Orchestrate Your Platform: Architectural Challenges for Different Types of Ecosystems for Mobile Devices

Herman Hartmann[1], Jan Bosch[2]

[1]University of Groningen, Groningen, The Netherlands
j.h.hartmann@rug.nl
[2]Chalmers University of Technology, Gothenburg, Sweden
jan@janbosch.com

Abstract. The introduction of smartphones and tablets has led to a fast growing industry in which most firms have started an ecosystem-centric approach. In this paper three types of ecosystems are identified: Vertically integrated hardware/software platforms, closed source software platforms and open source software platforms. These ecosystems differ in the scope of the platform, i.e. covering both hardware and software, and the technology design, i.e. whether the software can be altered by the complementors. In this paper the challenges for each type of ecosystems are identified from an architectural point of view. Platform leaders can use our analysis to orchestrate their platform by proactively addressing the challenges that we identify and properly evolving the scope and technology design of their platforms.

Keywords: Platform leadership, mobile ecosystems, consumer electronics, software architectures, embedded systems.

1. Introduction

The last decade has seen a revolution in the consumer electronics industry in the wake of the introduction of smartphones and tablets. This industry consists of hardware and software vendors, handset makers and application developers [1]. Furthermore, smartphones have caused a renewed interest in tablets that are often replacing personal computers as the consumers preferred computing platform.

Initially this industry was dominated by a handful of firms that developed the hardware and software and created the handsets, e.g. Nokia, Siemens, Motorola and RIM. In a later stadium newcomers, such as Apple, entered the market because hardware platforms became commodities. Existing software companies entered the market, e.g. Microsoft and Google, and created a software platform that acts as a spanning layer between the applications and the hardware platform [2].

Due to the increasing development effort, many firms have adopted an ecosystem centric approach [3]. In this paper we identify the three types of ecosystems that have emerged. These ecosystems differ in the scope of the platform and whether the complementors can alter and contribute to the platform. We analyze these types with the framework of Gawer and Cusumano [4] by using a number of factors that influence the optimal scope and technology design. These factors cover general competitive criteria, design challenges that originate from the nature of embedded consumer electronic devices and factors that originate from the use of ecosystems. For each type of ecosystem we identify which factors are challenging, from an architectural point of view, and describe how the actors mitigate these challenges.

The key contributions of this paper are:

- Three types of ecosystems are identified and the factors that determine the optimal scope and technology design of the platform.
- Each type of ecosystem is evaluated and the factors are identified which are challenging for each type of ecosystem, from an architectural point of view.
- Case studies are presented that show how the actors address these challenges.
- An analysis of the market share is presented with future scenarios.

Platform leaders can use our analysis to orchestrate their platform by proactively addressing the challenges that we identify and properly evolving the scope and technology design of their platforms. The research in this paper is both descriptive, since it provides a classification of the existing situation, and explanatory because it describes the conditions under which the different types of ecosystem prosper.

The remainder of this paper is structured as follows: Section 2 provides background and Section 3 describes the type of ecosystems. Section 4 presents the factors and the evaluation for each type of ecosystem. In section 5 cases studies are presented, while section 6 gives a historical perspective and presents future scenarios. This paper ends with a comparison with related art in Section 7 and by our conclusions and recommendations for further research in section 8.

2. Background

Smartphones and tablets have characteristics of embedded devices as well as general purpose computing devices [1]. As an embedded device they are aimed to perform dedicated functions, such as making telephone calls, recording videos and playing music, which have real-time performance requirements. As a mobile device they have constrained computing resources and should use as little energy as possible because they operate on batteries and have few options for heat dissipation. These devices also have characteristics of general purpose computing devices because mobile phones and tablets are meant for a variety of tasks such as Internet browsing, reading and writing documents, accessing social media and playing games.

As an example look at Flash, an Internet browser plugin for watching videos. Apple has not allowed the use of Flash on their devices because it causes a significant shortened battery life [5], but has promoted the use of the H.264 video encoding standard for which their device is optimized. This example shows the conflicting requirements between a dedicated embedded device and a general purpose device.

The speed of innovation occurred at a high pace and the commercial lifetime of a product is often less than two years and therefore do not require backwards compatibility. A firm that is lagging behind may lose a market share rapidly.

2.1 System Architecture

In figure 1 a high level architecture of smartphones and tablets is presented, showing some of the actors. The architecture consists of a hardware platform an OS-kernel and middleware that offers a framework for application developers [6].

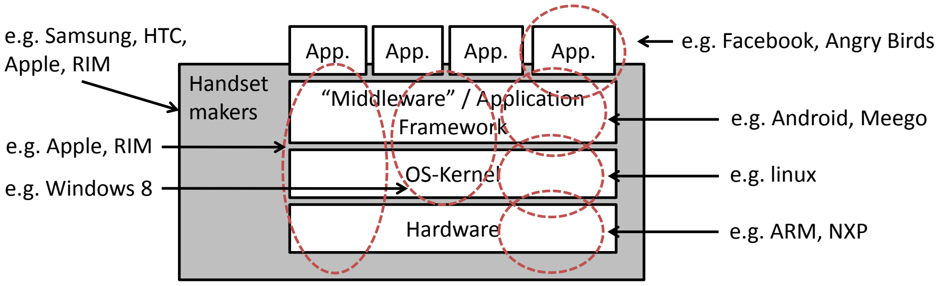

Fig 1. System architecture of mobile phones and tablets with some actors

The hardware layer consists of System on Chips (SoCs) with peripheral Integrated Circuits (ICs). A SoC contains several dedicated building blocks on a single IC such as audio decoding, memory and a configured CPU. A dedicated SoC, which is common for embedded systems, is better able to meet the performance requirements at lower power consumption than using a general purpose CPU and separate ICs, as in the PC industry. The design of the software and the SoC are tightly coupled, e.g. for controlling the power consumption of separate hardware building blocks.

3. Growing Software Size: Move towards Ecosystems

In an earlier paper we analyzed the transition in the mobile phone industry from 1990 until 2010 [1]. Initially the mobile phones were developed by a small group of vertically integrated companies that developed the hardware, firmware and applications as well as created the handsets, e.g. Nokia, Siemens, Ericsson and Motorola.

Smartphones introduced the need for dedicated application processors, operating systems and software applications. The development investments, both for hardware and software, became significantly higher and therefore the companies needed to focus their activities. The mobile phone companies made their software platforms available to external parties. In this way these organizations transitioned from software product line engineering in an intra-organizational context to software ecosystems [3].

Separate hardware platform suppliers were created as spin-offs of the vertically integrated companies, such as Qualcomm [7] and Freescale. Some of these suppliers focused on a hardware platform only while others also integrated software platforms on these platforms. When hardware platforms and other components become commodities it was possible for newcomers to enter the market without large investments. Examples of these newcomers are TCL, HTC and Apple.

Several software platforms entered the market and firms were aiming to create a common software platform in an attempt to replicate the Wintel model and create an ecosystem [8]. Similarly as the case with Microsoft Windows, a firm controlling the spanning layer can earn most of the revenues since it controls the interface towards the hardware as well as towards the applications [10, 2]. The number of competing mobile platforms that entered the market was huge, e.g. WebOS, Android, LiMo, Symbian, Windows Mobile, MeeGo etc. Until 2009 most of these attempts didn't gain industry wide adoption and therefore the industry was highly fragmented.

Starting in 2010 many of the vertically integrated firms lost market share and Android, an open source software platform, gained a share rapidly. Furthermore the dominant player in the PC market, Microsoft, increased their efforts to gain a dominant position. As a result in 2013 there is a fierce battle between different ecosystems.

3.1 Classification of ecosystems with their complementors

This paper focusses on the ecosystems that are centered around the software platforms and the vertically integrated firms. For a comparison of these ecosystems we use a classification that is based on two properties: (1) is the software proprietary or open source, and (2) are the hardware and handset included, see figure 2.

	Proprietary closed Source	Open Source
Software Platform only	**(2)** Windows WebOS (2010-2012)	**(3)** Android, Tizen, Firefox OS
Hardware and handset included	**(1)** Apple, RIM Nokia (< 2013)	**(4)** Not present in the market

Fig 2. Classification of ecosystem types

This classification results in four possible ecosystems:

1: Vertically integrated proprietary hardware/software platform. This platform consists of the hardware, proprietary closed source software and includes the handset. The complementors are the App developers. Examples of platforms and their leaders are Apple with the IPhone, Rim with Blackberry and Nokia with Asha.

2: Proprietary, closed source software platform. This platform consists of a proprietary closed source software platform. The complementors are the suppliers of hardware platforms, system integrators, handset makers and app developers. Examples of such platforms are Windows Phone and WebOS (2010-2012).

3: Open source software platform. This platform consists of an open source software platform. The open source software platform is based on the concept that multiple parties can contribute to share development effort and since the source is open, the handset makers can change, add or remove functionality. The complementors are the suppliers of hardware platforms, system integrators, handset makers and app developers. Examples are Android, Tizen and Firefox OS.

4: Open source software and hardware platform. This platform would consist on an open source software platform including the hardware and handset. The users of such a platform, or better handset, would be able to change the code of the software platform and add their own functionality. We have excluded this type from this paper as this type, to our knowledge, is not available in the market.

Note that in the past some handset makers used an open source platform of which they acted as the orchestrator or platform leader, e.g. Nokia that used Symbian when it was open source and Motorola of Google that used Android. In this situation the handset makers are in-house complementors of the open source software platform and therefore the combination is not a separate type of ecosystem.

4. Defining the Platform Scope

Gawer and Cusumano designed a framework for a company to become a platform leader. A platform leader is driving the innovation in the industry, ensures the integrity of the overall system and creates opportunities for firms that create complementary products [4]. This framework includes (1) how a company defines the scope of the platform, i.e. what part of the product is developed in-house and what it leaves for complementary firms, (2) the technology design, i.e. to what degree the platform should be open to the complementors [12]. The scope and technology design highly determine the success of a platform. It should support the capabilities of the platform leader, allow for complementors to contribute and be able to respond to a changing environment. Therefore the scope and design of a platform evolve over time [4].

4.1 Factors that influence the optimal scope of the platform

For evaluating the scope and technology design, we use the following factors, each of which will be elaborated in more detail in the remainder of this section:

Cost, Quality, Flexibility, Innovation and **Variability**: These factors originate from the analyses of Bolwijn and Kumpe [13], who identified the main competitive forces that determine the success of a consumer electronics firms. In a later analysis Sheate [14] identified variability as an additional criterion, related to flexibility, since individual users increasingly have specific demands.

Efficient use of system resources and **Hard real time requirements:** These factors originate from the nature of embedded systems and consumer products [15].

Stability of the interface and **Effort for system integration**: These factors are specific for ecosystems since complementors have to rely on a stable and, ideally, backwards compatible interface [2], and for the set-makers the integration of the hardware and software is a substantial proportion of the development effort.

Other factors that determine the success of an ecosystem, such as business models and actions [16, 12] are not covered since this paper focusses on architectural aspects.

4.2 Evaluation of the factors

In this section we analyze each type of eco-system using the factors from the previous section. Here we will "score" each factor on whether it is particular challenging or easier to address. Here + means: easier to address, 0 means: neutral, and - means: particularly challenging. The "scores" do not represent absolute values, but are solely used to express the differences between the types of ecosystems. This analysis is based on the experience of the authors, the information in literature and the case studies that are presented in section 5.

A: Costs. The larger the scope of the platform, the more development and maintenance effort is required by the platform owner. When multiple parties are involved these costs can be shared between the participants [17]. Because hardware and software development is costly, this is the main reason why ecosystems became widely adopted. Firms that aim for low costs specialize on a few products so that tasks become routine [13]. Consequently products that are developed in consort by specialized firms, especially when interfaces are pre-defined, can be made at lower costs.

For the vertically integrated type this factor is particularly challenging since they develop the entire platform including both hardware and software and can only amortize their investments over their own handsets (hence score = -). The open source platform can more easily address this, especially when the complementors contribute to the platform and often individual developers develop code in their free time [12] (hence score = +). For the proprietary software platform we evaluate this as neutral (score = 0): Although this platform owner has to develop the platform on its own, the costs can be amortized over multiple products from different handset makers.

B: Control over Quality. The overall product quality depends on the combination of software from the different contributors and failures often occur because of component interaction, unclearly documented Application Programming Interfaces (API) or unknown use of the system [18, 19]. A firm that controls a wide scope of the architecture can guarantee the product quality more easily. In the situation where multiple firms are involved, and especially when the interfaces are not clearly defined, the quality can easily break down and externally developed code could access data in the system, causing malfunctioning or security problems. Furthermore, applications developed by complementors may not follow the UI standards, as set by the platform owner, thereby causing a reduced user experience [20].

The quality can be controlled easier by the vertically integrated platform type as they have full control over the end-product. Furthermore these firms are able to test the externally developed applications on their handsets (hence score = +). For the proprietary platforms we evaluate this as neutral (score = 0): although they have no control over the hardware, they can control the API and, because of its closed nature, can easier avoid that code with defects is added to the platform. For the open source platform we also regard this as neutral (score = 0). Because of its open nature, faulty code can be added or code can be changed incorrectly by the complementors and applications may compromise the security. An advantage however, is that the software is tested by a variety of firms and open source developers.

C: Flexibility and Variability: Customers increasingly want the software and the handset that serves their specific needs. For instance users have different requirements for the product, e.g. the size of the screen, a keyboard or prefer a low cost device. Specialized firms can often add this variability to the platform more easily [21], e.g. because they have the required knowledge, can reuse existing hardware and software components and they can have a more intimate contact to the end-users [3].

Flexibility and variability is easier to achieve through the open source platform type, since complementors can add or remove functionality without the need to involve the platform owner (hence score = +). For the proprietary platform type we evaluate this as neutral (score = 0): The handset makers can create differentiating products with different hardware configuration; however this is limited to that which is supported by the software platform. As a comparison look at the PC industry where the different OEM suppliers of a Windows based PC can only compete on price, service and hardware quality, since the functionality is large determined by the proprietary software platform. For the vertically integrated platform owner it is far more difficult to cover a wide range of products. In order for a firm to be flexible and respond quickly, it needs to focus on a number of core activities [13] and the development of all the required hardware and software components would simply be too costly. We therefore evaluate that this as challenging for this type (score = -).

D: Freedom to innovate: The optimal definition of the boundaries depends on where the major innovation steps in the architecture take place. When innovation takes place across the boundaries of the platform the integrity of the platform is compromised and the complementors need to be involved thus slowing down the speed of innovation [2]. Therefore, a wide scope allows for larger innovation steps more easily.

The introduction of multi touch screens is such an example. Due to this innovation specialized hardware was needed; the interface towards the user has changed and a new API towards the application developers was required. Such a large innovation step couldn't be done through small changes to an existing platform.

Large innovation steps are easier to establish by the vertically integrated platform type (score = +) because these firms control the entire architecture and complementors do not need to be involved. In the proprietary platform type the innovation is restricted because hardware is not part of the scope and the platform supplier has to involve hardware suppliers and handset makers for major steps (hence score = -). For the open source platform the hardware is also not part of the scope, however, the complementors may change the code and thus has the possibility for innovations, independently from the platform owner, for which the architecture does not have to be changed. Therefore we evaluate this as neutral (score = 0).

E: Efficient use of system resources and hard real time requirements: Due to the need for optimal resource utilization and low power consumption a direct control of the hardware is required. Furthermore, embedded devices have hard real time requirements, e.g. for audio playback and telephone conversations. An embedded device usually contains a System on Chip (SoC) and each component is controlled separately. For instance, for audio playback a separate building block of the SoC is used which can operate with low power consumption and is only active when needed. Furthermore, by controlling each part of the IC separately it can be avoided that processes interrupt each other. Therefore the design the hardware and software are developed in parallel and require close co-operation [6].

As a comparison, look at the Wintel framework; the dominant ecosystem of Windows and Intel in the PC industry [11]. This is a modular architecture with stable interfaces between the hardware and the software layer. Both Microsoft and Intel can independently innovate on their part of the architecture. Such a modular interface is possible because most demands of the end user can easily be met by existing hardware. For mobile phones such a modular interface is not yet possible since there are still large innovation steps that involve changes to hardware and software together and a modular architecture would lead to less efficient resource utilization [22].

Since the vertically integrated platform type has both control of the hardware and software this can controlled more easily (score = +). In the open source type the complementors can also change the code to accommodate the hardware and vice-versa and therefore this can also be controlled (hence score = +). For a proprietary platform type this is particularly challenging since changes to hardware may require changes to the proprietary platform and vice versa (hence score = -).

F: Stability of the Interface: The success of a platform relies on attracting 3rd party application developers [20]. It is important for a platform to maintain a (sufficiently) stable API, thus avoiding that interoperability problems exist where applications do not work on the variety of devices based on different versions of a platform. This fragmentation is seen as the major challenge by the application developers [23].

Vertically integrated platforms can more easily avoid fragmentation since they have full control over the API (hence score = +). This also holds for the closed source proprietary platform, similarly as the case in the PC industry, which has proven to be the major advantages and success of this type of platform [21] (hence score = +). For an open source platform this is challenging since fragmentation can easily occur because handset makers can change the API or the hardware (hence score = -).

G: Effort for system integration: The time needed to integrate the hardware and software components is taking an increasing amount of time [24]. This is especially the case when components from different parties are used, e.g. with different technologies, non-matching interfaces or heterogeneous architectures [15]. When a modular architecture exists with stable interfaces, the integration time would be substantially less. Furthermore, when multiple parties are involved, the time for interaction, definition of requirements etc., is forming a substantial part of the development effort.

For the vertically integrated platform type the integration is easier to manage since no complementors are involved and the integration is done with in-house development for which the interfaces can be defined (hence score = +). For the proprietary platform type this is particularly challenging since this may require that code has to be altered or glue components have to be developed [25] (hence score = -). In the open source platform type the handset makers or the system integrators can perform the hardware/software integration without (intensive) support of the platform supplier. Therefore we evaluate this as neutral (score = 0).

4.3 Overview of ecosystems and their challenges

Table 1. Overview of types of ecosystems and their challenges

Type / Challenge	A: Costs	B: Quality	C: Variability	D: Innovation	E: System resources	F: Interface Stability	G: Integration effort
1:Vertically Integrated	-	+	-	+	+	+	+
2:Proprietary Software	0	0	0	-	-	+	-
3:Open Source Software	+	0	+	0	+	-	0
In this overview: + means: easier to address, 0 means: neutral, - means: particularly challenging							

The overview in Table 1 shows that for the vertically integrated platform type most factors are easier to address. However, variability is more difficult to achieve and that it has to bear all the development costs.

In the open source software type the complementors can contribute to the platform thus offering better possibilities for flexibility and variability. Furthermore, this type gives the handset makers and system integrators the possibility to tailor the software for specific hardware thus being able to optimize on the system resources. However, fragmentation of the platform can occur more easily.

In the proprietary software platform it is particularly challenging to create innovative products and products with optimal system resources. The latter is a major challenge since a smartphone is an embedded device and a consumer electronics product.

5. Case Studies

This section contains case studies that cover the three types of ecosystems and show how each company addresses the challenges that are described in the previous section. These case studies are based on publicly available information.

5.1 Vertically integrated hardware/software platforms

Nokia was a vertically integrated firm with the largest market share until 2008. The products were renowned for its quality and long battery lifetime. Nokia offered a large variety of feature and smartphones.

To share the development effort with other handset makers, Nokia's adopted and supported the Symbian platform. The Symbian platform was initially a closed-source collaborative platform of a small group of players and later turned into an open source platform. However, in this ecosystem there was not a strong platform owner, but a divided leadership [8]. This led to a fragmented platform with different branches and poorly documented APIs and therefore many 3rd party developers abandoned the platform [9]. Although Nokia, being a vertically integrated player, had the possibilities to innovate, they insufficiently took advantage of this because the Symbian platform required too much development effort and many contributors of the platform later opted for Android [8].

The market share of Nokia dropped rapidly since 2008. In 2010 an alliance was created with Microsoft, but no large market share was regained.

RIM entered the market in 1999 with the BlackBerry, one of the first smartphones. Being vertically integrated they could create a differentiating and innovative product, by combing functionality, implemented both though hardware and software, from a PDA, pager and mobile phone. This product was attracted by the professional market because of its functionality and services.

In the years that followed this differentiating functionality became part of products of the competitors: the touch screens took over the QWERTY keyboards and WhatsApp created an alternative for RIMs messaging service. RIM continued to offer a wide variety of handsets for which the development costs could not be amortized with a small market share which has reduced gradually over the last few years.

Apple entered the market in 2007 when hardware components in the mobile domain become commodities, e.g. the first two types of the IPhone used a SoC that was developed by Samsung. The product immediately attracted many customers. Since

Apple was already leading in MP3 players and had an online store they could offer the users functionality that went beyond the smartphones that were available.

A strong focus is on better performance and power consumption; especially the latter was a major concern for the earlier handsets. Since they had little experience with developing SoCs and, rather than using a complementor, acquired SoC design firms [26] and other hardware design firms that develop new technologies [27].

Apple uses the advantages of being a vertically integrated player, by focusing on usability and quality, which was considered better than that of the competitors, especially in the early years, thus creating a loyal group of buyers [28]. Furthermore Apple has a strict control over the applications that are offered through their app store by validating each application, prior to making them available in the app store.

Apple avoids development costs and fragmentation by offering only a small amount of variants, by using the same platform for the IPhone, IPod Touch and the IPad, and by restricting the backwards compatibility to two generations of products. Because Apple only develops a small range of variants, their market share is under pressure as many consumers that are attracted to a product with different features, such as a large screen, or a lower cost handset. However, Apple's does not aim for a large market share, but rather focuses on a small set of products for which they can obtain high profit margins [29].

5.2 Proprietary, Closed Source Software Platforms

Microsoft became active in hand held devices in the early 2000s through pocket-PCs and PDA's . At that time there were no other firms that offered a software platform for 3rd parties so the handset makers were willing to pay the license fees.

With each new version of a platform additional hardware is supported, e.g. different screen resolutions and new application processors. Here we see a contrast with the open source approaches where most additions are done by the complementors. To reduce the integration effort Microsoft started a close cooperation with a small number of hardware platform suppliers and integrators. In 2009 Microsoft closely cooperated with HTC and 80% of the Windows Mobile phones were based on a platform of HTC [30]. In 2010 a close alliance was formed with Nokia to create the hardware platform and handsets. This combination allows for a close co-operation between the hard- and software development that, as shown in the previous section, is required for embedded products and for the speed of innovation. In 2013 Windows acquired Nokia.

Microsoft has created one platform for tablets and PCs and many components are shared with smartphones so that applications can be ported to smartphones as well. Given their dominant position in the PC industry this gives a competitive advantage, but at the same time causes risks because the speed on innovation for smartphones and tablets occurs at a much higher pace then for PCs, for which a stable API is preferred rather than a high degree of innovation.

5.3 Open Source Software Platforms

Google with Android provides a spanning layer that consists of an open source application framework and base applications and uses the open source Linux operating system kernel. In this way Google entered the market with little investments but, since it offers a spanning layer, could become a platform leader. In Android some proprietary Apps are included, such as searching and YouTube, to ensure revenues.

Linux is a real time operating system that is widely used in embedded systems. It offers the developers the possibility to optimize on system resources [31] and supports hard real time requirements [12]. The handset makers were quickly interested in the platform since it offered an alternative for a proprietary platform for which license fees had to be paid [32] and the functionality that Google offered, e.g. Google Maps, attracted many users. Because of the separation of the middleware layer the OS Kernel, the handset makers can make changes to the source code, e.g. to support different screen size, cameras and other ICs. Consequently a large variety of handsets could be developed, resulting in a wide range with different functionality and price settings.

Although Android is an open source system, the main contributions to the source code are done by Google. This gives Google more freedom to innovate because they do not aim for wide consensus. New versions of the platform are offered frequently without focusing on backwards compatibility.

To increase the innovation, Google has created an alliance with Samsung [33]. Samsung develops a wide range of consumer products and develops and manufactures its own hardware. Samsung was able to control the hardware architecture and create innovative products for different market segments. Therefore the speed of innovation and quality of many Android products was possible because an existing large, vertically integrated consumer electronics firm acted as complementor.

In 2011 Google acquired Motorola [34], which was operated by Google as an independent company, to have a complementor that can bring Google's innovation to the market. However in 2014, when already having a large market share was obtained, Motorola was sold again [35]. Only for first-of-a-kind products, such as Google Glass, both the hardware and software are developed by Google.

Fragmentation of the Android platform is seen as a major concern by the developers [23]. As an example, look at the game "Angry Birds" that was initially introduced for the iPhone and following its success developed for other platforms. When developing a version for Android the developers ran into performance problems of different devices. As a result the game was not available on a large group of devices, i.e. devices with different hardware or based on different versions [36]. Google is taking measures to avoid fragmentation by defining a set of baseline compatibility standards. To have a better control over the quality, in recent versions Android also incorporates stricter security control [37] and provides a compatibility test suite [38]. Furthermore Google is making agreements with handset makers to use the standard UI and set of Googles Apps, which are more consistent of that of the complementors, by adding tighter conditions for the use of the App store by the handsets [39,35].

Meamo/Tizen, Firefox OS, Ubuntu OS, Sailfish OS Over the last few years several other attempts have been made to create an open source platform ecosystem to replicate the success of Android, all of which are based on Linux. An example is Tizen which was born out of MeeGo, which was itself was combination of Moblin and Maemo. None of these initiatives created an attractive alternative because there was no firm taking a role as orchestrator. Firefox OS, also based on Linux, aims for a better support of the increasing amount of HTML5 websites through a more direct control of the hardware [40]. However, Firefox OS also offers an API for native Apps since HTML5 doesn't support the required functionality, especially for games.

In 2013 Samsung abandoned Bada, their proprietary operating system, to contribute to Tizen [41]. When Samsung would replace Android by Tizen for all their products, this could shift the balance for this type of ecosystem.

5.4 Summary of the case studies and the mitigation strategies

The table below shows how the key players, i.e. the platform leader with the largest market share of each ecosystem type, mitigate the challenges as described in Table 1.

Table 2. Mitigation actions on the main challenges

Ecosystem type and main challenges	**Mitigation actions of the key player**
(1) Vertically Integrated: Costs & Variability	Apple limits the variability and development costs by focusing on a particular market segments and it only supports the latest versions of the handsets.
(2) Proprietary Software: Innovation, System resources & Integration effort	Microsoft closely cooperates with a small group of HW platform suppliers and recently acquired Nokia, a previously vertically integrated firm.
(3) Open Source Software: Interface stability & Fragmentation	A compatibility test suite is offered with baseline compatibility standards. Agreements are made with the handset makers to use standard Google Apps and User Interface.

6. Historical Perspective and Scenarios for the Future

The table below shows the market share of the three types of platforms. This shows that the vertically integrated platforms initially had the largest market share, but in later years the open source platform.

Table 3. Market share of different ecosystem types for smartphones

	2007	2008	2009	2010	2011	2012	2013
Vertically integrated hardware/software platforms	87 %	88 %	87 %	73 %	51 %	31 %	18 %
Proprietary, Closed Source Software Platforms	13 %	12 %	9 %	4 %	2 %	3 %	3 %
Open source software platform	0 %	0 %	4 %	23 %	47 %	66 %	79 %
Total sales in millions of units	122	139	172	298	471	675	967

(Note: This overview is for smartphones only, since for tablets insufficient data was available. This overview is compiled from reports from Gartner [42]).

This change in market share can be explained because in the early years the vertically integrated firms were able to innovate faster and the end-users were willing to pay for new functionality. When the innovation slowed down, the open source software platforms gained market share because handsets were offered with more variability and with a lower price.

When the degree of innovation further slows down, the computing requirements may meet the user's demands, allowing for a modular architecture to be created that provides a clear separation between the software and hardware layer [43]. The advantage of such a modular architecture is that integration will become easier which will especially benefit the proprietary software platform types. Therefore this type of ecosystem might gain market share, as was the case with the PC industry in the 80s [11]. An important uncertainty is whether tablets will replace PCs in a business environment. When this happens, the main supplier, i.e. Microsoft, has a strategic advantage since they are dominant in the PC market.

When PCs and tablets remain consumer devices and the speed of innovation does not allow for a more modular architecture, the types of ecosystems that are now dominant are likely to keep their dominant position.

7. Comparison with Related Art

In our previous work [1] we analyzed the transition of mobile devices from 1990 until 2010 and described a model for different industry structures. The current paper analyses the situation of today. Some of the forces that have caused the transition are used in the current paper as factors to compare the three types of ecosystems.

Another previous work of the co-author [20] describes architectural challenges. Some of these challenges have been used in the current paper. However in that work the challenges have not been identified for mobile devices or embedded systems.

The work of Gawer and Cusumano [4, 12] discusses the scope and technology design of a platform but does not discuss architectural implications nor does this work identify or compare different types of ecosystems for mobile devices.

Other related work identified different types of ecosystems [3, 16], but this work did not compare ecosystems in a particular domain nor did it identify factors that determine the challenges from architectural perspective.

Several predictions on market shares have been done by industry analysts, e.g. by the Gartner group [42], but none of these used architectural aspects as a basis.

8. Conclusions and Further Research

In this paper we identified three different types of ecosystems for smartphones and tablets and identified the factors that determine the optimal scope and technology design. Based on this analysis we described the challenges for each type of ecosystem from an architectural point of view.

The case studies demonstrated how different firms act on the factors identified in this paper: Apple takes over existing hardware design firms and focusses on a limited variety of products. Microsoft, who did not have a hardware department nor created an end product, took over an existing hardware firm to be able to act as a vertically integrated HW/SW ecosystem type. Google, who is faced with a high degree of fragmentation, is offering a compatibility test suite and makes agreements with the handset makers to use standard Google Apps and User interface.

Our analysis shows that when the speed of innovation slowed down, most of the vertically integrated firms were not able to amortize the development investments and an open source platform obtained the largest market share, which, in comparison with the closed proprietary platform, allows more flexibility and variability and efficient use of the system resources. In this paper we predicted that when the market requires a more stable interface, e.g. when smartphones and tablets need to support business applications, the proprietary platforms can obtain a larger market share.

We think that the types of platforms that are identified in this paper and the factors that determine the scope, is applicable to the wider domain of consumer electronics and embedded systems, but further research is necessary.

The analysis in this paper is based on information in literature and the experience of the authors. Further research, e.g. with multiple experts, may give a more precise comparison and may reveal more relevant factors.

The results of this paper can be used by the platform owners to reconsider the scope of the platform, e.g. by expanding the scope to areas where more control over the quality is preferred, or to publish part of the platform as open source so that handset makers can add variability more easily. On this topic further research is needed.

The analysis used in this paper can provide to the academic community a lens to evaluate the strategies in the market of mobile devices from an ecosystem and architectural perspective.

9. References

1 H. Hartmann, T. Trew, J. Bosch: “The changing industry structure of software development for consumer electronics and its consequences for software architectures”. The Journal of Systems & Software 85 (2012), pp. 178-192

2 D. G. Messerschmitt, C. Szyperski: “Software Ecosystem“, MIT Press, 2004

3 J. Bosch: “From Software Product Lines to Software Ecosystems”, SPLC2009, Sheridan.

4 A. Gawer, M. Cusumano: “Platform leadership”. Harvard Business School, 2002.

5 S. Jobs: “Thought on Flash”, http://www.apple.com/hotnews/thoughts-on-flash/, April 2010, accessed November 24 2011.

6 M. Halasz,” Menu expands at the OS diner”, Timesys Corporation, February 14, 2011

7 http://www.qualcomm.com/solutions/operating-systems, accessed January 2014.

8 J. West, D. Wood: "Tradeoffs of Open Innovation Platform Leadership: The Rise and Fall of Symbian Ltd.", Social Science and Technology Seminar Series 2010-2011.

9 D. Gilson: “The History of Symbian's Secret Fragmentation” http://www.allaboutsymbian.com/, March 12th, 2012.

10 C.Y. Baldwin, K. B. Clark. “Managing in the Age of Modularity,” Harvard Business Review 1997, Sept/Oct: 81-93.

11 A.S Grove: “Only the Paranoid Survive: How to exploit the crisis points that challenge every company and career”. Currency Doubleday, October 1996.

12 A. Gawer, M. Cusumano: “‘How companies become platform leaders’’, MIT Sloan Management Review, 2008

13 P.T. Bolwijn, T. and Kumpe: “Manufacturing in the 1990s—Productivity, flexibility and innovation”. Long Range Planning, 23 (4). pp. 44-57. ISSN 0024-6301

14 W.R. Sheate: "Tools, Techniques and Approaches for Sustainability", World Scientific Publishing Company (September 30, 2010)
15 T. Henzinger, J. Sifakis. " The embedded systems design challenge", Proceedings of the 14th International Symposium on Formal Methods (FM), August, 2006.
16 K. Popp, R. Meyer: "Profit from Software Ecosystems", Books on Demand, 2010
17 M. van Genuchten: "The Impact of Software Growth on the Electronics Industry". IEEE Computer, 2007
18 T. Trew, G. Soepenberg, "Identifying Technical Risks in Third-Party Software for Embedded Products", Fifth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, 2006.
19 R. Oberhauser, R. Schmidt: "Improving the Integration of the Software Supply Chain via the Semantic Web", International Conference on Software Engineering Advances, 2007.
20 J. Bosch: "Architecture challenges for software ecosystems", ECSA 2010: 93-95
21 J. F. Moore: "Businesses ecosystems and the view from the firm": The Antitrust Bulletin, March 2006, American Antitrust Institute, 2006.
22 C. Christensen, M. Verlinden, G. Westerman:"Disruption, disintegration and the dissipation of differentiability", Industrial and Corporate Change 2002, Volume 11, pp 955-993
23 B. Gadhavi, K. Shah:"Analysis of the emerging android market", San Jose State, 2010.
24 M. Underseth: "Verifying Embedded Software Supply Chains", EETimes, April 2007.
25 H. Hartmann, M. Keren, A. Matsinger, J. Rubin, T. Trew, T. Yatzkar-Haham: "Using MDA for integration of heterogeneous components in software supply chains", Science of Computer Programming, Volume 78, Issue 12, 1 December 2013, Pages 2313–2330
26 http://www.eetimes.com/document.asp?doc_id=1168401, retrieved December 24 2013.
27 http://www.reuters.com/article/2013/11/25/us-primesense-offer-apple-dUSBRE9AO04C20131125, accessed January 2014.
28 C. Jones:"Apple Vs. Samsung: Who Could Win The Smartphone War?", Forbes, August 20 2013
29 S. Grobart: "Apple Chiefs Discuss Strategy, Market Share and the New iPhones". http://www.businessweek.com, September 19, 2013.
30 Tricia Duryee: "We Learned Just How Great Of A Partner HTC Is To Microsoft", Feb. 17, 2009, http://paidcontent.org/, accessed January 2014.
31 W. Weinburg: "Time and Space: Optimizing Boot Up and Footprint in Linux –Based CE Devices", Open Source Development Lab, 2006
32 S. Hansel: "Microsoft, Google and the Bear", New York times, October 26 2009.
33 http://www.samsung.com/us/news/3475, accessed January 2014.
34 http://www.google.com/press/motorola/, accessed January 2014.
35 http://www.uswitch.com/mobiles/news/2014/01/samsung_and_google_thrash_out_deal_over_android_bloatware/, accessed January 2014.
36 http://www.techhive.com/article/211152/angry_birds_devs_angry_at_android_fragmentation.html, accessed January 2014.
37 Sierraware: "Android: Is It Secure Enough?", http://www.sierraware.com/SE_Android_Security_Integrity_Management.pdf
38 Android compatibility test suite, http://source.android.com/compatibility/cts-intro.html, accessed January 2014.
39 L. Gannes, I. Fried:"After Google Pressure, Samsung Will Dial Back Android Tweaks, Homegrown Apps", January 29 2014, http:// http://recode.net/2014/01/29.
40 http://www.mozilla.org/en-US/firefox/os/
41 S. Byford: "Samsung finally folding Bada OS into Tizen", http://www.theverge.com/, February 25 2013.
42 Gartner reports on smartphone market shares 2007 – 2013, e.g. http://www.gartner.com/newsroom/id/2623415, accessed January 2014.
43 C. Christensen, S. Anthony, E. Roth:"Seeing what's next", Harvard Business School, 2004.