GStreamer Daemon: Project Update

Miguel Taylor-Lopez

RidgeRun

Agenda

- Introduction
- Refcount Commands
- Action support
- New GstD clients
- Other changes
- Conclusion and Q&A

Introduction

RidgeRun

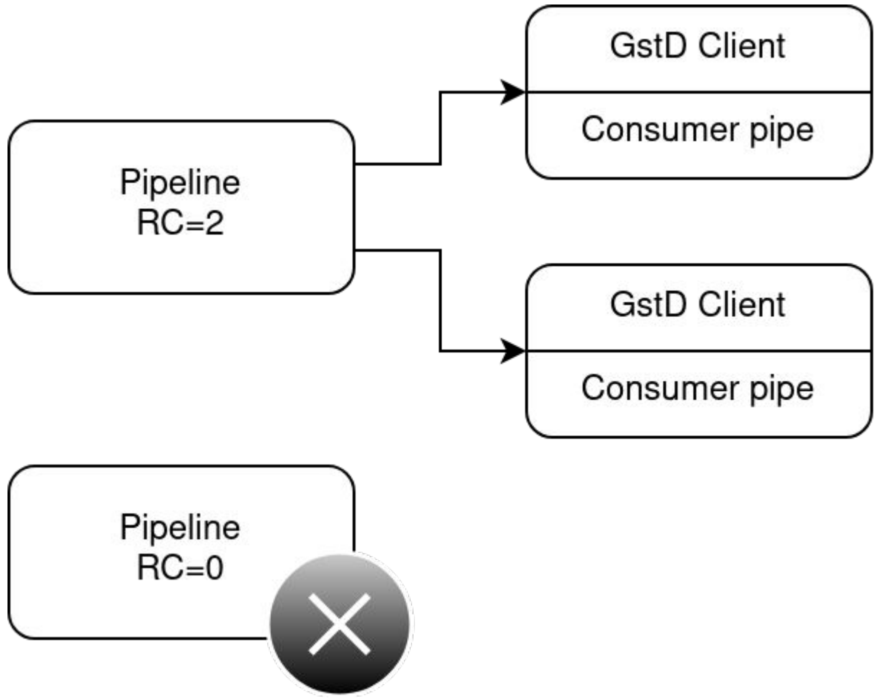
- RidgeRun is a software development and service integration company that specializes in embedded systems across various industries.
- Our areas of expertise include:
  - Embedded Linux.
  - Artificial Intelligence.
  - Computer Vision.
  - FPGA.
  - **GStreamer.**

gstd

- GstD is a GStreamer framework for controlling audio and video streaming via InterProcess Communication (IPC).
  - Versatile control
  - Production deployment
  - Facilitates automation and remote control

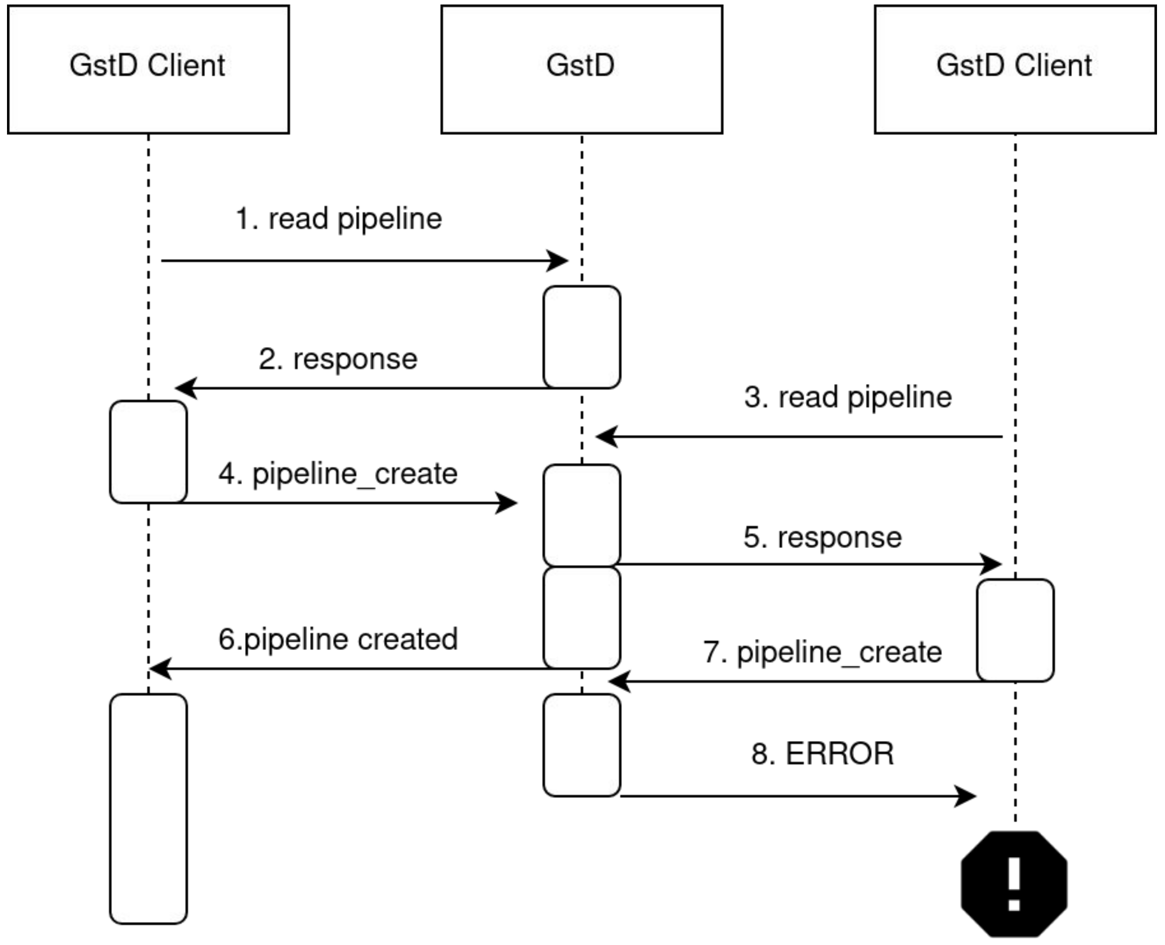
Refcount Commands

- The GstD refcount commands provide an alternative method to interact with pipelines, simulating the use of reference counters.
- It is essential to avoid mixing these commands with their regular counterparts to prevent unexpected behavior.
- Currently, GstD implements the following commands based on the refcount concept:
    - **pipeline_create_ref**
    - **pipeline_delete_ref**
    - **pipeline_play_ref**
    - **pipeline_stop_ref**



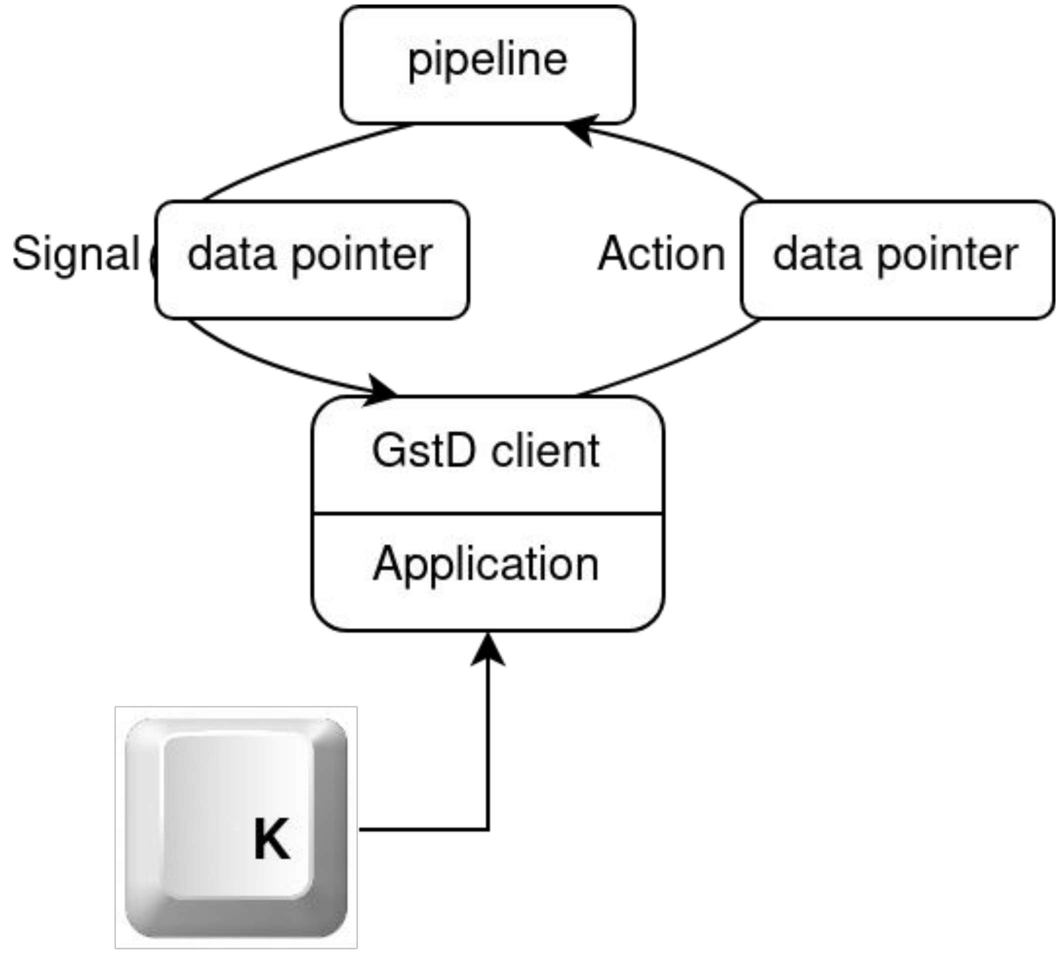
Thread Safety

- The refcount commands offer enhanced thread safety when multiple processes share a single pipeline.
- In a scenario where multiple processes share a pipeline, basic commands can lead to issues like double creation, premature deletion, or unexpected stops.
- Even with inter-process communication, basic commands do not guarantee thread safety.



Action Support

- GstD has extended its functionality with enhanced Action Support.
- Actions in GstD represent signals from applications to elements.
- These signals trigger specific behaviors or operations within elements.



Application Scenarios

- Media Streaming: Dynamic control of media streams.
- Automation: Implement complex workflows using signals and actions.

Python Client for GstD

- Python package.
- Can be installed standalone with .deb or pip.
- Enables communication with GstD via TCP socket.
- Includes a versatile logger class based on Python logging module.

```python
from pygstc.gstc import *
from pygstc.logger import *

gstd_logger = CustomLogger("gstd", loglevel="DEBUG")
gstd_client = GstdClient(logger=gstd_logger)

gstd_client.pipeline_create(...)
...
```

Other Changes

- Migration to Meson.
- Debian package generation.
- More clients: HTTP, javascript
- Deep notify support
- Run GstD as an application

Conclusions

Features Recap

- **Thread-Safe Implementation**: Ensures safe interaction with pipelines from multiple clients.
- **More Client Options**: Offers new methods for controlling GstD, including Python, HTTP, and JavaScript.
- **Improved Communication**: Provides a more flexible communication approach with pipelines through actions.
- **Simplified Distribution**: Transitioned to the Meson build system and .deb packaging, simplifying GstD distribution.

GStreamer Daemon Project Update

Miguel Taylor-Lopez

RidgeRun