Ray Alfano

4/26/2013

Revised 5/4/13

Network Computing

Draft Project Writeup for Synchronous Multi-User Music Production Via the Internet

The purpose of the project is to provide two musicians the ability to perform together over an HTTP connection with a high level of audio fidelity.

Vocabulary used:

PCM: Pulse-Code Modulation, the digital representation of an analog audio signal, the fidelity or quality of which depends on the number of digital samples taken of it in a time frame and also the bit depth of the representation of the digital samples.

Opus format: A new, lightweight, and relatively high-quality encoding format for audio that I attempted to use for my program.

Sub-goals:

- In abstract, educate myself on the nature of Internet audio.
- Gain significant experience with Python's capabilities for digital signal processing and audio.
- Provide the greatest level of audio fidelity possible for a connection's speed over time.
- Emphasize real-time nature of music performance balanced against audio fidelity.
- Overcome echo and latency issues over an Internet connection.
- Research the latency issues involved in RTP transmission of PCM audio data to gain perspective for my potential master's thesis that would expand on this project.

Libraries used:

- GStreamer for Python, for the channel between clients.
- GStreamer Tools, for debugging and informational purposes.

Data requirements as observed:

- For sending, the worst-case PCM WAV data is 1.35Mbits/sec to achieve the target high-quality benchmark for my program, which is CD-quality audio (44100Hz at 16-bit resolution)
- If possible I will also include more complex DVD-Audio quality (48000Hz at 16-bit resolution), although this is not supported by all sound cards and may not be useful. Still

undecided.

- See the additional document labelled “data requirements” which was part of progress report 1. This document will be integrated into the narrative of the final write-up.

Performance requirements:

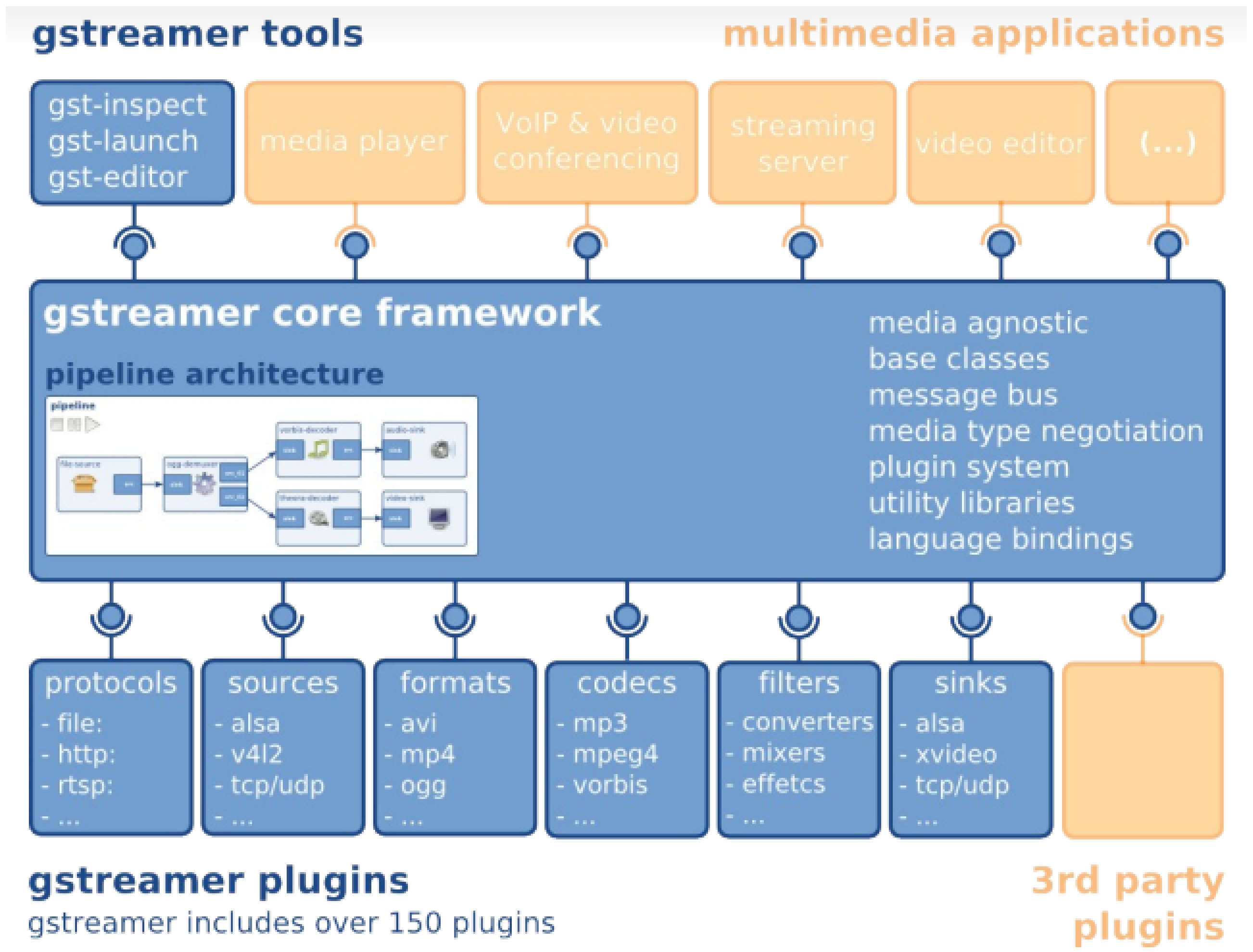
- Minimum requirements to use my program at all is a 56kbps connection.
- A fixed IP address for each host is also required for this program to work in the multiple-performer context.
- RTP and RTCP access will be necessary for the executing host and client.
- These requirements will be reformed as I get definitive statistics generated from my final program runtime.

Audio fidelity requirements:

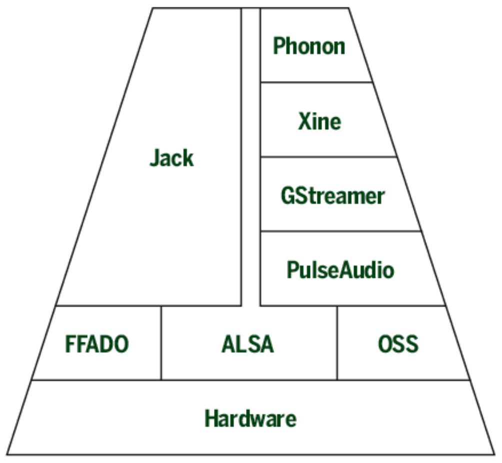
- At this stage, and it may change by the final project, input PCM audio data is given at CD quality (44.1kHz/16 bit). If possible, higher quality inputs will be tested, but some systems are unable to handle higher resolution audio and it can disrupt the usability of my program.
- In situations where the connection speed is variable and inconsistent, the program will adjust the sound sampling quality downward to a minimum of FM radio quality (12kHz sample rate) to try and maintain the real-time nature of the co-performance.
- The default Linux audio test is a flat tone at 8kHz, the ability to play that tone is the minimum system requirement for the audio system as detected by my program.
- If possible, I intend to add code to detect the capabilities of the input device being used. Some laptop microphones are intended only for speech and are therefore limited to 8kHz. The program should take that into account.icrStill needed for the final report:
- Examples of my final program runs illustrating the latency in the functioning system.
- Detailed description of where latency is introduced in the signal path from end-to-end.
-Detailed discussion of why my original plan to use Opus audio format was unproductive and a section on why “throttling” the encoding bitrate is more costly in time requirements than it is to simply transmit the raw PCM data.

- Charts and graphs showing latency for typical test cases.

Images to be integrated into final draft narrative:
Gstreamer at a glance:



Linux audio stack, used to explain why I switched to Linux to take advantage of the very low-level support for audio in ALSA:


Phonon
Xine
Jack
GStreamer
PulseAudio
FFADO
ALSA
OSS
Hardware