AUGMENTED REALITY BROWSERS: A PROPOSAL FOR ARCHITECTURAL STANDARDIZATION

Fernando Sambinelli[1] and Cecilia Sosa Arias[2]

[1]Federal Institute of Education, Science and Technology of Sao Paulo, Brazil
[2]Methodist University of Piracicaba, Brazil

ABSTRACT

The technology evolution of smartphones, systems and the growing of telecommunications, allow the use of Augmented Reality in the exploration of geo-referenced information, complementing the real environment of the users with various types of content displayed on robust mobile cameras through applications called augmented reality browsers. Although this type of application is being in development and growing use by society, the technology, and especially its software architecture do not have any kind of standardization. This work presents concepts about augmented reality browsers for mobile devices. It shows the main aspects and applications of those types of reality. Besides that, the specific features of those types of architecture are discussed and compared, and new architecture, whose most relevant feature is the interoperability of applications in various platforms for mobile devices, are presented. This work has the objective to develop an architectural framework for the development of these browsers.

KEYWORDS

Augmented reality, augmented reality browser, software architectural.

1. INTRODUCTION

There are currently several languages and platforms for the development of applications for cellphones. As a result, the cellphone, more specifically, smartphones have become a new and attractive platform for Augmented Reality systems (AR) [3]. Its applications are quite diverse, from the use with educational purpose, games, engineering, even military. We currently have applications that explore the potential offered by the smartphones and by the technological advances of the AR, the AR browsers [18] [4]. They allow the search for geo-referenced information, that together with the real environment of the user, present several types of content visualized in the cellphones' camera. There are several technological options for developing a browser, from creations of authorship based in extensions and adaptations of libraries known by the market and by researchers, such as ARToolKit [2], or, more recently, using new development platforms. Some are developed by companies, such as Layar [12] Wikitude [23], Sekai [21]; others by partnership between companies and researchers, like LibreGeoSocial [15].

According to Jeon, Kim and Lee [10], despite the recent growth of AR applications based in smartphones, some problems have been identified for its effective and full utilization by the society, something already surpassed by other technologies such as Web browsers and Internet browsers. Among the problems highlighted by the authors are the lack of standardization in the development and the consequent lack of interoperability of the applications in the multiple

platforms of the mobile devices.

Lechner and Tripp [14] claim that the lack of interoperability between the platforms is one of the main problems that prevent the massive use of the AR browsers, i.e., its wide use by the society. An application, with its information services developed for Layar, for example, is not compatible for use in the Junaio [11]. This problem increases the cost of the browser development when the goal is to reach the audience of all platforms, since several versions of the same application will be necessary.

Some initiatives of working groups of the W3C confirm this concern of the researches and the industry, in relation to the open patterns for AR, as is the case of the *Points of Interest Working Group*, working group of the W3C for the standardization of the points of interest used by several applications, including RA browsers.

2. ARCHITECTURE OF THE CURRENT AR BROWSERS

Butchart [5] identifies three distinct architectural patterns for the three main AR browsers using the model proposed by Reicher [19], they are: Gateway/Platform, Standalone and Web. Most platforms of the AR browsers are a slight variation of the architectural standard Gateway/Platform, shown in Figure 1, mainly because it offers a way of generating financial profits to the platform providers. They also offer some benefits to the users, such as allowing them to find points of interest channels, and for the developers, tools to create and test these channels.
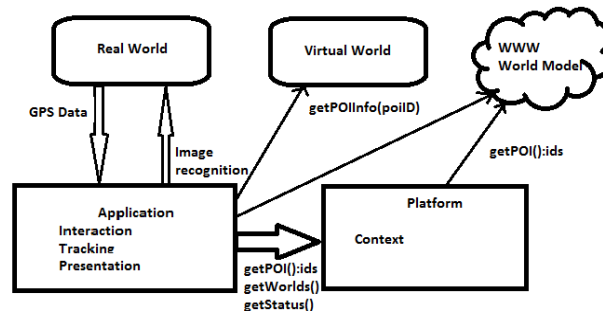

Figure 1. Architectural Pattern: Gateway/Plataform [5]

In a brief comparison, we highlight the question of the dependency of the architectural patterns Gateway/Platform and Standalone, in relation to the providers of the platforms of the AR browsers, a fact that does not occur in the Web pattern. When it is intended to propose a architectural framework for AR browser that can be used by all developers, without restrictions and technological impositions, and that allows the creation and customization without limitations, the only pattern that answers this expectations is the Web.

Another positive point in relation to the Web architectural pattern is the questions related to the development. The technologies, such as development languages, are the same ones used in other types of applications that are not AR browsers. Examples of these technologies are the HTML and Java Script languages. This strongly minimizes the learning time of these technologies by the developers when compared to the other propriety patterns of the development platforms of the AR browsers.

3. PROPOSED ARCHTECTURAL FRAMEWORK

The proposed architectural framework is based in the Web architectural style identified by Butchar [5]. A variation is considered in the application subsystem, inside the Application component, which is the AR browser itself, being this subsystem a local mobile Web application, shown in Figure 2.
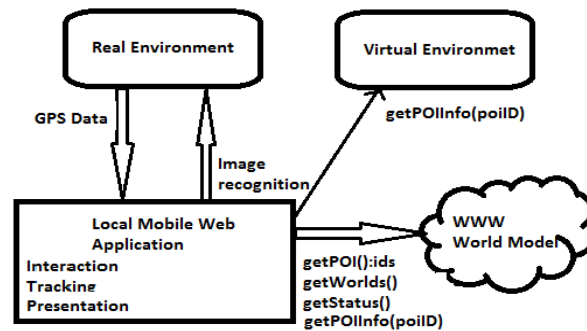

Figure 2: Local Mobile Web Architecture

According to Oehlman and Blanc [17], a mobile Web application can be thus defined: a mobile Web application is an application that is built with the core technologies of the client Web, HTML, CSS and JavaScript and is designed specifically for mobile devices.

In the proposed framework, in the logical application subsystem, the mobile web application is an application executed in a mobile device under a Web browser, using known technologies, such as HTML, CSS and JavaScript. Unlike a common web application, in which the Web server provides the HTML pages requested by the "client" Web browsers, this application will be hosted locally in the mobile device and will use a type of "slim" Web browser or generic HTML renderer.

The Webkit [21] is a generic HTML renderer compatible and used by the latest versions of the main Web browsers present on the market, such as Safari (Apple), Internet Explorer (Microsoft) and Firefox (Mozilla). The option to store the application locally allows for part of the AR browser to work offline, but it can (and should) be added to online services, required by a service or content server. The remaining subsystems, present in the Application component, are similar to those described in the architectural Web pattern by [5].

The mobile native applications are made for mobile devices, using specific programming languages for each smartphone platform, while the web mobile applications are web applications in mobile devices, using a web browser. The key difference is to leverage and use the Web pattern in the mobile applications.

Liu mentions some advantages of the mobile Web applications over the native ones [16]:

- Single version: the native applications require several versions to become compatible with the different mobile operating systems of its users. A mobile Web application uses the Web browser as a universal client program over all smartphones.
- Low cost: simplifying the development of the mobile Web applications as a single version, reduces the costs related to the tests and to the quality of the product.

- Business model: with the advance of the Web and the mobile technology, the applications are moving towards a business model in cloud computing, i.e., software as a service. Instead of developing mobile applications, a company can become a Web services provider and reach the costumers directly and not just company clients.

The use of a mobile Web application in the architecture of a RA browser allows unlimited access to the Internet, offering to the developers a bigger aggregate resource potential for their applications when compared to the current architecture of this kind of software. This makes possible to integrate functions of the AR browser with social networks, places information sites, weather prediction services, multimedia services or any other kind of internet service, in a manner already known by the developers.
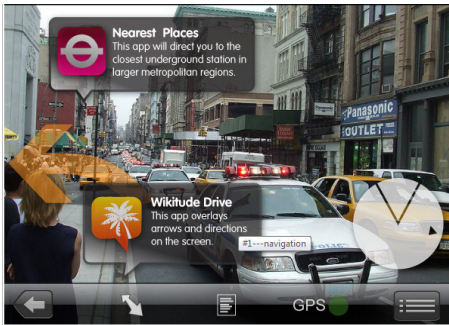

Figure 3: Wikitube running on smartphone [22]

This option also offers to the developers more freedom to create and customize the AR browsers, when compared to the current architectures. The use of the languages for Web development, such as HTML, CSS and JavaScript, don't offer restrictions or predetermined propriety patterns in the creation of the applications, as is observed in the current development platforms for AR browsers, used by Wikitude, Layar and Sekai Camera, for example. In Figure 3 is presented the Wikitube [22] running in mobile phone.

In Figure 4 the local mobile Web application of the subsystem application of the proposed architecture is presented. It is composed by the application, which will be the AR web browser in the mobile device itself and a JavaScript library that will guarantee the interoperability between various mobile platforms, called API Bridge. Besides these, a generic HTML renderer and the mobile device operating system compose this detailed view of the framework.
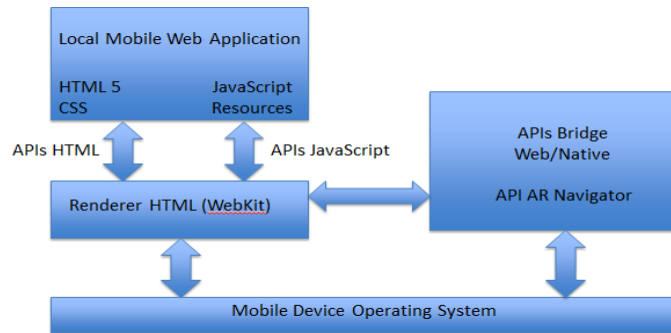

Figure 4: Local Mobile Web Application Subsystem

3.1 Local Mobile Web Application

This subsystem of the framework is the AR browser, client version, which was implemented by the developer. Taking advantage of the Web architecture, it was built using the HTML 5, CSS and JavaScript languages, and a data structure called "resources", responsible for storing some configuration information of the browser.

3.2 Generic HTML Renderer

According to Ferreira and Eis [7], there is a great diversity of devices that access the internet. Among them, there's a series of tablets, smartphones, personal computers, etc. Each one of these uses a certain Web browser. There is no way for the developers to maintain a good level of compatibility with all these browser taking into consideration the particularity of each one. A safer way to maintain the code compatible is to level the development by the HTML rendering or renderer engines. Each browser uses a rendering engine which is responsible for processing the Web page code.

It is important to make the source code in accordance with these renderers. Focusing the compatibility in the rendering engines a greater Web browser range is possible. In the case of smartphones, according to the latest researches, the sum of devices that run the OSX (Apple), Blackberry and Google Android operating systems reach 83% of the market [8]. In all these devices the standard HTML renderer is the Webkit. To ensure the greater compatibility possible among the mobile devices, it was opted to consider Webkit as the most adequate renderer for the architectural framework for the proposed AR browsers.

3.3 API Bridge (Web/Native)

This item of the architecture is the most important in regards to the solution interoperability. The goal is to offer to the developers a set of APIs in JavaScript language that can be used by the final application, the AR browser. These libraries cover the common functions in mobile devices, such as the use of the camera, geo-referencing resources, reading the sensors (accelerometer, gyroscope, etc), file handling, data storage, network access, among others (Figure 4). With the new features present in the new HTML version and with these APIs JavaScript, it's possible to access these features of the mobile devices without any native codification, thus guaranteeing the interoperability of the applications.

An important aspect in this architecture item is in regard to the implantation of the applications in the mobile devices. Each operating system for these mobile devices treats and organizes the structure of files and folders of the Web mobile applications in a different manner. For this reason, and in order to facilitate the development process, programmers are currently offered some solution of APIs Bridge (Native/Web), including open-source. These are also known as multiplatform frameworks for mobile devices. Are examples of this kind of APIs: PhoneGap, from the Nitobi Software company and Titanium Mobile, from the Appcelerator Inc. company. PhoneGap offers to the developer a set of packages compatible with each target operating system, which are basically composed by source files built for the chosen programming language, ready to be compiled in a specific platform. The developer needs only to move the specific files to a pre-defined directory, in which the Web content must reside and compile the file. Currently, PhoneGap supplies all the mobile operating systems available in the market. The disadvantage of the Titanium in relation to PhoneGap is that the first currently does not offer APIs to all the mobile operating systems available, but it offers features and works very similarly to PhoneGap.

However, for the development of AR browsers are necessary to the developer a set of other libraries that are not so common in standard mobile Web applications. As an example, there are libraries that deal with two-dimensional or three-dimensional markers, navigation, overlapping location, geo-positioning service information or games [10]. Quiros et al. [18] presents the ARViewer, a project developed by the Rey Juan Carlos University, in Spain, as a fine option of open code of an enhanced reality Web browser. In this project the AR libraries were implemented using JavaScript language. This library was used as an Enhanced Reality API of the architecture of this work, but other options can be considered or supplemented in the future.

3.4 Mobile Devices Operating Systems

The mobile devices operating systems in this architecture act with little or almost no influence. The goal is to raise the level of abstraction of the AR browsers, of the operating systems of the Web browsers, thus ensuring, with the support of some technologies, the interoperability of this kind of application to any mobile device with any mobile operating system. The mobile operating system will execute the functions necessary to the AR browsers, but the applications developed under this architectural framework will not need to relate directly with the native functions present in the operating system, for the JavaScript libraries of the API Bridge (Native/Web) are responsible for that, being transparent to the developer this relation.

Another aspect of this proposed framework is the server side of the architecture. This component, called "WWW", is similar to the one presented by Butchar [5], without modifications. Inside this Web architecture, completely eliminates the dependency on third parties, that in a certain way, made the developers use their respective development platforms, and with that they increased their revenues. In this proposed architecture, the AR browser becomes a genuine Web browser with unlimited access to the World Wide Web. This architectural "Web" pattern frees the developer from any dependency on the platform provider, but demands from them the task to make their content uncovered and to have the access control for the model "world" (Figure 5). The server part is responsible for providing the services available to the AR browsers in the mobile devices. The content services, world or channel models, the points of interest (POI), or details of these points, in addition to the context of the users, are examples of implementations that must be made on the server. Any other kind of customized service can be created by the developers, including integrating their solutions with other Web services, for example, social networking services or weather prediction ones.
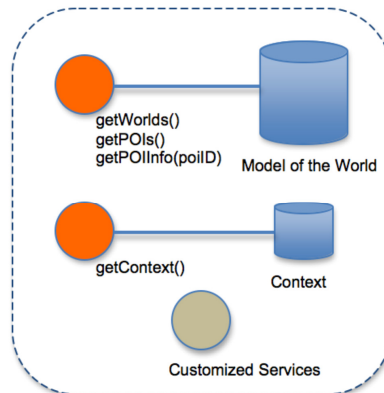

Figure 5: WWW Component

4. EXPERIMENTS

The architectural framework was submitted to three experiments inside an environment defined by a case study. To this end, it was considered the task of Augmented Reality in its subtask of guiding, as proposed by the Points of Interest Working Group of the W3C. It was used a database with points of interest related to cities in Piracicaba Region, Sao Paulo State. When navigating through the RA application the user visualizes the points of interest without any specific categorization, containing the respective titles and an image that characterizes the point. When selecting in the device screen one of these points of interest, detailed information is presented to the user, such as relative distance of Piracicaba, a title and an image.

The experiments completely implemented the proposed framework for this work for Google Android 2.2 and Apple OS X 4.3 smartphones. All the experiments used the libraries from PhoneGap 1.0 and ARViewer 1.0. The last of these experiments evaluated the effectiveness of the framework for multiple mobile platforms, using the same encoding to generate a AR browser and for a Galaxy Tab tablet running Google Android 3.1. Figure 6 presents the prototype of this last experiment in use.
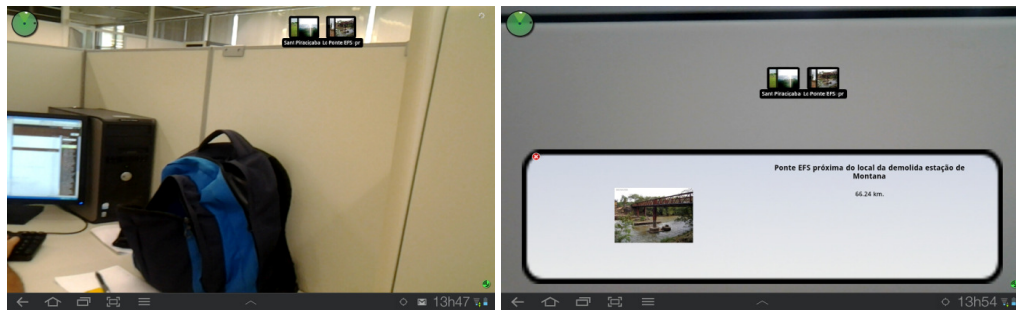

Figure 6: Prototype of an Experiment.

With the completion of these three experiments as a prototype implemented on the proposed architectural framework, some issues that were previously faced only as possibilities could be clarified.

The effectiveness of the item API Bridge (Web/Native) was noted in all of the experiments. This ensures the development of the AR browsers is supported by all development patterns already known by the software industry, and mainly, by the developers. The Web programming languages and its possibilities are known, practically, since the beginning of the internet.

In all of the experiments, containing different platforms and devices, a single local mobile Web was developed, and it worked with the same functional behavior. The interoperability of the application was ensured in terms of source code, i.e., after it was encoded, it can be reused to generate the same application in other platforms.

Another result of these experiments is the prove that it is possible to develop a AR browser without depending on third party industries, suppliers of AR development platforms, as Layar and Junaio, and without having to create basic resources of encoding in a AR application. This is possible, for example, through open-source libraries, in this case, ARViewer. In addition, with the Web architecture, the developer has a larger freedom to create and integrate his AR applications.

One point that needs to be analyzed in future work is the question of HTML rendering integrated with smatphones and tablets cameras. In all experiments, a small rendering delay could be observed. Nothing that hinders the user or that occurs all the time. This resource of camera integration is relatively new in HTML, being added in version 5, and should be improved by W3C, just as it occurred with the other resources of this language since its creation.

5. CONCLUSION

The AR browsers are a type of application developed recently to explore this potential offered by smartphones and by the technological advances of the AR. They allow the exploration of geo-referenced information, complementing the real environment of the users with several types of content visualized in the robust cellphone cameras.

However, despite the recent growth of this industry of AR application browsers, based mostly in smartphones, some problems have been identified, such as difficulties in its full and effective use by the society, something already surpassed by other technologies, such as Web browsers. Among the problems highlighted by the main researches are the lack of standardization and the consequent lack of interoperability of the applications in the multiple platforms of the mobile devices, what in some way does not contribute for a larger mass use of this technology by the society. Schmalstieg and Langlotz [20] emphasize that, for the AR to be implanted in a large scale, several aspects of the technology need to improve, including the lack of a low cost platform to develop applications and the lack of easy to use authorship tools. Specifically for AR browsers, the options to create applications that propose to be easy to use are proprietary and non-open technologies.

This work considered these points mentioned, applied to some case studies and experiments, to propose an architectural framework for the development of AR browsers that have as main target characteristics: the interoperability guarantee, the use of already established patterns, the freedom to create customized applications and the low cost.

The architectural framework proposed in this work presents a real possibility to implement AR browsers absent of the problems mentioned earlier, which can provide an effective implantation option by the software industry. The characteristics present, in actuality, use technologies already known and developed in the market and composed a more adequate framework software for this application. It also offers to the developers and final users, quality of software characteristics similar to those found in AR browsers based in development platforms of the market.

References

[1]   Andel, M. & Petrovski, A., (2006) "Interactive Collaborative Augmented Reality using a Distributed Scene Graph on Mobile Phone". United States of America: Linkoping University.

[2]   Artoolkit, Artoolkit Site. Available in: http://www.hitl.washington.edu/artoolkit/. Accessed on 20 nov. 2014.

[3]   Billinghurst, M. & Henrysson, A., (2006) "Research directions in handheld AR". The International Journal of Virtual Reality, v.5, n.2, p.51-58.

[4]   Butchart, B., (2011) "Augmented Reality for smartphones: A guide for developers and content publishers". Techwatch Report, JISC Observatory.

[5]   Butchart, B., (2011) "Architectural Styles for Augmented Reality in Smartphones", 4th International AR Standards Meeting, Basel, Switzerland.

[6]   Correia, N., (2003) "Mobile Augmented Reality". Master's thesis. Portugal: Faculdade de Ciência e Tecnologia de Évora.

[7]   Ferreira, E. & Eis, D., (2011) "HTML 5: Curso W3C". Sao Paulo, Brazil: W3C Brazil Office.

[8]   Gigaon, (2014) "Marketshare das plataformas móveis". Available in: http://gigaom.com/2011/03/03/android-pulls-ahead-while-htc-enjoys-the-ride/. Accessed on 20 nov. 2014.

[9]   Hurn, J., (1989) "GPS - A guide to the next utility". Trimble Navigation. Sunnyvale, United States of America.

[10] Jeon, J. & Kim, S. & Lee, S., (2010) "Considerations of Generic Framework for AR on the Web". Proceedings of W3C Workshop: Augmented Reality on the Web, Spain.

[11] Junaio, Junaio Site. Available in: http://www.junaio.com/. Accessed on 15 nov. 2014.

[12] Layar, Layar Site. Available in: http://www.layar.com/. Accessed on 10 nov. 2014.

[13] Lechner, M.,(2011) "A Proposed Step-By-Step Guide to an AR Standard". 10th IEEE International Symposium on Mixed and Augmented Reality, Proceeding of the ISMAR 2011, Switzerland.

[14] Lechner, M. & Tripp, M., (2010) "ARML: An Augmented Reality Standard". Mobile World Congress 2010, Mobile AR Summit, Spain.

[15] Libregeosocial, LibreGeoSocial Site. Available in: http://www.libregeosocial.org/. Accessed on 15 nov. 2014.

[16] Liu, K., (2009) "A Taxonomy and Business Analysis for Mobile Web Applications". United States of America: Massachusetts Institute of Technology.

[17] Oehlman, D. & Blanc, S., (2011) "Android Web Apps: Develop for Android Using HTML5, CSS3 & JavaScript". Ed. Apress.

[18] Quiros, P. et. al., (2010) "Mobile Augmented Reality browsers should allow labeling objects". W3C Workshop: Argumented Reality on the Web, Proceedings. Barcelona, Spain.

[19] Reicher, T., (2004) "A Framework for Dynamically Adaptable Augmented Reality Systems". Munich, Germany: Technical University Munich.

[20] Schmalstieg, D., Langlotz, T., (2009) "Augmented Reality 2.0". ISMAR 2009, Innovation Workshop Proceedings. Orlando, United States of America.

[21] Sekai, Camera, Sekai Camera Site. Available in: http://www.sekaicamera.com/. Accessed on 13 nov. 2014.

[22] Webkit, The Webkit Open Source Project. Available in: http://www.webkit.org/. Accessed on 10 nov. 2014

[23] Wikitube, Wikitube AR Site. Available in: http://www.wikitude.org/. Accessed on 15 nov. 2014