Evaluating Architectural Openness in Mobile Software Platforms

Mohsen Anvaari
IT University of Gothenburg
mohsen.anvaari@gmail.com

Slinger Jansen
Utrecht University
slinger@slingerjansen.nl

ABSTRACT
Software platform developers do not have insight into the consequences of architectural decisions made in regards to openness, while openness determines whether and how third-party developers adopt the software platform. For mobile communication devices, however, the adoption by third-party developers is essential for the growth of an ecosystem, which has proven to be a successful strategy for mobile communication device ecosystems. In this research the openness and architectures of five mobile platforms are compared, to provide further insight into successful openness strategies for software ecosystem growth.

Keywords
Mobile Software Platforms, Openness Strategy, Platform Architecture, Platform Accessibility, Qualitative Interview.

1. INTRODUCTION
Open source or proprietary; which of them is more successful? Which strategy has received more attention from the developers? Which does lead to more innovative applications? One can say the open source strategy is more successful due to its creativities and inventions [1] and on the other hand, one can believe that the proprietary strategy is more successful because it will lead to more qualified products due to strong control. The reality, however, is not that black and white, especially when it comes to the software on the smartphones called mobile software platform. Considering a platform as open or closed is rarely a binary decision and generally is a question of "how open" [2]. The answer to the question deals with openness strategy of the platform.

Openness strategy is the degree to which a platform supplier allows the platform users to interact with the platform, view, extend or change its components and depends on different technical and commercial aspects such as platform architecture, platform accessibility, platform transparency, licensing state, marketing policy, etc. Mobile software platform means the overall structure of the software on the mobile devices [3]. The openness strategy is different among various mobile software platforms of smartphone ecosystems. A software ecosystem is "a set of actors functioning as a unit and interacting with a shared market for software and services, together with the relationships among them" [4]. When the definition comes to the smartphone area software platform suppliers, device manufacturers, operators, application developers, device customers, etc are considered as the actors and participants of the ecosystem.

In the current smartphone ecosystem Symbian, Windows Mobile, iPhone, BlackBerry, and Android are the main software platforms competing for superiority [5]. Some of these mobile platforms are more successful than others and many differences between these platforms exist. Some are available for any hardware such as the Android platform, whereas others are only available on limited hardware such as the iPhone platform. To implement the openness strategy that platform suppliers have made, the architecture of their platform should support proper platform accessibility for their application developers and device manufacturers.

This research studies the openness strategy of the different mobile platforms. The scientific contribution of this research lies in the identification of the openness strategy of the main mobile platforms based on their architecture. The platform suppliers define their openness strategy based on their business model by considering the architectural aspects of the platform. The strategy they choose affects device users, application developers, device manufacturers and operators. In this research, the architectural aspects of the platforms and some licensing aspects related to platform accessibility are studied. Platform architecture is the structure of the software platform compromises its components and the relationship between them [6] and platform accessibility means the methods and points that developers can use to extend or modify the platform. Since other aspects of the business model such as marketing are not in the scope of *software engineering* they are not studied in this research. To validate the openness strategy in different mobile platforms, only application developers are chosen to be interviewed. This is due to time limitation, although the device manufacturers are also in the scope of this research.

The remainder of this paper is structured as follows: In section 2, the research questions of the study are clarified and a summary of the research methods is presented. Before describing the activities of the study, a summary of related works is presented in sections 3 and in section 4 results of the study are described. Section 6 describes the openness strategy in main mobile platforms. Sections 6 and 7 discuss the research results and argue how the research methods provide an answer to the research questions. Part 7 summarizes all sub-questions and answers, and provides some pointers for future research.

2. RESEARCH APPROACH
The research question and sub-questions are defined as follows:

RQ: How does the software architecture expose the openness strategy of mobile software platforms?

- SQ1: Can a model be developed that describes the architectural openness of mobile platforms?
- SQ2: How would the licensing aspects of the platforms relate to such a model?

- SQ3: How open are the five main mobile platforms?

In the following section the methods chosen for this research are discussed.

To answer above questions, literature reviews of architectural descriptions and qualitative interviews with third-party developers are held. The literature review has the aim of finding the relationship between openness strategies and the software architecture of mobile platforms. Furthermore, the literature study aims to find out how application developers can extend the platform. The second research method is that of qualitative interviews. These semi-structured interviews with mobile application developers are held to verify the platform accessibility in the five main platforms that enables application developers to extend the software platform, which previously has been captured from the literature and platform documents. An architectural reference model is created to connect between the openness strategy and the software architecture of a mobile software platform.

In qualitative researches, the two main types of interviews are semi-structured interview and unstructured interview. "Researchers sometimes employ the term qualitative interview to encapsulate these two types of interviews" [6]. For this research semi-structured interviews have been chosen. The main objective of this study is validating the openness strategy of mobile platforms by gaining experiences of mobile platform developers. To prepare the questionnaire, some general questions were created based on the objectives of interviews. Some more specific questions about the favorite mobile platform(s) of interviewees were added to the list of questions. Simultaneously, several application developers in the Netherlands were contacted to see whether they are interested to be interviewed or not. The plan was to find one application developer for each platform. After finalizing the questionnaire and making appointments with interviewees, the interviews are conducted. The interviews are recorded by a voice recorder and all interviews have been transcribed. Finally for analyzing the interview data, the proposed stage-by-stage method by Burnard is applied. The aim of this method is "to produce a detailed and systematic recording of the themes and issues addressed in the interviews and to link the themes and interviews together under a reasonably exhaustive category system" [8].

3. RELATED WORK

The most recently published article in the area of smartphone ecosystem states that few attempts have been made to demonstrate the comparison of smartphone OSs [9]. It is also true particularly about comparing the openness strategies in different mobile software platforms. Lin and Ye, themselves, have compared the ecosystem of iPhobe, Symbian, Windows Mobile and Blackberry by discussing about the value chain (they call it "food web") of the ecosystems, but they have not discussed the openness strategy of the platforms based on the architectural aspects. Further, Android is not in their comparison. Cho and Joen have discussed and compared Symbian, Linux and Rex [10], which, except Symbian, are not the current major platforms. Besides that, although they have compared the architecture of the platforms and also the openness strategy of the platforms, but not a connection between two subjects has been settled. Yamakami has covered Android, Symbian, iPhone and Windows Mobile in his competitive analysis [10]. But the factors he has considered in his analysis are governance, ease of maintenance, interoperability, ecosystem, cost reduction and reliability, which are not related to architectural aspects. Constantinou has discussed open source in mobile platforms and shows the current state of openness in software stack of mobile platforms [12]. It is very general and not applied to any specific platform.

Beyond the software platforms on mobile devices, in the software area in general, there is again a lack of works that discuss how software architecture is treated in open source projects [13]. Nakagawa et al. have shown that software architecture has an important role in open source projects via a case study, but they have not discussed how the openness strategy affects the architecture. Prehn has concluded in his article that one characteristic that is shared by the largest and most successful open source projects is a software architecture and it has to be very modular [14]. Arief et al. in a similar conclusion shows that for open source projects the architecture must be modularized [15]. Both are not applicable results for the case of comparing the openness strategy in different mobile software platforms.

4. RESULTS

Openness is the "quality or condition of being open" [16]. The software openness, in the scope of this research, means the degree to which a software platform approaches to open characteristics which depend on accessibility and licensing as key aspects. There might be other aspects which determine openness in a software platform but only these two are researched here as explained in the introduction part. Alspaugh et al. [17] in their research mention several software elements included in common software architectures that affect the openness of architecture. The elements are software source code components, executable components, application program interfaces (APIs) and configured system or sub-system architectures [17]. The ways that architectural elements of a software system affect the openness of the system are being expressed as software extension mechanisms and approaches. Klatt defines software extension mechanism based on extension points. An extension point is the definition of the provided interface for extensions. An extension itself is an implementation according to an extension point. An extension mechanism includes everything about an explicit extensible part of a software" [18]. Jansen et al. mention several mechanisms in their article that extend software functionality. The mechanisms are component calls, service calls, source code inclusion, and shared data objects [19]. To analysis more specifically the extension in mobile software platforms, a discussion of extension mechanisms on operating systems is needed since software platforms on mobile devices are expressed as operating system of the devices [20]. Alexandrov et al. present different approaches for extending the operating system. The ways they have mentioned are extension or modification of different levels of an operating system from the lower levels to the higher levels. The approaches are changing operating system (kernel) itself, modifying device drivers, installing a network server, adding user level Plug-Ins, making changes to user level libraries, applications specific modifications and intercept system calls [21]. Their point of view about extension approaches is employed to develop the architectural openness model for this research and is discussed in the following chapters

4.1 Mobile Software Platforms and Their Architecture

In the last decade, mobile phones have become programmable handheld computers which have internet connectivity, computing power and open application programming interfaces (APIs) providing prospective platforms for an infinite set of new mobile services and applications [20]. The new mobile phones which are usually called smartphones have both hardware and software parts the same as all computing systems. The software part is called mobile software platform. Some authors like Verkasalo use software platform as a synonym to operating system of mobile devices [20]. Some others like Cho and Jeon believe that software platform of a system means the overall structure of the software on the system and operating system is a part of it [10]. Cho and Jeon consider a layered architecture for the software platform of typical mobile devices consists of operating system layer, middleware layer and applications layer [10]. Layered architecture is an architectural pattern helps to structure systems that can be decomposed into groups of subtasks in which each group of subtasks is at a particular level of abstraction [22]. The software platform of mobile devices is such a system, therefore the layered architecture is applicable to the architecture of mobile software platforms. The suggested model for architecture of mobile software platforms for this research is the same as proposed architecture by Cho and Jeon with some modifications. In their model, they have not separated the default applications which are set by device manufacturer from the applications developed by third-party community and installed by users. Since third-party applications have a main role in defining the openness of a platform, the proposed model for this research has two different sub-layers in the applications layer: native applications and extended applications. Native applications are those developed by device manufacturers and in some platforms are not modifiable. Extended applications are those developed by application developers and installed by device users, so these applications extend the applications layer of the platform. Middleware layer consists of main libraries and services of the platform like data storage, virtual machine, multimedia libraries, etc. When application developers create extended application, they usually call this layer in their applications instead of calling the core libraries of the platform. The kernel layer, which in Cho and Jeon's model is called operating system layer, is the core the platform. It consists of the lower level components of the platform such as device drivers, power management framework, security framework, etc. Figure 1 shows the proposed architecture for mobile software platforms.

The architecture of main mobile platforms include Android, iPhone, Symbian, Blackberry and Windows Mobile is looked by the lens of this architecture and the results are presented in the following parts. In the next section, the architectural openness model which is built based on the proposed architecture is discussed.

4.2 Architectural Openness Model and Factors

As presented in the previous part, a mobile software platform like other software systems has an architecture which is the structure of the platform. A proposed mobile software architecture that is a general model and can be applied to different mobile platforms was shown. To discuss the openness strategy of mobile platforms based on their architecture, the proposed model is not sufficient and it should be expanded to an "architectural openness model" to demonstrate platform extension mechanisms and platform accessibility since these notions have a connection with the openness concept as discussed before.

The architectural openness model to accommodate the platform accessibility and platform extension methods, and in a higher view the platform openness, should illustrate how much and under which conditions the platform extenders (application developers, device makers, customers…) can access to different layers and components of the platform and extend its functionality. Two online resources of Google Android have used and defined *integrate, extend* and *modify* concepts to clarify the openness notion in the architecture of Android platforms [23][24]. Sim et al. mention similar meanings when consider *integration* and *customization* as issues in mobile operating systems [25]. To expand the "mobile software architecture" presented in the previous part to "architectural openness model", the same terms and definitions are used here:

Integrate a layer: To use the existing components of a layer in a mobile application via API, Service Call, source code inclusion, shared data object and other software extensions mechanisms.

Extend a layer: To enhance the functionality of the components of a layer. The application uses the built-in Google map application and adds its own functionality on top of Maps is an example.

Modify a layer: To replace or change the components of a layer. Writing your own device driver is an example.

The architectural openness model to support the openness strategy in mobile software platforms is illustrated in the Figure 2.

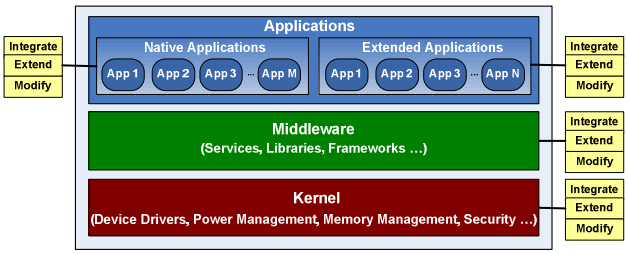

Figure 1. Architectural Openness Model for Mobile Software Platforms

Although the model shows the platform access and extension methods in different levels, but to demonstrate the openness degree of a platform, licensing aspects of mobile platforms should also be considered. Different mobile platforms have various licensing considerations. Besides the possibility to integrate, extend or modify the components of a platform, the permission of these activities depends on the platform supplier's licensing policy. In some platforms you don't need any permission to extend the platform whereas in others you need to submit the change to the platform supplier first and if they accept, the changes will be confirmed.

Based on the proposed architectural openness model and possible states of licensing, the Table 1 shows the architectural openness factors for mobile software platforms. Applying this table to a

particular mobile platform, the result for instance for middleware layer of the platform can be like this: Integrate the components of the layer is allowed and doesn't need any permission. Extend the components of the layer is allowed but needs permission from the platform supplier. And modify the layer is allowed only for some components and needs permission.

These factors and their statuses provide a continuum to determine how much a platform is open based on the architecture of the platform. A platform is considered entirely open if in all the architectural layers of the platform integrate, extend and modify the components are possible without any permission from the platform supplier. On the other end, a platform is totally closed if within all layers of the platform integrate, extend or modify the components are not possible. Other platforms are situated between these two boundaries. In the following part, the situation of five main mobile platforms being Android, iPhone, Symbian, Blackberry and Windows Mobile in the openness continuum based on the architectural openness model and the openness factors is discussed.

Table 1. Architectural openness factors for mobile software platforms

Layer	Factor	Possibility	If possible → Licensing status
Extended applications	Integrate extended applications	Possible/ Possible for some components/ Not possible	Permission is not needed/ In some situation permission is needed/ Permission is always needed
	Extend extended applications		
	Modify extended applications		
Native applications	Integrate native applications		
	Extend native applications		
	Modify extended applications		
Middleware	Integrate middleware		
	Extend middleware		
	Modify middleware		
Kernel	Integrate kernel		
	Extend kernel		
	Modify kernel		

4.3 Openness in a Mobile Platform from the Application Developers' Point of View

The main objective of qualitative interviews in this research is verifying the openness strategy of main mobile platforms based on the interviewees' experiences. The main results of the interviews are presented in the next section where openness strategy of main mobile platforms is discussed. In the following sub-sections the openness in a mobile platform, as a general concept, from the interviewees' point of view is discussed which is the implicit result of the interviews. First, the opinion of interviewees about the relationship between openness in a mobile platform and architectural and licensing aspects of the platform is presented. Then the importance of openness in a mobile platform for application developers is discussed.

4.3.1 Relationship of Openness in a Mobile Platform with Architectural and Licensing Aspects of the Platform

All of the interviewees believe that architectural and licensing aspects of a mobile platform affects its openness degree. One of them counts the programming language, the way libraries of a platform work, if the frameworks are object oriented or not and the SDKs provided to extend the platform as the architectural aspects of the platform and believes that "a lot of openness depends on the architecture". Another interviewee thinks that if the architecture of a platform is messy it will be very hard to make it open and on the other hand, if the architecture is modular and has layers and subsystems then it will be easy to open it up. In their point of view, the licensing aspects also affect the openness of a platform since to access the different layers of the architecture of a layer or to modify the components of a platform you need to have a license.

But in their opinion, the architectural and licensing aspects are not the only parameters that affect the openness degree of a platform. "You can do a lot with architecture to open a platform but it is not the only aspect", one of the developers believes. He explains that the available samples and examples for developers, the provided community and the platform documentation are some other aspects that affect the openness of a platform.

4.3.2 Importance of Openness in a Mobile Platform for Developers

Overall, for most of interviewees the openness degree of a mobile platform is not such important and they care less about the openness of a platform. Since they are commercial developers, for choosing a platform the financial aspects of the platform is more important and determinant for them. One of them believes that managers should more care about openness and they should be aware what is allowed to do and what is not allowed. Another explains that if a platform is open enough to make a lot of money for him, then the platform is interesting for him.

Nevertheless, openness in higher levels of a mobile platform is a considerable aspect for interviewees. For example, although Apple has supported some APIs in the kernel layer of the iPhone and RIM has not supported any API in the kernel layer of the Blackberry, one of the interviewees considers the Blackberry more open than the iPhone, because native applications in the Blackberry are allowed to integrate but in the iPhone are not. All of the interviewees are more confident about the openness degree of higher layers in their favorite platform than the lower layers. They are not sure about kernel layer and most of them about middleware layer because they have not tried to integrate, extend or modify components of these layers. One of them explains that "the middleware you want to integrate is huge, so … you as a business developer don't care about extending or modifying it". Most of them believe that even if a platform opens up its middleware or kernel layer, the people who benefit from the openness of the platform and are interested to access or modify the components of the lower layers are device manufacturers not application developers. All of the interviewees are almost satisfied by current openness degree of their favorite platforms and just one of them would like to see a small part of his favorite platform to see opened up which is Wi-Fi library in Blackberry.

5. OPENNESS IN MAIN MOBILE PLATFORMS

In this section, based on the architectural openness factors, documentation of the mobile platforms and results of qualitative interviews, the openness strategy in five main mobile platforms being Android, iPhone, Symbian, Windows Mobile and Blackberry is discussed

5.1 Android

In November 2007, Google formed a group of mobile and technology companies called The Open Handset Alliance. The aim of this conglomerate is improving mobile phones to change the mobile experience for customers by increasing the openness in the mobile ecosystem. Their first joint project is Android which has been released officially on October 2008 [26]. Android is a Linux-based OS that's geared to run on lightweight devices like mobile phones [27]. Software architecture of Android platform is a layered architecture. It is possible for a developer to code in any of the layers, one of the Android developers explains. He clarifies his opinion about openness in Android more by separating the situation in literature and in reality based on his experience: According to the official documents "Android is open source. If a developer wanted to make changes to something in the Kernel, they could submit it to Google who manages the project, and Google would approve the change, and allow it into the framework. In practice, I think it is difficult for anyone outside of Google to get changes into the core of the framework". All of the interviewees believe that even if in practice Android is totally open and everyone can modify the source, this openness makes sense for device manufacturers not commercial developers. In Table 2 the openness degree of Android in different architectural layers is shown.

5.2 iPhone

In January of 2007, Apple announced iPhone at the MacWorld expo in San Francisco [28]. The iPhone OS platform was built on Mac OS X. Despite its similarity to Mac OS X, there are some technologies available only on iPhone OS such as Multi-Touch interface [29]. iPhone software platform is built on a layered architecture. Apple has published some public APIs for the iPhone that are the access points of the platform and developers can integrate them in their applications, as the iPhone application developer explains. There are also some private APIs which, if developers use them in their applications, the applications will be rejected when developers want to publish them on AppStore. Table 2 demonstrates the openness degree of different architectural layers in iPhone.

5.3 Symbian

Symbian OS is still at the highest position in the worldwide smartphone market [5]. Its creator is Symbian Limited founded in 1998 by Ericsson, Nokia, Motorola and Psion. But in 2008, Nokia acquired all the remaining shares of Symbian Limited [9]. After releasing the Android, Nokia wanted to compete with it in the openness degree [28], therefore has formed Symbian Foundation in 2008. Symbian is a layered software platform. After launching Symbian Foundation and releasing the source code of the platform, the openness degree of the platform has been risen up. But the extension of the platform is still under control of Symbian Foundation and the same as Android device manufacturers are the main beneficiaries of openness of the platform. In Table 2 the openness degree of different layers of the platforms is presented.

5.4 Blackberry

"In 1999, Research In Motion (RIM) introduced the Blackberry which started as a simple two-way pager, but quickly became one of the most widespread of mobile computing devices … The device became so widely adopted, that PC magazine ranked it the 14th most important gadget invented in the past 50 years" [28]. There is no significant information about the structure of the platform in either in literature or in official documents of the platform. So the architecture of the platform is get from the interview with a Blackberry application developer. The same as other mobile platforms, Blackberry software platform has a layered architecture. The same as iPhone, Blackberry is strongly controlled by RIM and developers are not allowed to extend or modify most components of the platform. The native applications of the platform are more open than iPhone to be integrated into the developers' applications. Table 2 shows the openness situation in different layers of the platform.

5.5 Windows Mobile

Around the same time the Blackberry was gaining traction, Microsoft released its first OS targeted at the mobile device market [28]. Microsoft licenses Windows Mobile to any mobile phone maker who is interested in launching Windows Mobile on its device [9], therefore it is not a proprietary OS. Operating system of Windows Mobile is built based on Windows CE which is an operating system developed by Microsoft handheld computers and embedded systems. So the architecture of Windows Mobile is the same as Windows CE which is a layered architecture. Developers have permission to extend or modify some components of the lower layers of the platform such as device drivers. But native applications are not allowed to be integrated, extended or modified. The openness degree of different layers of the platform is shown in Table 2.

6. ANALYSIS

The explicit results of the research process show the different openness strategies in the main platforms. In theory, the Android and Symbian platforms are almost completely open since Google and Symbian Foundation have released the whole source code of the platforms and device manufacturers, developers and users can download the source code and do everything they want with the components of each layer of the platforms. In this case even Symbian is more open since there is no limitation set by Symbian Foundation about integrating, extending or modifying of any components of the platform, where in kernel layer of the Android there is some components like power management which users are only allowed to integrate it and not extend or modify. But the situation in practice is different and the experiences of developers show that these two platforms are not as open as they seem since there are some controls governed by Google and Symbian Foundation which do not allow commercial developers to make every desired change to the platform and the target people of releasing the source code are device manufacturers like HTC. In this case Android is more open since there is less restriction set by Google for submitting an extension or modification of the platform. However, even in practice Android and Symbian are the most open platforms among current main mobile platforms, although they are not completely open. After these two platforms,

Windows Mobile and Blackberry are situated near together. There are not enough materials on the documentation of these platforms to discuss about accessibility to different layers of the platforms, but according to the experiences of the developers, Windows Mobile is more open than Blackberry since kernel layer in Blackberry is almost close and users cannot access to it directly but in Windows Mobile users even can write their own drivers for the device. But on the other hand native applications users can do more with Blackberry than Windows Mobile. In Windows Mobile, even integrate the native applications is not possible, but in Blackberry some native applications that have APIs can be used by users. The licensing situation for both platforms is almost the same and in both cases there is some situation that people can use unsigned applications. The big difference here is that Windows Mobile can be installed on different devices, but Blackberry is a proprietary software platform that is installed only on Blackberry devices. So totally the conclusion is that Windows Mobile is more open than Blackberry. And finally in the spectrum of openness, after these two platforms the iPhone is situated which the least open platform among the main mobile platforms. Although developers can integrate the kernel layer of the platform in their applications and in this sense it is more open than Blackberry, but on the other hand developers can integrate several native applications in the Blackberry which is almost not possible for any native applications of the iPhone. So in the case of accessibility of architectural layers, Blackberry and iPhone are situated nearly in the spectrum but the main thing that distinguish the iPhone from Blackberry and brings it to the end of the spectrum is licensing situation which is very controlled for the iPhone and restriction set by Apple is much more than other platforms since every application before submitting to the application store should be signed by Apple and the process of quality testing of the application is strongly controlled. Table 2 in Table 2 summarizes the comparison of openness strategy in the main mobile platforms based on the architectural aspects and licensing situation of the platforms.

This study has also some implicit results which are mainly achieved from the interviews with developers. The interviews show that most of developers, which are typically commercial developers, do not care about architectural openness of a platform. For developers the tools and languages are supported to develop applications, guides and documentation for developing and financial aspects of the platforms are more considerable. When they want to consider the architectural openness of a platform they more care about higher layers of the platform. The reason is that first of all they think even if the platform is mostly open, the lower layers provided by the platform supplier work well and they do not need to spend time to extend or modify it. The second reason is that if they modify lower layers such as components of the middleware or kernel, the application users need also to change the middleware or kernel of their devices which is not a practical work. So in their point of view increasing the openness of a platform and releasing the source code of lower layers make sense for device manufacturers who want to customize the platform for their own devices. This result brings some limitation for this study which is argued in the discussion section.

Table 2. Comparison of Openness Strategy in the Main Mobile Software Platforms

	Android		Symbian		Windows Mobile		Blackberry		iPhone	
Factor	P	L	P	L	P	L	P	L	P	L
Integrate extended applications	Pc	Pn	Pc	Ps	Pc	Ps	Pc	Ps	Pc	Pa
Extend extended applications	Pc	Pn	Pc	Ps	Pc	Ps	Pc	Ps	Pc	Pa
Modify extended applications	Pc	Pn	Pc	Ps	Pc	Ps	Pc	Ps	Pc	Pa
Integrate native applications	Po	Pn	Pc	Ps	Np		Pc	Ps	Pc	Pa
Extend native applications	Po	Pn	Po	Ps	Np		Np		Np	
Modify native applications	Po	Ps	Po	Ps	Np		Np		Np	
Integrate middleware	Po	Pn	Po	Ps	Po	Ps	Po	Ps	Po	Pa
Extend middleware	Po	Pn	Po	Ps	Pc	Ps	Np		Po	Pa
Modify middleware	Po	Ps	Po	Ps	Np		Np		Np	
Integrate kernel	Po	Pn	Po	Ps	Po	Ps	Np			Pa
Extend kernel	Pc	Pn	Po	Ps	Pc	Ps	Np		Pc	Pa
Modify kernel	Pc	Ps	Po	Ps	Pc	Ps	Np		Np	

P = Possibility, L = Licensing Status, Po = Possible, Pc = Possible for some components, Np = Not possible, Pn = Permission is not needed,

Ps = Sometimes permission is needed, Pa = Permission is always needed

7. DISCUSSION

The first limitation of this research is realized in the literature review stage. As discussed in related work part, although there is some literature compares most of studied mobile platforms of this research together, even regarding the openness of the platforms, but none of them discuss the openness strategy of the platforms based on technical aspects such as architecture of the platforms or software access points of the platform. Even literature about general software platforms do not discuss the openness of platforms based on the architectural aspects. The architectural openness model developed in this study is a tool to aim identification of the openness strategy in mobile platforms based on their architectural aspects. Despite this model is based on typical layered architecture of mobile platforms and is applicable to all main mobile platforms, but the efficiency of the model to identify the openness strategy of the mobile platforms would be confirmed by an empirical study which is not done here due to restricted time of the research. Another boundary in literature

review part is incompleteness of documents of the mobile platforms to finding out the openness strategy and access points of the platforms. It is not easy to contact technical engineers and architects of the companies like Google, Microsoft, RIM, etc. but if they accepted to be interviewed, the results of the study would be more reliable.

The second limitation of this study as shown in results of the interviews is that architectural openness of the studied platform is not important for interviewees which are commercial developers. Since they do not care about the openness especially in lower layers of the platforms, they have not tried to extend or modify the components of the lower layers. As a result they are not certain about accessibility of all layers of their favorite platform and it could affect the reliability of the results. As interviews show, device manufactures would benefit more from openness in mobile platforms since they can customize even lower layers of open platforms like the Android and Symbian and to some extend Windows Mobile for using the modified platform in their devices. So the way to improve the validity of research results would be conducting interviews with some technical engineers in device manufacturers if it is not very difficult to contact them.

8. CONCLUSIONS AND FUTURE WORK

The overall aim of this research was identification of openness strategy in mobile platforms based on the software architecture of the platforms. This section revisits the specific objectives of this research, summarizes the findings of the research, offers conclusions based on the findings, and finally suggest some recommendations for future works based on the limitations of the research work.

Research Objective 1: Building a model to describe the architectural openness of mobile platforms - Architectural openness model is built based on a typical layered architecture which is applicable to all main mobile software platforms and compromises applications layer, middleware layer and kernel layer. Applications layer itself includes two parts, extended applications and native applications. Besides the layered architecture, the model shows three way of accessibility to the platform for each layer. These ways are integrating, extending or modifying a layer.

Research Objective 2: Defining architectural openness factors by considering the licensing aspects of mobile platforms in the model - Although the model shows the platform access and extension methods in different levels, but to demonstrate the openness strategy of a platform, licensing aspects of the platform should also be considered. By considering licensing situation of mobile platforms, some factors are defined to identify openness strategy of platforms and presented in table 2. For example for kernel layer of a platform, the result of applying the openness factors can be: Modifying the kernel layer is possible for some components of the layer but it always needs permission from the platform supplier.

Research Objective 3: Looking at main mobile platforms by the lens of developed model and factors to determine how open the platforms are - The openness strategy of main mobile platforms include Android, Symbian, iPhone, Windows Mobile and Blackberry is discussed by applying the architectural openness model and factors in the architecture and licensing aspect of the platforms which are gained from the literature. The results are demonstrated by a figure for each platform. Due to insufficiency of literature and documents of the platforms, some factors needs to be confirmed by interviews.

Research Objective 4: Conducting some qualitative interviews with application developers of the platforms to confirm the results of previous step - For each platform except the Android, one qualitative interview with an application developer of the platform is conducted. The developers are asked to explain about the openness of their favorite platform based on the accessibility of each architectural layer of the platform and also licensing situation of the platform. The results show that for some layers especially lower ones, commercial developers are not sure about openness degree of the platform because they don't care about the openness in the lower layers and have not tried to extend or modify them. But about the higher layers and licensing situation of the platform the results of the interviews are valuable. In some cases the results confirmed the previous results gained from documents of the platforms, and in some cases the experience of developers are different from results of literature. So they believe that even for platforms like Android and Symbian, the openness degree in the reality is fewer than what is claimed in the theory.

Finally a comparison of openness strategy in five main mobile platforms is presented in this research which is based on the results of looking at the platforms by lens of architectural openness model and factors and evaluation of results by conducting interviews.

As argued in the discussion section insufficiency of literature and documents, lack of time to do more interviews with other application developers and interview with some engineers from devices manufacturers, restrictions on access to technical engineers in platform suppliers such as Google and Microsoft are the main limitations of this research and affect the reliability of the results. The architectural openness model and factors are valuable results of this research. It is recommended, for increasing the reliability of the model and factors, that some expert people from software architecture and software openness areas also be interviewed. For improving the validation of openness strategy of mobile platforms, interview with technical engineers of mobile device manufactures is also recommended.

9. ACKNOWLEDGMENTS

Our thanks to mobile application developers for their contribution in the qualitative interviews. Their efforts have influenced positively the quality of the results of this research and it was impossible without their support.

10. REFERENCES

[1] Paulson, J.W., Succi, G., and Eberlein, A. *An Empirical Study of Open-Source and Closed-Source Software Products, IEEE* TRANSACTIONS ON SOFTWARE ENGINEERING, 30, 4 (2004), 246-256.

[2] Maxwell, E. *Open Standards, Open Source and Open Innovation: Harnessing the Benefits of Openness,* Innovations: Technology, Governance, Globalization 1, 3 (2006), 119-176.

[3] Cho, Y.C. and Jeon, J.W. *Current Software Platforms on Mobile Phone*, International Conference on Control, Automation and Systems. Seoul (2007), 1862-1867.

[4] Jansen, S., Finkelstein, A., and Brinkkemper, S. *Business network management as a survival strategy: A tale of two software ecosystems.* In Proceedings of the First Workshop on Software Ecosystems. CEUR–WS, vol. 505, (2009)

[5] *Canalys research release 2010*, Available at: http://www.canalys.com/pr/2010/r2010021.html, Last retrieved: 2010-04-21.

[6] Bass, L., Clements, P. and Kazman, R. *Software Architecture in Practice,* Second Edition. Addison-Wesley, Boston, (2003), p 21.

[7] Bryman, A. and Bell, E. *Business Research Methods,* Oxford: Oxford University Press, (2007), p 104.

[8] Burnard, P. *A Method of Analysing Interview Transcripts in Qualitative Research,* Nurse Education Today 11 (1991), 461-466

[9] Lin, F. and Ye, W. *Operating System Battle in the Ecosystem of Smartphone Industry,* In Proc. of 2009 International Symposium on Information Engineering and Electronic Commerce, (2009), 617-621.

[10] Cho, Y.C. and Jeon, J.W. *Current Software Platforms on Mobile Phone*, International Conference on Control, Automation and Systems. Seoul, (2007), 1862-1867.

[11] Yamakami, T. *Foundation-based Mobile Platform Software Engineering: Implications to Convergence to Open Source Software,* ACM International Conference Proceeding Series; Vol. 403, In Proc. of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human, (2009), 206-211.

[12] Constantinou, A. *Mapping open source into mobile: who, where and how*, VisionMobile Ltd., available via: http://www.visionmobile.com/blog/2008/12/mapping-open-source-into-mobile-who-where-and-how/, Last retrieved: 2010-02-19

[13] Nakagawa, E. Y., Souza, E. P. M., Murata, K. B., Andery, G. F., Morelli, L. B., Maldonado, J. C. *Software Architecture Relevance in Open Source Software Evolution: A Case Study,* In Proc. of Annual IEEE International Computer Software and Applications Conference, (2008), 1234-1239

[14] Prehn, S. *Open Source Software Development Process,* Term Paper in AG Software Engineering Seminar SS07 (2007).

[15] Arief, B. Gacek, C. and Lawrie, T. *Software Architectures and Open Source Software – Where can Research Leverage the Most?*, In 1st Workshop on Open Source Software Engineering:Making Sense of the Bazaar (part of the 23rd ICSE, 2001) 3–5.

[16] Oxford English dictionary – *openness definition*, Available at: http://dictionary.oed.com/cgi/entry/00332458?single=1&query_type=word&queryword=openness&first=1&max_to_show=10, Last retrieved: 2010-04-07.

[17] Alspaugh, T.A., Asunction, H.U. and Scacchi, W. *Analyzing Software Licenses in Open Architecture Software Systems,* In FLOSS '09: Proceedings of the 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development, (2009), 54–57.

[18] Klatt, B. *Software Extension Mechanisms*, Fakultt fr Informatik,. Karlsruhe, Germany, Interner Bericht, (2008), Available: http://www.bar54.de/benjamin-klatt-software-extension-mechanism.pdf, Last retrieved: 2010-04-23.

[19] Jansen, S., Brinkkemper, S., Hunink, I., Demir, C. *Pragmatic and Opportunistic Reuse in Innovative Start-Up Companies*, IEEE Software, 25, 6 (2008), 42-49.

[20] Verkasalo, H. *Open Mobile Platforms, Modeling the Long-Tail of Application Usage,* In Proc. of Fourth International Conference on Internet and Web Applications and Services, (2009), 112-118.

[21] Alexandrov, A.D., Ibel, M., Schauser, K.E. and Scheiman, C.J. *Extending the operating system at the user level: the ufo global file system*, In 1997 Annual Technical Conference On Unix and Advanced Computing Systems (USENIX' 97).

[22] Buschmann F., Meunier R., Rohnert H., Sommerlad, P., and Stal, M. *Pattern-Oriented Software Architecture: A System of Patterns*, John Wiley & Sons, (1996), p 31.

[23] Chen, J. *An Introduction to Android,* Available at: http://sites.google.com/site/io/an-introduction-to-android, Last retrieved: 2010-04-09.

[24] *Live from Google I/O – Android: Integrate, Replace and Extend*, Available at:

http://mobilementalism.com/2008/05/28/live-from-google-io-android-integrate-replace-and-extend/, Last retrieved: 2010-04-09.

[25] Sim, N., Turnbull, R. and Walker, M.D. *Open devices — their role in supporting converged services,* BT Technology Journal, 24, 2 (2006), 200-204.

[26] *Open Handset Alliance – Overview*, Available at: http://www.openhandsetalliance.com/oha_overview.html, Last retrieved: 2010-04-09.

[27] Childers, B. *Android Everywhere!,* Linux Journal, (2009), 186.

[28] Hall, S. P. Anderson, E. *Operating Systems for Mobile Computing.* Consortium for Computing Sciences in Colleges, USA, (2009).

[29] *iPhone OS Overview*, In iPhone OS Reference Library, Available at:

ttp://developer.apple.com/iphone/library/referencelibrary/GettingStarted/URL_iPhone_OS_Overview/index.html, Last retrieved: 2010-04-13.