AN EFFICIENT INTEROPERATION BETWEEN MOBILE APPLICATIONS USING PHONEGAP WITH LCIM MODEL

C. Shanthi¹, M.S. Josephine² and V. Jeyabala Raja³

¹. Ph.D., Research Scholar, Vels University, Chennai, India. Email: shan_c08@yahoo.co.in
². Asst Professor, Department of MCA., Dr. MGR Educational and Research Institute, Chennai, India. Email: josejbr@yahoo.com
³. Professor, Department of Computer Science and Engineering, Velammal Engineering College, Chennai, India. Email: jeyabalaraja@gmail.com

Abstract

Mobile communication is everywhere around us, the essential to communicate between the variety of the devices which is available. The major obstacle to achieve communication between the available devices is the incompatibility between the operating systems operation on these devices. In this paper we propose the possible ability to inter-operate between processes running on different mobile operating systems. The interoperability make use of any communication atmosphere which is made available by the mobile devices where the processes are installed. The communication environment is preferred so as the process is most advantageous in terms of transferring the data between the mobile devices, main problems that arise in development of cross platform mobile applications are concerned which create solution that will work on different mobile operating systems like Android, Windows Phone and iOS. The mobile applications are created by PhoneGap cross platform tool which helps to avoid developing same type of application separately and this is best cross platform tool for achieving interoperability using LCIM model.

I. Introduction

Mobile applications follow the same stringent business as Web sites; it is natural that users have the same expectations of them. Providers of mobile applications will make further efforts to gain a better understanding of users such as effectively to sell products, services and advertising for mobile devices [1]. Mobile applications have a specific set of rules defined and rigorously chosen as running operations on mobile devices [2]. If a new application is integrated with other applications, this application and other applications are built using a set of specific rules for all mobile applications. The new application is taken directly from other applications because it follows the same set of rules as the other [3]. Thus a mobile application is fully deployable if it is built by the same pattern as the applications they want to be integrated.

The need for cross-platform development approaches arises from the incompatibility of today’s platforms for mobile devices. The available technological options for developing cross-
platform mobile applications are web-based apps and hybrid apps. Web-based apps (or ‘web apps’) are basically web sites, which are optimized for mobile devices. Because of this, they are said to lack device-specific functionality and performance [4]. Whereas performance issues are slowly decreasing due to optimizations, missing functionality is not an option for today’s sensor-packed mobile devices [5]. As opposed to this, hybrid apps, though also relying on standardized web technologies, are bundled within a native app “container”, which serves as a bridge to access device hardware and functions like ringing, vibrating, and notifications, making calls, using the camera, GPS sensor and accelerometer [6]. The web standards mainly used are JavaScript for business logic and HTML5 and CSS3 for presentation (forms, styling, transitions etc.) and offline storage. These technologies are used by most modern web browsers and therefore they are supported more and more completely by all vendors [7]. As a result of the web technology-based approach, cross-platform apps also circumvent the vendor-specific development process (special programming languages and IDEs)[8].

2. Levels of Conceptual Interoperability Model (LCIM)

Interoperability makes systems and organizations work together (inter-operate) and writing cross-platform code that provides native functionality, performance, and look-and-feel across Android, iOS and Windows operating system. When the system follows many levels for interoperability between mobile apps cross-platforms the best exchange of data can be done using Level of Conceptual Interoperability Model (LCIM) shown in figure 1 [9]-[10].

![Figure 1. The Levels of Conceptual Interoperability Model](image_url)
TABLE 1: Levels of Conceptual Interoperability Model – Descriptive Model

<table>
<thead>
<tr>
<th>Level</th>
<th>Layer Name</th>
<th>Description of level</th>
</tr>
</thead>
<tbody>
<tr>
<td>L6</td>
<td>Conceptual</td>
<td>Interoperating systems at this level are completely aware of each other’s information, processes, contexts, and modeling assumptions.</td>
</tr>
<tr>
<td>L5</td>
<td>Dynamic</td>
<td>Interoperating systems are able to re-orient information production and consumption based on understood changes to meaning, due to changing contexts.</td>
</tr>
<tr>
<td>L4</td>
<td>Pragmatic</td>
<td>Interoperating systems will be aware of the context (system states and processes) and meaning of information being exchanged.</td>
</tr>
<tr>
<td>L3</td>
<td>Semantic</td>
<td>Interoperating systems are exchanging a set of terms that they can semantically parse.</td>
</tr>
<tr>
<td>L2</td>
<td>Syntactic</td>
<td>Have an agreed protocol to exchange the right forms of data in the right order, but the meaning of data elements is not established.</td>
</tr>
<tr>
<td>L1</td>
<td>Technical</td>
<td>Have technical connection(s) and can exchange data between systems.</td>
</tr>
<tr>
<td>L0</td>
<td>No</td>
<td>NA</td>
</tr>
</tbody>
</table>

3. Proposed methodology

Cross-platform interoperability problems tend today, to be specific to ad-hoc mobile devices. These devices typically run proprietary operating systems upon which a whole new class of native applications have been developed using proprietary software development tool.

There are several ways for developing apps for multiple platforms. In general, cross-platform and native development can be distinguished. Native development uses the platform specific software development kits (SDKs) along with the applicable development tools. The programming language is bound to the platform; for example, Java is used for Android apps whereas Apple requires usage of Objective C (or, newly, Swift). Consequently, development is carried out for each platform separately. Development effort increases almost linearly with each additional platform of the typical activities carried out in software engineering. Only requirements engineering is more or less shared while design, implementation and testing are carried out independently on each platform. In sharp contrast, cross-platform approaches follow an “implement once run everywhere” principle.

Cross-platform apps can be developed following one of two main paradigms. A runtime environment abstracts from the native interface of a platform. Generative approaches allow developing an app once and then to generate native source code for each supported platform. Generative approaches can be realized in two possible ways. Firstly, Model-Driven Software Development (MDSD) can be employed. This app is described using a (typically textual or graphical) modeling language, which is independent of an actual platform. Such a model is transformed to native, platform-specific code. Transformation is usually done by using tools.
Secondly, transpiling translates the single used programming language to platform-specific ones. Finally, a self-contained runtime allows development based on Web technology but offers a bridge to native GUI elements. Thereby, apps become closer to a native look & feel at the price of a performance penalty.

**Cross-Platform framework:**

As incompatibilities of mobile phone platforms, it is hard to develop mobile application for developers on various platforms [11]. By using cross platform framework, the system can develop quality mobile apps for our clients which can serve their needs successfully and can run on a variety of platforms. Now, there are many cross-platform frames in the market, Phonegap is considered the best one. PhoneGap is called as Apache Cordova is a mobile development network that is both open source and free and also enables you to generate cross-platform mobile apps easily. It will be used to package and build the application into native application for different mobile platforms. PhoneGap allows development of mobile applications using web technologies by providing an interface to a web view component and tools to create platform-specific project files and initial source code that shows the web view. PhoneGap offers generic access to platform specific features on all major mobile platforms.

The PhoneGap architecture is composed mainly of three layers: Web Application, PhoneGap, and OS and native API’s. In Figure 2 the top layer represents the application source code. The central layer is composed by JavaScript and native API’s. Mainly, this layer is responsible for the interfacing between web application and PhoneGap layers. Furthermore, it also takes care of the interfacing between JavaScript API’s that are used by the application with native API’s that are used by mobile OS’s. The functionality of this layer is to maintain the relationship between JavaScript API’s and native API’s of each mobile OS. PhoneGap supports most of the mobile operating systems like iPhone, Windows Mobile, BlackBerry, Android Symbian, and WebOS which can be worked along with other operating systems like Linux, Mac and Windows shown in figure 3. It also supports scripting languages like HTML, JavaScript, and CSS which makes it an easy tool to work with. PhoneGap provides JavaScript API’s to developers that allow the access to advanced device functionality, such as Accelerometer, Barcode, Bluetooth, Calendar, Camera, Compass, Connection, Contacts, File, GPS, Menu, NFC, etc.
Figure 2. Interfacing Layers of the PhoneGap Architecture

Figure 3. Mobile operating systems supported on PhoneGap framework
PhoneGap performance can be fast enough for many types of applications if the right web framework is chosen. In order to create applications with PhoneGap, the user will need to first install the standard SDK for the mobile platforms. This is because the PhoneGap actually use these SDKs when compiling your app for that platform. Figure 4 shows all the APIs supported by the framework.

![PhoneGap API's](image)

**Figure 4. PhoneGap API's**

The working theory of PhoneGap is not complex. The application’s user interface consists of essentially a unique screen that contains a unique web view that consumes all of the
available space on the device’s screen [12]. When the application launches, it loads the web application’s startup page (typically index.html but easily changed by the developer to something else) into the web view and then passes control to the web view to allow the user to interact with the web application. As the user interacts with the application’s content (the web application), links or JavaScript code within the application can load other content from within the resource files packaged with this application or can reach out to the network and pull content down from a web or application server.

The PhoneGap tool is utilized for best interoperability using LCIM model. The LCIM model of the system engineering domain is the method and model towards enhancing interoperability among apps. The LCIM is used in both descriptive and prescriptive form. In descriptive role, the LCIM can be used to describe the levels and properties of interoperability a given app can exhibit that is rate the apps based on interoperability. In the prescriptive role, the model prescribes the methods and requirements that must be satisfied during engineering phases of app development to achieve a desired level of interoperability. LCIM model not only enhances the interoperability but it also provides a better way to rate the mobile apps. The mobile apps are currently rated based on content. The designed system presented a LCIM model for its best interoperability between mobile apps cross-platform for getting conceptual interoperability between mobile apps. Overall, the level by level approach was found using LCIM model and suitable for best interoperability among the cross platform apps.

Benefits of developing Cross-Platform applications using PhoneGap are as follows:

- A major advantage of the cross-platform frameworks is code reuse.
- Learning web technologies might be less challenging that learning an all new platform
- Cross platform mobile apps written using one SDK, operating System and development environment requiring less financial investment
- Cross platform apps try best to emulate natively written applications
- Cross platform apps might exploit well the native controls

The weakness of developing Cross-Platform applications using PhoneGap is as follows:

- Recommended as a contender for applications which are heavily web dependent
- Lack of support for native UI components, design patterns and development tools
- Another important issue concerning cross-platform apps seems to be a divergence in terms of user interface and its challenges.
4. Performance analysis

In this work, by using LCIM model with Phonegap there is difference in processing data. The interoperation data level is high using this model with phone gap.

Table 1: Interoperability of mobile applications

<table>
<thead>
<tr>
<th>Interoperability of mobile applications that provides the output text or an image is reflected by</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send by e-mail</td>
<td>0</td>
</tr>
<tr>
<td>Send by SMS</td>
<td>5</td>
</tr>
<tr>
<td>Post on social network</td>
<td>10</td>
</tr>
<tr>
<td>save in a file system on the phone</td>
<td>15</td>
</tr>
<tr>
<td>Save in a picture file o the phone</td>
<td>20</td>
</tr>
<tr>
<td>Save in a file on the extern server</td>
<td>30</td>
</tr>
<tr>
<td>Save in a picture on the extern server</td>
<td>50</td>
</tr>
<tr>
<td>Save in a database on the extern server</td>
<td>80</td>
</tr>
<tr>
<td>Save in clipboard memory</td>
<td>100</td>
</tr>
</tbody>
</table>

The low energy consumption for PhoneGap is a consequence from it having much shorter execution time than the other two (Native and Codename one) during the Write to SQL phase.
The energy consumption chart shows, figure 5, PhoneGap used the least energy during its execution and Codename One had highest. The biggest reason for this is that it had a much faster execution time. It would have been interesting to see the energy consumption if they had similar execution time. Then the execution time wouldn’t have such big impact on the result and we would be able to see how they used memory and CPU impact the energy consumption.

![Energy Consumption Chart](chart.jpg)
The small energy consumption by PhoneGap correlates with the high CPU use it had when writing to SQL which made it have much shorter execution time and thus the energy consumption is less than the other two.

Figure 6. Execution Time

Figure 7. Total memory usage
When measuring CPU usage it is very important to measure execution time. Otherwise the result can be interpreted in different ways. Phone gap achieves higher memory use compared with other two.

5. Conclusion

The cross platform mobile application development frameworks are benefiting developers to build applications for multiple platforms. PhoneGap tool is utilized for best interoperability using LCIM model. PhoneGap is a mobile cross-platform development framework invented to solve the problem of different programming languages that developers challenge/encounter when building applications for different mobile platform. PhoneGap API is supported for famous mobile operating systems such as iOS, Android, Black Berry, Symbian, WebOS and Windows 7 Phone. By using LCIM the apps can be rated based on interoperability. Apps of different vendors can interoperate efficiently when LCIM is followed while implementation. It is quite popular among users mainly because of its flexibility, straightforward architecture and ease of use.

References


