

Permissions, Isolation and Streams:

Redesigning WirePlumber's Core

Julian Bouzas
Senior Software Engineer

GStreamer Conference 2026
Prague, CZ

October 10th - 11th



COLLABORA

Hi, I am Julian

- From Spain
- Multimedia Team at Collabora since 2019
- Mostly PipeWire and WirePlumber developer
- julian.bouzas@collabora.com



- PipeWire and WirePlumber
- What's New?
 - The Permission Manager
 - The Client Context
- What's Next?
 - Drop Session Item Design
 - Introduce Stream Design



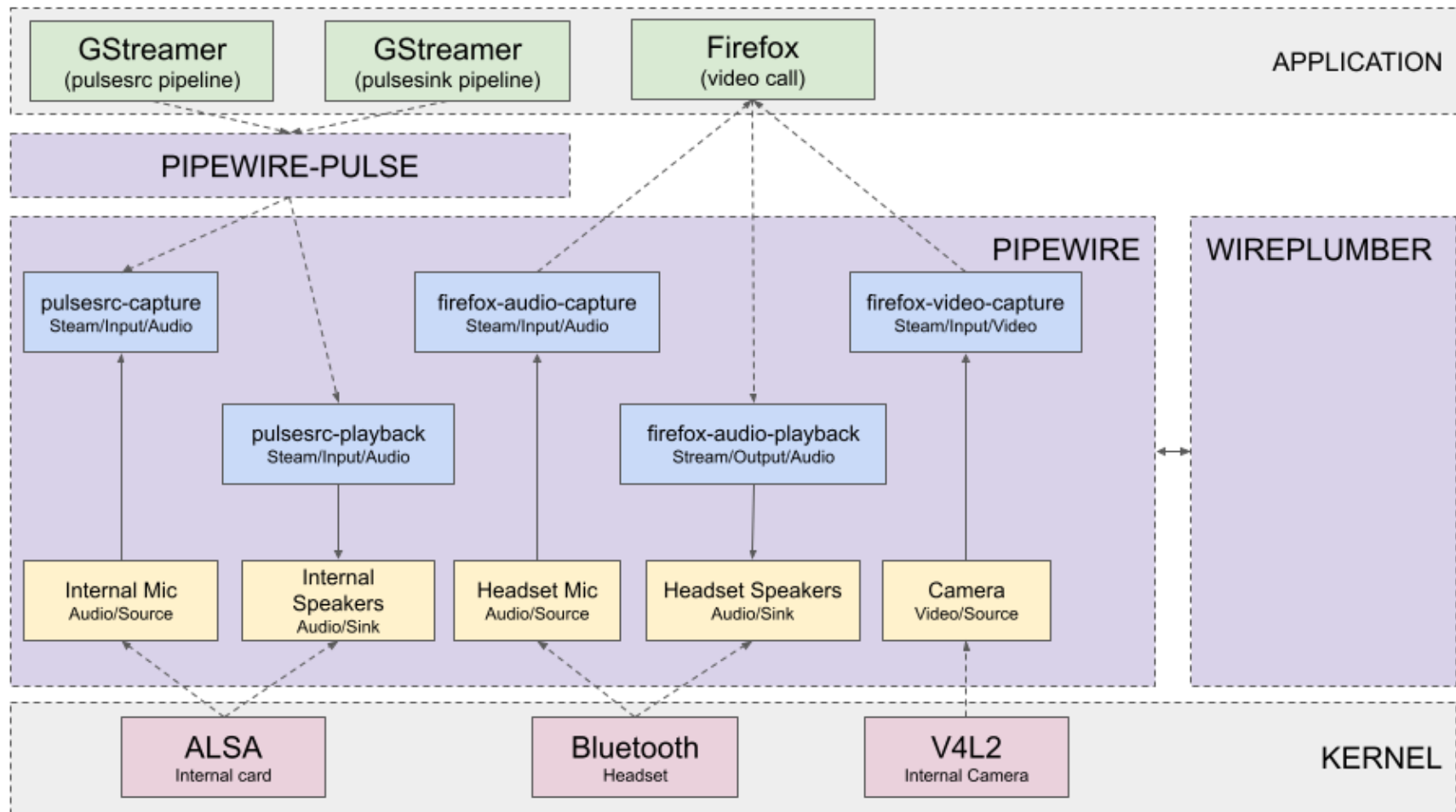


Part 1:

PipeWire and WirePlumber

PipeWire and WirePlumber

- They are Multimedia daemons to handle audio and video on Linux
 - Already included by default in most Linux distributions
 - There are 2 separate daemons because they have different purposes
- PipeWire only provides the mechanism
 - Applications connect to it
 - Notifies the session manager (WirePlumber) when an event happens, and handles its requests
- WirePlumber only decides the policy
 - The Policy defines:
 - Who may see and use which multimedia devices
 - What gets linked to what
 - How devices and nodes are configured
 - The Policy is written as Lua scripts and has a extensive JSON configuration



COLLABORA

Open First



COLLABORA

Part 2:

What's New?



COLLABORA

The Permission Manager

How PipeWire permissions work

- Each client-object pair has a permission bitmask:
 - r = visible
 - w = modify
 - x = call methods
- PipeWire only enforces these bits
- WirePlumber supplies them
- Now centralized with permission managers

The Permission Manager

- Reusable object that computes and updates the permissions of all clients attached to it
 - A client connects
 - A matching object is added or removed
- One instance per category: Config, Flatpak, Snap, Portal, etc...
- Fallback mechanism to avoid conflicts
 - Config → Flatpak → Snap → Portal → Default

Configuration

- Users can define custom permission managers
- Can be applied to any client
- Example
 - Remove all permissions on all capture devices for paplay

```
1  access.permission-managers = [  
2    {  
3      name = "no-mic"  
4      default_permissions = "all"  
5      core_permissions = "rx"  
6      rules = [  
7        {  
8          matches = [  
9            { media.class = "Audio/Source" }  
10         ]  
11         actions = { set-permissions = "-" }  
12       }  
13     ]  
14   }  
15 ]  
16  
17 access.rules = [  
18   {  
19     matches = [  
20       { application.name = "paplay" }  
21     ]  
22     actions = {  
23       update-props = {  
24         permission_manager_name = "no-mic"  
25       }  
26     }  
27   }  
28 ]
```





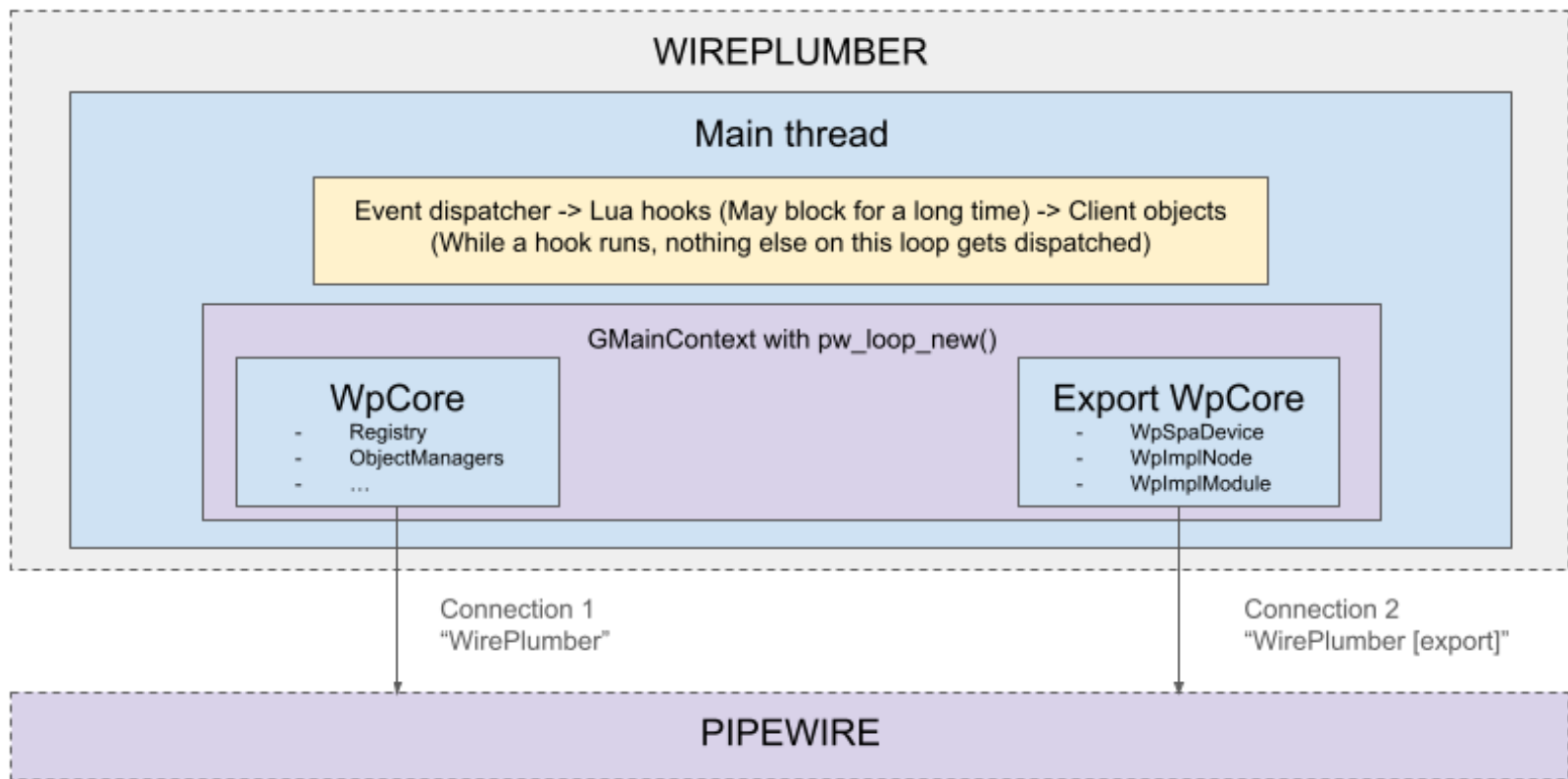
COLLABORA

The Client Context

What WirePlumber runs in-process

- WirePlumber doesn't only defines the policy
- Act as a client to run local media processing objects:
 - For example: Loopback and Filter-Chain objects
- Those local objects use a 2nd pw_core connection
- No isolation with Lua Scripts and Event Dispatcher
 - Share the main pw_context → Single-Threaded





Problem with single-thread

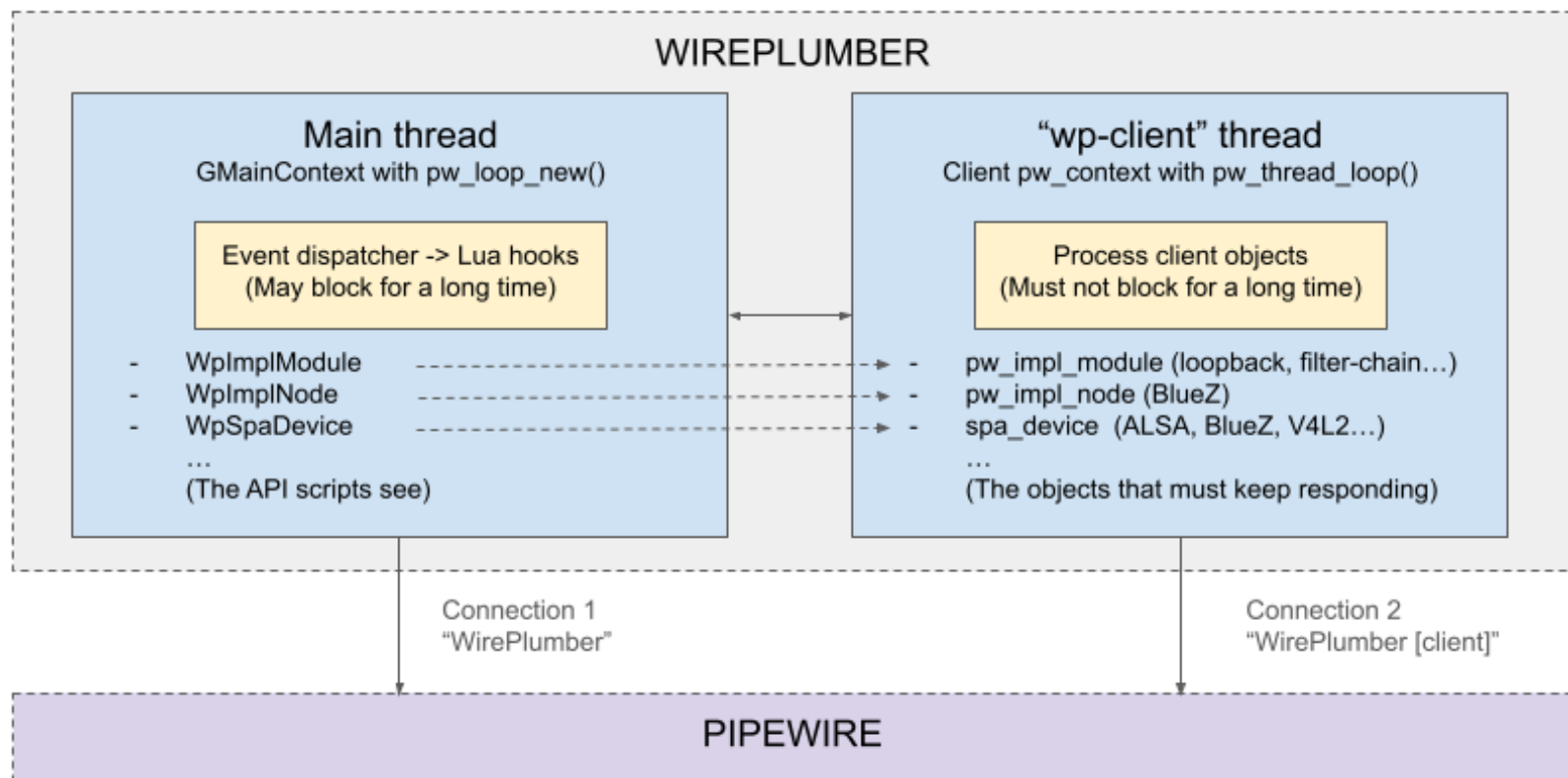
- Lua scripts can take a noticeable amount of time:
 - For example, rescans, big rule sets, D-Bus calls...
- The control path of every processing object can stall:
 - State changes, format negotiation, param updates, etc...
- A visible symptom
 - Loopbacks can lose their first periods of audio

Solution with WpClientContext

- Isolate media processing objects with a 2nd context
 - Main pw_context: event dispatcher and Lua scripts
 - Client pw_context: media processing objects
- Multi-threaded

Keeping it safe

- All public (global) objects stay on the main thread
- Calls into local (private) objects take the thread-loop lock
 - Only done when creating and destroying them
- No public API changes
 - No need to port any Lua script





Part 3: What's Next?



COLLABORA

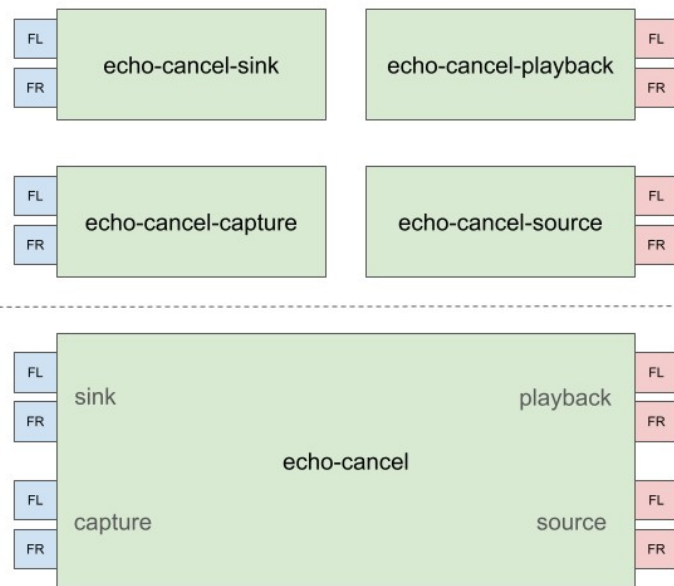
Drop Session Item Design

WpSessionItem

- Linking 2 nodes in PipeWire is not simple
 - Need to first configure ports if the number of audio channels don't match
 - Might need to create more than one link between 2 nodes
 - For example: link a mono node with a stereo node
- Session items wrap nodes and add extra functionality to easily do that
- Worked well until...

PipeWire introduces complex nodes

- Nodes with multiple directions:
- Example: echo-cancel:
 - From: 4 nodes without streams
 - To: 1 node with 4 streams



Why refactoring?

- Too generic and hard to manage:
 - Need to constantly check properties to know what type of item we have
 - Lua scripts are responsible to create and destroy them
- Old codebase:
 - Originally designed for other ideas that never really happened
- Cannot have multiple Session Items per node
 - Still need refactoring even if we decide to keep the Session Item concept
- More experience with PipeWire
 - Good opportunity for a better design



COLLABORA

Introduce Stream Design

WpStream

- A group of ports on a node
 - Same direction and media type
 - Very similar to a Pad in GStreamer
- First class citizens in WirePlumber
 - Lua scripts can handle them in the same way as any other object
- Subclasses:
 - WpAudioStream: handles audio formats and configures ports
 - WpVideoStream: handles video formats

WpStreamManager

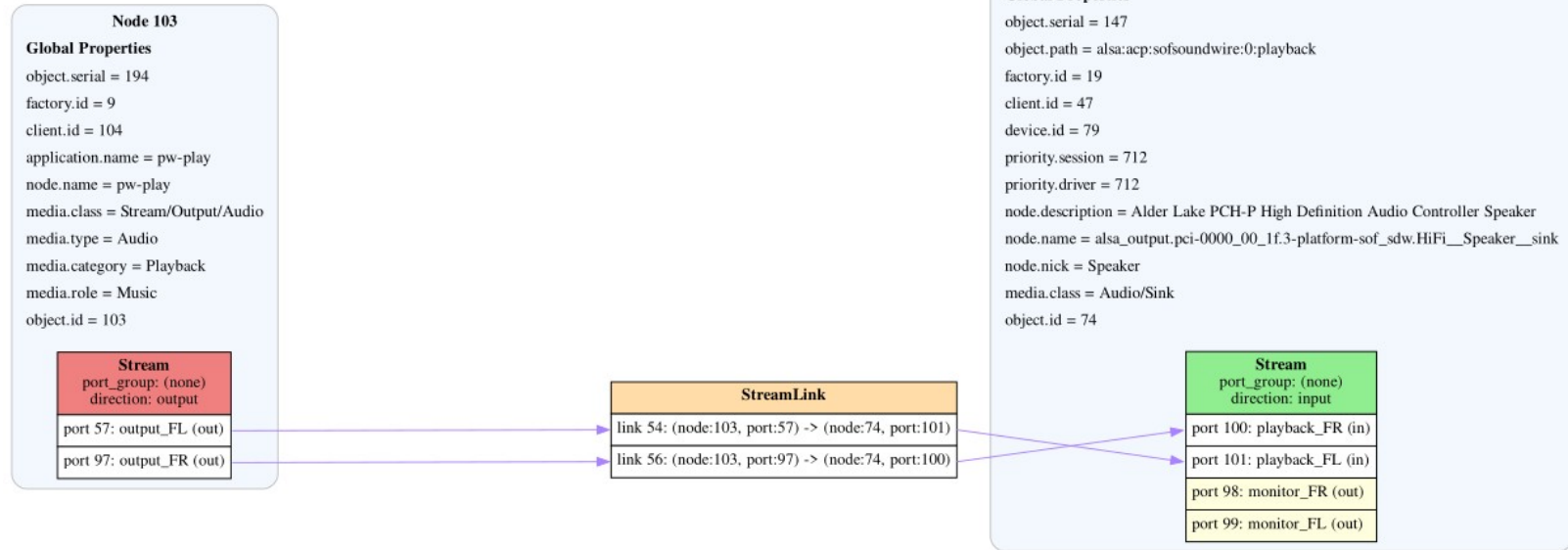
- Manages automatically streams when nodes are added and removed
- Users don't need to worry about creating and destroying streams
- The create-item.lua script is gone

WpStreamLink

- A group of links that connect 2 streams
- Owns the individual PipeWire port links
- 2 sub-classes:
 - WpAudioStreamLink: configures ports and creates links
 - WpVideoStreamLink: creates links
- Able to recover if an audio stream reconfigures its ports
 - No need to destroy and create it again
 - For example: if switching between mono and stereo

WpStreamLinkManager

- Creates and tracks all stream links
- Also adopts links that WirePlumber didn't create
 - For example: links created with pw-link or by patchbays
- Cleans up automatically when a node, port or link disappears
 - No need to handle them in Lua scripts either



What this means for you

- This is an API break: WpSessionItem is removed entirely
- Will be part of the next major release 0.6.0
- The linking scripts are ported to streams and stream links
 - Any custom scripts that use session items need to be ported
- Upstream MR:
 - https://gitlab.freedesktop.org/pipewire/wireplumber/-/merge_requests/895

Status

- Hosted on Freedesktop
 - PipeWire: <https://gitlab.freedesktop.org/pipewire/pipewire>
 - WirePlumber: <https://gitlab.freedesktop.org/pipewire/wireplumber>
- Welcome to contribute





Questions?



COLLABORA

Open First