



HbbTV Test Applications Webinar

Mark Riley & Alex Russell

1st July 2025

Secondary Video Window OpApp

Mark Riley

1st July 2025

Introduction

This simple opapp is intended to help the manufacturer commence initial development.

- First created during spec development of feature – to bootstrap manufacturer implementation
- Prototyping / PoC
- Exercising New Features
- Dev and Debugging

Main Window (index.html)

```

ApplicationManager object. Error: Cannot read properties of
undefined (reading 'isObjectSupported')
000295ms:WARN: (hbbtvOpApp)Exception when getting the owner
Application object. Error: Cannot read properties of null
(reading 'getOwnerApplication')
000295ms:WARN: (hbbtvOpApp)Exception when creating creating
oipfConfiguration Object. Error: oipfObjectFactory is not
defined
000296ms:WARN: Exception when creating creating
oipfCapabilities Object. Error: oipfObjectFactory is not
defined
000301ms:WARN: DRM: DRM: application/oipfDrmAgent is NOT
supported
000302ms:WARN: DRM: setActiveDRM: not supported
000310ms:Elapsed: OpApp loaded (v4.33)
000607ms:RX: (Video) Child msg:
{"jsonrpc":"2.0","id":"rpcId1","method":"childInitialised"}
000608ms:TX: (Video)
{"jsonrpc":"2.0","id":"rpcId1","method":"SetupVideo","params":
{"hbbtvCfg":"null","jsonRPCServerURL":""}}
000842ms:RX: (Video) Child msg: {"id":"3db4577a-a030-4662-b557-
ad42955bb8b5:1","result":null}}
001152ms:WARN: (hbbtvOpApp)No Broadcast Channel List present!
002371ms:TX: (Video)
{"jsonrpc":"2.0","id":"rpcId2","method":"Play","params":
{"src":"https://refplayer-
dev.cloud.digitaluk.co.uk/dynamic/ngp-ssai.mpd"}}

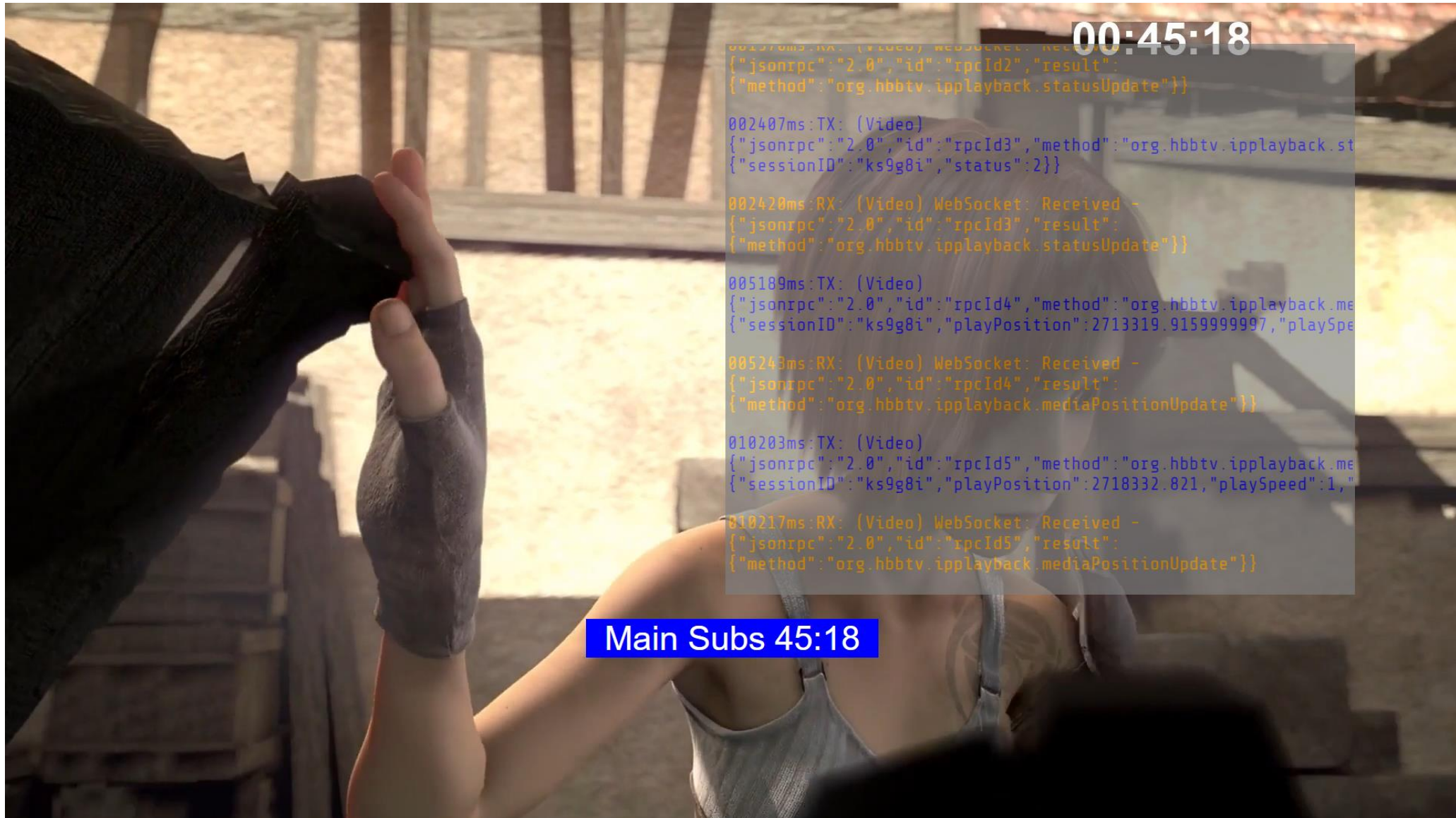
```



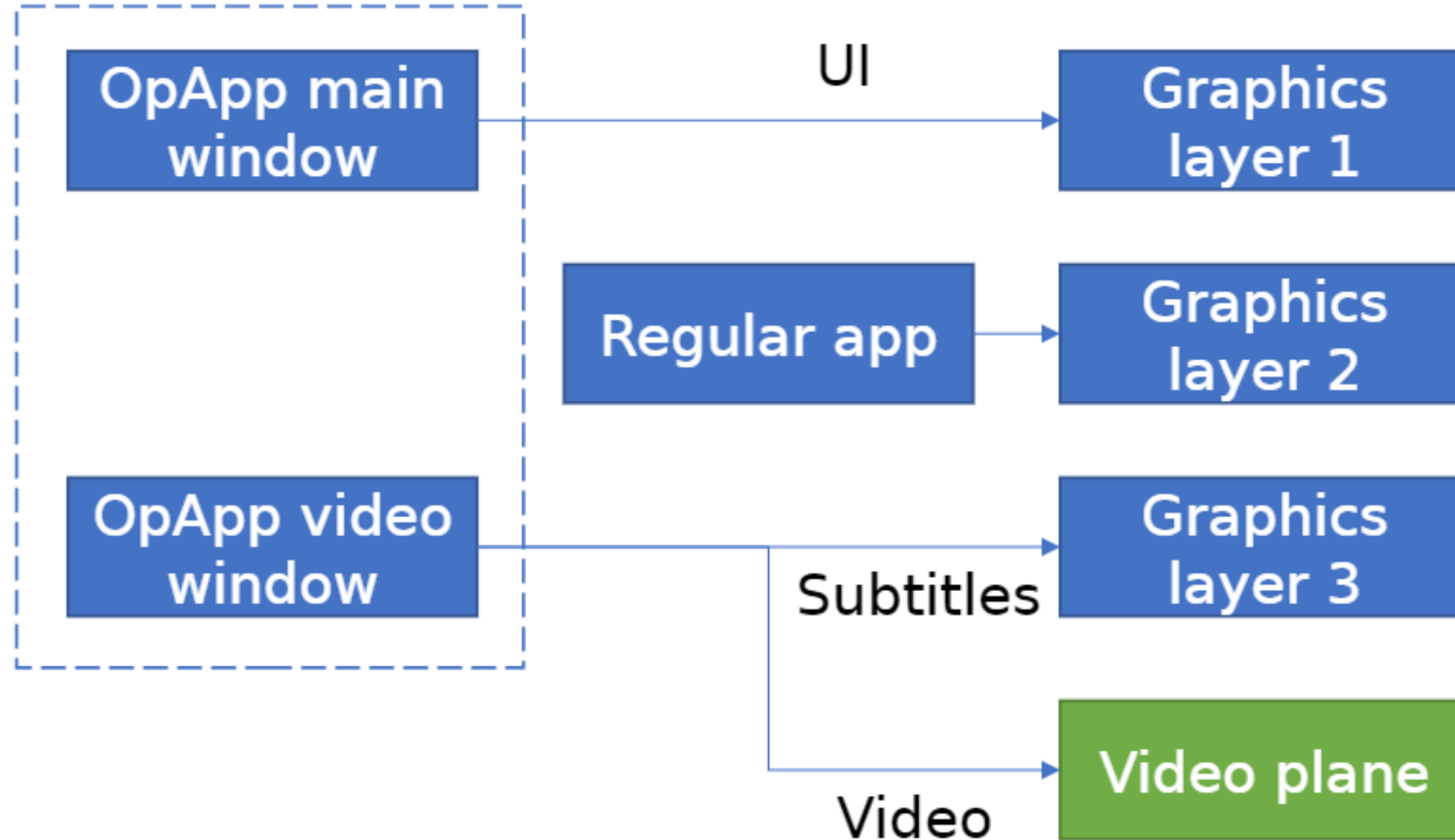
DUK SSAI

47:09 / 1:00:00

Video Window (video.html)



Plane / Layer Ordering



Composited Layers

00:48:33

```

ApplicationManager object. Error: Cannot read properties of
undefined (reading 'isObjectSupported')
000304ms:WARN: (hbttvOpApp)Exception when getting the owner
Application object. Error: Cannot read properties of null
(reading 'getOwnerApplication')
000305ms:WARN: (hbttvOpApp)Exception when creating creating
oipfConfiguration Object. Error: oipfObjectFactory is not
defined
000306ms:WARN: Exception when creating creating
oipfCapabilities Object. Error: oipfObjectFactory is not
defined
000311ms:WARN: DRM: application/oipfDrmAgent is NOT
supported
000313ms:WARN: DRM: setActiveDRM: not supported
000314ms:Elapsed: OpApp loaded (v4.33)
000490ms:RX: (Video) Child msg:
{"jsonrpc":"2.0","id":"rpcId1","method":"childInitialised"}
000491ms:TX: (Video)
{"jsonrpc":"2.0","id":"rpcId1","method":"SetupVideo","params":
{"hbttvCfg":"null","jsonRPCServerURL":""}}
000697ms:RX: (Video) Child msg: {"id":"65a37845-89c2-4ae0-b9c8-
3b8965486015:1","result":null}
000764ms:WARN: (hbttvOpApp)No Broadcast Channel List present!
001855ms:TX: (Video)
{"jsonrpc":"2.0","id":"rpcId2","method":"Play","params":
{"src":"https://refplayer-
dev.cloud.digitaluk.co.uk/dynamic/ngp-ssai.mpd"}}

```

```

020115ms:RX: (Video) WebSocket: Received -
{"jsonrpc":"2.0","id":"rpcId7","result":
{"method":"org.hbttv.ipplayback.mediaPositionUpdate"}}
025115ms:TX: (Video)
{"jsonrpc":"2.0","id":"rpcId8","method":"org.hbttv.ipplayback.me
{"sessionID":"ks9g8i","playPosition":2901252.132,"playSpeed":1,"
025179ms:RX: (Video) WebSocket: Received -
{"jsonrpc":"2.0","id":"rpcId8","result":
{"method":"org.hbttv.ipplayback.mediaPositionUpdate"}}
030103ms:TX: (Video)
{"jsonrpc":"2.0","id":"rpcId9","method":"org.hbttv.ipplayback.me
{"sessionID":"ks9g8i","playPosition":2906240.241,"playSpeed":1,"
030140ms:RX: (Video) WebSocket: Received -
{"jsonrpc":"2.0","id":"rpcId9","result":
{"method":"org.hbttv.ipplayback.mediaPositionUpdate"}}
035107ms:TX: (Video)
{"jsonrpc":"2.0","id":"rpcId10","method":"org.hbttv.ipplayback.m
{"sessionID":"ks9g8i","playPosition":2911244.196,"playSpeed":1,"
035164ms:RX: (Video) WebSocket: Received -
{"jsonrpc":"2.0","id":"rpcId10","result":
{"method":"org.hbttv.ipplayback.mediaPositionUpdate"}}

```

Main Subs 48:33



DUK SSAI

48:50 / 1:00:00

Playback

- Playback via video window
- Plays DASH content
- DRM
 - SL2000 / SL3000
 - Temporary / Persistent licences
- MSE / EME API
- Uses dash.js v4.7.4 (and v5.x currently being tested)



JSON RPC

OpApp to Terminal calls:

- `org.hbbtv.negotiateMethods`
- `org.hbbtv.ipplayback.statusUpdate` (where: 2 presenting, 3 stopped)
- `org.hbbtv.ipplayback.mediaPositionUpdate`
- `org.hbbtv.ipplayback.setComponents`

Terminal to OpApp calls:

- `org.hbbtv.ipplayer.selectChannel`
- `org.hbbtv.ipplayer.play`
- `org.hbbtv.ipplayer.stop`
- `org.hbbtv.ipplayer.pause`
- `org.hbbtv.ipplayer.resume`
- `org.hbbtv.ipplayer.setVideoWindow`
- `org.hbbtv.ipplayer.seek`
- `org.hbbtv.ipplayer.selectComponents`

Socket

The WebSocket properties are exposed via the **application/oipfCapabilities** xml doc.

The app checks for the following element:

```
1 <json_rpc_server url="ws-url" version="spec-version" opapp_version="spec-version"/>
```

```
1 videoWindow = window.open(window.globalURL + "video.html", "_opappvideo");
```

 Note the second parameter `_opappvideo` (this is how the terminal knows that this is the OpApp Video Window)

The App creates a WebSocket, using the standard JavaScript API, eg:

```
1 __websocket = new WebSocket(cfg.jsonRPCServerURL);
```

Negotiation

Example of list
of methods
required by the
OpApp video
window

```
"method": "org.hbbtv.negotiateMethods",
"params": {
  "appToTerminal": [
    "org.hbbtv.ipplayback.statusUpdate",
    "org.hbbtv.ipplayback.mediaPositionUpdate",
    "org.hbbtv.ipplayback.setComponents"
  ],
  "terminalToApp": [
    "org.hbbtv.ipplayer.selectChannel",
    "org.hbbtv.ipplayer.play",
    "org.hbbtv.ipplayer.stop",
    "org.hbbtv.ipplayer.pause",
    "org.hbbtv.ipplayer.resume",
    "org.hbbtv.ipplayer.seek",
    "org.hbbtv.ipplayer.setRelativeVolume",
    "org.hbbtv.ipplayer.setVideoWindow",
    "org.hbbtv.ipplayer.selectComponents"
  ]
}
```

Select Channel

Some example messages...

```
"method": "org.hbbtv.ipplayer.selectChannel",  
"params": {  
  "channelType": 0,  
  "idType": 41,  
  "ipBroadcastID": "dvb://233a..d04f"  
},
```

Terminal >>> Video
Window

Video Window >>>
Terminal

```
"method": "org.hbbtv.ipplayback.statusUpdate",  
"params": {  
  "sessionID": 1513807178,  
  "status": 2  
},
```

Scaled
Secondary
Video
Window

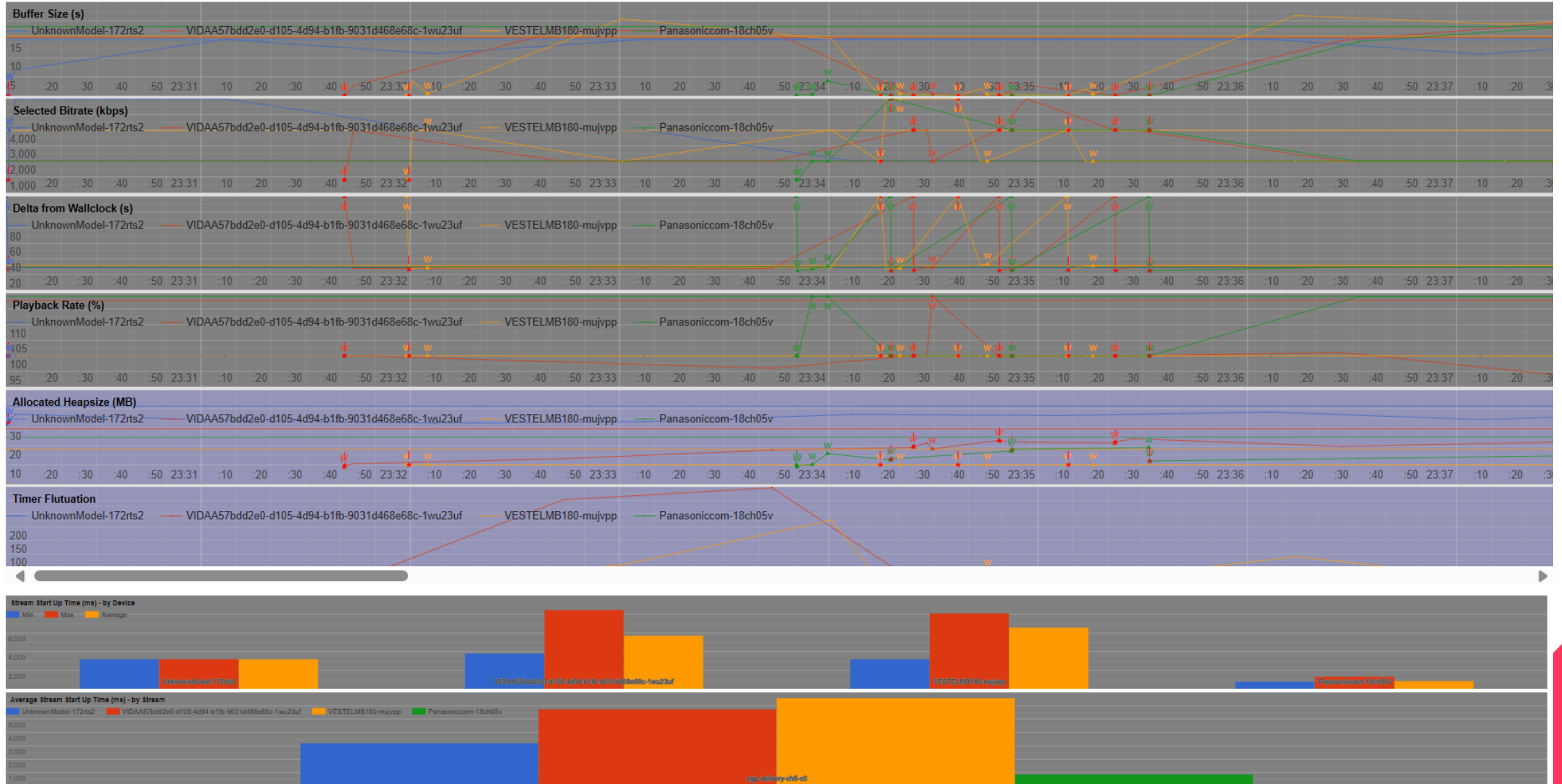
```
TEST: Windowed/Fullscreen

00000000: INFO - width: 40
00000001: INFO - vid: 55112
00000002: INFO - wid: 3818
00000003: INFO - hgt: 4244
00000004: INFO - vid: 62224
00000005: INFO - name: 887, Dev: Eazzy Testing
00000006: INFO - no [w/Channel] 443
00000007: INFO - termWin[Channel] 446
00000008: INFO - ipBroadcastIR: Web // 233a [118
00000009: INFO - *** Run Test: Windowed/Fullscreen ***
00010010: INFO - --- 1 of 18 ---
00010011: INFO - Step: 0
00010012: INFO - IsIn: boScaleWinDef()
00010013: INFO - Create

00010014: onPlayStateChange state: 1:connecting
00010015: onPlayStateChange state: 1:presenting
00010016: onPlayPositionChanged position: 3230246
00010017: onPlaySpeedChanged speed: 1
00010018: onPlayPositionChanged position: 3235248
00010019: onFullscreenChange
```

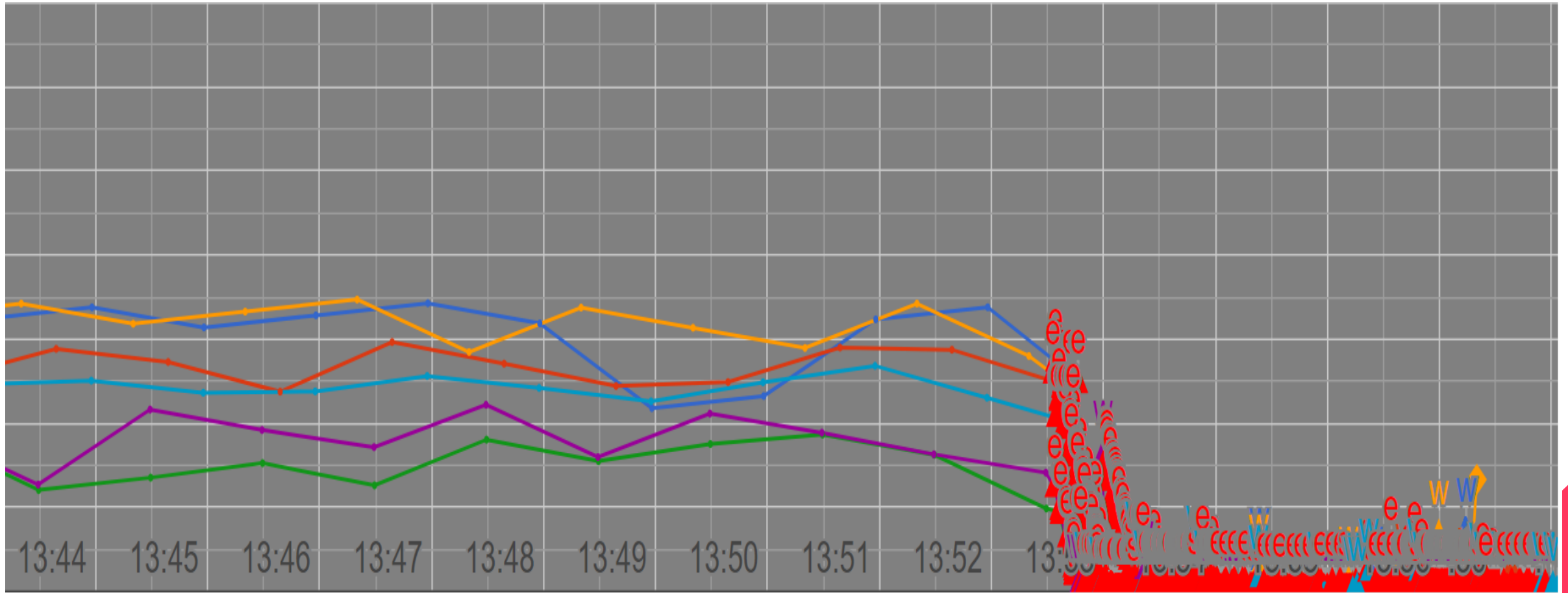
OpApp Main Window in background state

Playback Metrics



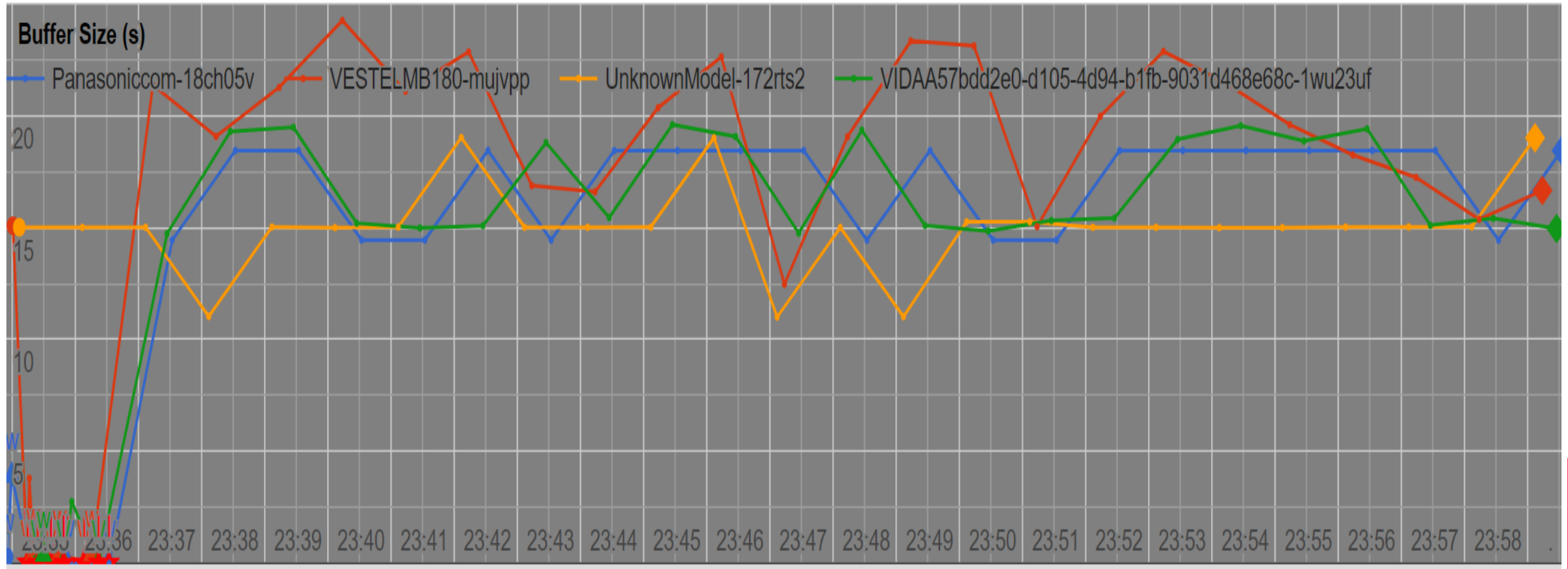
Capturing Fundamental Stream Issues

Eg: Incorrect CORs headers for Ads



Capturing Buffering Levels

Buffering levels over time – for a contended network (5 devices, capped at 16mbps)



Test and Conformance OpApp

Alex Russell

1st July 2025

API testing with an OpApp

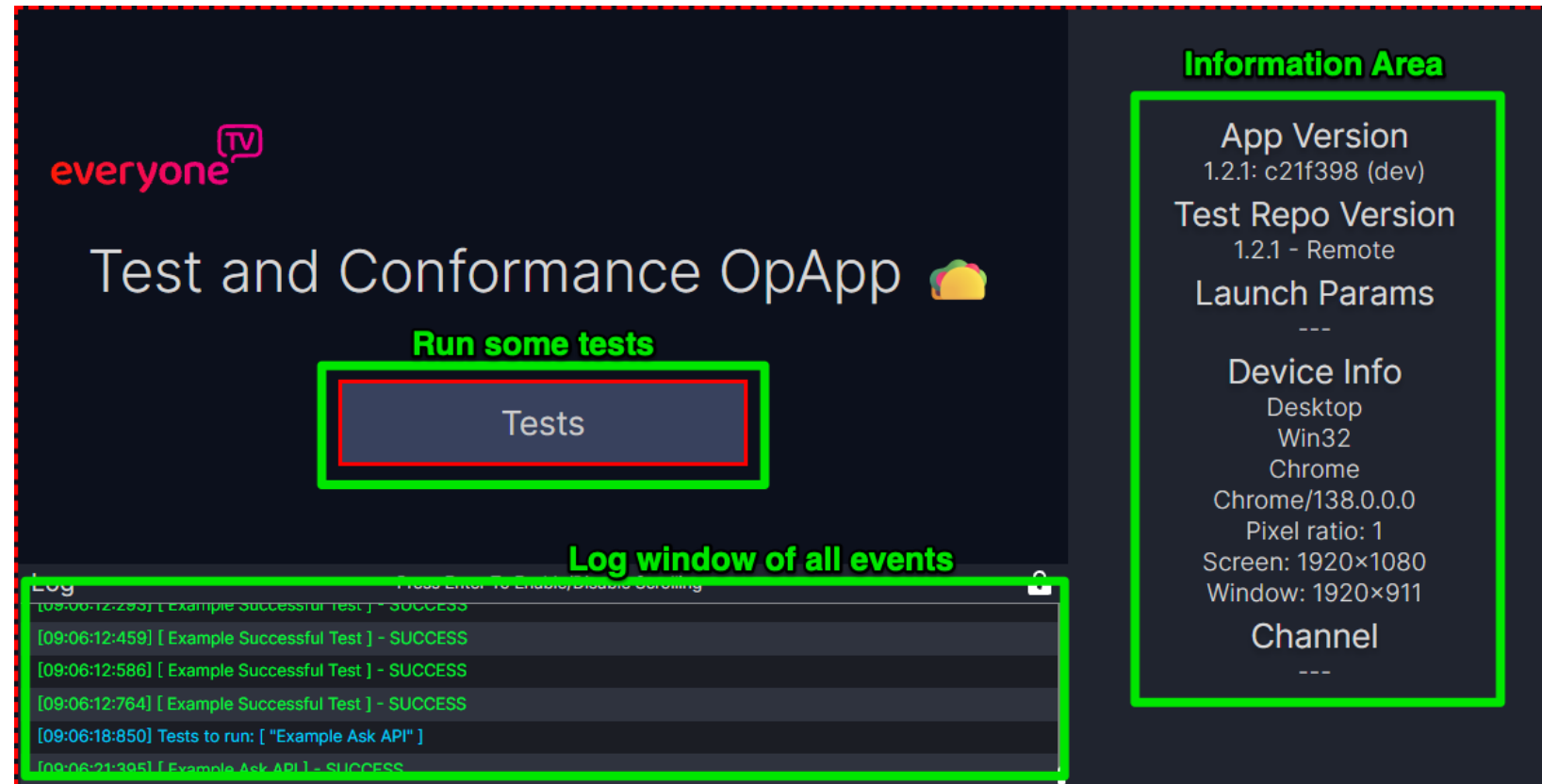
ETV also developed the Test and Conformance OpApp - TaCO

- Focusses on validating the API integration between an OpApp and the terminal
- Does not attempt to demonstrate video playback (as far as possible) as SVW OpApp provides this testing
- TaCO was created in two parts:
 - A reference OpApp with a set of features implemented
 - A set of tests downloaded at run-time using these features

TaCO launch screen

TaCO is designed to:

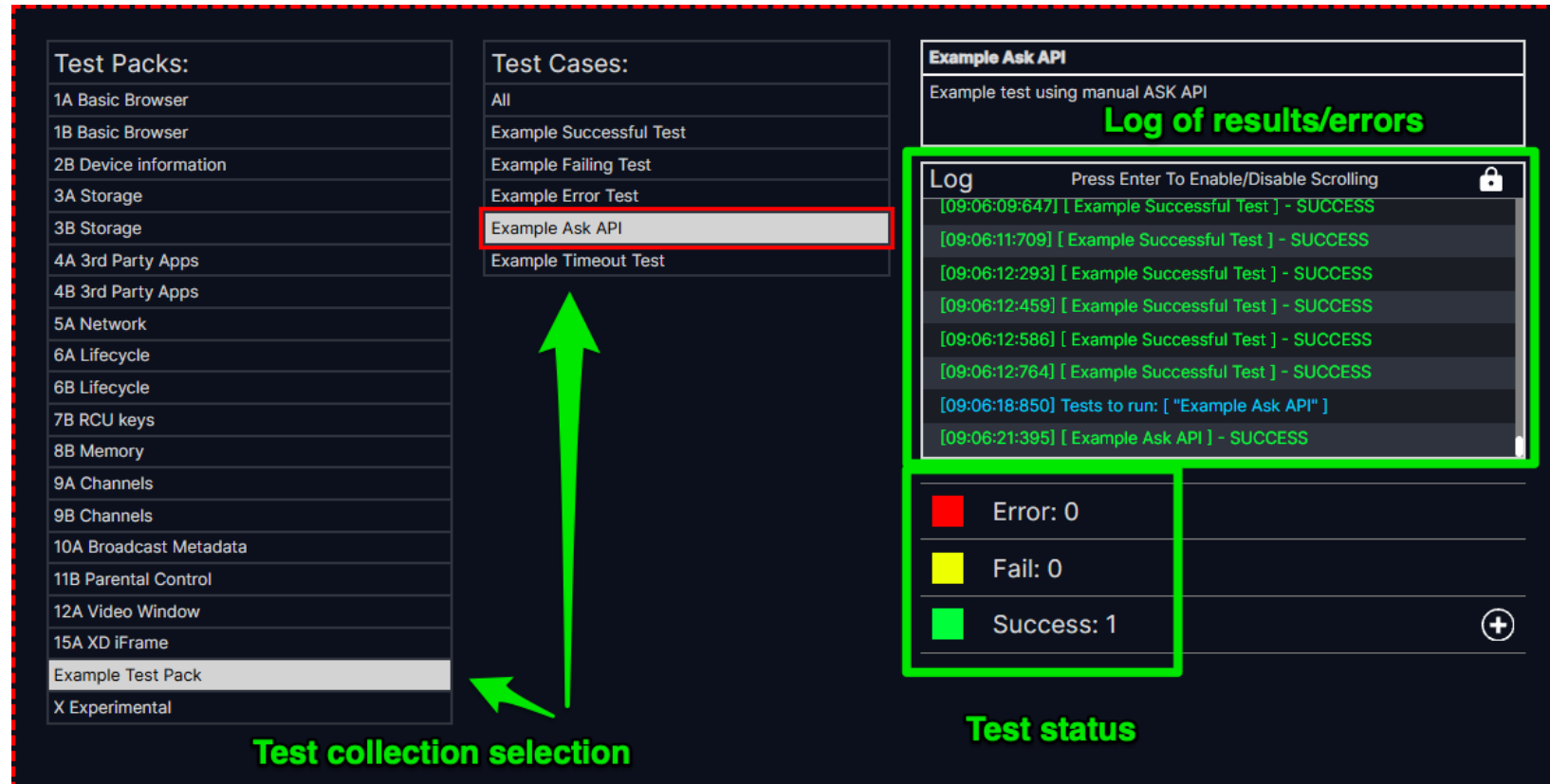
- Provide as much on screen and console feedback to the developer/tester as possible
- Simply launching TaCO on a real Freely device provides useful info on the right hand side
- Allows SLOC/LLOC combinations to be validated, check UA etc.



TaCO test selection

A test pack contains:

- Groups of related tests which can be run individually or as a set
- Results and errors are shown on screen for immediate feedback
- Further automation hooks are planned but not yet implemented
- ETV would like to consult with partners on how best to allow remote automation of TaCO



The screenshot displays the TaCO test selection interface with three main sections:

- Test Packs:** A list of test packs including 1A Basic Browser, 1B Basic Browser, 2B Device information, 3A Storage, 3B Storage, 4A 3rd Party Apps, 4B 3rd Party Apps, 5A Network, 6A Lifecycle, 6B Lifecycle, 7B RCU keys, 8B Memory, 9A Channels, 9B Channels, 10A Broadcast Metadata, 11B Parental Control, 12A Video Window, 15A XD iFrame, Example Test Pack, and X Experimental. The 'Example Test Pack' is highlighted.
- Test Cases:** A list of test cases including All, Example Successful Test, Example Failing Test, Example Error Test, Example Ask API, and Example Timeout Test. The 'Example Ask API' is highlighted.
- Log of results/errors:** A log showing the results of the 'Example Ask API' test. The log entries are:
 - [09:06:09:647] [Example Successful Test] - SUCCESS
 - [09:06:11:709] [Example Successful Test] - SUCCESS
 - [09:06:12:293] [Example Successful Test] - SUCCESS
 - [09:06:12:459] [Example Successful Test] - SUCCESS
 - [09:06:12:586] [Example Successful Test] - SUCCESS
 - [09:06:12:764] [Example Successful Test] - SUCCESS
 - [09:06:18:850] Tests to run: ["Example Ask API"]
 - [09:06:21:395] [Example Ask API] - SUCCESS

Below the log, the **Test status** is displayed:

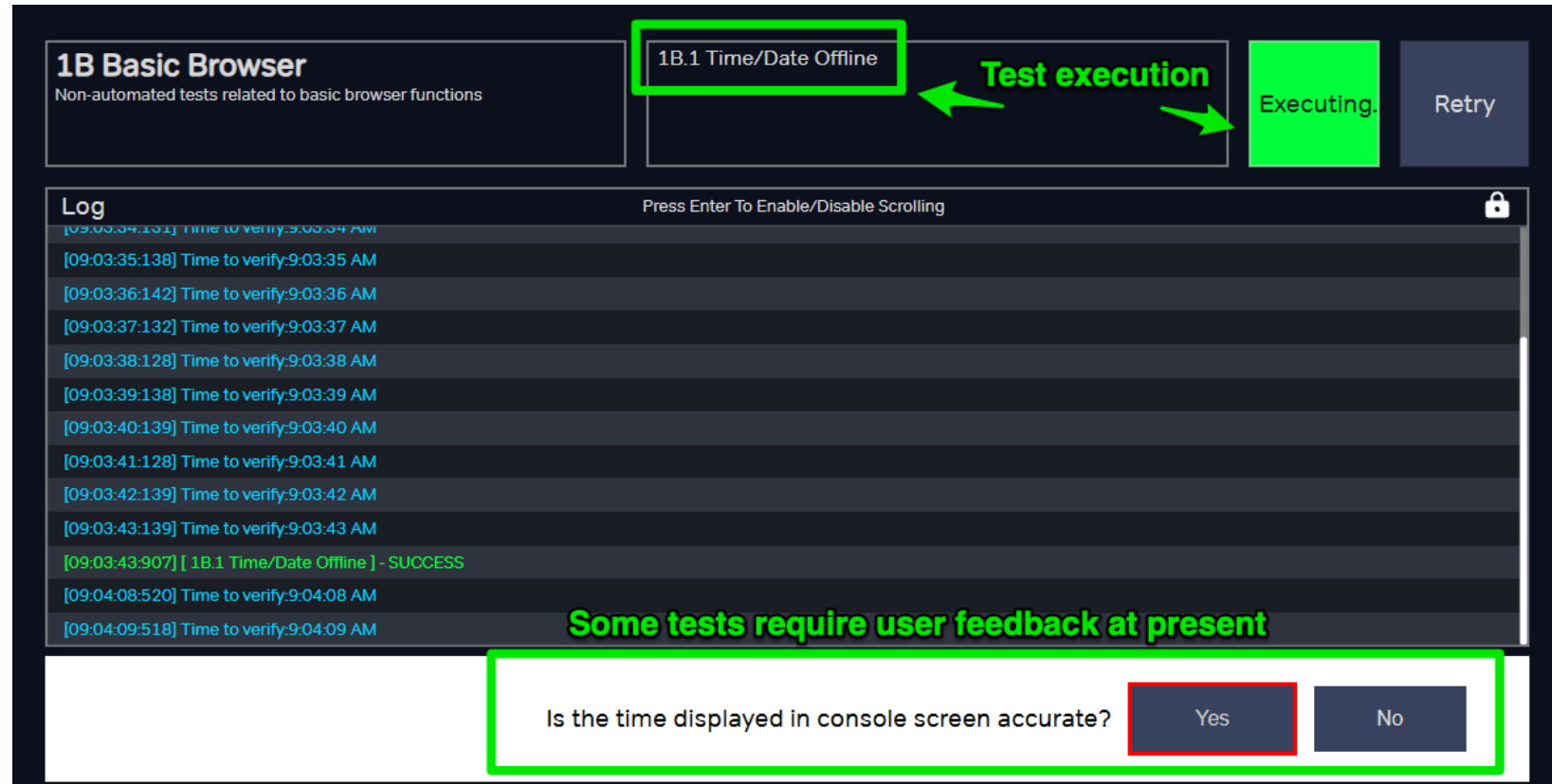
- Error: 0
- Fail: 0
- Success: 1

Green arrows indicate the flow from the 'Example Test Pack' in the Test Packs list to the 'Example Ask API' in the Test Cases list, and then to the 'Log of results/errors' section.

TaCO test execution

During test execution:

- TaCO checks for, queries and validates the response for APIs in a specified sequence allowing for threaded execution and server response timings.
- ETV identifies inconsistencies, problems and lack of support for required OpApp APIs
- Some APIs are stubbed out only, some return invalid responses under certain conditions
- Manufacturers are required to pass TaCO tests as part of the conformance regime for Freely.



The screenshot displays the TaCO test execution interface. At the top, there is a section titled "1B Basic Browser" with the subtitle "Non-automated tests related to basic browser functions". To the right of this section, there is a status indicator "1B.1 Time/Date Offline" which is highlighted with a green box. A green arrow labeled "Test execution" points from this status indicator to a green button labeled "Executing.". To the right of the "Executing." button is a grey button labeled "Retry". Below the status indicator, there is a log window titled "Log" with a subtitle "Press Enter To Enable/Disable Scrolling". The log contains a list of test results, including timestamps and status messages. One of the log entries is "[09:03:43:907] [1B.1 Time/Date Offline] - SUCCESS", which is highlighted with a green box. Below the log window, there is a feedback prompt: "Is the time displayed in console screen accurate?". This prompt is also highlighted with a green box. To the right of the prompt are two buttons: "Yes" (highlighted with a red box) and "No".

TaCA – Study Mission Group

To further consider the role test and reference applications play, HbbTV has convened the Study Mission Group (SMG) for Test and Conformance Apps (TaCA).

- Review and summarise the current set of test apps
- How can test/ref. apps be incorporated into HbbTV
- Consider business aspects of development, licensing and validation

First meeting 30th July 11:30am CET
(see HbbTV e-mails for details)

A large, thick, pink speech bubble with a rounded rectangular shape and a small tail pointing towards the bottom left. Inside the bubble, the text 'Thank-you' is written in a bold, pink, sans-serif font.

Thank-you