cause distribution problems mostly due to licensing problems.

The RDK requires only the "base" and "good" plugin sets. Additionally some SOC-specific plugins plugins may be required for hardware-accelerated audio /video playback.

Component Interactions



Exported APIs

Applications can interact with GStreamer by using a GObject-based API, which implements basic object-oriented programming using pure C language. This is similar to what is used in all Gnome-based components such as Glib or GTK+.

A detailed documentation for this API can be found in the GStreamer Core Reference Manual on the project website.

Required APIs

GStreamer is built on top of the GObject framework provided by the Glib library (not to be confused with glibc, which is a standard C library).

Porting

The GStreamer core is portable and usually won't require any changes in order to build for a target platform. The availability of plugins will vary depending on the dependencies installed.

GStreamer comes with a number of audio and video playback plugins, however in order to optimally use the capabilities of the target chipset the SOC vendor is required to create a set of plugins to provide audio and video playback using the SOC-specific methods with dedicated hardware acceleration.

GStreamer in RDK

RDK Media Framework (RMF) uses GStreamer as a playback mechanism for various types of media available on CPE devices. RMF acts as primary application for managing GStreamer based playback. RMF, with help of gstreamer can be used to play linear QAM video, Home Network Content, HTTP Dynamic/Live/Smooth Streaming and other media streams.

RMF makes use of gstreamer elements. Constructing an RMF media pipeline by connecting various RMF elements results in the formation of a gstreamer pipeline made up of the gstreamer elements used to implement each RMF element.

The document **RDK GStreamer Guidelines.pdf** describes the guidelines for GStreamer plugins on CPE devices which support RDK Media Framework.

RDK GStreamer Plugins

Based on the device type and capabilities RMF requires the support of several GStreamer plug-in elements.

Source Elements

- httpsrc (Generic Gstreamer plugin)
- qamtunersrc (Device-specific)

Parse/Filter elements

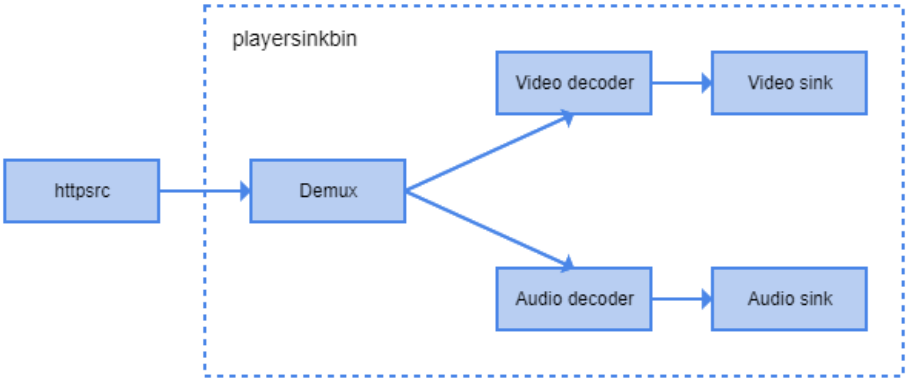
- rbifilter (Generic)
- dtcp (Generic + SoC)

Sink elements

- playersinkbin (Generic + SoC)
  Bin which includes demux, decoder and sink elements
- httpsink (Generic)

Sample GStreamer Pipeline in RDK

An RMF pipeline of hnsource playersinkbin will invoke the following gstreamer pipeline of httpsrc playersinkbin



AAMP

AAMP (Advance Adaptive Media Player) is an application which uses gstreamer to present IP Video Streams.This player is built on top of GStreamer framework and is capable of playing Apple HLS & MPEG DASH contents over IP.

AAMP as a Plugin

- Enables HTML5 based playback
- WebKit manages the GStreamer pipeline
- AAMPs GStreamer plugin ‘Gstaamp’ is loaded in the GStreamer pipeline
- The ‘Gstaamp’ creates AAMP player instance
- Audio /Video buffers are pushed to ‘Gstaamp’s srcpads