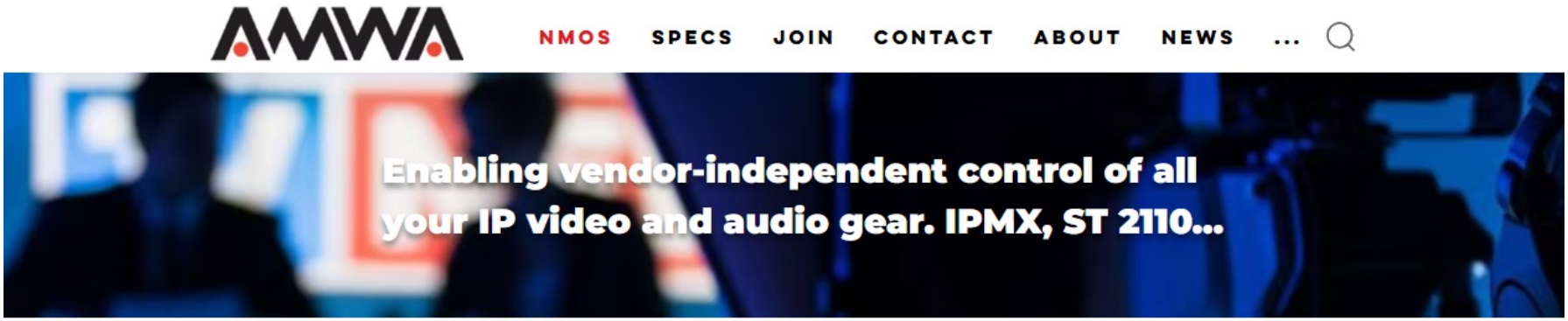
AMWA NMOS OSSRF

Open-Source Sender/Receiver Framework

Pedro Ferreira – BISECT, Founder & CEO

pedro@bisect.pt



networked media NMOS open specifications

A family of open-source, free-of-charge specifications that enable interoperability on the control layer for media devices on an IP infrastructure.

If you use ST 2110 or IPMX, it's likely you already rely on NMOS.

🎯 NMOS is the open-source "special sauce" that enables connection, management, and control of your IP video and audio devices from different manufacturers – in a common eco-system and in an interoperable way. NMOS works whether you're using ST 2110 or IPMX, locally or in the cloud.

🎯 NMOS (networked media open specifications) lets equipment from different companies all work together. Since AMWA's NMOS is an open set of specifications, it gives you a significant advantage by preventing vendor lock-in. It means you can choose the equipment you prefer, and

VID25 TRANS

OSS to Kickstart your NMOS Implementation

	Media Node Development	Testing
What		
Who		
Why		
Where		
How		

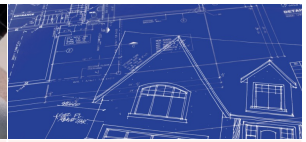
Special thanks to Gareth Sylvester-Bradley

OSS to Kickstart your NMOS Implementation

	Media Node Development	Testing
		
What		NMOS Testing Tool
Who		AMWA
Why		Conformance
Where		AMWA-TV/nmos-testing
How		Apache-2.0

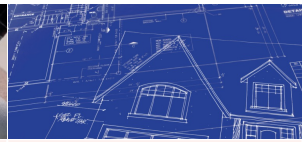
Special thanks to Gareth Sylvester-Bradley

OSS to Kickstart your NMOS Implementation

	Media Node Development	Testing	
			
What		NMOS Testing Tool	Easy NMOS (Registry, Controller, mock Node)
Who		AMWA	Joint
Why		Conformance	Interoperability
Where		AMWA-TV/nmos-testing	rhastie/easy-nmos
How		Apache-2.0	Apache-2.0

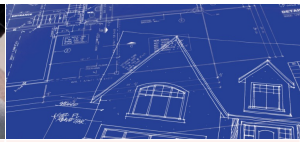
Special thanks to Gareth Sylvester-Bradley

OSS to Kickstart your NMOS Implementation

	Media Node Development	Testing	
			
What	nmos-cpp	NMOS Testing Tool	Easy NMOS (Registry, Controller, mock Node)
Who	Sony	AMWA	Joint
Why	All the low-level building blocks	Conformance	Interoperability
Where	sony/nmos-cpp	AMWA-TV/nmos-testing	rhastie/easy-nmos
How	Apache-2.0	Apache-2.0	Apache-2.0

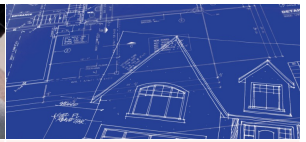
Special thanks to Gareth Sylvester-Bradley

OSS to Kickstart your NMOS Implementation

	Media Node Development		Testing	
				
What	nmos-cpp	NvNmos	NMOS Testing Tool	Easy NMOS (Registry, Controller, mock Node)
Who	Sony	NVIDIA	AMWA	Joint
Why	All the low-level building blocks	Common use cases, preassembled	Conformance	Interoperability
Where	sony/nmos-cpp	NVIDIA/nvnmos	AMWA-TV/nmos-testing	rhastie/easy-nmos
How	Apache-2.0	Apache-2.0	Apache-2.0	Apache-2.0

Special thanks to Gareth Sylvester-Bradley

OSS to Kickstart your NMOS Implementation

	Media Node Development			Testing	
					
What	nmos-cpp	NvNmos	Sender/Receiver Framework	NMOS Testing Tool	Easy NMOS (Registry, Controller, mock Node)
Who	Sony	NVIDIA	AMWA	AMWA	Joint
Why	All the low-level building blocks	Common use cases, preassembled	Higher-level building blocks	Conformance	Interoperability
Where	sony/nmos-cpp	NVIDIA/nvnmos	AMWA-TV/nmos-sender-receiver-framework	AMWA-TV/nmos-testing	rhastie/easy-nmos
How	Apache-2.0	Apache-2.0	Apache-2.0	Apache-2.0	Apache-2.0

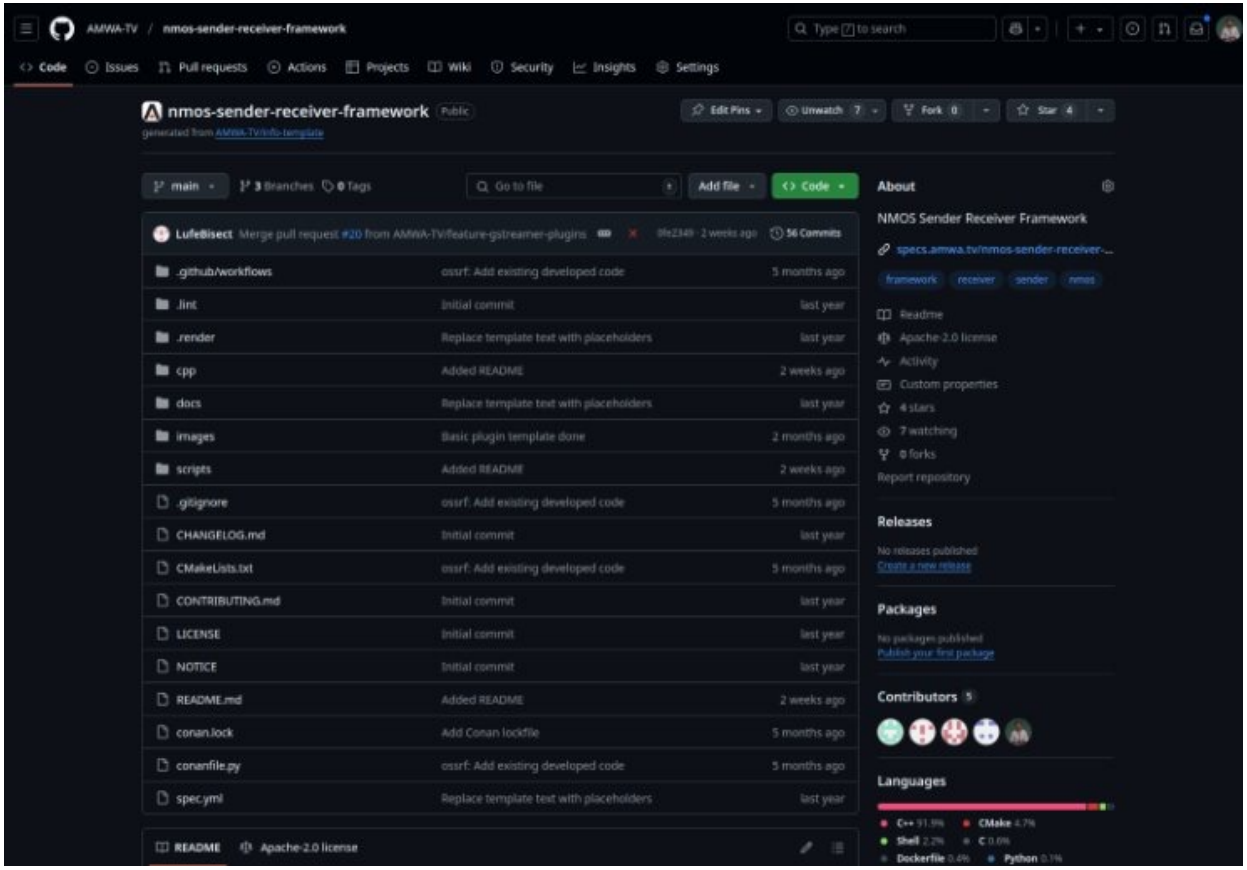
Special thanks to Gareth Sylvester-Bradley

NMOS Open-Source Sender/Receiver Framework

- **What**
  - A library, a set of GStreamer plugins and demo applications to simplify the implementation of NMOS media nodes
- **Where**

  http://github.com/amWA-TV/nmos-sender-receiver-framework/
- **When**
  - Now!

http://github.com/amWA-TV/nmos-sender-receiver-framework/



NMOS Open-Source Sender/Receiver Framework

- **Phase 1**
  - Library with a simple API to create senders and receivers
  - Simple low-res ST2110-20 and –30 sender and receiver
  - Demo applications
  - Launched summer 2025

GStreamer

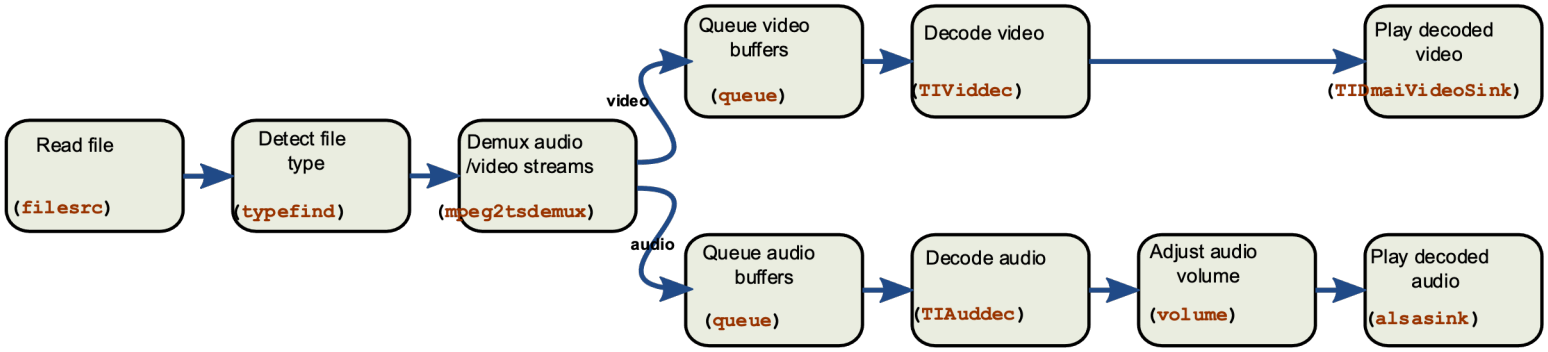
- https://gstreamer.freedesktop.org/

- Media processing pipelines
- Can be launched from the command-line
- Can be embedded in an application

Example GStreamer Pipeline

The goals of GStreamer are to seprate the application (e.g. Video player, Video editor, etc.) from the streaming media complexity (e.g. hardware acceleration, remoteness)
GStreamer – streaming media
D-Bus – inter process communication
Use **`gst-launch`** command to create the GStreamer pipeline

```
gst-rtsp-server - v4l2src ! video/x-raw,width=1280,height=720 ! omxh264enc ! video/x-h264,profile=baseline ! h264parse config-interval=1 ! rtph264pay name=pay0 pt=96
```




```
gst-launch filesrc location="video.ts" ! typefind ! mpeg2tsdemux name=demux \
       demux. ! 'video/x-h264' ! queue ! TIViddec ! TIDmaiVideoSink \
       demux. ! 'audio/mpeg'   ! queue ! TIAuddec ! volume volume=5 ! alsasink
```

by ScotXW - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=46518853

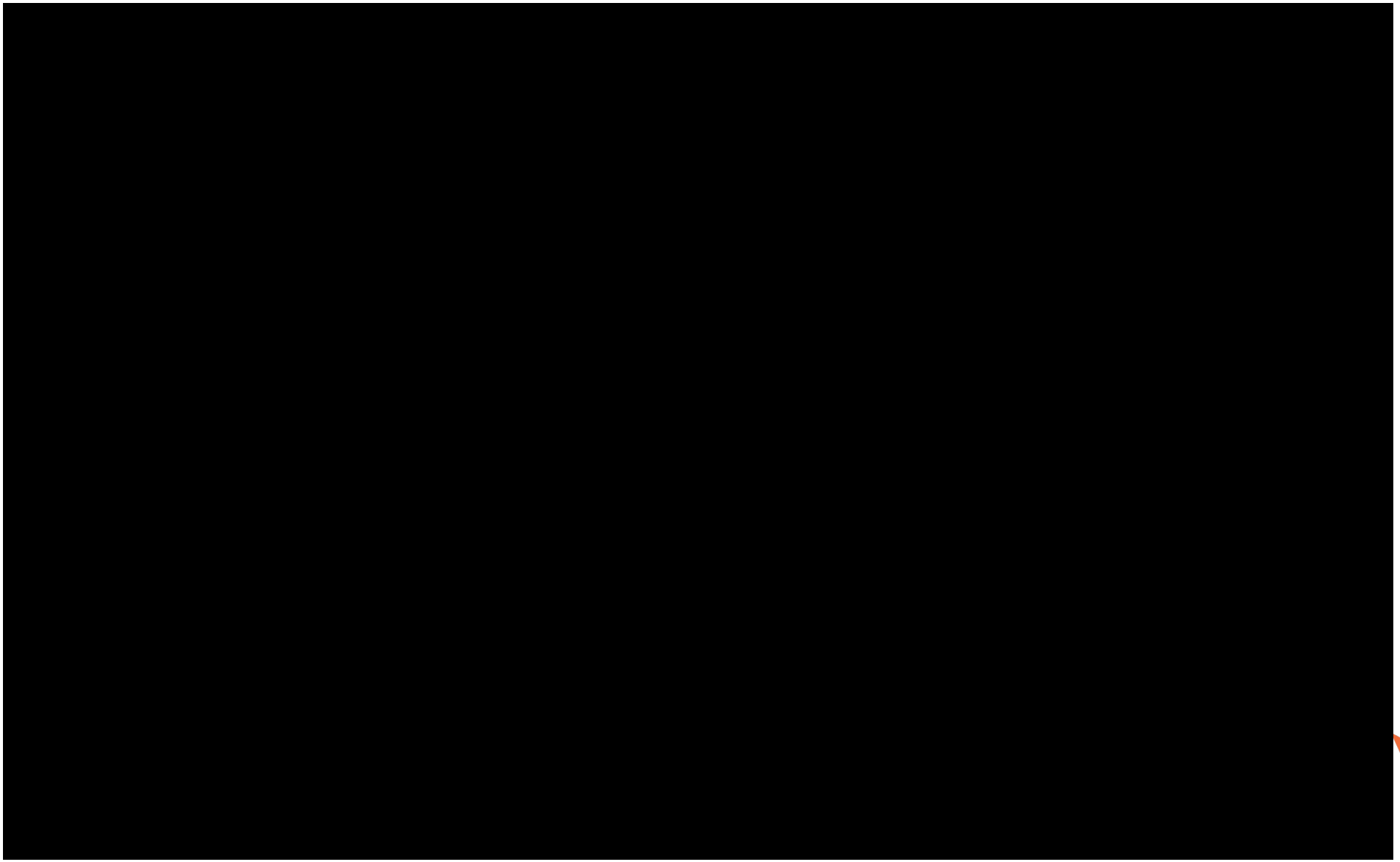
NMOS Open-Source Sender/Receiver Framework

- **Phase 2**
  - GStreamer plugins for sending and receiving audio and video
  - Just released

Demo#1

- **Use command line to launch 3 GStreamer pipelines:**
  - 2 video senders
  - 1 video receiver

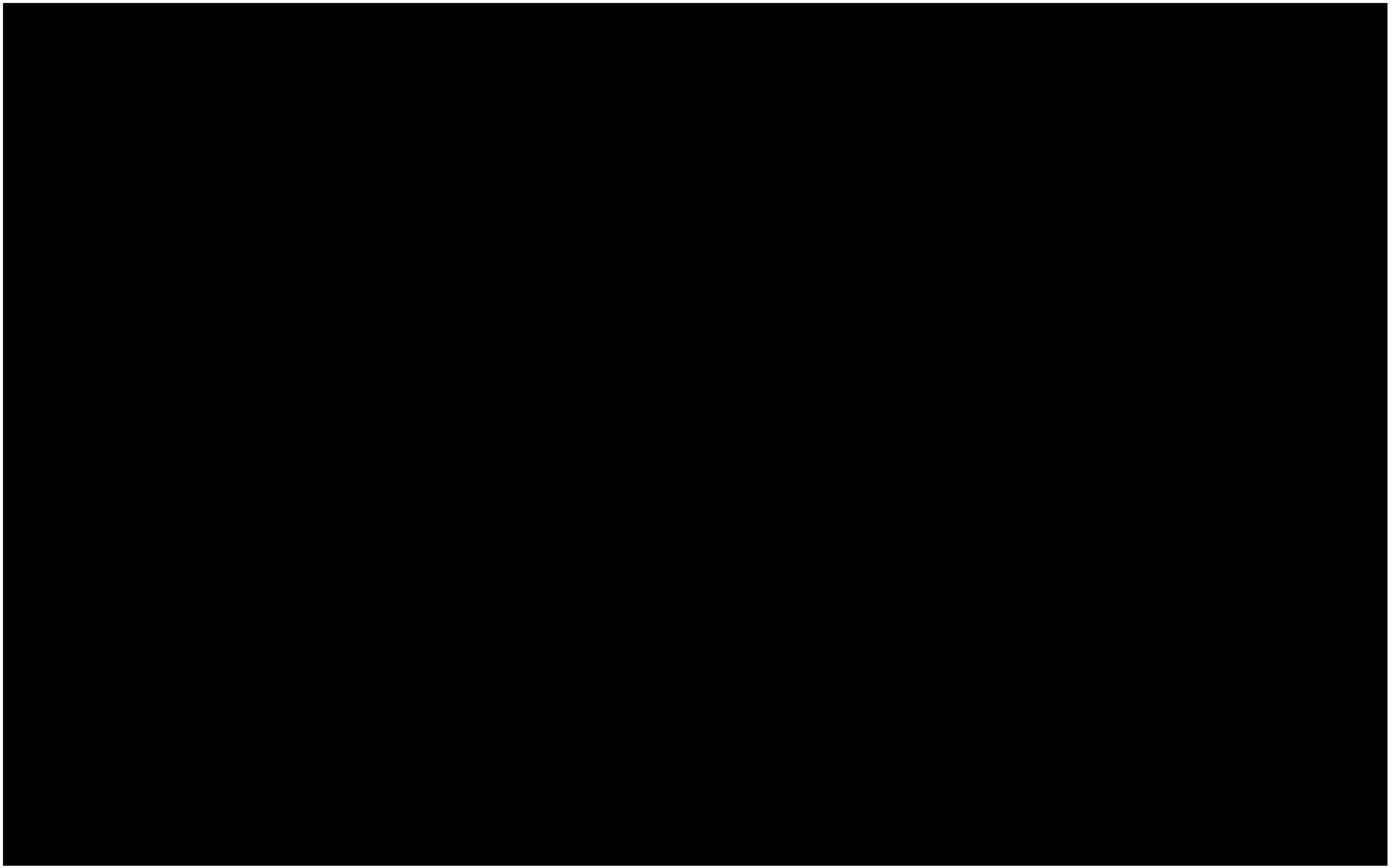
- **Control using Matrox ConductIP**
  - Using NMOS IS-04 and IS-05



Demo#2

- **Use command line to launch 4 GStreamer pipelines:**
  - 2 video senders
  - 2 video receivers

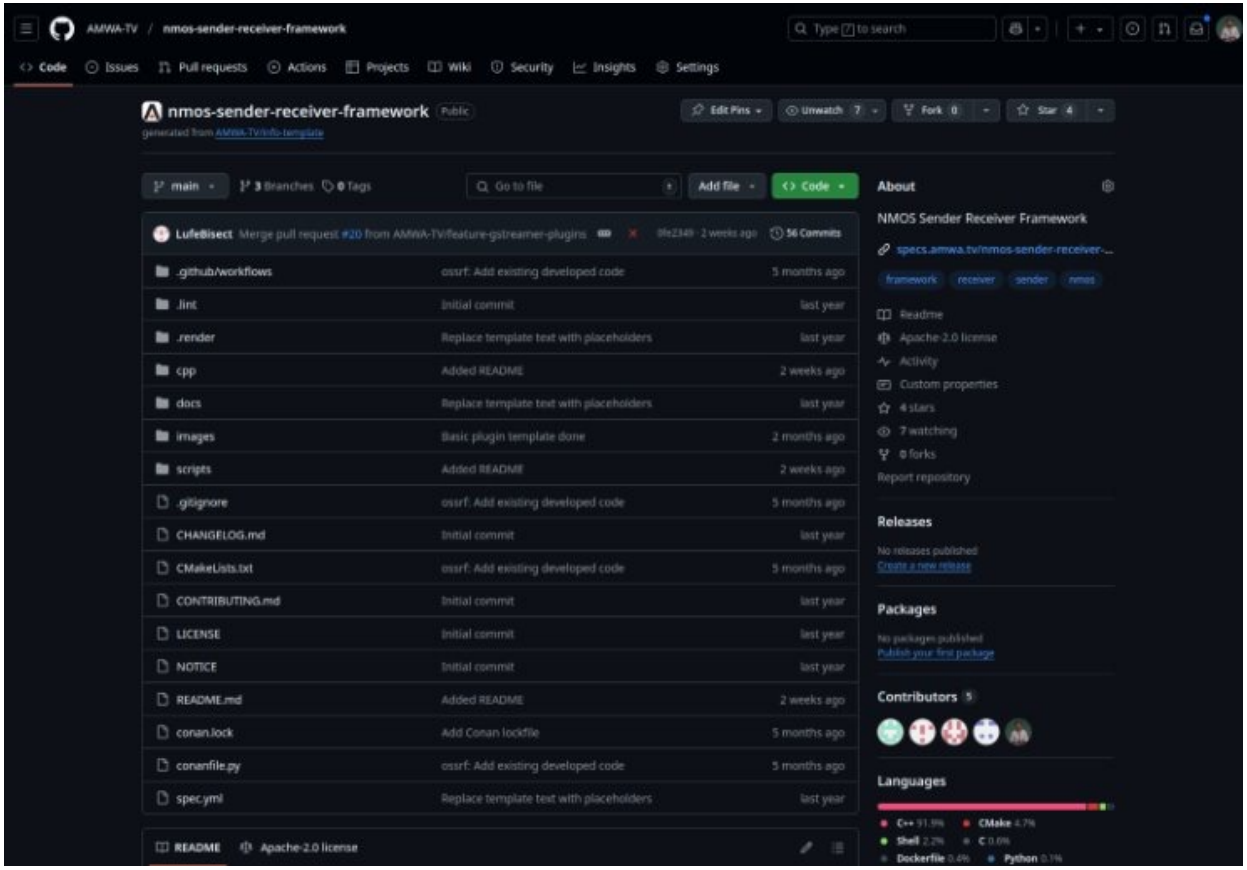
- **Control using nmos-js / Easy NMOS**
  - Using NMOS IS-04 and IS-05



Next Steps

- **Need feedback from the community**
- **Contribute!**

http://github.com/amWA-TV/nmos-sender-receiver-framework/



Thank you!

NMOS

AMWA NMOS OSSRF

Open-Source Sender/Receiver Framework

AMWA

VID25 TRANS