Mobile Web Browsers in Android Deriving Reference Architecture

Ashraf Abdulmunim Abdulmajeed
College of Computers Sciences and Mathematics
University of Mosul

ABSTRACT
One of the most interesting changing in the today's web users is relying heavily on the mobile web browsers to do all their daily life activity. There are many numbers of web browsers who have their mobile versions released and which work as effectively and efficiently as the stationary web browsers although they do have certain limitations. The reference architecture provides a template solution for the architecture of a particular domain. In this paper, we present new proposed reference architecture for web browser in the mobile domain based on the analysis of three selective mobile web browsers. The work includes (1) the extraction of the conceptual and concrete architectures for three selected well-known mobile web browsers namely: Firefox, Google Chrome and Dolphin, (2) the analysis and discussion of the different extracted architectures and their components as well as their various dependencies, (3) the deriving of the proposed architecture based on the resulted analysis and the evaluation of it using existing stationary web browser reference architecture.

The study is restricted only to open-source and closed-source browsers with limited access to the code. It would be interesting to check derived reference architecture for mobile web browsers against commercial mobile web browsers and refine it accordingly. Another interesting point is to derive reference architecture for other mobile operating systems and compare the derived reference architectures with the one in Android OS.

Keywords

Web browser, Google Chrome, reference Architecture, mobile users, reusability

1. INTRODUCTION
Today, web browser plays an essential rule in the daily life of every web user. Big software companies such as google, Mozilla, and apple are providing their own web browsers' applications that based on the latest architecture governance. This allows their web users to have the fastest and more enjoyable browsing experience. They are mainly relying on utilizing the right combination of architecture, interface, and networking principles to provide a successful product and achieve the best results.

In the recent years, the mobile web browsing has more features for all smart cell phones. With the current stage of development, mobile users are allowed to easily browse the World Wide Web with fully supported features and high compatibility options making successful experiences for mobile users. The easy accessibility and comfortable use of mobiles as well as the provided services that matched the one in normal web browser allows more dynamic shift in the web users toward the use of mobile browser. Reference architecture for mobile web browsers would provide valuable information on the structure of these applications for the purposes of evolution and maintenance. It is essential feature that allows any software developer to understand a software system even if that particular system did not have reports and documentations related to the specification architecture. It represents a model template for the construction of new systems as it could be used to identify sections and areas where reusability could be achieved at different stages and levels of abstractions. Also, it could serve as a guidance to support the process of understanding the legacy code and provides insight into evolutionary trends in mobile web browser domain [1]. To the best of our knowledge, mobile web browsers (including popular ones such as Chrome from Google and Firefox from Mozilla) are still lacking reference architectures for their mobile application.

In this paper, an extraction of the conceptual architectures is carried out using various mobile web browsers for an android mobile (two well-known open source and one closed source mobile browsers, Chrome, Firefox, and Dolphin respectively). Then, this is followed by an evaluation and comparison with the existing concrete architecture. Finally, proposed mobile web browser reference architecture was derived based on the analysis results and compared with the existing stationary web browsers.

The reminder of this paper is structured as follow: Section 2 describes the methodology for deriving the reference architecture. Section 3 explains the extraction of the architectures for the selected mobile web browsers and the derived reference architecture. Section 4 discusses and compares the resulted derived reference architecture of the mobile web browser and the existing reference architecture for stationary web browsers. Section 5 describes the background of the work and section 6 concludes the paper.

2. RELATED WORK
There has been a huge interest in determining the reference architecture of stationary standalone web browser and many research has been carried on to achieve this purpose.. Yet, there is a lack of the conducted works related to the reference architecture in the mobile platforms for the web browsers applications. When it comes to the differences of the study of Modern Web Browser and Architectural among various web browsers, extensive study and historical evaluation, source of occurrence, experiments and statistical analysis of results are carried on and then a practical conclusions to understand the Modern web browser are achieved.

One of the most noted works in the field of modern web browser is the work conducted by Grosskurth and Godfrey [1] where they present the concepts of the history and evaluation of the Modern Web Browser. The study is examined the architecture of the browsers and explored the areas of Reference Architecture through the use of Open Source methodologies. Also the authors tried to understand the

underlying subsystems and inter-relationships among them. The work includes the study of the architectures using architectural implementations of five different web browsers namely Epiphany, Safari, Lynx, Mosaic, and Firefox. This followed by the validation process which is based upon the browsers' specific subsystems' components and their popularity using several factors such as Open and Closed Methodologies, Primitive Browsers, UI Capabilities, etc.

Another study by Vrbanec, Kiric and Varga [3] is conducted to compliment the study of Grosskurth and Godfrey [1] and shared the most useful insights specifically the one related to the evolution of the web browser architecture. Focusing on the efficiency and the timing factor needed in the development of web browsers and to help developers reducing development time and ensure best serving to fit the needs of a web user, the work proposed the concepts of web browser architecture that would lead to reduce development costs, ensure best cross-browser compatibility and lead to rich user experience.

Marin Šilić [2] performed an extended research on the security vulnerabilities in the modern web architecture. The study covered a large number of security factors and external threats. Moreover, the analysis of a number of the popular web browsers such as Chrome architecture is provided in accurate details about the security loopholes and the browser's behavior to different web threats.

Roy T. Fielding [4] performed a similar research on the modern web architecture and introduced the concepts of Representational State Transfer (REST) and extended the concept of Web architecture by offering a new abstract model of the web architecture to lead the redesign and the definitions of the HTTP protocols and URL stages. It is suggested that the study is best conducted with its comparative study of the abstract model defined with the present web architecture to obtain discrepancies in the web protocols.

The study of Campos, *et al.* [7] shares valuable information about "Firefox" which is currently one of the most popular web browsers. The work includes analysis of Complete architecture, Browser Components and other essential factors to give a broad overlook to the browser in the modern era. Furthermore, the conceptual architecture is studied. The work concluded that (1) the browser's architecture resembles the Layered architecture and (2) is less affected by code changes and is therefore is easily maintainable and reusable.

There has been a lot of research conducted for deriving the reference architecture of the stationary/ standalone web browser. Yet, to the best of our knowledge, the problem still unexplored and no research has been conducted on the reference architecture for web browsers in the mobile editions.

3. DERIVING A REFERENCE ARCHITECTURE

Reference architecture is basically an abstract architecture of a system that brings incredible information for the purpose of application domain and maintenance of the software [3]. For studying a domain system, reference architecture plays a major role in defining the common subsystems in that domain and the relationships between them. It also acts as a baseline for the implementation of new systems for which architecture and proper documentation are not available or reengineering of existing systems and aids during the design of new systems and the maintenance of existing ones.

The conceptual architecture can be defined as a high level description of a system that concentrates on how developers understand and think about a software system or how they would like the architecture to be. This does not involve intricate details at either the Interface or Procedure level. On the other side, the concrete architecture can be defined as an actual structure of the implementation of a system. It gives a broader perspective on the relationships between the subsystems. It is worth mentioning that the main difference between the concrete and conceptual architectures is the number of extra dependency between subsystems in concrete architecture, this may exist due to implementation constraints that they don't affect the overall understanding of system's functionalities [2].

In this paper, three known web browsers namely Google Chrome, Mozilla Firefox and Dolphin have incorporated and analyzed to closely understanding their architectures in details leading to derive and propose new reference architecture. Out of the three browsers, Google Chrome and Mozilla Firefox have both mobile and standalone versions available while the Dolphin web browser was limited to mobile edition only. It is worth mentioning that Dolphin browser has been the selected to ensure the consistency in deriving the proposed architecture and to ease the validation process of the architecture against other browsers.

The presented paper closely follows the methodology of extraction proposed by Godfrey and Grosskurth's study on the web browser reference architectures [1].

To explore, understanding and deriving the reference architecture for mobile web browsers, few steps are applied for each of the selected browsers:

i. Obtaining the source code of each browser from its repositories.
ii. De compiling each browser application using existing analysis tools such as *Apktool*, *dex2jar*, and *JArchitect*.
iii. Gathering relevant documentations related to each one of the browsers.
iv. Analyzing and examining both the conceptual and concrete architectures of each of the browsers.
v. Deriving the reference architecture of each one of them.

It is worth mentioning that the selections of the used analysis tools are based on their functions to cover all the required aspects of the analysis which briefly includes the use of *Apktool* for decoding resources to nearly original form and rebuilding them with few modifications, *dex2jar* to convert the application APK binary code to JAR file format so that *JArchitect* could be used to visualize the source code dependencies graphs and its metrics to ease the understanding and analyzing steps.

The derivation process of reference architecture for mobile web browser is described here:

3.1 We choose three Mobile web browsers for the derivation process. Two of them namely Google's Chrome browser and Mozilla's Firefox browser are Open Source browsers and highly popular for Android application and available for both Mobile and Desktop, while the third browser namely Dolphin Browser developed by Mobotap, is only available for Mobile version.

3.2 For each of the web browsers, the following steps are carried out to derive the conceptual and concrete architecture:

3.2.1Conceptual architecture of each browser is derived using available documentations, source code, and gathered general domain knowledge.

3.2.2 Concrete architecture is derived using the conceptual architecture framework, subsystems that presented in the conceptual architecture where an instance for each subsystem is created to analyze the source code of each browser application.

3.2.3 Refinements and adjustments of the derived architectures are applied using an iterative review process and compared it against the architectures of other android web browsers.

3.3 The reference architecture for Android web browsers is then derived basing on the similarities and common subsystems in concrete and conceptual architectures for the selected browsers.

3.4 A comparison is carried on between the reference architecture with the architecture presented by Grosskurth and Godfrey [1] to ensure the consistency and validity of the results.

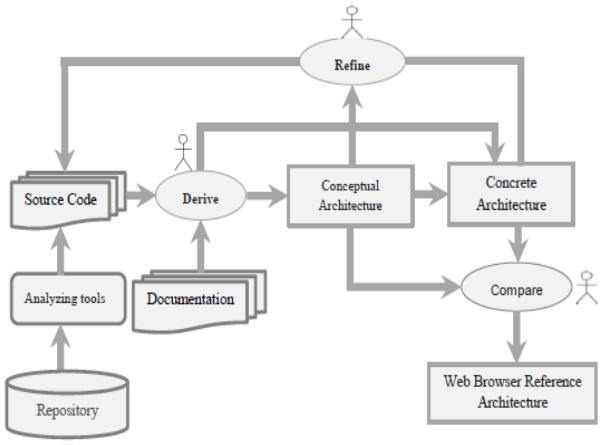

Fig 1: Extraction Methodology for Reference Architecture

The main extraction process of the reference architecture is shown in Figure 1. From this Figure, it could be noticed that the extraction process is mainly achieved manually and the automatic process is only at the initial stages of the process (code decomposition from android install files). Also, during the extraction and exploration process of the source code and by using some of the high level source code components a quick mapping could be achieved. This is done by aligning the extracted instanced subsystems based on the name of packages or the classes in the packages. Other components are hard to recognize by their names especially in dolphin browser as it is not open sourced application. Because of that, an extra step has been carried on by determining their functions and which subsystem they are belonging to. This can be help in identifying and allocating the dependencies between these components and comparing them between the concrete architectures with the ones that exist in the conceptual architectures for each one of the targeted browsers.

4. CHROME ARCHITECTURE

Chrome for Android is an open source web browser that is developed by Google. The application is launched for the first time in 2012 and went through various iterations. The mobile edition of the web browser application still lacks a number of features that existed in the standalone web edition such as supporting apps and/ or some extensions and plug-in. On the other hand, it has a number of prominent features such as page rendering, desktop browser synchronization open tab, hardware acceleration , and the ability to see what tabs you have opened on any other chrome-running device [12][10].

Both mobile and standalone editions of Google Chrome web browser applications are derived from chromium project. Chromium has a multi process architecture which is the key feature that recognizes it from other web browsers. The main idea behind this architecture is that each tab or window will be opened in a new process to protect the overall application from bugs and glitches in the rendering engine. An access restriction from Google developer is applied to each rendering engine process with the others and the rest of the system. This allows the web browser to have a number of benefits including memory protection, access control and browser security which one of the top priorities for chromium development. They already introduced sandbox mechanisms which reduce the rendering engine and limit the access to the operating system [11][13].

Chrome leverages the same multi process architecture in both android and standalone web browser application. One essential difference between the two editions was the memory constraints as Chrome may not be able to run a dedicated renderer for each open tab. In this case, Chrome determines the optimal number of renderer processes based on available memory, and other constraints of the device, and shares the renderer process between the multiple tabs [14].After analyzing the internal implementation of Google chrome, the identification of the key aspects of Google chrome's conceptual and concrete architecture are extracted shown in Figure 2. The extracted information shows that conceptual architecture for google chrome consists of eight subsystems: User Interface, Browser engine, Data persistence, Networking, JavaScript engine V8, Display backend, the rendering engine and Base/utilities.

Subsystem can displayed as following:

4.1 User Interface (UI) is a subsystem that has many abilities such as allowing the user to explicitly see and includes all the active elements like windows, buttons, menus, text fields, etc. Also the view which is a component that is responsible for organizing all elements in the user interface.

4.2 browser engines, it is the most central component of google chrome browser and for all other browsers which acts as a mediator. It has the responsibilities of delegating activities with other subsystems, windows management, handling the user input and communication with the network.

4.3 Data persistence represents the database for Google chrome and includes the details of the browsers' activities such as cookies, password database, browsing history and all other data that the browser intends to maintain.

4.4 Networking (Network stack) was a library that provides the browser with access to network resources and has the responsibility of handling Universe Resource Locator (URL) and fetching resources from the network.

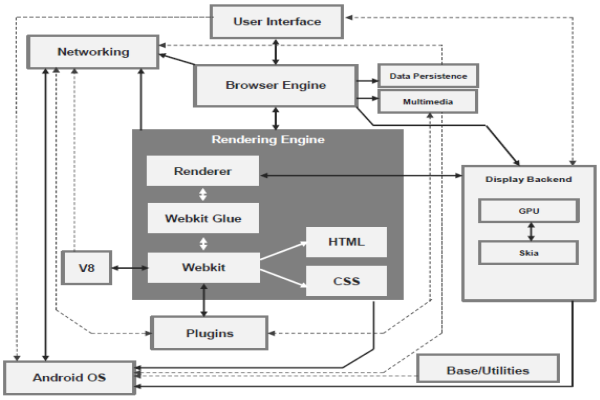

Fig 2: Information extraction for the architecture of Chrome browser.

4.5 JavaScript engine V8 which is one of the most efficient java script interpreters. It is developed by Google to just improve the browser performance.

4.6 Display backend was a system component that provides all the inessential drawing, windowing, and fonts in order to properly display the webpage on the User Interface. It has two main libraries: The first library was *Skia* which is a 2D graphics library for drawing text, geometric, and images, and the second library was the *Graphical Processing Unit* (GPU) which is a unit code that is used to accelerate the composition of 3D graphics.

4.7 The Rendering Engine which has three main components:

4.7.1 The *renderer* component, this is a chrome specific code that runs in the renderer process and talks directly to the browser engine.

4.7.2 The *webkit* which is an open source project developed by Apple to layout webpages in the web browsers and has three essential cores: the *JavaScript Core*,andthe *Webcore*. The *webkit JavaScript Core* is a framework that is used to provide the JavaScript codes and currently not being used anymore as they have JavaScript engine V8. On the other hand, the *webkit Webcore* is responsible of handling web components such as CSS, DOM, HTML, and XHTML.

4.7.2 The *Webkit glue* refers to the embedding API layer that converts between *Webcore* types and *Chromium* types.

4.8 Base/utilities subsystem was added to the conceptual architecture subsystems to represent common codes that are shared between all the subsystems such as string manipulation, generic utilities, etc.

The task of identifying the dependencies among these subsystems and comparing them with the existing one in the conceptual architecture become an applicable task by utilizing the existing analysis tools to view, explore and compare the source code and the dependences of the browser. It has been noticed that the concrete architecture has a few more dependencies compared to the conceptual architecture. Even though the new dependencies don't add logical explanation when the analysis is carried on using the conceptual architecture view, but it has more value and logic when it is being viewed in the concrete architecture. These missing dependencies are mainly due to misunderstanding and/ or overlooked functionalities such as the dependency between data persistence and user interface that is related to functionality such as URL autocompleting or login data. Another worth mentioning point of view that among the concrete architecture dependencies, few are reasonable and can be justified for example the dependency between plug-in component and network component that allows the reading of data and information to be carried on. Other dependencies are not logically connected and probably constructed in a rush to quickly fix bugs or sudden issues.

The conceptual architecture of Google Chrome tends to follow both a layered approach and an object oriented approach whereas the concrete one tends to follow only the object oriented approach. The reason behind that is the dependencies among the subcomponents themselves and the main components in the architecture which violate the rules of layered style architecture. The layered approach style of the conceptual architecture represents the browser as the communication or the center that allows and interacts with the other components of the system, while the concrete architecture represents the opposite approach by using other components within the system for interactions.

5. FIRFOX ARCHITECTURE (Fennec)

The architecture of the Firefox is based on layered architecture. The first layer is user interface that renders to the browser and allows a web user to interact [7][8]. The second layer is Gecko main subsystem which is a complex system that has other subsystems under its umbrella such as plugins, data persistence, browser engine, layout engine, document object model (DOM), XML user interface language (XUL), XML, HTML, and CSS. The advantage of layered architecture is the possibility of reusing subsystem such as Gecko for the mobile edition of the browser. The architecture of the Firefox for Android OS has been profoundly affected by stationary Firefox browser architecture; for example, version 1.0 used the same Gecko engine as Firefox 3.6. Also, the mobile edition of the browser followed the design patterns: Observer, Proxy, Abstract Factory, Singleton and Façade. From version 2.0 to version 4.0, the same rendering engine is used for both mobile and standalone editions of Firefox browser which creates a higher similarity and nearly matched results regardless of the edition of the browser. Figure 3 shows the subsystem of mobile edition of Firefox browser.

Subsystems of mobile Firefox:

i. *Analytics*: the main responsibility of this subsystem is to collect statistics and send feedback from web users to the Firefox such as the number of crashes, the reasons behind that and user's feedback.
ii. *Utility* - this is a subsystem that provides utility functions such as converting symbols from one Unicode to another one.
iii. *User Interface* - it is a subcomponent that offers methods to allow a web user to interact with the Browser Engine [5][6].
iv. *Gecko* - is the complex subsystem that is responsible of the rendering and displaying webpages to users through User Interface subsystem and it is following the object-oriented architecture [5][6].
v. *JavaScript Interpreter* - is a subsystem that is in charge of interpreting java script files.
vi. *Networking* subsystem - is a library that supports security-enable communication of the browser as well as manages protocol handlers to create URI object from URI string.

vii. *Android OS* - this is included as subsystem as it provides a large number of libraries to support the browser GUI.

Gecko subsystems:

i. *Browser engine*: this is responsible of coordinating other internal modules [5][6].*Data persistence*: this is responsible of saving.
ii. browser data through session storage or global storage [7]**.**
iii. *Layout engine*: this is in charge of translating the loaded DOM and Style information into display.
iv. *Plugins*: this is a subsystem is responsible to manage plugins, which extends the functionality of the web browser. Unlike Chrome, Firefox plugins have full access to all web browser resources [5].
v. *Document object model* (DOM): is an interface that provides an abstraction layer above underlying components representing web page content [7].
vi. *XML user interface* language (XUL): is the XML based user interface-rendering framework that allows rendering Rich Internet Applications (RIA).
vii. *XML* parser: this parses XML format files**.**
viii. *HTML* parser: this parses HTML documents received from the network.
ix. *CSS* parser: this parses CSS files and applies style to the parsed html.

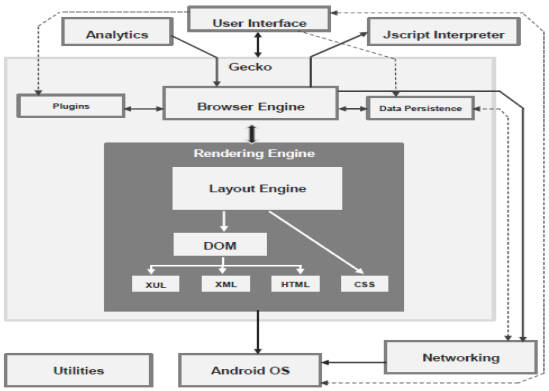

Fig 3: Information Architecture of the Firefox browser

6. DOLPHIN ARCHITECTURE

The Dolphin Browser is a mobile browser for Android Operating System developed by Mobotap. It is one of the first alternative browsers for Android platform[9]. It covers the basic features and has few additional features such as Tabbed browsing, Dolphin Sonar, Gesture Browsing to open bookmarks and navigate around web pages. This browser supports third party plug-ins and tools to extend its features. The architecture of Dolphin browser is similar to one in Firefox and Chrome. The subsystems in the dolphin browser are almost the same except that it has more subsystems such as Multimedia subsystem as well as Social subsystem to support Flash and other additional features. Figure 4 presents the subsystems of Dolphin browser for the mobile platform. The Tunny Browser in the Figure is a complex subsystem that includes Browser Engine, the Core, the Data Persistence, Downloads Subsystem, the Plugins and the Rendering Engine. The Plugins subsystem is an isolated subsystem as it could contains many features provided from third-party developers in the form of add-ons and plugins. It is worth mentioning that the Dolphin browser uses the same Rendering Engine as in Chrome: *webkit* Engine.

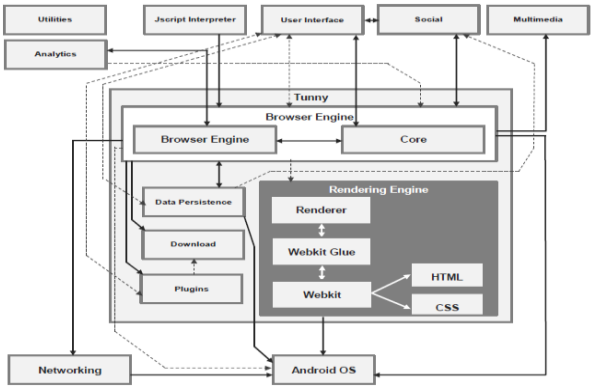

Fig 4: Information architecture of the Dolphin mobile browser

7. CONCLUSION AND FUTURE WORK

The current change in the web users' trends in surfing the internet has a great influence in our daily life. Web users are relying more on the mobiles and their web browsers to sort out almost everything. The current issue is that the browsers in the mobile platforms did not have reference architecture to be used as a guided reference. This provides a good opportunity to conduct a feasible study. The work is conducted by using two of the most popular open-source browsers and one private browser. They are deeply analyzed to develop accurate reference architecture for web browser in the domain of mobile platform. The examination and evaluation of the mobile web browser showed that the stationary/ standalone web browser domain has profoundly affected mobile web browser domain, for example, Firefox's *Gecko* subsystem core part has been used across the domains almost without change. However, other subsystems have been adapted to the mobile devices because of hardware restrictions like process frequencies, memory capacity, and screen size and so on.

The study is restricted only to open-source and closed-source browsers with limited access to the code. It would be interesting to check derived reference architecture for mobile web browsers against commercial mobile web browsers and refine it accordingly. Another interesting point is to derive reference architecture for other mobile operating systems and compare the derived reference architectures with the one in Android OS. Future work will focus on validate current reference architecture against other mobile web browsers, further refine reference architecture using additional mobile web browsers, and refine reference architecture using browsers from different mobile platforms.

8. REFERENCES

[1] Grosskurth A. and Godfrey M. W "A Reference Architecture for Web Browsers", Proceedings of the 21st IEEE International Conference, Software Maintenance,. ICSM'05, pp. 661 664, 2005.

[2] Silic, M., Krolo, J.and Delac, G.,"Security vulnerabilities in modern web browser architecture",33rd International Convention pp 1240-1245, 2010.

[3] Vrbanec T., , Kirić .and Varga M., "The evolution of web browser architecture". pp 472-480, 2013.

[4] Fielding, R. and Taylor, R., "Principled design of the modern web architecture". ACM Transactions on Internet Technology (TOIT), pp 115-150, 2002.

[5] Dr. Ahmed and E. Hassan, "Concrete Architecture of Mozilla Firefox (version 2.0.0.3)" July 4, 2007.

[6] Brereton J. , Krull G., Staalduinen R. and Tanner K. "Conceptual Architecture of Mozilla Firefox 6", Prepared For CISC 322, Queen's University at Kingston, Ontario, Canada, 2011.

[7] Lane B. and et al.,"Conceptual Architecture of Firefox", June 2, 2007.

[8] Grosskurth A. and Godfrey M. W., "Architecture and evolution of the modern web browser" University of Waterloo, Waterloo, ON N2L 3G1, Elsevier Science, Canada, 20 June 2006.

[9] Paul R., "Hands on: Dolphin HD browser for Android is swimmingly good". Ars technica, 2011.

[10] https://sites.google.com/site/cisc322project/conceptual architecture.

[11] Adam B. et al., "The security architecture of the Chromium browser".pp11-18, 2010.

[12] http://en.wikipedia.org/wiki/Google_Chrome_for_Android.

[13] http://www.chromium.org/developers.

[14] http://aosabook.org/en/posa/high-performance-networking-in-chrome.html.