HTML5 apps on AGL platform with the Web Application Manager

Julie Kim
jkim@

Automotive Linux Summit
Japan
July in 2019
Agenda

● Introduction
  ○ About Igalia
  ○ Motivations
  ○ Chromium
  ○ Web Application Manager (WAM)

● What we’ve done
  ○ Integration with AGL
  ○ Try WebApps

● Plans
Introduction
About Igalia

- Open Source Consultancy
- Based in Galicia, Spain. Global team and customers
- ~81 employees around the world
- Areas
  - Chromium/Blink, WebKit and Servo; W3C member
  - Compilers, JavaScript engines (V8, JSC)
  - Multimedia, Kernel, Networking, Graphics
  - Virtualization & Cloud
A GLOBAL COMPANY
Motivations

- Web apps should be first class citizens in AGL; bring flexibility and power for a big number of developers.
- Chromium has been verified on several embedded devices; many manufactures use it as an WebEngine.
- Chromium with wayland is working fine on upstream; Igalia has worked on it.
- LGE open-sourced Web Application Manager which they’re using on their product.
Motivations; Risks

● How to manage SMACK with Chromium Multi-process model?
  ○ Using proxy process

● How to integrate on AGL?
  ○ Porting it with libhomescreen and libwindowmanager

● How to support Wayland with ivi extension protocol?
  ○ Migration wayland port with chromium upstream
  ○ Adding ivi extension protocol
Chromium Multiprocess

- **Browser process**
  - handles all interactions with the disk, network, user input, and display.
  - manages windows and tabs.

- **Renderer process**
  - contains all the complex logic for handling HTML, JavaScript, CSS, images, and so on.

- **GPU process**
  - Manages GPU operations.

https://www.chromium.org/developers/design-documents/multi-process-architecture
Chromium with Wayland

- AURA: UI Framework
- Ozone: Platform Abstraction Layer
- Wayland port implemented under Ozone
  - Using XDG shell on desktop environment
  - Implemented event handling
  - Handled multi tabbed/windowed browsing
  - Supported separate GPU process
Web Application Manager (WAM)

- Launches Web page as an independent application
  - Allows Web page to run outside of WebBrowser.
- Manages Web App’s Life-cycle
  - Resume/Suspend
  - Memory Management
  - Policy Management
- No Browser menu
  - Full screen view
  - No URL bar, No setting menu
What we’ve done
Systems level and Yocto recipes

- Reorganized and cleaned up Web App Manager Yocto recipes and layers.
  - Meta-html5-framework: [https://gerrit.automotivelinux.org/gerrit/#/c/19601/](https://gerrit.automotivelinux.org/gerrit/#/c/19601/)

- Update to the available stable version of AGL, which was Flucky Flounder for the goal of CES.
  - Available on Grumpy Guppy.

- Integrated all the changes for the different builds on the reference hardware platforms.
  - Renesas m3 board, Minnowboard, and RPi 3
Chromium with Wayland port

- Previous Wayland port vs Upstream Wayland port
Chromium Ozone/Wayland Core Design
Chromium

- Backporting Wayland port
- Supported IVI extension protocol
- Added APIs for supporting WAM.
  - SetSurfaceID to set ivi surface ID for each app.
  - SetProxy to manage network requests.
QT-less Web App Manager

- [meta] Qt-less WAM
  - https://jira.automotivelinux.org/browse/SPEC-1871
    - Replaced qt basic data structures
    - Moved from QMake to CMake
    - Removed QJson in favor of a Jsoncpp-based implementation
    - Dropped QObject + Q_SLOT + Q_SIGNALS
Integration with AGL Framework

- Integration WAM with homescreen and windowmanager.

- Registers event callbacks for HomeScreen, WindowManager and notification for ILMControl.
- Activates WebApp window, when it gets Event_TapShortcut.
- Manages WebApp states for Event_Active/Event_Inactive.

![Diagram showing integration with AGL Framework with icons and processes]

[Chromium]
Integration with AGL Security model

- Set proxy module to WAM to manage SMACK label

1. Sets Proxy with port
2. Sends port info
3. Requests data
4. Requests data to network
5. Gets response from network
6. Gets responses
7. Delivers data
8. Parses DOM, Sets Layout
Try WebApps

● Build the image with html5 framework.
  ○ Feature: agl-html5-framework
  ○ Target: agl-demo-platform

● Package WebApp as following https://www.w3.org/TR/widgets/
  ○ Refer to prebuilt WebApps from https://github.com/jaragunde/agl-html5-launcher or https://github.com/jaragunde/agl-html5-hvac

● Install and Start WebApps
  ○ afm-util install xxx.wgt
  ○ afm-util start <app name>
  ○ Refer to
Web Apps and Chromium Browser
Debug WebApps

Create preferences folder and the flag file.

1. `mkdir -p /var/agl_devel/preferences`
2. `touch devmode_enabled`
3. Open `<board_ip>:9998` in chrome browser.
AGL F2F meeting at Igalia HQ in May
Discussion related to WAM

- WebApp Packaging
  - Performance issue due to the proxy process path.
  - Complicated architecture with proxy process.
- Converting current QT based Apps to Web based Apps.
  - Discussion about using system APIs.
Current status

- Reorganized and cleaned up Web App Manager Yocto recipes and layers.
  - AGL has Web Application Manager at [meta-html5-framework](https://meta-html5-framework).
    - It doesn’t bundle any widgets. Try it with installing your own widget.
    - Or, you could also try [meta-agl-lge](https://meta-agl-lge) on Funky Flounder branch. (Deprecated)
- Web Application Manager works with Chromium68 on FF and GG.
- Chromium68 works with upstream wayland port.
  - It worked with the wayland port released by Intel and maintained by LGE.
  - Upstream has well-designed wayland port now.
- Worked on Renesas m3 board, Minnowboard, and RPi 3.
- Presented WAM Demo at CES 2019.
Current status

- Completed QT-less WAM.
- Under discussion
  - Security model without proxy
  - Converting QML based apps to HTML based apps.
  - Web App Packaging
First day in #CES2019 @igalia at the AGL booth.

Automotive Grade Linux (AGL)
- A collaborative open source project that is bringing together automakers, suppliers and technology companies
- A Linux-based, open software platform
- Hosted at Linux Foundation
- Focused on rapid innovation of vehicle software
- https://www.automotivelinux.org/

Visit @igalia at the @autogradelinux #embeddedworld2019 booth in Hall 4-171 to see our demos of the Chromium based Web Application Manager #ew19

Motivations
- Web apps should be first class citizens in AGL; living flexibility and power by a big number of developers.
- Chromium has been verified on several embedded devices; many manufactures love it as a Webengine.
- Chromium with wayland is working fine on upstream; igalia has worked on it.
- LGE open-sourced Web Application Manager which they're using on their product.
Plans

- Identified several aspects that required more work:
  - Continuous stabilization and maturity. Although the prototype was functional, it was needed to keep integrating it on the newer AGL versions, and fix and improve some components.
  - Keep updating fresher WAM WebOS, and reducing delta with upstream.
  - Margin for improvement of the performance of the demos.
  - Review memory management and evaluate providing a proper app lifecycle.
  - Targeting to full HTML5 demo.
Plans

● Chromium
  ○ Rebase after latest commits available at upstream WebOS Chromium WebView (from @15.agl.flounder to @39.agl.flounder)
  ○ Upstream the AGL changes into LG webosose
  ○ Update to more recent Chromium version expecting a 7x update–Additional bug-fixing and performance improvements.
  ○ Better IPC for communication between WAM launcher and browser process.
  ○ Make WAM work without in-process-gpu
Plans

● Web Application Manager:
  ○ Rebase the latest commits available at upstream WAM (from @2.chromium68.5.agl.flounder to @6.agl.flounder)
  ○ Integration of WAM into new HMI architecture
  ○ Launcher process permission control (SMACK permissions holder)
    ■ Defining a new protocol communication between launcher and browser process and lightweight http proxy implementation
    ■ Review token logic for HTML5 apps and remote clients/apps
  ○ Improve integration with the new audio framework
  ○ Application life cycle on WAM with the AGL life cycle and memory management.
Questions

Lorenzo Tilve (ltilve@)

Jacobo Aragunde (jaragunde@)

Antia Puentes (apuentess@)

Julie Kim (jkim@)
Reference

- https://www.chromium.org/developers/design-documents/multi-process-architecture
- https://docs.google.com/presentation/d/13D5M9ZDGM-i33GDjMFLFrPHWRvYJcZvai3zl4RQ0tDM/edit?usp=sharing
- https://github.com/webosose/meta-agl-lge/tree/flounder.chromium68