Browser Technology Evaluation

Rationale

The existing proof-of-concept browser is based on QtWebkit. This technology has been deprecated and needs a replacement.

Market trends seem to lean towards Chromium-based solutions, so several of them were analyzed in the GENIVI AMM meeting that happened in Stuttgart in April 2015.

A report was written as an input for this meeting:

This wiki page takes the report as a starting point and expands the information found there for a more complete evaluation.

**WARNING:** This is a work-in-progress effort! The analysis will change as more information is gathered.

Available solutions

**Chromium**

Chromium is the open-source basis of Google's Chrome products: browser and OS.

Chromium is a vertical solution comprised by many layers: user interface, multiprocessing, web content sandboxing, window management and graphics rendering among others. It is designed to use a minimum number of external dependencies, so it can run on a light stack on its OS mode and be more easily portable as a browser.

(Extended information in the report above)

**Crosswalk**

Crosswalk is an HTML application runtime based on the Chromium code. It is available for Android as an embeddable webview container and for Tizen as the system-wide application runtime.

Crosswalk aims to provide a web runtime to develop applications based on web technologies instead of native toolkits, replacing the browser concepts with application development ones. Crosswalk provides interfaces for communication with the hardware and other applications. Chromium's multiprocess architecture is wrapped by Crosswalk's own, focused on applications and services.

(Extended information in the report above)

**Chromium Embedded Framework (CEF)**

The Chromium Embedded Framework (from now on, CEF) is a simple framework for embedding Chromium-based browsers in other applications.

CEF is intended to provide a stable API for development of applications with embedded browsers. All browser abstractions are preserved, and the multiprocess architecture of Chromium is preserved and properly interfaced.

(Extended information in the report above)

**QtWebEngine**
QtWebEngine is an implementation of Chromium into the Qt platform. It technically replaces the QtWebkit element, provides a stable Qt API to the Chromium web engine.

QtWebEngine plugs to Qt's own graphic implementation, so it should run on any platform where Qt runs. This means that QtWebEngine does not rely on the Ozone-Wayland layer for Wayland support.

It is released under the terms of the LGPLv3 license. This license imposes some restrictions to embedders, who must provide means for users to replace the default QtWebEngine with custom implementations. Commercial licenses can be bought from Digia to overcome this restriction.

QtWebEngine is supposedly not intended to be used as a webapp runtime or a fullscreen browser, although is technically possible. Direct use of Chromium is recommended for those cases.

This project is developed and maintained mainly by Digia, as a part of the Qt platform. We could suppose the support is strong, but there is always the risk of the component falling out of the priorities of the Qt project and being left unmaintained, like the QtWebkit component before.

Evaluation items

- HTML5 support
- Browser features: bookmarks, history, tabs and tab management, incognito mode, etc.
- GENIVI API: how easy would be to implement the GENIVI DBUS API on top of that solution.
- Portability: should work on different Linux-based stacks on top of ARM or Intel boards.
- Wayland support
- Maintainability: effort required to maintain the solution in sync with upstream development.
- Community: activity of the community, upstream development and maintenance.
- Licensing: degree of permissivity of the software license, from the point of view of the integrator.

Evaluation

Summary matrix

<table>
<thead>
<tr>
<th></th>
<th>Chromium</th>
<th>Crosswalk</th>
<th>CEF</th>
<th>QtWebEngine</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Browser features</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GENIVI API</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Portability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wayland support</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maintainability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Community</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Automotive API</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Licensing</td>
<td>BSD+third party</td>
<td>BSD</td>
<td>BSD</td>
<td>LGPLv3</td>
</tr>
</tbody>
</table>

**HTML5**

Any of these options are based on modern versions of Chromium so they have a high support for HTML5 features.

Score: *** for all

**Browser features**

Chromium is the most complete solution with regard to browser features (bookmarks, history, tabs and tab management, incognito mode, etc.); of course, that was expected because Chromium is exactly a browser. The other solutions hide or remove the browser features to build different things: a runtime (Crosswalk), a library (CEF) or a widget (QtWebEngine).

Score: *** for Chromium, * for other solutions.

**Automotive API**

The only of the solutions already providing support for Automotive APIs is Crosswalk (source).

Work to implement these on the other options should be done from scratch or reusing the support found in Crosswalk. We don't know yet what would be the effort to reuse Crosswalk implementation in other solutions.
Score: *** for Crosswalk, * for other solutions.