PipeWire Portal System

31th September 2018 **By: Saurav Malani**

Do you confirm that you are eligible for this internship as stated in our eligibility rules at https://wiki.gnome.org/Internships#Eligibility?.

Answer: yes.

Do you confirm that you have read and agree with the internship contract linked at https://wiki.gnome.org/Internships#Eligibility?.

Answer: yes

Personal Details

Name: Saurav Malani

Preferred pronoun: He

E-mail address: sauravmalani1@gmail.com

Blog URL: https://medium.com/sauravmalani1

IRC nick: malani

Twitter URL: https://twitter.com/SauravMalani

Website or Portfolio URL: https://researchweb.iiit.ac.in/~saurav.malani/

GitHub: https://github.com/saurav-malani

Gerrit: https://git.opendaylight.org/gerrit/#/q/malani (Linux Foundation)

LinkedIn URL: https://www.linkedin.com/in/saurav-malani-686854114/

Any other online presence URL you wish to provide: Quora

Location: Hyderabad, India (Time Zone- IST)

Education Details:

College: IIIT-Hyderabad

Branch: Computer Science

Degree: B-tech+MS by Research

Year: 4th-year

How did you hear about this internship?

I am an open source enthusiast, since my 1st year in college. As, from my childhood I always dreamt of being part of something great, creating dent in the society. But, before being introduced to open source I always wondered, how would I be able to achieve this alone. But, this is exactly how open source empowers us. Isn't it amazing that irrespective of from where you are, where you live and what you do, you can be part of this open source community and hack around stuffs that are open source to create new stuff, which further might solve problems of millions.

So, I follow all the cool open source organizations on twitter, facebook, read their blogs, to get updates. And always waiting for a chance to contribute to projects. I Previously I have done remote internships at Tor Summer of Privacy, Linux Foundation (Networking), by getting updates from there blogs. And, similarly I got to know about GNOME internship program from wiki.gnome.

Project Details

What project are you interested in?

Pipewire Portal System

Who is a possible mentor for the project you are interested in?

Wim Taymans

Please describe your experience with the GNOME community and GNOME projects as a user and as a contributor.

As a user, since last 4 year I have been using Gnome Desktop environment, Pulse Audio, GStreamer, keyring and many other applications. As a developer, I have developed a Tic-Tac-Toe game using GTK+ & a Chat Application which required the knowledge of Python, GTK+, Twisted and Crypto.

GOALS

The main goal of this project is to make the pipewire library use a dbus connection to the gnome portal to set up a session.

Project Overview

Pipewire is an API to deal with multimedia pipelines. In short, it is PulseAudio for audio/video, i.e. a middleware between applications and hardware devices.

Pipewire solves the following problems:

1. Sharing of Multimedia Devices: Previous architecture did not support sharing of multimedia devices i.e. previously it was not possible for two or more applications to share same video hardware device.
2. Provides an efficient method for sharing of multimedia between applications like for instance fullscreen capture from compositor (like Gnome Shell) to video conferencing application running in browser like Google Hangout.
3. Provides a security measure for device sharing.
4. Syncing of Audio/Video files was always a pain.
5. Latency: This new infrastructure provides a low very low-latency for both audio and video processing.
6. It makes interacting with audio and video devices from containerized application easy, by introducing security measures.
7. Also, Provides support for sandboxed applications.

Apart from all this pipewire also handles all the use cases currently being handled by PulseAudio and Jack.

Problem with Pipewire

Till now pipewire uses local socket to connect to the daemon. But, desktop applications are moving towards primarily being shipped as containerized Flatpaks i.e. sandboxed. But, sandboxed applications cannot use local socket. So, we need to use some other mechanism to connect to the daemon in case of Sandboxed applications.

Suggested Solutions

We can use D-Bus to solve this problem.

D-Bus is an inter-process communication mechanism (IPC), a medium for local communication between processes running on the same host. D-Bus is fast and lightweight and is designed for use as a unified middleware layer underneath the main free desktop environments. D-Bus behaves as a remote procedure call mechanism and provides its own marshaling.

Two major components of D-Bus:

1. Dbus Library, used by any two processes for peer-to-peer communication.
2. Dbus daemon, to which any number of processes may be connected at any given time.

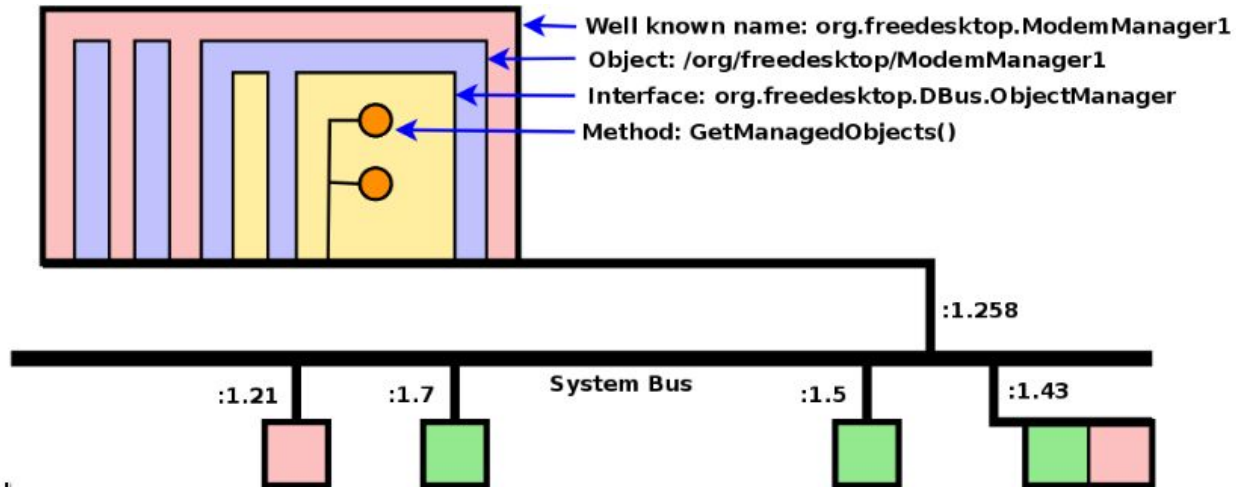
Here in case of sandboxed application, we will use Dbus daemon. Bus daemons are started using the d*bus-launch* command, which in turn runs *dbus-daemon.* And the start of the bus is indicated by configuration file.

In general message exchange on protocols like TCP or UDP is symmetric i.e. data is always transferred from one port to another. D-Bus presents a more sophisticated model, where the sending and the receiving side of the message are never quite of the same type. It is more like OOPs.
So, while using D-Bus we need to know the following details:

1. Unique name
    a. Servers: well-known name
2. Objects
3. Interface
4. Methods
5. Properties
6. Signals

Eg. When a client invokes a method or listens for a signal, it must indicate the object and the member it is referring to. In addition to object and member, the client may also name the

interface in which that member was specified.



Brief implementation idea:

1. the PipeWire client load the portal module which hooks into the pw_remote_connect() method
2. When a client does a pw_remote_connect() the portal module intercepts the call and does a DBus call to the portal. The client needs to give hints about what it wants to do in the properties.
3. the portal gets the client request and opens a PipeWire connection.
4. the portal configures the session, it can list the devices and present those in a user interface. The user grants access to selected devices.
5. the portal can choose to hide objects from the session as well
6. the portal returns the PipeWire connection socket to the client
7. the client continues the PipeWire session on the socket

Rough Timeline

Date	Target
2018-12-01 (1st two Week)	Setup the development environment i.e. making everything work with the script and pipewire and getting everything set up.
2018-08-15 (2 week)	1. Learn to write DBus code. 2. Write some sample programs. 3. Understanding the working of existing pipewire code.
2019-01-01 (4 weeks)	1. Implement the portal calls. 2. Learn to handle the asynchronous

	replies and keeping it's track, cleanups and handling errors.
2019-02-01 (2 weeks)	Integration to pipewire.
2019-02-14 (10 days)	Testing on some test apps and use the gstreamer elements.
2019-02-24	Finish the remaining work and submit the final report.

Please describe any relevant projects that you have worked on previously and what knowledge you gained from working on them (include links):

I have done remote internship twice, at TOR Project, and at Linus Foundation (Networking).

Brief Project Details of FOSS Project

1. TOR Browser: In this I worked with core Tor-dev team to add Metadata Stripper feature in Tor Browser. This involved stripping Exif, IPTC, XMP metadata from image/audio/video files before uploading with the help of exiv2. Check this more details of Project
2. Linux Foundation (Networking): In this I developed a Data Visualization Tool for Bottlenecks Project for log files. This involved searching in NOSQL database, appropriate data transformation and visualization of data. Check this more details of Project
3. OpenDayLight: volunteered by solving few bugs.