

DIPLOMARBEIT

Titel der Diplomarbeit

„Concept of a distribution and infrastructure model for
mobile applications development across multiple
mobile platforms“

Verfasserin / Verfasser

Peter Bacher

Angestrebter akademischer Grad

Magister der Sozial- und Wirtschaftswissenschaften
(Mag. rer. soc. oec.)

Wien, 2011

Studienkennzahl lt. Studienblatt:

A 157

Studienrichtung lt. Studienblatt:

Diplomstudium Internationale Betriebswirtschaft

Betreuer/:

Univ.-Prof. Dr. Dimitris Karagiannis

Erklärung

Hiermit versichere ich, die vorliegende Diplomarbeit ohne Hilfe Dritter und nur mir den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Wien, 05.12.2011




Table of Contents

Table of Contents.....	3
1 Introduction.....	10
1.1 Motivation	10
1.2 General problem statement	11
1.3 Goal of the Thesis.....	12
1.4 Approach	12
2 Historical development of mobile telecommunication	16
2.1 The history of the mobile phone.....	16
2.2 The history of telecommunication networks.....	18
2.2.1 First Generation	18
2.2.2 Second Generation	18
2.2.3 Generation 2.5G.....	19
2.2.4 3G	19
2.2.5 3.5G	20
2.2.6 4G	21
3 Characteristics of the mobile phone market.....	23
3.1 Classification of operating systems for mobile devices.....	24
3.1.1 Overview of Operating Systems and Languages	24
4 Characteristics of the mobile application market.....	26
4.1 Mobile Application Ecosystem	26
4.2 Differentiation of mobile native applications, mobile widgets, mobile browsers applications.....	27
4.2.1 Mobile Widgets.....	28
4.2.2 Mobile Native Applications	30
4.2.3 Mobile Browser applications.....	32

4.3	Application Store Definition.....	33
4.4	Distribution channels for mobile application.....	33
4.5	Numeric overview of application stores:.....	37
4.6	Business Models.....	39
4.6.1	Pay Per Download Model.....	39
4.6.2	Advertising Model.....	39
4.6.3	Freemium Model	39
4.6.4	Data Usage Model.....	40
4.6.5	Subscription Model.....	40
5	Developer Communities	41
5.1	JIL — Joint Innovation Lab	41
5.2	BONDI from the Open Mobile Terminal Platform.....	42
5.3	Wholesale Application Community WAC	43
5.3.1	Widget Definition WAC.....	44
5.3.2	WAC Technologies.....	44
5.3.3	Key Milestones of WAC.....	45
5.3.4	WAC Development Environment.....	45
5.4	Conclusion	47
6	Distribution Model	49
6.1	Approach	49
6.1.1	BOC Management Office	50
6.2	Meta Model Apps Distribution	51
6.2.1	Cross Platform distribution	52
6.3	Platform Distribution Classification	54
6.3.1	Enabler Platform	54
6.3.2	System Integrator Platform.....	55

6.3.3	Neutral Platform	56
6.3.4	Broker Platform	57
6.3.5	Telco centric model	59
6.3.6	Device centric model	59
6.3.7	Aggregator centric model	60
6.4	Distribution characteristics	61
6.4.1	Mobile application development tools.....	61
6.4.2	Mobile application portals.....	62
6.4.3	Device set	62
6.4.4	Platform integration	63
6.5	Aggregated Distribution Model – Concept Criteria Analysis	63
6.6	Aggregated Distribution Model - Business Process Analysis.....	74
6.7	Infrastructure Analysis	94
6.7.1	Approach.....	94
6.7.2	Open Model Infrastructure Concepts.....	94
6.7.3	Aggregated Infrastructure Model.....	100
7	Multiple Platform Application Development	104
7.1	Platform Approach	104
7.2	SWOT Analysis of native vs. alternative application development.....	105
7.2.1	Software development platforms.....	106
7.2.2	Portable Applications.....	109
8	Perfect Egg - Application implementation	113
8.1	Installation – Developer Environment	113
8.2	Platform requirements	113
8.2.1	WAC SDK	114
8.2.2	Eclipse.....	114

8.2.3	Typical widget structure.....	115
8.2.4	ID Attribute	115
8.2.5	Perfect Egg – An application description.....	116
8.2.6	Development and source code.....	117
9	Conclusion.....	122
10	List of Figures	124
11	List of Tables.....	128
12	Bibliography	129
13	Appendix.....	135
13.1	Appendix 1: Mobile Operating Systems Detail.....	135
13.1.1	Android	135
13.1.2	Blackberry OS.....	135
13.1.3	iPhone OS.....	136
13.1.4	Symbian.....	136
13.1.5	Windows Phone 7	137
13.2	Appendix 2: Business Process Models.....	138
13.2.1	Apple Appstore	138
13.2.2	Android Marketplace	148
13.2.3	Wholesale Application Community.....	157
13.3	Appendix 3.....	166
13.3.1	Abstract.....	166
13.3.2	Zusammenfassung	168
13.3.3	Lebenslauf	170

Abbreviations

1G	First Generation
2G	Second Generation
3G	Third Generation
4G	Fourth Generation
AAPT	Android Asset Packaging Tool
ADB	Android Debug Bridge
ADC	Android Developer Challenge
ADOit	IT Architecture- & Service Management Toolkit
ADONIS	Business Process Management Toolkit
ADT	Android Development Toolkit
AIDL	Android Interface Description Language
AMC	Adaptive Modulation and Coding
AMOLED	Active Matrix Organic Light Emitting Diode
AMPS	Advanced Mobile Phone Service
AMPS	Advanced Mobile Phone System
Android	Google's Open Source mobile Operating System
ANT	Another Neat Tool
API	Application programming interface
APK	Android Package
AWT	Abstract Window Toolkit
BPMN	Business Process Modelling Notation
BPMS	Business Process Management software
BREW	Binary Runtime Environment for Wireless
CATT	Chinese Academy of Telecommunications Technology
CDC	Connected Device Configuration
CDMA	Code division multiple access
CDR	CDR
CLDC	Connected Limited Device Configuration
CobiT	IT governance framework and toolset
CSS	Cascading Style Sheets
CVS	Concurrent Versions System
Dalvik VM	Dalvik Virtual Machine
DDMS	Dalvik Debug Monitor Service
DOM	Document Object Model
DRM	Digital Rights Management
DSP	Digital Signal Processing
DX	Dalvik Cross-Compiler
E-BPMS	Business Process Management Systems for e-Business Applications
ECMAScript	JavaScript
EDGE	Enhanced Data rates for GSM Evolution
EPC	Evolved Packet Core
ERM	Enterprise risk management
eTOM	Enhanced Telecom Operations Map
FOMA	Freedom of Mobile Multimedia Access
GPRS	General Packet Radio Service

GPS	Global Positioning System
GSM	Global System for Mobile Communications
GSMA OneAPI	GSM Association's ONE Application programming interface
GUI	Graphical user interface
GUI	Graphical User Interface
HARQ	Hybrid Automatic Request
HSCSD	High-speed circuit-switched data
HSDPA	High-Speed Downlink Packet Access
HTML	Hypertext Mark-up Language
HTTP	Hypertext Transfer Protocol
IDC	International Data Corporation
IDE	Integrated Development Environment
IDE	Integrated development environment
IDL	Interface Description Language
IMT-A	International Mobile Telecommunications - Advanced
iPhone	Apple Inc. Mobile Device
ISO	International Organization for Standardization
ITIL	Information Technology Infrastructure Library
JAD	Java Application Descriptor
JAR	Java Archive
Java EE	Java Platform, Enterprise Edition
Java ME	Java Platform, Micro Edition
Java SE	Java Platform, Standard Edition
JIL	Joint Innovation Lab
JSP	Java Server Pages
JVM	Java Virtual Machine
KVM	Kilobyte Virtual Machine
LCD	Liquid Crystal Display
LOVEM	Line of Visibility Enterprise Modelling
LTE	Long Term Evolution network
LTE-A	LTE-Advanced
MAC	Macintosh
MIDP	Mobile Information Device Profile
MIMO	Multiple-Input Multiple-Output
MNO	Mobile network operators
MTJ	Eclipse Mobile Tools for Java
NGOSS	New Generation Operations Systems and Software
NMT	Nordic Mobile Telephone
OFDM	Orthogonal frequency-division multiplexing
OHA	Open Handset Alliance
OMA	Open Mobile Alliance
OMTP	Open Mobile Terminal Platform
OS	Operating System
OTA	Over The Air
OVI	Nokia Appstore
PaaS	Platform-as-a-Service

PC	Personal Computer
PDA	personal digital assistant
PDC	Personal Digital Cellular
PIM	Personal information manager
PSTN	Public Switched Telephone Network
RFID	Radio Frequency Identification
RIM	Research in Motion Inc.
RPC	Remote Procedure Call
SAX	Simple API for XML
SCOR	Supply-Chain Operations Reference-mode
SD (Memory Card)	Secure Digital (Memory Card)
SDK	Software Development Kit
SIM (Card)	Subscriber Identity Module (Card)
SMS	Short Message Service
SOX	SOX is an alternative syntax for XML
SQL	Structured Query Language
SSL	Secure Sockets Layer
SVN	Subversion
SWT	Standard Widget Toolkit
TACS	Total Access Communications System
TDMA	Time division multiple access
UML	Unified Modelling Language
UMTS	Universal Mobile Telecommunications System
UMTS-TDD	Universal Mobile Telecommunications System - time-division duplexing
USD	United States Dollar (\$)
UTF-8	8-bit Unicode Transformation Format
VM	Virtual Machine
W3C	World Wide Web Consortium
WAC	Wireless Application Community
WAP	Wireless Application Protocol
Wi-Fi	Wireless Fidelity
WiMax	Worldwide Interoperability for Microwave Access
WIN	Windows Operating System
WTK	Sun Java Wireless Toolkit for CLDC
WYSIWYG	What You See Is What You Get
XML	Extensible Mark-up Language
XSD	XML Schema Definition
XSL	Extensible Style sheet Language
XSLT	XSL Transformation

1 Introduction

1.1 Motivation

The mobile phone market and especially the mobile content industry has evolved massively over the last years from the early days of WAP portals, monophonic ringtones and JAVA games to an ever developing environment of smart phones with increasing capabilities and processor power to deliver a mobile internet experience and multimedia applications. I've been working in the telecommunication industry since 2003 on the mobile network operator side (T-Mobile / Deutsche Telekom) managing, mobile content and multimedia platforms. Since then I followed the development of the mobile and mobile content industry closely. Up until the introduction of the Apple iPhone in 2007 the market for mobile content was dominated by the mobile network operators due to their infrastructure and a strong established billing relation with their customer base. The launch of the iPhone and further the release of the iPhone SDK and the opening of their App Store has changed the classic operator centric content model to a manufacturer centric one within month. From that point onwards the race by handset manufacturers, OS developers, mobile network operators, and third party mobile content platforms started to copy the "App store" model, which was predominantly driven by the huge success of mobile application sales and downloads. A snapshot of the 2010 application market shows numerous app stores, (Wiki2) lists currently 29 official application stores not including all carrier stores on the globe, some open for all application types some with their own SDK's and programming language. It's getting increasingly difficult for developers of mobile applications to prioritise their resource in terms of which platform to focus on to reach the most customers with their application. This work provides an overview of the existing mobile application and app store market, investigating in business models, processes and infrastructures to develop and distribute mobile applications across multiple platforms.

1.2 General problem statement

The mobile application market continues to grow drastically due to the explosion in the sales of mobile device. According to (Daum, 2010), the smart phone market alone grew by 64% annually worldwide in Q2 2010 and is forecast to grow to over 576 million shipments annually by 2014. The key driving force is the increasingly popular application stores provided by handset makers and operators (Anar, et al., 2010). According to the study (ABIresearch, 2011) approximately 40% of US mobile business customers, stated that mobile data services were more important than mobile voice services. A Market research from IDC suggests that smart phones are now outselling PCs for the first time ever, whereas the transition from laptop for the usage on the road by business users moves to the usage of tablets or smart phones (Stephen Pritchard, 2011). *These findings suggest that the importance of the original use case of mobile voice services is reduced in the favour of data services. However, this large customer base is not homogeneous. There are several factors which fragment that initial customer base. These factors range from unavoidable human preferences to technological issues. While the basic needs of customers may limit the development of a mobile application, technical issues in the mobile industry pose another burden. Users who have the same needs regarding their mobile application will often operate different devices. These differing devices contain competing operating systems, development platforms, physical characteristics, and network infrastructures. This competition creates a large degree of uncertainty in the industry on a strategic, technological, and demand level for mobile developers. (Qusay H., et al., 2010). Currently developers need to ask themselves the question concerning which platform to develop and distribute. This depends on a number of factors including target market, compatibility issue, development time, hardware requirements and the desired level of scalability. There is a large literature on architectures and tools that are proposed to solve the challenges of mobile application development like the cross-platform compatibility (MING-CHUN, et al., 2007). However, the subject of cross-platform distribution is still in development stage and presents an opportunity for further research to limit the resource effort in the development stage and publishing of applications.*

1.3 Goal of the Thesis

The goal of this thesis is to find an aggregated¹ infrastructure and distribution model for mobile applications on multiple platforms by giving a comprehensive overview of the mobile phone and mobile application market by analysing the key success factor from a developer's point of view. The result should be useable as a reference for the development of future mobile content distribution systems. In this paper, the basic components of the mobile market are outlined and a better insight into the mobile application development and distribution process is provided as well as a discussion regarding business models and value chains in the mobile market.

1.4 Approach

As the goal is to find an aggregated model for the distribution of cross-platform applications I will start with a top-down approach to identify the existing distribution and infrastructure landscape, therefore I will conduct a research of the literature, internet i.e. Application store developer sites, specialized press and expert talks. The modelling of the business processes will be done with ADONIS® Business Process Management Toolkit and the modelling of infrastructures with ADOit® IT Architecture- & Service Management Toolkit. This will be followed by an analysis of these models to find common characteristics as outlined in the following graphic:

¹ In context of this thesis aggregated means the model with the maximum fulfilment of the common characteristics of cross platform distribution in the view of the developer as described in Figure 1

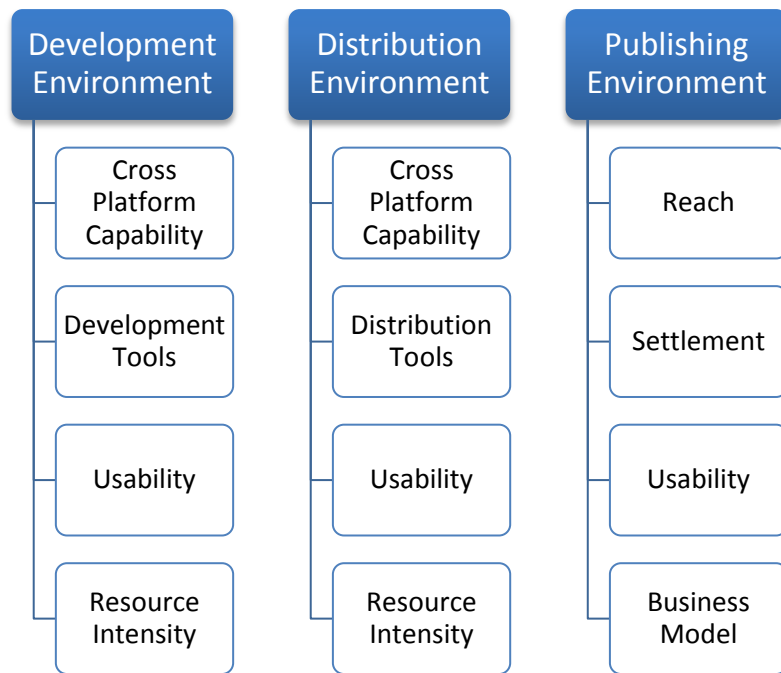


Figure 1: Common characteristics of application distribution

To determine an aggregated model a 2-way approach is executed by conducting on the one hand a research of literature and expert talks to find strengths and weaknesses with each model and preferences among the stakeholders (developers, operators, application stores) and on the other hand starting to develop the prototype application to analyse and compare the findings in the development, porting and distribution process. The usage of the developer environment and SDK is dependent on the release timings of the same. After mapping of the characteristics and outlining an aggregated model, I am going to model the business processes and the infrastructure model of an aggregated model for cross platform mobile application distribution.

This paper is organized as follows. The first section gives an overview of the mobile telecommunication history, phones and the mobile applications market. An overview of current business process models for application distribution is given in the second part by analysing the current application stores by breaking down the business processes from development to the distribution of the applications along the value chain. A comparison of common characteristic among the current application stores that would enable a cross platform distribution should be achieved by modelling the individual business processes, analyse their cross platform capabilities and connect the process to suit an outcome of an aggregated

model for the distribution across multiple platforms. Third part of the paper will give a general view on current infrastructural frameworks for cross-platform distribution and the derivation of a model based on literature and in-life architecture. Fourthly the sample application will be developed with an openly available SDK that enables cross platform distribution. The conclusion will be derived from the previous findings in this work as well as the latest industry developments and trends, and will be presented in section five.

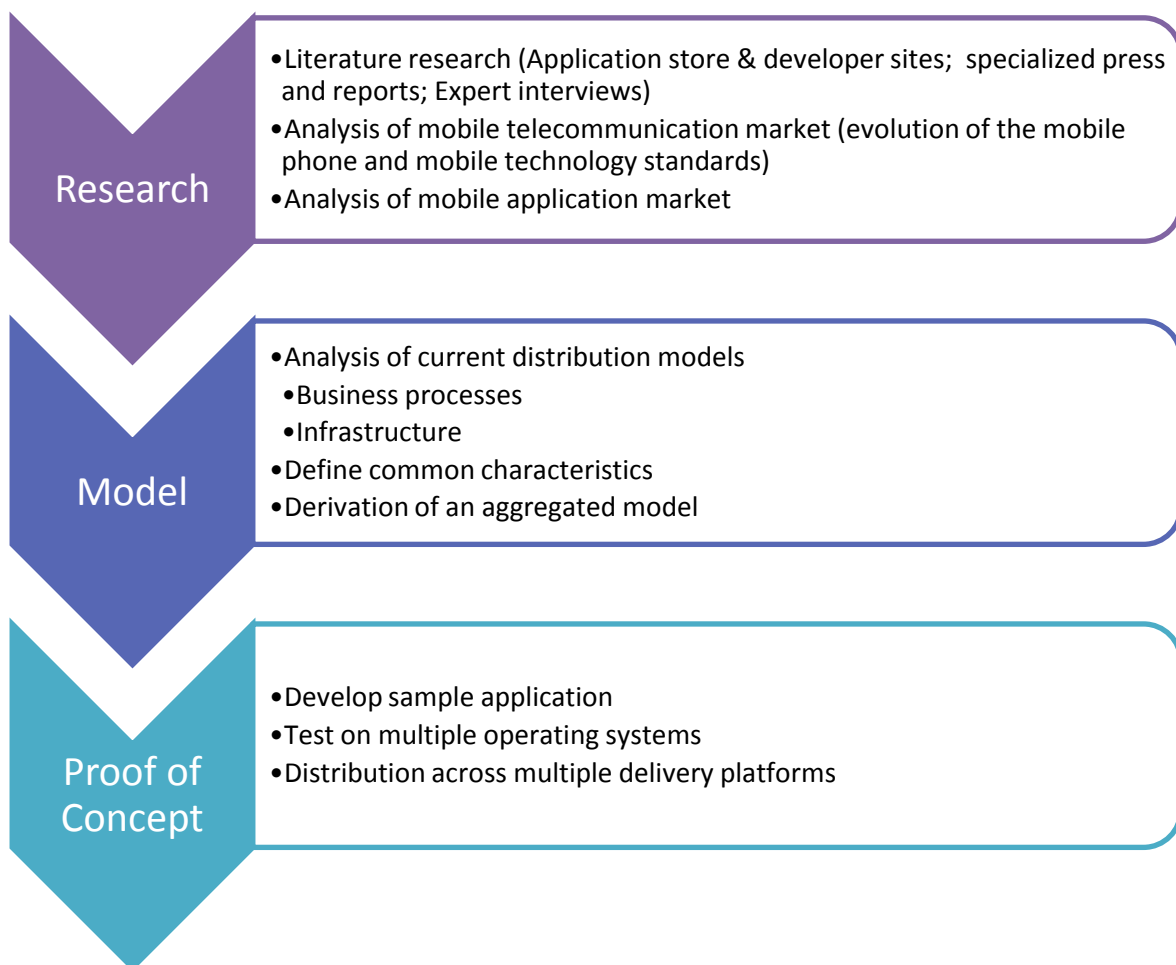


Figure 2: Structural approach of the thesis

Based on the available information and literature as well as on the importance in terms of reach and market share, the following applications stores and their environment will be analysed:

- Apple AppStore
- Android Market Place

- WAC Environment

For the analysis of their common characteristics and the suitability as cross platform distribution environment, each application store will be analysed through the modelling of their business processes in the light of their development, distribution and publishing environment.

2 Historical development of mobile telecommunication

The last few years have seen an increase of new technologies for the distribution of multimedia content like mobile applications towards consumers as constantly new mobile and wireless technologies are developed. *This is because mobile broadband evolution is not only linked to moves of various competing carriers towards mobile cellular technologies such as GPRS, EDGE, UMTS and/or HSDPA, but also to 'alternative' wireless technologies such as Wi-Fi, WiMax, UMTS-TDD or Flash OFDM, that provide mobile and nomadic users with high bandwidth wireless access*(Pieter, et al., 2006). The next chapter will in give an overview of the history of the mobile phone to the current Smartphone's as well as the network technologies used. Chapter 2 of this work should give a better understanding of the fast development of the mobile phone industry and the evolving capabilities of mobile devices to the current technology standards that enable mobile computing and the usage of mobile applications that take advantage of features of consumer electronics and computing. The development of mobile phones developed hand in hand with the performance of the telecommunication networks thus an overview of their evolution is given as well in the second part of this chapter.

2.1 The history of the mobile phone

According to Dunnewijk in (Dunnewijk, 2006) does the history of mobile phones begin with early efforts to develop mobile telephony concepts using two-way radios and continues through emergence of modern mobile phones and associated services . Radiophones have a long and varied history going back to Reginald Fessenden's invention and shore-to-ship demonstration of radio telephony, through the Second World War with military use of radio telephony links and civil services in the 1950s (Wiki1). The first worldwide mobile network was introduced by the USA in 1946 and could only be used in the USA at this time, mostly for military purposes. Not until the end of the 1950's was this technique replaced by the Analog network (A-network) (Speckmann, 2008).

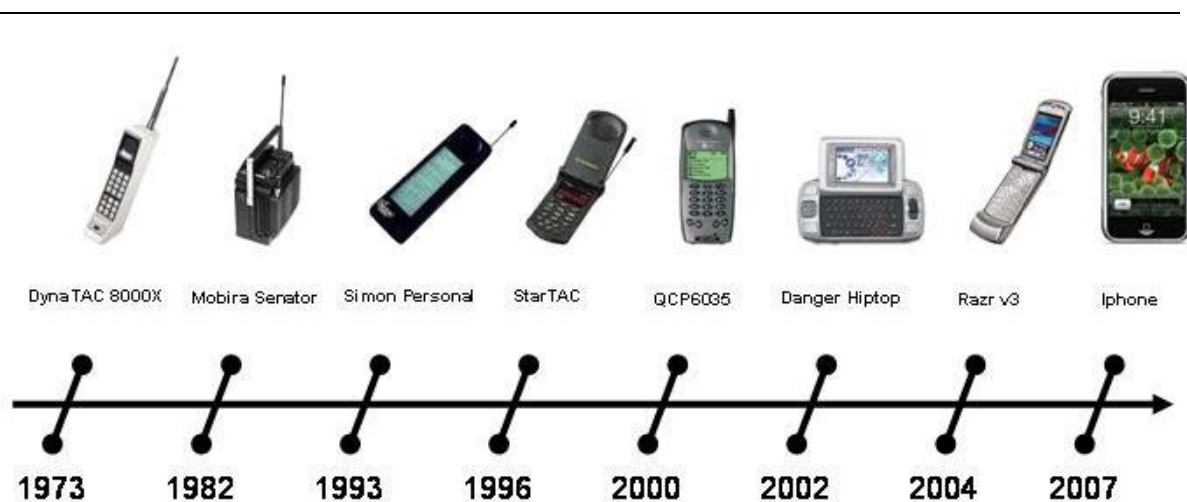


Figure 3: Historical development of cell phones (Speckmann, 2008).

Martin Cooper who was a Motorola researcher and executive, made the first analogue mobile phone call on a prototype model in 1973. This cell phone became commercially available in 1983 and provided one hour of talk time and could store 30 phone numbers. Nokia a Finnish handset maker introduced Mobira Senator in 1982 - its first Mobile phone. The first cell phone with PDA features was introduced in 1993 by Bell South/IBM. It included phone and pager functionalities, calculator and calendar applications as well as fax and e-mail capability. The weight was about 18 pounds and it sold for \$900. In 1996 Motorola launched the StarTac phone. Kyocera introduces its QCP6035 mobile phone in year 2000. It was the first widely available Palm OS based phone. In 2002 the Danger Hiptop was introduced. It was one of the first mobile devices to include a Web browser, reliable e-mail access and instant messaging. The RAZRv3 Motorola later was the first mobile device which many people wanted to have because of its. In 2007 there was an innovation presented by Apple Inc. – the iPhone - a stylish mobile phone that featured an innovative touch screen navigation interface (Morisson, 2007).

2.2 The history of telecommunication networks

Mobile telecommunication technologies have developed in successive generations. The first generation (1G) appeared in the 1950's. The second generation (2G) or GSM technology is used massively, but challenged globally by the next (third) generation (3G) technologies (Dunnewijk, 2006). This sequence of generations is characterised by increasing capacity through higher transmission speeds and bandwidth and richer content of the message. Further penetration of 3G depends critically on the integration of telecommunication services and multimedia services, like mobile applications (Dunnewijk, 2006).

2.2.1 First Generation

The first generation of mobile cellular telecommunication systems appeared in the 1980s. The first generation was not the beginning of mobile communications, as there were several mobile radio networks in existence before then, but they were not cellular systems either. In mobile cellular networks the coverage area is divided into small cells, and thus the same frequencies can be used several times in the network without disruptive interference. The first generation used analogue transmission techniques for traffic, which was almost entirely voice. There was no dominant standard but several competing ones. The most successful standards were Nordic Mobile Telephon (NMT), Total Access Communications System (TACS), and Advanced Mobile Phone Service (AMPS) (Korhonen, 2003).

2.2.2 Second Generation

The second-generation (2G) mobile cellular systems use digital radio transmission for traffic other than the first generation which used analogue transmission. 2G technologies can be divided into TDMA-based and CDMA-based standards depending on the type of multiplexing used. The main 2G standards are: Global System for Mobile (GSM) communications and its derivatives; digital AMPS (D-AMPS); code-division multiple access (CDMA) IS-95; and personal digital cellular (PDC). Originally GSM was designed as a pan-European standard but it was quickly adopted all over the world except the USA where the CDMA was in the dominant position. (Korhonen, 2003).

2.2.3 Generation 2.5G

2.5G is a step between 2G and 3G cellular wireless technologies. The term "second and a half generation" is used to describe 2G-systems that have implemented a packet switched domain in addition to the circuit switched domain. *It does not necessarily provide faster services because bundling of timeslots is used for circuit switched data services (HSCSD) as well. 2.5G provides some of the benefits of 3G (e.g. it is packet-switched) and can use some of the existing 2G infrastructure in GSM and CDMA networks. The commonly known 2.5G technique is GPRS. Some protocols, such as EDGE for GSM and CDMA2000 1x-RTT for CDMA, officially qualify as "3G" services (because they have a data rate of above 144kbps), but are considered by most to be 2.5G services (or 2.75G which sounds even more sophisticated) because they are several times slower than 3G services. 2G is the current generation of full digital mobile phone systems. It transmits primarily voice but is used for circuit-switched data service and SMS as well (Wiki1).*

2.2.4 3G

3G is short for third-generation mobile telephone technology. The services associated with 3G provide the ability to transfer both voice data (a telephone call) and non-voice data (such as downloading information, exchanging email, and instant messaging)(3GE11).

3G Standards

3G technologies are an answer to the International Telecommunications Union's IMT-2000 specification. Originally, 3G was supposed to be a single, unified, worldwide standard, but in practice, the 3G world has been split into three camps. According to (3GE11) there are the following standards of 3G networks.

UMTS (W-CDMA)

UMTS (Universal Mobile Telephone System), based on W-CDMA technology, is the solution generally preferred by countries that used GSM, centred in Europe. UMTS is managed by the 3GPP organization also responsible for GSM, GPRS and

EDGE. FOMA, launched by Japan's NTT DoCoMo in 2001, is generally regarded as the world's first commercial 3G service (3GE11).

CDMA2000

The other significant 3G standard is CDMA2000, which is an outgrowth of the earlier 2G CDMA standard IS-95. CDMA2000's primary proponents are outside the GSM zone in the Americas, Japan and Korea. CDMA2000 is managed by 3GPP2, which is separate and independent from UMTS's 3GPP (3GE11).

TD-SCDMA

A less well known standard is TD-SCDMA which was developed in the People's Republic of China by the Chinese Academy of Telecommunications Technology (CATT), Datang and Siemens AG. TD-SCDMA stands for Time Division Synchronous Code Division Multiple Access. The launch of a TD-SCDMA network was projected by 2005 but commercial trials took place only in 2008 and finally an official license was given to China Mobile in 2009. (Michael Wei, 2009)

2.2.5 3.5G

High-Speed Downlink Packet Access or HSDPA is a mobile telephony protocol. Also called 3.5G (or "3½G"). High Speed Downlink Packet Access (HSDPA) is a packet-based data service in W-CDMA downlink with data transmission up to 8-10 Mbit/s (and 20 Mbit/s for MIMO systems) over a 5MHz bandwidth in WCDMA downlink. HSDPA implementations includes Adaptive Modulation and Coding (AMC), Multiple-Input Multiple-Output (MIMO), Hybrid Automatic Request (HARQ), fast cell search, and advanced receiver design. In 3rd generation partnership project (3GPP) standards, Release 4 specifications provide efficient IP support enabling provision of services through an all-IP core network and Release 5 specifications focus on HSDPA to provide data rates up to approximately 10 Mbit/s to support packet-based multimedia services. MIMO systems are the work item in Release 6 specifications, which will support even higher data transmission rates up to 20 Mbit/s. HSDPA is evolved from and backward compatible with Release 99 WCDMA systems (Wiki1). 3G radio has evolved from WCDMA to 3.5G radio. By 2011 it is approaching to the final stage, 3.9G radio the Long Term Evolution network short LTE. The 3G core network has also evolved from the GSM circuit-

switched network/GPRS to the IP-based soft switch network/GPRS+IMS (IP Multimedia Subsystem). Now, it is approaching the All-IP network the EPC (Evolved Packet Core). In real commercial networks, the 3G network is being overlaid, not replaced, by LTE/EPC (Yabusaki, 2010).

2.2.6 4G

4G (or 4-G) is short for fourth-generation the successor of 3G and is a wireless access technology. It describes two different but overlapping ideas, which is on the one side high-speed mobile wireless access with a very high data transmission speed, of the same order of magnitude as a local area network connection (10 Mbits/s and up). It has been used to describe wireless LAN technologies like Wi-Fi, as well as other potential successors of the current 3G mobile telephone standards (Wha11).

On the other side pervasive networks are amorphous and presently entirely hypothetical concepts where the user can be simultaneously connected to several wireless access technologies and can seamlessly move between them. These access technologies can be Wi-Fi, UMTS, EDGE, LTE or any other future access technology. Included in this concept is also smart-radio technology to efficiently manage spectrum use and transmission power as well as the use of mesh routing protocols to create a pervasive network (Wiki1). ITU-R has assigned frequencies for 4G radio IMT-A (International Mobile Telecommunications - Advanced). 3GPP has started the standardization of LTE-A (LTE-Advanced) as a promising candidate for IMT-A (Yabusaki, 2010).

In terms of mobile network infrastructures, recent revolution in mobile phones came with the new mobile telephony communication protocols, i.e., 3G and 3.5G. These new protocols significantly increased the mobile network bandwidth. However, upcoming protocols such as 4G could pose a new trend in the area of mobile applications. With increasing bandwidth and more powerful and user-friendly hardware e.g., larger screen estate, Internet browsing becomes more accessible to mobile users. These evolutions in mobile technologies have renewed the interests of developers. As a result, innovative applications/services are created in order to tap on the increasingly sophisticated capabilities of mobile devices (Anar, et al., 2010).

After the description of the historical development of the mobile telecommunication in terms of devices and networks, the following chapters 3 and 4 will give an overview of the characteristics of the current mobile phone and application market by examining the dominant operating systems for mobile phones that resulted in the fragmentation of the phone market and the mobile application market.

3 Characteristics of the mobile phone market

In this chapter insights into the current mobile phone market with the focus on dominant mobile operating systems according to their importance and market share. Further I'm going to look closer into their history and their OS characteristics from a technological point of view and development environment for applications.

The evolution of the mobile phone from a bulky communication device to a form of compact powerful device has occurred within the last twenty years. This evolution, with the increasing infrastructure and decline in cost, has made mobile phones an essential component for millions around the world. The current mobile phone is more than a device for voice communication; it is a device that can truly become an entertainment centre on one end, keeping individuals happy and occupied for hours, and a productivity juggernaut on the other end, capable of granting access for individuals to large knowledge bases of information in the palm of their hands. *The ability of the mobile phone to perform these wide varieties of tasks relies on both, the hardware found on mobile devices and the creation of functional and useful applications.* (Qusay H., et al., 2010).

The main concern today of manufacturers is to provide a bigger screen, greater processing power, lighter weight and a longer battery life. The engineering limitations which they have been constantly stretching include the maximization and balancing between processing power and battery life as well as the screen and device size. The processing power is continuously enhanced, and some of the current phones are already comparable to notebooks. *Apart from improving the quality of existing components such as camera resolution and wireless range, manufacturers also incorporate new technologies into their offerings such as Radio Frequency Identification (RFID).* In addition to these features of mobile devices, another important dimension that is often highlighted in comparison websites for mobile devices is the mobile applications/services (Anar, et al., 2010)

3.1 Classification of operating systems for mobile devices

According to the Canals report (Can10), the worldwide smart phone market grew an impressive 95% in Q3 2010 over Q3 2009 to 80.9 million shipped units. Nokia despite its leadership position has lost a lot of margin, with a 33% share of the market. Apple is becoming an increasingly strong player with 17% market share, just ahead of RIM, with 15% market share. The Android platform that proved the greatest driver of growth in the worldwide market increase its market share by 1,309% year-on-year from 1.4 million in Q3 2009 to more than 20.0 million units in Q3 2010. Vendors are now delivering Android devices across a broad range of price points, from high-end products such as the Sony Ericsson, HTC or Samsung range, to lower priced LG Optimus or the Huawei devices.

Today mobile operating systems of five companies dominate the mobile application development and distribution market. These products are Nokia (Symbian), Microsoft (Windows Mobile), RIM (BlackBerry OS), Apple (iPhoneOS), and Google (Android). Each mobile OS offers a software development kit (SDK) which consists an integrated development environment (IDE), an emulator, specific libraries, and other tools like certification, or testing tools. Those environments are commonly based on platforms that are used for Personal Computer development such as xCode for Apple Mac OS or Microsoft Visual C++ for Windows. Except for some unique cases (e.g., xCode), these native development environments can also be replaced by open and generic development platforms such as Eclipse and NetBeans. (Anar, et al., 2010)

In the next section the dominant mobile operating systems are explained in terms of their development environment (see (Allan, et al., 2010) (Holzer, et al., 2010):

3.1.1 Overview of Operating Systems and Languages

As mentioned above for most of the dominating operating systems in the market, there is a native development language, which is required to develop optimally for that platform, as illustrated in Table 1. While it is possible to develop using web or generic development languages like Java or HTML, there are some limitations compared to native languages, which will be discussed in a later chapter.

Table 1 Smartphone Operating Systems and Development Environments (Kirk Knoernschild, 2010)

Platform vendor	Operating system	Supported development kits	Supported languages	Distribution restrictions
Apple	iPhone OS	iPhone SDK	Objective-C	Yes
Google	Android/Dalvik Virtual Machine (VM)	Android SDK Android Native Development Kit (NDK)	Java C/C++	No
Microsoft	Windows Mobile	Windows Mobile SDK .NET Compact Framework (CF) JavaFX Mobile	Visual C++ C# and Visual Basic .NET JavaFX Script	No
Nokia	Maemo	Maemo SDK	C/C++ Python	No
Palm	webOS	Mojo SDK	HTML, CSS, and JavaScript	No
RIM	BlackBerry OS	BlackBerry Java Development Environment (JDE)	Java	No
Symbian	Symbian OS	Flash Lite Java ME Web Runtime Widgets C/C++ Python Ruby	ActionScript Java HTML, CSS, and JavaScript C/C++ Python Ruby	No

Further details on operating systems and their history and development environments are discussed in appendix 2.

4 Characteristics of the mobile application market

In this chapter the mobile application market will be examined. Firstly the mobile application ecosystem will be discussed and then a definition of an application and an application store will be give as well as the most important application stores in the current market analysed

4.1 Mobile Application Ecosystem

Knoernschild describes in (Kirk Knoernschild, 2010) the mobile application as an ecosystem which consists of six interdependent components. All six aspects of the ecosystem influence the development and management of mobile applications:

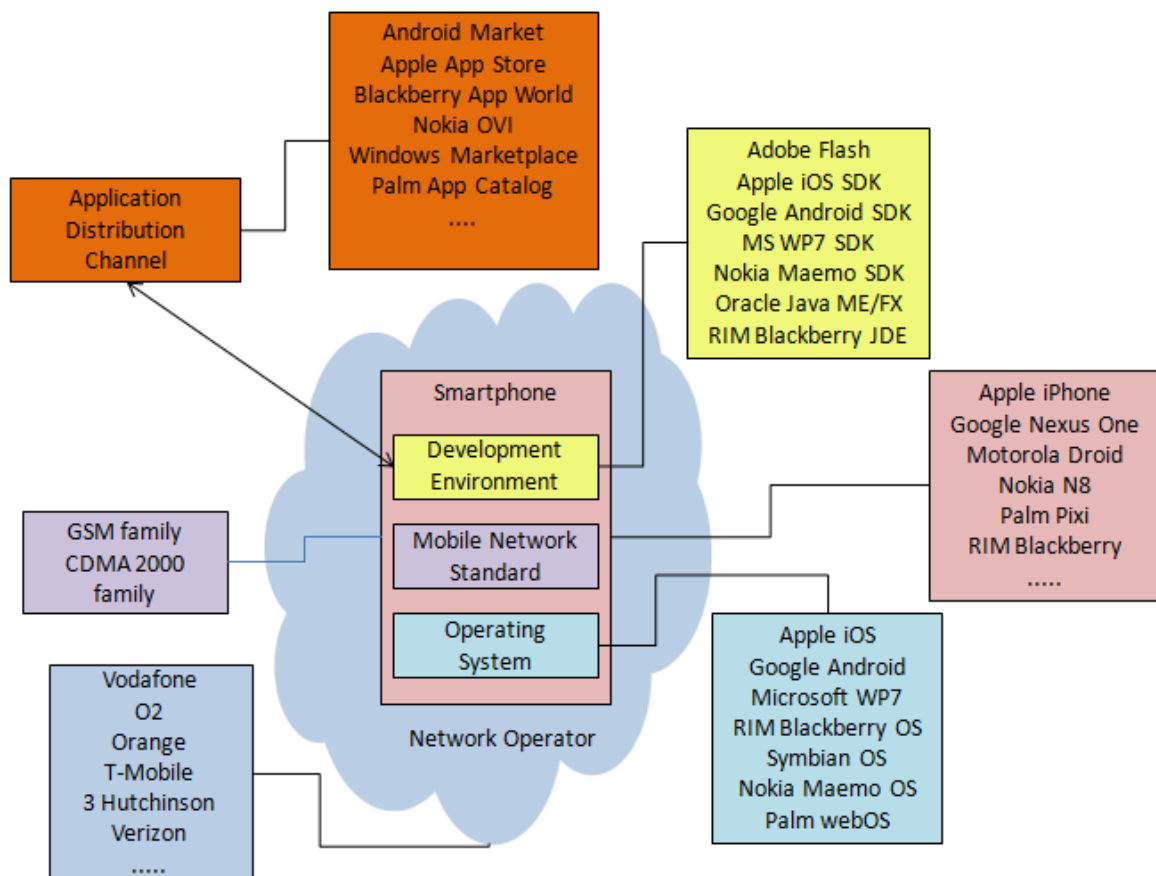


Figure 4 Mobile Application Ecosystem (compare: (Kirk Knoernschild, 2010))

Application distribution channel: Some vendors restrict application distribution to their certified distribution channel, while others are more open to external distribution

Application platform: The platform affects the programming experience and determines what features and capabilities are available to an application. Each application platform is available for a limited set of smart phones and operating systems.

Mobile OS: The system utilities and graphical user interface (GUI) framework supplied by the OS impact the look and feel of an application and the way the user and the applications interact with the phone. Each OS supports a limited set of smart phones and application platforms.

Smartphone: The form factor and processing power of the hardware device on which the application runs constrains the usability and capabilities of the application. Each device type supports a limited set of OSs, application platforms, and mobile network standards.

Mobile network standard: The network standards supported by the device limit the mobile network operators that can support the device, and they impact the geographical reach of the application and its communication bandwidth.

Mobile network operator: Operators control which devices can use their networks, and they influence the kinds of applications that may run on their networks.

4.2 Differentiation of mobile native applications, mobile widgets, mobile browsers applications

This chapter provides an overview of the different ways to develop applications for mobile phones. Research shows that there are three main categories of mobile application development approaches, 1) mobile widgets, 2) native applications and 3) browser applications (Kaar, 2007), (Cáceres, 2010) (Kirk Knoernschild, 2010) (Anar, et al., 2010).

This table gives an overview of the different development approaches:

Table 2 Comparison Table Mobile Widgets, Browser Applications, Native Applications

	Mobile Widgets	Mobile Browser Applications	Mobile Native Applications
Languages	HTML, CSS, XML JavaScript; Packaging: ZIP, WGT	HTML, CSS, XML JavaScript	Native Languages per OS usually SDK is provided (JAVA, Objective-C etc)
Accessibility	Hosted by widget engine and installed on device	Hosted by Mobile Web Browser accessed over mobile internet	Installed on device
Cross Platform Capability	Deployable across multiple platforms;	Viewable across multiple browsers	Limited within OS
User Experience	User experience not optimised for all devices; design limitations; small – task specific software	User experience not optimised for all devices; design limitations; limited to web content	User experience optimised for each OS; optimisation within product ranges of OS needed (i.e. iPhone vs. iPad)
Support of advanced hardware capabilities	Limited to common APIs – standardization ongoing	Limited to browser capabilities	Full support of hardware capabilities; image processing

4.2.1 Mobile Widgets

Widgets are a way of developing applications for mobile phones. By using mark-up and scripting languages, widget development is on a higher abstraction layer than developing with native code like Java ME or Android. This translates into the user interface design that is simplified, the development process is shorter and development costs decrease and furthermore does the similarity to web development result in a huge increase in the number of potential developers for mobile applications. Widgets are well suited for applications which access services or resources on the web. The development of such applications as widgets is

faster and more efficient than with other mobile platforms. But there are other types of applications where widgets are unsuitable, e.g. games and applications which make use of device specific functions like the camera or GPS (Kaar, 2007). W3C defined a widget in their “Widgets 1.0 Requirements” as “A widget is an interactive single purpose application for displaying and/or updating local data or data on the Web, packaged in a way to allow a single download and installation on a user's machine or mobile device. A widget may run as a stand-alone application (meaning it can run outside of a Web browser), and it is envisioned that the kind of widgets being standardized by this effort will one day be embedded into Web documents.” (Cáceres, 2010) A mobile widget is therefore a small, specialized mobile application that executes outside the browsers and provides access to the mobile internet. Widgets can provide better user experiences than a browser and are more flexible than mobile applications (Peter, et al., March 2010) in terms of their portability on different platforms.

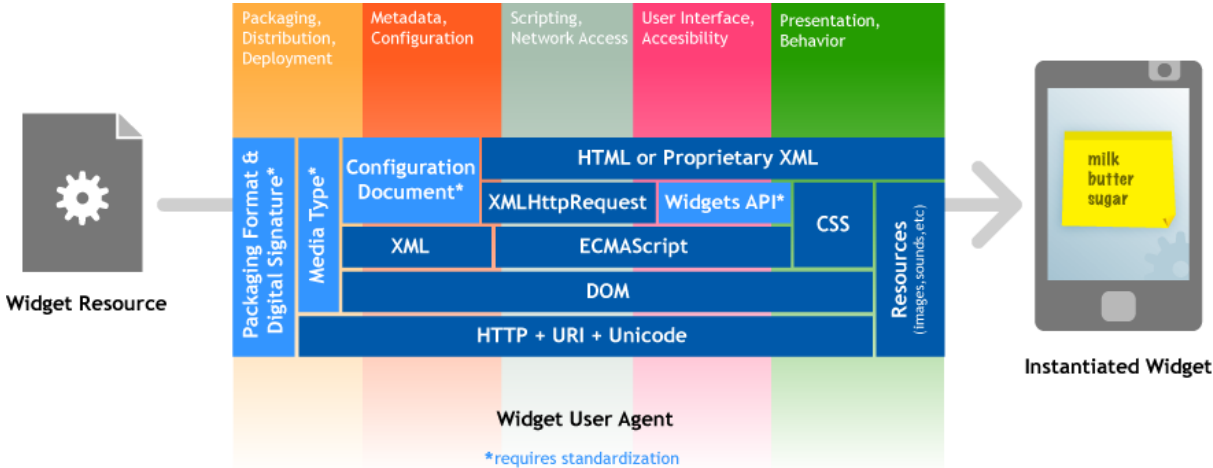


Figure 5 Widget Landscape W3C (Marcos Cáceres, 2008)

As shown in figure 2, a widget is instantiated on a widget user agent and can make use of a number of technologies that serve a different role (e.g. distribution and deployment, etc). However, some of those technologies have not yet been formally standardized (items marked with an asterisk), which has contributed to fragmentation across the widget space (Marcos Cáceres, 2008).

4.2.2 Mobile Native Applications

Knoernschild defines in (Kirk Knoernschild, 2010) mobile applications as mobile software applications that provide a rich user experience by integrating with and leveraging mobile device features, such as telephony, cameras, accelerometers, and global positioning system (GPS) chips. These mobile applications execute on the smart phone and connect using mobile phone networks and other wireless technologies like Wi-Fi.

Development and deployment of native mobile applications use a mobile application platform. These platforms are dynamic, programmable environments with sophisticated user interaction and device interface capabilities. A native platform has three components:

- Software development kit (SDK): The SDK includes tools for writing, compiling, debugging, testing, and packaging applications. Often times, a simulation environment aids testing by providing an execution environment on the desktop that mimics application execution on the device.
- Runtime environment: The runtime environment is the container in which the application executes on the device. The runtime environment may be preinstalled by the device manufacturer or mobile service provider or installed by the device owner.
- Server-side software: application platforms or stores supply server-side software that eases application distribution and device management. Some vendors also provide middleware and packaged applications that ease integration with applications.

List of mobile application categories

From its naive inception as an accessory for mobile phones, e.g., for personal gaming and music listening purposes, mobile applications further can be seen as well as a platform for social and commercial purposes. The concept of being connected “anytime, anywhere” has led to a plethora of mobile applications which target at satisfying a wide variety of requirements and use cases. According to

(Anar, et al., 2010) is the development of mobile applications is characterized by three factors: 1) maturity of the mobile network infrastructures,

2) advanced mobile hardware, and

3) increasing demand for mobile applications

There are numerous classification concepts in the literature about mobile applications reaching from their extent of connectivity, their intended usage, their involvement of billing, private or corporate etc. (Anar, et al., 2010) lists five classes of applications in his work: Transactional, Content dissemination, Social networking, Personal productivity, Leisure.

Table 3 Classification of Mobile Applications (Anar, et al., 2010)

Category	Domain	Key characteristics	Implementations
Transactional	Conversational	Short term, as-needed conversations	Nambuzz (nimbuzz.com), eBuddy (ebuddy.com), Xumii (xumii.com), Truphone (truphone.com), Fring (fring.com)
	Commercial	Information and/or monetary exchanges involving one or more parties	Digby (digby.com), Mobiqa (mobiqa.com), PicTicket (matrixsolutions.com), Luupay (luupay.de)
	Tracking and tracing	User tracking and tracing via location updates from mobile, satellite, or home location register (HLR)	Trace A Mobile (traceamobile.com), Child Locate (childlocate.co.uk)
Content dissemination	User-requested Information dissemination	Delivery of information requested by users	Google Maps (www.google.com/mobile/gmm), Yahoo Weather, Stock, Mail (mobile.yahoo.com), Shazam (shazam.com),

	Advertising	Information delivery for marketing purposes	Cellfire (cellfire.com)
	Mobile content sharing	Data exchange using Internet, WLAN or near field communications such as peer-to-peer (P2P) file sharing	Miraveo (miraveo.com), eMule (mobile.emule-project.net)
Social networking	Networking	Facilitate the forming and maintenance of social relationships, and organizing social activities	Twitter (twitter.com), Plazes (plazes.com), Facebook (m.facebook.com), Xing (mobile.xing.com),
	Entertainment	Collective-based leisure activities undertaken using mobile device	The Club (www.apple.com/store/iphone/appstore)
Personal productivity		Support work related activities to improve personal productivity	Microsoft Office Mobile, Remember The Milk (www.rememberthemilk.com) , OmniFocus (www.omnigroup.com/applications/omnifocus)
Leisure		For personal leisure activities that do not involve social exchanges with others	Pandora (pandora.com), Books (textonphone.com)

4.2.3 Mobile Browser applications

Organizations can support mobile interactions via web applications that are specially designed for access via a mobile web browser. Mobile browser applications, which run on a web server, support any device with a mobile web

browser, but they have limited access to a specific device's capabilities. Developers have a wide choice of development frameworks, including open source frameworks such as iWebKit and PhoneGap. A number of commercial mobile application vendors, such as Usablenet and Volantis, specialize in delivering Mobile browser applications.

4.3 Application Store Definition

According to the application store definition in (Copeland, 2010) an app store may be defined as a “digital facility to browse and download published applications that were developed with an SDK compatible with a terminal OS”. As this is a paraphrase of the Wikipedia definition, Copeland adds that an app store can be a distribution mechanism with no SDK as well.

The development and improvement especially of the hardware of mobile phones like the processing power, wireless network bandwidth, improve the capabilities these. Therefore *mobile devices can run rich stand-alone applications as well as distributed client–server applications that access information via web gateways. Lately, the development of mobile applications has generated more interest among the independent and freelance developer community. This has opened up new avenues for future mobile application and service development. The potential of the mobile application market is expected to reach \$9 billion by 2011, according to Compass Intelligence (Holzer, et al., 2010).*

4.4 Distribution channels for mobile application

Once a mobile application has been developed, the developer faces the task of successfully promoting their applications and thus generating revenue. The mobile application market offers several channels to distribute and market a mobile application. These channels have both benefits and drawbacks. Copeland in (Copeland, 2010) categorises four different distribution channels which are device manufacturers, vendors of mobile operating systems, independent 3rd party aggregators and major mobile operator. Qusay in (Qusay H., et al., 2010) adds to these furthermore the distribution channels, developer sites and customer. These

different distribution channels are described in the following part. Device Manufacturers

The Apple App Store is an example of a device manufacturer's mobile application distribution system. The App Store, like others, comes loaded onto the new iPhone sold to all customers and allows the mobile device user to search, find, purchase, and download mobile applications directly to their phone. In their distribution system the developer sets the price of the application, receives a revenue share, and does not pay for marketing, hosting, credit card, or charges for free applications. An up to date and exhaustive list of all application distribution platforms can be found in (Wiki2). In detail I will give an overview over the most important ones in terms of volume and influence on the mobile applications market:

Apple App Store

Apple App Store has redefined the mobile application space, with applications acquired like commodities off the shelf and revolutionised the apps market by opening up the marketplace so that anyone can be a provider or a consumer, or both. Apple opened this tantalizing proposition to developers by allowing them to earn 70% of the app sales proceeds. By June 2010, over 225,000 apps were listed on the Apple App Store, with 500 millions downloads. Using these statistics, it can be calculated that Apple receives payments for applications worth \$2.4 billion USD per year (Copeland, 2010).

RIM App World

Research in Motion the developer of the blackberry also created an application store called the Blackberry App World. It's becoming increasingly better in terms of the offer and customer experience. The number of applications and support for it's blackberry devices is improving to compete with the competitors applications stores like the Apple App Store.

Nokia OVI

Nokia has also created its own application store which is called OVI store, which is translated as "door". The OVI store can be accessed by Nokia's Symbian and Maemo devices.

Vendors of mobile operating systems

The dominant OS vendors in the market are currently Google's Android and Microsoft Mobile Windows, with the Android Market (Android1) and the Windows Marketplace respectively (Win1). They look to gain momentum via developers and to be utilized on as many handsets as possible. Software vendors profit only from the software and seek to maximize the number of devices that can utilize their OS.

Android Market

Android Market is an online software store developed by Google for Android devices. An application program called "Market" is preinstalled on most Android devices and allows users to browse and download apps published by third-party developers, hosted on Android Market (Wiki2). The Android Market was announced on 28 August 2008 and was made available to users on 22 October 2008. Developers of software receive, 70% of the application price, with the remaining 30% distributed among carriers if authorized to receive a fee for applications purchased through their network and payment processors. Revenue earned from the Android Market is paid to developers via Google Checkout merchant accounts. T-Mobile, the first carrier with an Android device, recently updated the market to allow Google to directly bill app purchases to a customer's cell phone account that show up as a charge on the bill. In keeping with the Open Handset Alliance goals of Android being the first open, complete, and free platform created specifically for mobile devices. The Android Market offers the ability for developers to create any application they choose with the community regulating whether the application is appropriate and safe, as opposed to relying on a formal screening process(Wiki2).

(Windows) Zune Marketplace

Zune Marketplace is an online store that offers music, podcasts, TV shows, movies, music videos, and mobile applications. Content can be viewed or purchased on Windows PCs with the Zune software installed, Zune devices, the Xbox 360, Windows Phone 7 phones, or the Microsoft Kin phones. The Windows Phone Marketplace was launched along with Windows Phone 7 in Oct 2010 in

some countries. It was reported on October 4, 2010 that the Windows Phone SDK has been downloaded over half a million times. The Marketplace is about to reach 4000 applications. Windows Phone Marketplace has support for credit card purchases, operator billing, and ad-supported content(Wiki2). The Marketplace also features a "try-before-you-buy" scheme, where the user has an option to download a trial or demo for a commercial app. You can download games and apps from the Windows Phone Marketplace and if you have an Xbox live account you will be able to access this remotely from your Windows Phone 7 enabled handset (Win10).

Network Operators

Infrastructure providers operate sites that are used to display, distribute, and sell mobile applications. Operators such as Vodafone, Verizon or Telefonica operate currently own mobile application portals. Carriers generally open their 'stores' only to their own subscribers, and offer apps for a range of handsets that they approve and sell in their networks.

Third-Party Distributors

Third-party distributors act as a middleman for the developer and the customer. Sites such as Handango (www.handango.com) and Getjar (www.getjar.com) distribute content on behalf of a developer. These sites handle the marketing, distribution, sales, and reporting processes of mobile applications on behalf of the developer. In exchange for promoting specific software on their site, the respective websites take a percentage of any revenue generated by the sales of the products. In case of Handango they take 40% of any revenue generated through their website (Qusay H., et al., 2010). A full list of all 3rd party application distribution platforms can be found in (Wiki2). According to (Copeland, 2010) aggregators offer a variety of apps that are not bound by a single handset, that are suited for a niche market or that have been censored by the mainstream app stores. Some of their items are considered as pirated apps for 'jailbreak' handsets that have been 'unlocked'.

Developer Sales Sites

Developer sales sites are directly controlled by the mobile application developer. The developer keeps all the revenue obtained through the distribution, sales, and marketing of their applications but also bears all the costs associated. Sites such as Gameloft or Smackall Games are an example of developers marketing their own products. This method is more appropriate for larger developers than for individual developers who may find the costs too high to set up their own websites and secure payment systems (Qusay H., et al., 2010).

Customer

The customer can be used as a distribution channel. In this case the developer can provide freeware or shareware versions of an application. Based on the value of the application to the customer, and its perceived value to their friends, customers can pass on trial versions or free versions of software. This situation may assist developer to spread their application. Customers can often utilize peer-to-peer or torrent applications to distribute mobile applications. Caution must be given when designing applications, as the proliferation of full applications by individuals could cause security issues concerning the applications themselves (Qusay H., et al., 2010).

4.5 Numeric overview of application stores:

The following table should give a numerical overview of the most important manufacturer, OS vendor and operator application stores to date (see (Wiki2); (Stanley, 2009) (Giz1).

	Device Manufacturers							OS Vendors		Network Operator
	Apple App Store	Nokia OVI	Blackberry App World	Palm App Catalog	LG Application Store	Sony Ericsson Playnow Arena	Samsung Mobile Applications	Android Market	Windows Marketplace	Verizon Vcast App Store
Launch Date	7/08	5/09	4/09	6/09	7/09	10/09	1/09	10/08	H1/10	11/09
Number of Apps (Nov/10)	300k+	28k	16k	6k	1.4k	4k	6k	170k	4k	-
Apps Downloaded (Nov/10)	7N	~3m daily	~1m daily	2.6m	-	-	-	2B	-	-
Exclusive Source	Y	N	N	N	N	N	N	N	N	N
Paid or free apps	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Phone client	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Desktop Client	Y	Y	N	-	N	N	Y	N	Y	N
Multi-Device	N*	Y	Y	Y	Y	Y	Y	Y	Y	Y
Non-app content	Y**	Y	N	N	Y	Y	Y	N	Y	Y
Billing	iTunes	CC, Carrier	Paypal	CC	CC, Carrier	CC, Carrier	CC, Paypal	Google Checkout/Carrier	CC, Carrier	CC, Carrier
Carrier Stores	N	Y	N	-	-	-	-	Y	Y	-
Developer Share	70%	70%	80%	70%	80%	-	70%	70%	70%	70%
Developer Fee	\$99 /year	\$99 /year	\$200 once	\$99 once	-	free	free	\$25 once	\$99 /year	-
App Listing	Unlimited		10 for every \$200	-	-	-	-	Unlimited	5 then \$99 each	-
Minimum Price	\$0.99	-	\$2.99	-	-	-	-	\$0.99	-	-

*except iPod, iPhone, iPad differences

**Not on AppStore but iTunes is available

Figure 6: Numeric overview of application stores

As shown in the table above, the Apple App Store is so far the most successful one in terms of the number of apps downloaded, and on second place comes the Android Market. While the Windows Marketplace for Mobile just hit the market in 2010 and can't report so far any numbers, Nokia's OVI and Blackberry App World gain momentum with their new device models hitting the market in 2010. The App Store allows developers to set their own price, which can be completely free or anything within the prescribed range (\$0.99 to \$999). All other app stores have had to match the developer revenue share (70%), and RIM/BlackBerry has even exceeded it (80%). Developers also consider the initial uploading fee and how many apps can be uploaded, apps source exclusivity and limits on apps listing. Apple lowered the bar for developers' entry to the mobile apps market and offered substantial revenue share. Developers initially flocked to the iPhone because it guarantees wide market but are now seeking to expand to Android. Since an app store is only as good as the apps on it, the power of the web developer community is on the rise.

4.6 Business Models

To complete the overview of mobile application market this chapter describes the most used business models for the monetization of applications. According to Vânia in (Vânia, et al., 2010) business models can be classified in to the following five models.

4.6.1 Pay Per Download Model

This is the standard business model where the price of the application is paid upfront before downloading and usually entitles the customer to use the application as long as it is installed on their phone. Latest application portal offer a locker functionality that enables users to re-download applications if they lost or changed their phone.

4.6.2 Advertising Model

Advertisements play an important role in generating profit. Advertisements can be displayed in a number of locations or during a number of different points in time. Usually the applications are offered as a free or “lite” version without any upfront charges to the customer. Depending on the application the banner advertisement are running within the application are displayed through a code supplied by advertising networks the run their banners across different application. The application may be offered only as free advertisement funded application (like the popular Paper Toss game) or it can be used for a certain time or with limited functionality (like Shazam free version) to encourage the users to buy the full or advertisement free version of the application.

4.6.3 Freemium Model

This play of words come from FREE and PREMIUM and is the latest trend in monetization of application predominantly used in mobile games that utilize virtual currency in the game to enhance game play. Usually these applications are downloaded free of charge to gather the interest of users and to rank in the application stores in the top downloads lists. The in-application billing is managed through so called offline billing APIs that the application used to trigger billing calls to the application portal billing server.

4.6.4 Data Usage Model

A partnership could be established with the network operators to obtain some of the revenue generated by increased usage of the network in downloading application. In this scenario, the network operator pays whenever the framework is utilized to find a model application. The latest application stores do not offer this data usage model as operators have no or limited involvement in the commercials.

4.6.5 Subscription Model

This model requires that a fee be levied on the distribution channels in order to access the service. This fee would be best suited for the infrastructure and third-party distribution channels and might be less suitable for the independent developer distribution channel.

5 Developer Communities

According to (Anar, et al., 2010) there are at the moment four companies sharing over 83% of worldwide Smartphone market, Nokia, RIM, Apple and HTC. In addition Samsung, Sony Ericsson and some other companies are becoming increasingly important. These companies could cooperate in order to accelerate a standardization process for the development of mobile application and usage of common APIs. However, the complex structure of the mobile market involving the interests of the multiple stakeholders discussed in chapter 4 prevents this process.

The convergence of technologies has led to the integration of more Internet based features onto mobile devices. However, without a set of core standards defining the necessary interfaces and security to drive web services on mobile phones, fragmentation has been evident on many new devices. The progress of these initiatives has been significant, resulting in a massive convergence of manufacturers and operators towards a reduced set of technologies and standards. The success of these initiatives really is establishing the basis for a developer to write an application which can run on many different devices and across multiple Operating Systems.

Various initiatives have been started to remove or reduce this fragmentation including Joint Innovation Lab (JIL), Open Mobile Terminal Platform's (OMTP) BONDI, the Wholesale Application Community (WAC) and on the network standardization side the GSMA OneAPI (GSM Association's ONE Application programming interface). The next section describes each initiative and their outcome towards a unified application market and their successes.

5.1 JIL — Joint Innovation Lab

It's unclear exactly when JIL begun, though this commercial joint venture under Dutch law was founded by:

China Mobile

Vodafone

Softbank

Verizon Wireless joined as well which makes them 4 large mobile operators from each continent. JIL has adopted most elements of the W3C widget P&C specification and extended it in some ways. However JIL has not contributed any specification material back to W3C discussion. Collectively their aim was to launch a mobile applications store to compete against the Apple App store and its dominance.

5.2 BONDI from the Open Mobile Terminal Platform

BONDI is an initiative from the Open Mobile Terminal Platform (OMTP) that begun around March 2008. The need for device APIs on mobile devices was identified and BONDI was founded with members largely based in Europe. Opera is the latest member of the group. Both Opera and Access are browser vendors, with a lot of mobile experience involved in BONDI. The Open Mobile Terminal Alliance has launched version one of its Web-2.0-widget platform BONDI with a reference implementation, software developer's kit and endorsements from Opera and the other Linux consortium.

BONDI is a selection of extensions to ECMAScript (the scripting language formally, and informally, known as JavaScript) to give digitally-signed scripts access to phone functions, including location, contacts, camera and messaging functions - enabling a scripted application to integrate with the phone environment in just the way that iPhone WebApps failed to do. (Bill Ray, 2009) OMTP are members of the W3C.

BONDI achievements are listed below according to (OMT11).

Interface Requirements – A high level definition of the BONDI interfaces which include a dynamic API which is remotely updateable once the device is in the field

Security and Architecture requirements – Requirements for BONDI architectural constraints and for the security policy which protects the user from harm

API specifications – A set of Doxygen generated HTML pages that define the syntax and semantics of the BONDI APIs

Security Policy DTD – An interoperable XML description of the security policy which defines the access that a particular web application and widget will have to the BONDI APIs.

Reference Implementation (RI) – The RI is a real concrete example (using Windows Mobile as the platform) of how the interfaces and security specifications should be implemented. The RI SDK contains API documentation and example code – the initial alpha release is available [here](#).

Compliance Criteria – A set of criteria which may be used to judge compliance of implementation against the defined standard and RI. (OMT11)

5.3 Wholesale Application Community WAC

The Wholesale Applications Community (WAC) is an open, global alliance formed from the world's leading telecoms operators in their attempt to get a share of the applications market that is dominated by Apple and Google. WAC aim is similar to the other initiatives before to unite the fragmented applications marketplace and create an open industry platform that should benefit the entire ecosystem, including applications developers, handset manufacturers, OS owners, network operators and end users. The Wholesale Applications Community sets out to simplify application development by giving developers the opportunity to write applications that can be deployed across multiple platforms and multiple operators, and address a potential global market of more than 3 billion users. WAC is intended to increase the scale and scope of the core standards and interfaces(WAC1).

The Open Mobile Terminal Platform (OMTP) group is already part of WAC from July 1 2010, and Joint Innovation Lab (JIL) became part of WAC by September 2010. In this way it is assured that well developed and W3C compliant technologies for mobile web widgets, BONDI and JIL, will define the foundations of the new WAC platform for cross-device mobile development worldwide and all previous learning and developed standards will be integrated into the new specifications. WAC is therefore the only existing initiative of its kind for mobile application development and distribution with the largest members list across the mobile industry so far and with the proved achievements of its members initiatives

the most advanced standardization activity for mobile applications. A full list of members and anticipated members can be found on (WAC1; Rupp, 2010)

5.3.1 Widget Definition WAC

A WAC Widget is a composition of HTML, JavaScript and CSS combined as a package that is installed on the handset. The widget package is self contained. It includes all the support files that are needed by the widget. With this approach, the widget can become a complete standalone application that does not require any external resources. Any access issue in running a widget can be avoided. The HTML is based on standard HTML 4.0. It supports a rich set of JavaScript with WAC extensions. These extensions support integration with the handset device in the form of Messaging, PIM and Device information. It supports a rich set of robust Network Resource API's. It provides widget application access of device and network resources. With these capabilities, a widget can provide access to internet based data, information, and services. A widget can also provide access to existing enterprise applications (WAC10).

WAC Widgets allows developing applications that can be used on various handsets. These applications allow providing a simple and fast interface that resides on the handset. These applications have access to the handset resources such as the Contacts, Email and SMS system as well as other functions in the WAC Widget API (WAC10).

The WAC SDK enables to create mobile widgets for the WAC enabled mobile handsets. These mobile widgets can then be loaded onto the handset. WAC Widgets are portable to run on any WAC enabled handset (WAC10).

5.3.2 WAC Technologies

The Wholesale Application Community is according to their developers site (WAC10) creating a solution that uses existing technical standards, building initially on the work from OMTP BOND, JIL and the GSMA OneAPI activities. WAC 1.0 is already available and WAC Waikiki (the next generation of WAC specifications) has already gone through a full public review process. The Wholesale Applications

Community intends to work with W3C (World Wide Web Consortium), the standards body relevant to web runtime applications, to establish the "best of breed" converged solution/device APIs, which will then be standardised. The following technologies are used for a standardised development environment:

BONDI - OTMP APIs for mobile terminals

GSMA's OneAPI standard on the network side (Rupp, 2010) GSMA OneAPI is set of APIs that expose network capabilities over HTTP. OneAPI is developed in public and based on existing Web standards and principles. Any network operator or service provider is able to implement OneAPI. (gsm11)

5.3.3 Key Milestones of WAC

According to the WAC webpage ((WAC1) the following milestones have been achieved so far:

February 2010: 24 of the world's leading operators announce the Wholesale Applications Community

July 2010: The WAC Company was formed and the Board elected. Business models for the participating companies were announced.

September 2010: The Wholesale Applications Community published materials/documents for developers.

November 2010: The Wholesale Applications Community held its first developer events.

February 2011: The Wholesale Applications Community launches commercial activity at Mobile World Congress 2011. (WAC10)

5.3.4 WAC Development Environment

I will give an overview of the WAC specifications in this section according to (WAC10).

WAC applications, also known as WAC widgets, utilize Web technologies. The widget packaging format is based on W3C Widget Packaging specification and introduces some extensions to meet WAC requirements, such as specifications for billing. WAC widgets can optionally utilize a comprehensive handset API. A code-

signing security system ensures that widgets can only access APIs that are suitable to their level of trust.

Key components of the WAC architecture include:

Widget contents, which are the various widgets that are created by the developer community.

Widget platform, which is the software that renders widgets and addresses the requirements identified in the documents detailed above.

WAC Common Web Services (or, Network Resources), which provides capabilities hosted by the operator, such as billing, account balance lookup, etc. A common protocol is defined as a recommendation for communication between the widget platform and the WAC services gateway, although operators can decide to implement a different API.

In addition to these components, WAC also provides the following components in support of the widget ecosystem:

Developer Website, which provides access to SDKs, forums, widget uploading, code signing, and other developer support services. The sign up to the developer site is currently free of charge. Nevertheless there is a charge of annual US \$99, - to obtain a Publisher ID. A Publisher ID is a digital certificate that includes the needed data used to identify the author of widgets. Developers can obtain a Publisher ID from the third party Certificate Authorities which is designated by WAC. Prior to publishing the widget, developers are required to sign their widget by using the Publisher ID if their widgets use advanced APIs. With a Publisher ID, publishers are capable of:

Getting a test and/or production certificate.

Charging for your widgets globally.

Developing widgets with Advanced APIs.

Obtaining the "Identified" security domain.

Advanced APIs are APIs defined within WAC that can be used only by an identified developer. The restricted functions are the following:

To get the account info of the device such as “phone MSISDN” and “user account balance.

To get the owner info.

To delete all messages.

To delete all call records.

Widget SDK, which provides emulators, documentation and other tools required to develop WAC widgets. WAC 1.0 SDK supports MAC, Linux, Windows and comes with a transit Widget Emulator for Firefox. Further there are sample codes and manuals like JIL Widget Engine white paper, JIL Widget System High Level Tech Spec – Format and Packaging, JIL Web Widgets Developer's Guide, JIL SDK 1.0 Getting Started, JIL SDK 1.0 Overview, JIL SDK 1.0 Revision 1 Manual, JIL SDK 1.0 API Reference Documentation available.

Reference widget platform, which is a complete widget platform that serves as a definitive reference for the proper implementation of the specifications.

WAC Signing Server, which provides code signing services to widget developers through the developer web site.

5.4 Conclusion

Originally this section should have been a comparison of developer communities whereas due to the fact that JIL and BONDI with their technologies are part of WAC a comparison is not applicable. Therefore the WAC was examined in detail and will be the only developer community to be included in the following chapters where I will look closer into the distribution and infrastructure model and the application development. Part of this thesis will be the development of a sample application with a cross platform environment with the most suitable development environment for cross platform distribution of the application. Based on the above findings that

-
- WAC is the latest standardization project for a cross platform development and
 - WAC is the only available environment at the time of writing this thesis that offers a unified application distribution through their infrastructure

the application development will be done using the WAC specification and standards. In chapter 9 I will use the WAC application development environment to test the cross platform compatibility of such environment with the implementation of the “Perfect Egg” mobile application.

6 Distribution Model

In this chapter the distribution models of the earlier discussed applications stores will be examined in terms of their business models, the related business processes for the distribution of application and an aggregated model for the distribution cross platform should be derived. Chapter 7 will then investigate the related infrastructural models behind the different distribution models.

6.1 Approach

The research methodology employed is based on a synthesis of literature and case studies related to content distribution, application development and distribution, complemented by a number of in-depth, face-to-face interviews and workshops with business and system architects of content developers, portal and mobile operators. The methodology of the E-BPMS² framework according (Bayer, et al., 2001) will be applied through a top down approach by modelling the different stages of the framework depicted in the figure below using BOC's Management Office as a basis. As the developer needs to plan resources accordingly and make the appropriate platform decisions on how to reach the most customers with his application, the examination of the business processes will be predominantly from the developer's point of view.

² Business Process Management Systems for e-Business Applications

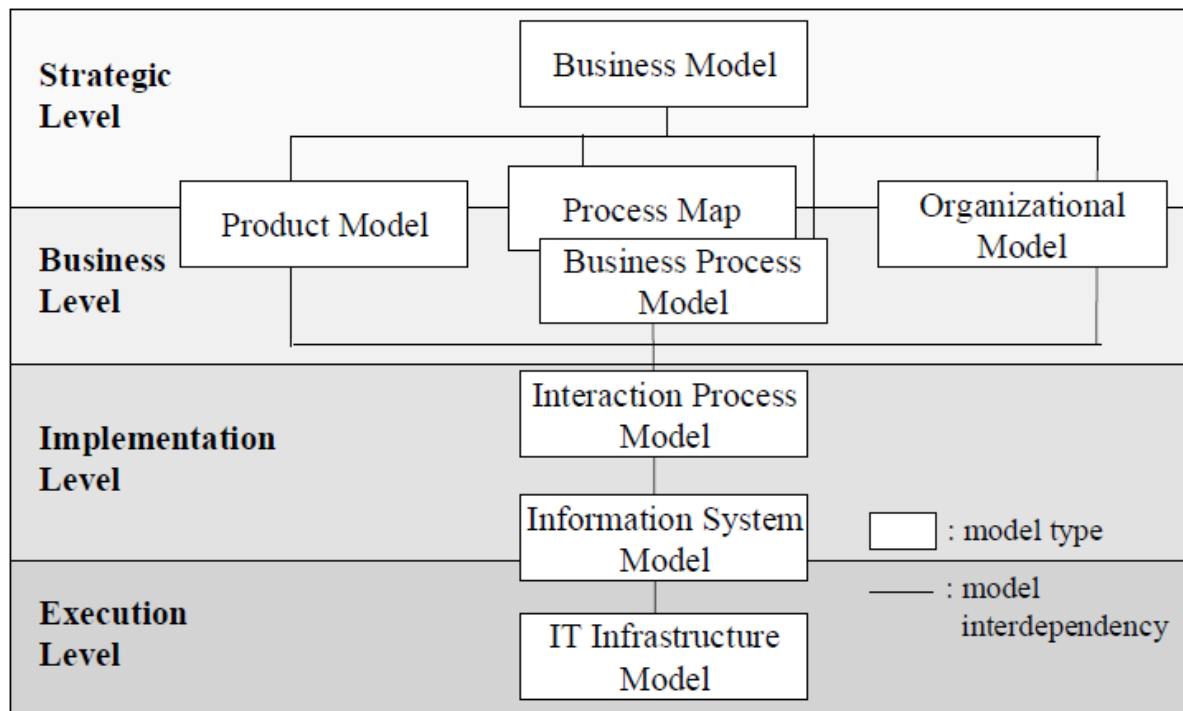


Figure 7 E-BPMS Metamodel – Conceptual View (Kühn, et al., 2001)

In the strategic level the different business models of application stores will be modelled by using the E-BPMS model type “Business model”. The different actors and the interrelating revenue streams will be discussed. In the business level, the processes for the distribution of the application from the developer to the end-user will be modelled as process maps and drilled down to the specific business process models within each application store environment. The common characteristics of these processes should be identified, evaluated and classified to find an aggregated model for the distribution process. The implementation level should bring together the findings of the strategic and business level to describe the information systems behind the chosen application stores. Finally the findings of the modelling efforts should be summarised by the creation of an aggregated model for application distribution across multiple platforms. The execution level will be discussed in Chapter 6.7 below where the underlying IT infrastructure will be examined.

6.1.1 BOC Management Office

The community edition (ADONIS:CE) of the ADONIS Business Process Management Toolkit is used in the following section to model the different mobile

application distribution processes. ADONIS is a key part of The BOC Management Office which also includes ADOit®, which is used in a later part of this these – a family of products for the IT supported management approaches for strategy and performance management, business process management, supply chain management and IT-service and architecture management. ADONIS:CE is a functional and feature rich stand-alone version of ADONIS, which is free to download on the ADONIS:CE website. *ADONIS supports standard modelling notations such as BPMS, BPMN, UML, EPC, and LOVEM. In addition, ADONIS provides an underlying meta-modelling technology that allows users to define new modelling notations and mechanisms for domain-specific or customer-specific needs. Various pre-defined reference models, templates, and meta-models are also available, including ITIL, CobiT, ISO 20000, SCOR, Six Sigma, SOX, NGOSS/eTOM and ERM. These are implemented as specific pre-built modules and templates designed to increase project efficiency and communication, reduce costs in developing procedures and to ensure a rapid return on investment (Hall, et al., November, 2005).*

ADONIS is separated into two toolkits: the Administration Toolkit and the Business Process Management (BPM) Toolkit. The Administration Toolkit provides multi-level user administration, meta-model administration, and configuration management facilities. The BPM Toolkit provides the end-user with configured application components. This division provides a clear separation between administrator and end-user tasks. All of the necessary components are directly available from a single user interface within each toolkit. For documenting processes, ADONIS uses the swim lane paradigm for its workflow model. ADONIS is written in C++, with add-on components in C and Java (Harmon, 2010).

6.2 Meta Model Apps Distribution

The mobile application distribution process is the process under an application is developed, distributed to the market and purchased and downloaded by customers, and used on mobile devices. This process involves three main components as shown in the model in Figure 8 Meta Model Application Distribution adapted from (Holzer, et al., 2010).

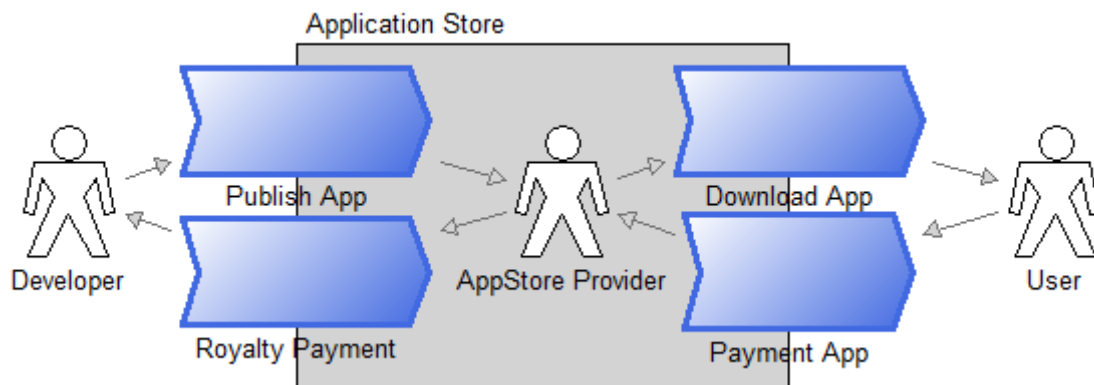


Figure 8 Meta Model Application Distribution adapted from (Holzer, et al., 2010)

First, the developer uses development tools to build its mobile application. Second, the developer publishes its application on a portal, from which the consumer can download the application onto its mobile device and pays for it. Then a revenue share is paid back to the developer. This approach is different from the walled garden approach which was popular until recently where mobile network operators (MNOs) were in charge of being the interface between customers and service providers. The application distribution model depicted in figure 8 represents a two-sided market with developers on one side and consumers on the other. In such a market, an increase or decrease on one side of the market induces a similar effect on the other side. In other words as the number of consumers increases for a given platform, portal, or mobile device, the number of developers attracted to this platform, portal or device will also increase (Holzer, et al., 2010).

6.2.1 Cross Platform distribution

In the case of a cross platform distribution the following workload currently applies where developers want to publish their applications on multiple platforms and application stores to reach as many customers as possible.

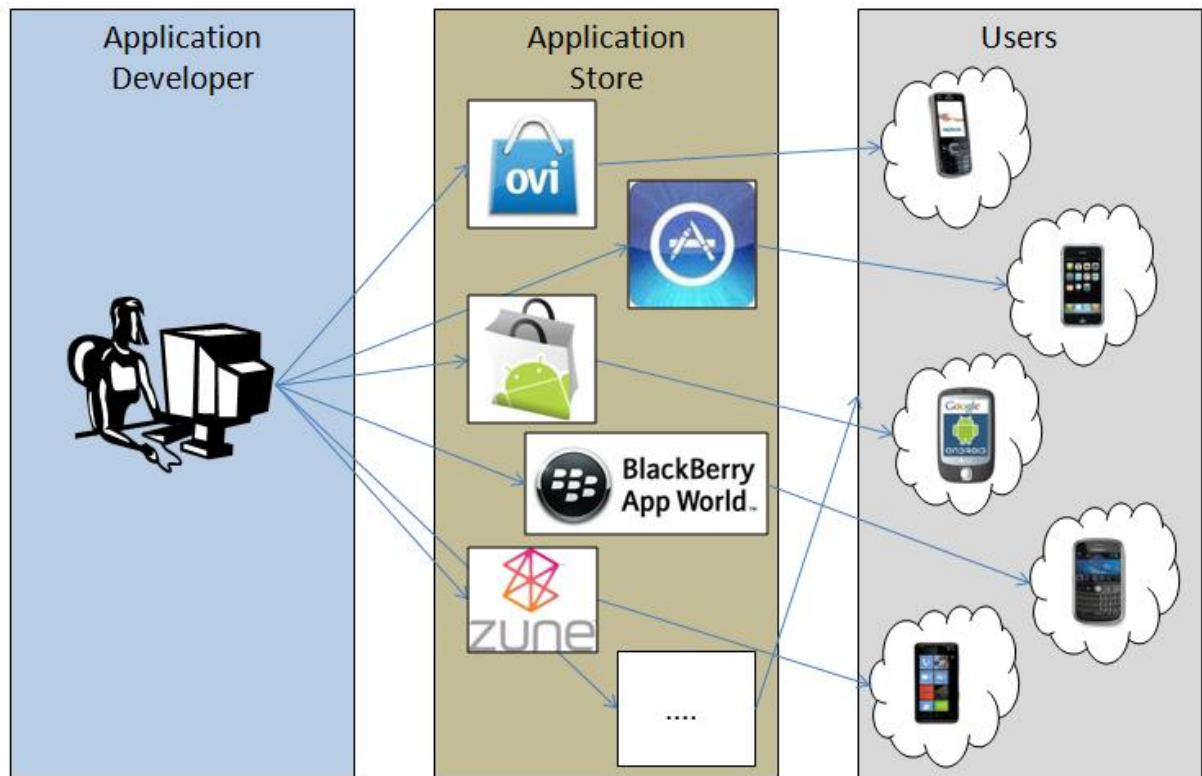


Figure 9 Current Cross Platform Distribution Scenario

Here the developer programs one application with different development tools to meet the requirements and a specification of the platform and application store or builds the application in an abstraction layer and ports them with different commercial tools the application to the destination format. An overview of such porting tools is given in Chapter 8: “Multiple Platform Application Development”. This application needs then to be published on each download portal that can be used by devices for each platform. The immense resource, cost and compatibility implications have been mentioned earlier in this work.

One approach to overcome the resource issue of the development and management of the distribution is the introduction of a “meta platform” where the developer uses a specified SDK to program the application, publishes the application once to the meta platform. Application stores that are connected to the meta platform can distribute this application in their environment to their users. Compatibility across multiple platforms could be achieved by using standardized technologies across the value chain. Developer communities like WAC are evangelizing this approach as discussed in Chapter 5.

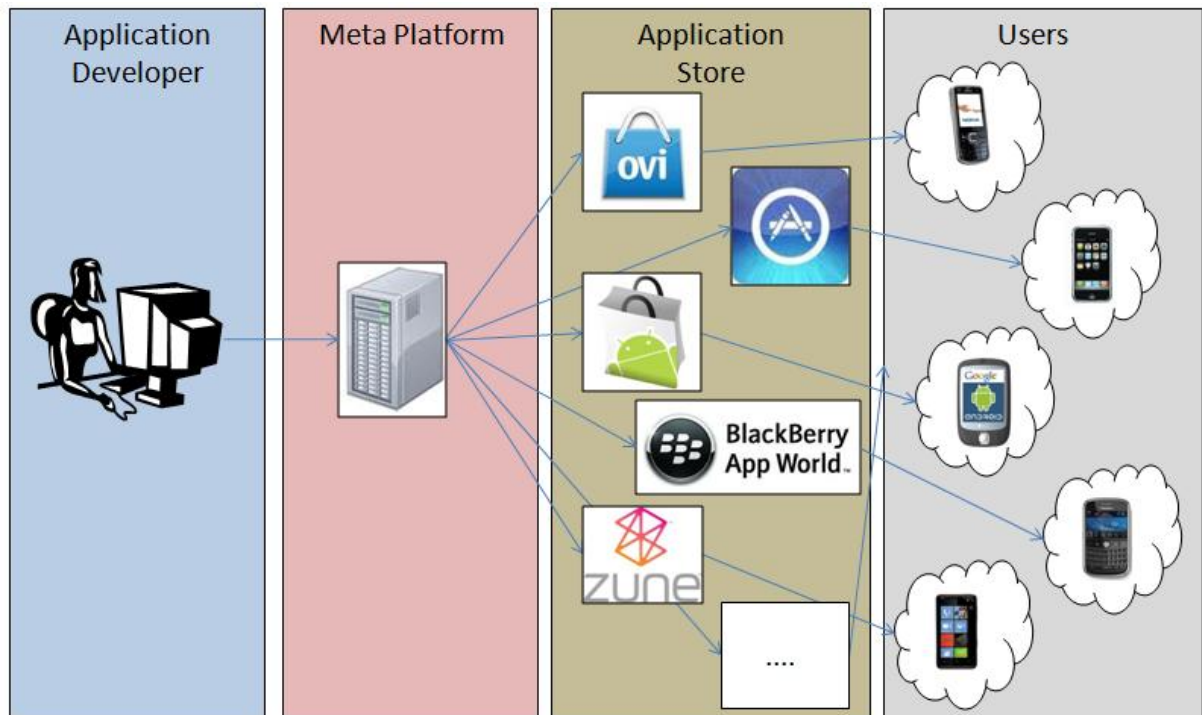


Figure 10 Cross Platform Distribution Scenario with Meta Platform

In the following chapter are the findings of the literature research discussed which helped to formulate the criteria for an aggregated model environment.

6.3 Platform Distribution Classification

The literature research showed that different models are discussed and classified in terms of their control ownership (Vânia, et al., 2010) and business model (Pieter, et al., 2006)(Pieter, et al., 2008) but never the distribution aspect of cross-platform applications. In short I will give an overview of the literature classification of mobile application platforms.

Vânia in (Vânia, et al., 2010) classifies the platform models in four different types with regards to their control ownership and core competencies, which are “enabler platform”, “system integrator platform”, “neutral platform” and “broker platform”

6.3.1 Enabler Platform

According to (Vânia, et al., 2010) in this platform *the owner controls many or most of the assets involved in mobile service provision, but leaves the customer*

relationship to third-party developers. Mobile operating systems such as Windows Mobile and Android can be placed with this platform type.

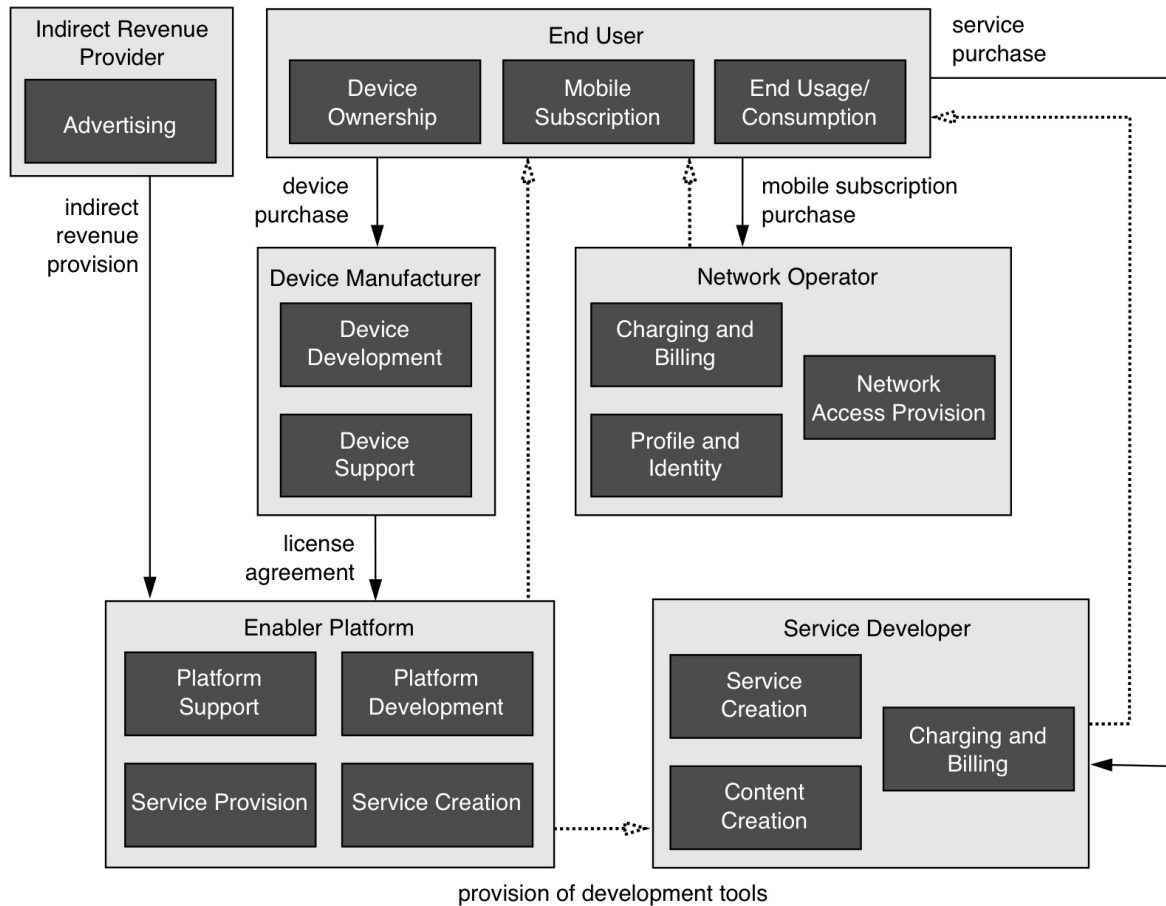


Figure 11 Enabler Platform Model (Vânia, et al., 2010)

Core competencies of this platform type are building and maintaining an IT infrastructure to support the services offered in the platform as well as supporting developers in developing and submitting their software. Furthermore development of attractive pricing for developers, customer support towards developers, involvement in standardisation activities and inducing innovation by incentivising developers.

6.3.2 System Integrator Platform

According to (Vânia, et al., 2010) *this represents the case where many or most of the assets related to the value proposition, as well as the customer ownership, is in the hands of the platform owner. This actor actively facilitates and encourages*

entry of 'third parties' to constitute a multi-sided market. It allows competing service providers to use its platform, in order to increase the value of both this platform and its own end-user service offering. Examples include Apple's Appstore and Nokia Ovi.

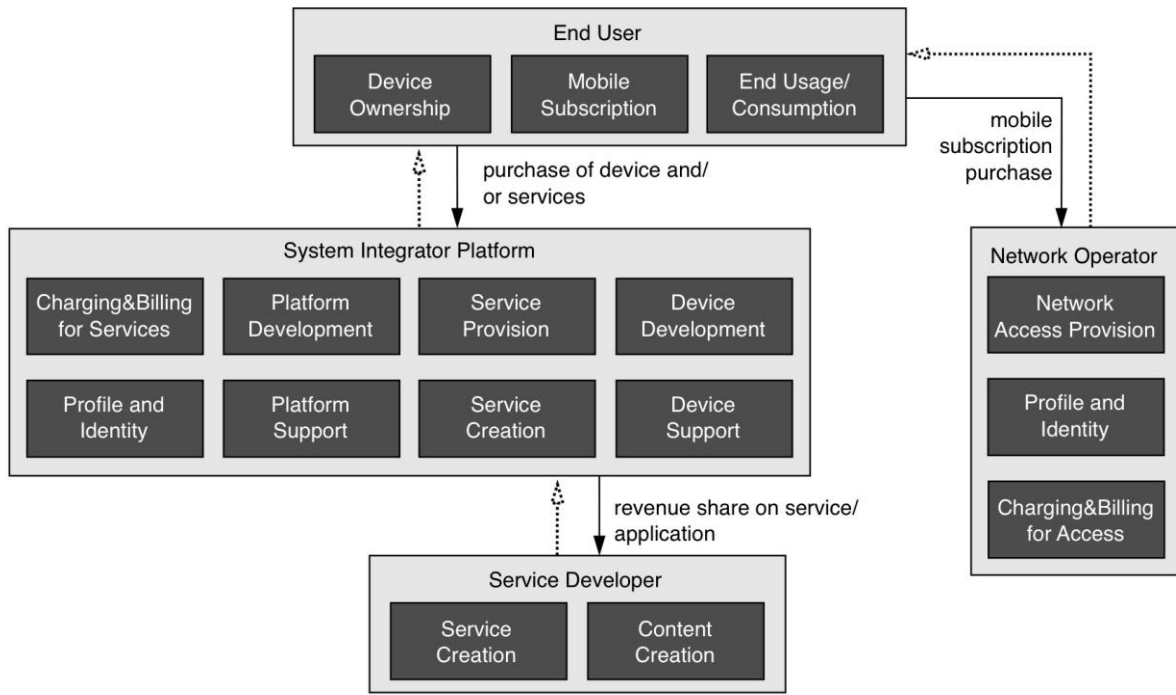


Figure 12 System Integrator Platform Model (Vânia, et al., 2010)

The enabler platform competences also apply here. This platform type successfully attracts both developers and end-users by setting up an attractive revenue share rate, a good development platform with possibility for feedback as well as competitive pricing and billing schemes.

6.3.3 Neutral Platform

According to (Vânia, et al., 2010) *this refers to a case in which the platform owner does not control most of the assets necessary for the value proposition and on top of this does not have customer ownership because it does not establish a billing relationship with the end-user and may be even invisible to the end-user.* WAC could fall into the classification of a neutral platform if seen as enabler platform between the developer and the final storefront.

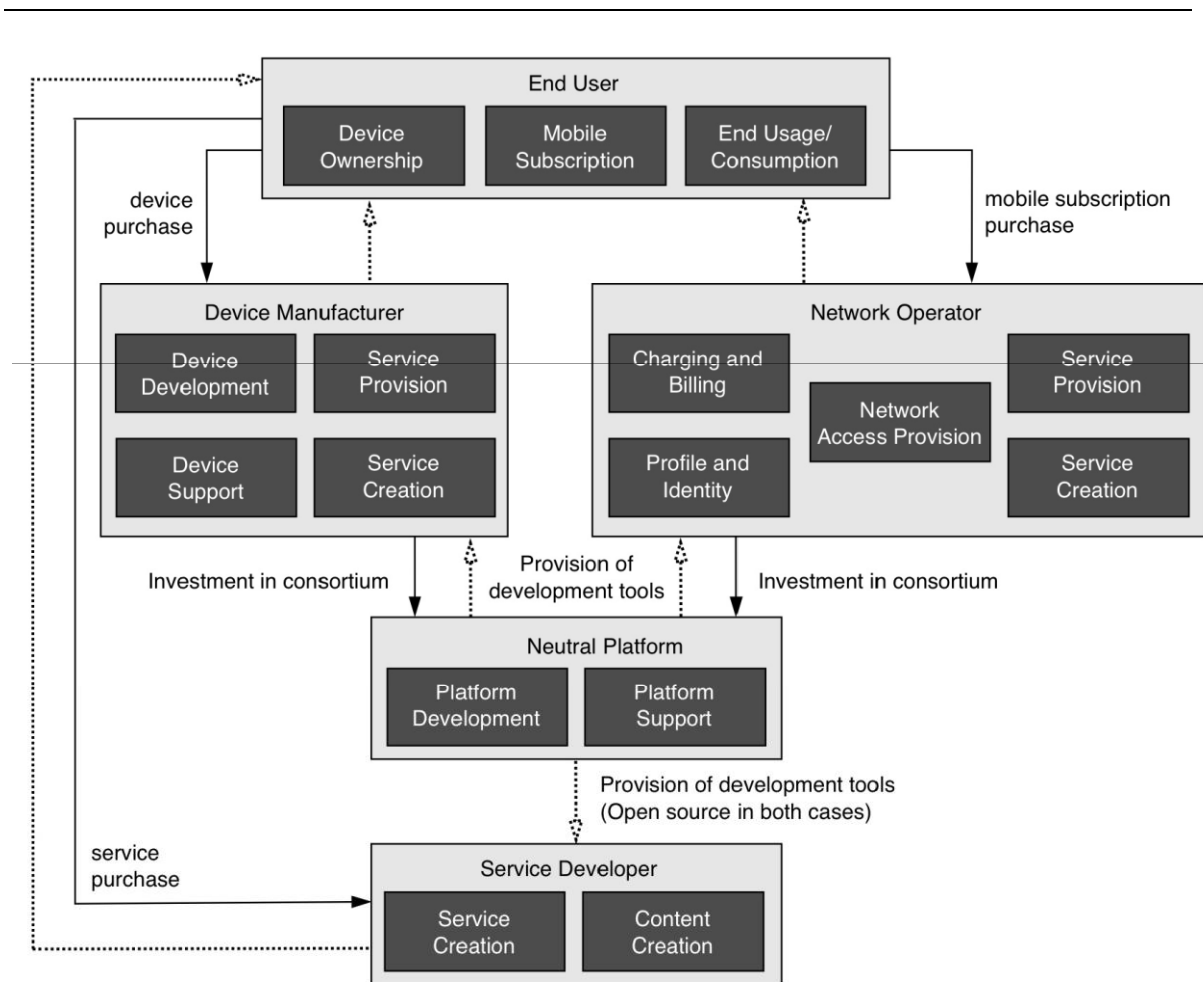


Figure 13 Neutral Platform Model (Vânia, et al., 2010)

The core competencies of the neutral platform are organisation and structure to facilitate efficient collaboration. Managing relationships and balancing the internal and external interests of partners. As well as setting up business-to-business public relations to create awareness of the platform.

6.3.4 Broker Platform

According to (Vânia, et al., 2010) *the broker platform relies on other actors that control most of the assets for establishing the value proposition, but does integrate customer ownership*. Typical examples of such a broker platform are the mobile storefronts GetJar and Handango. The broker platform can be represented as follows:

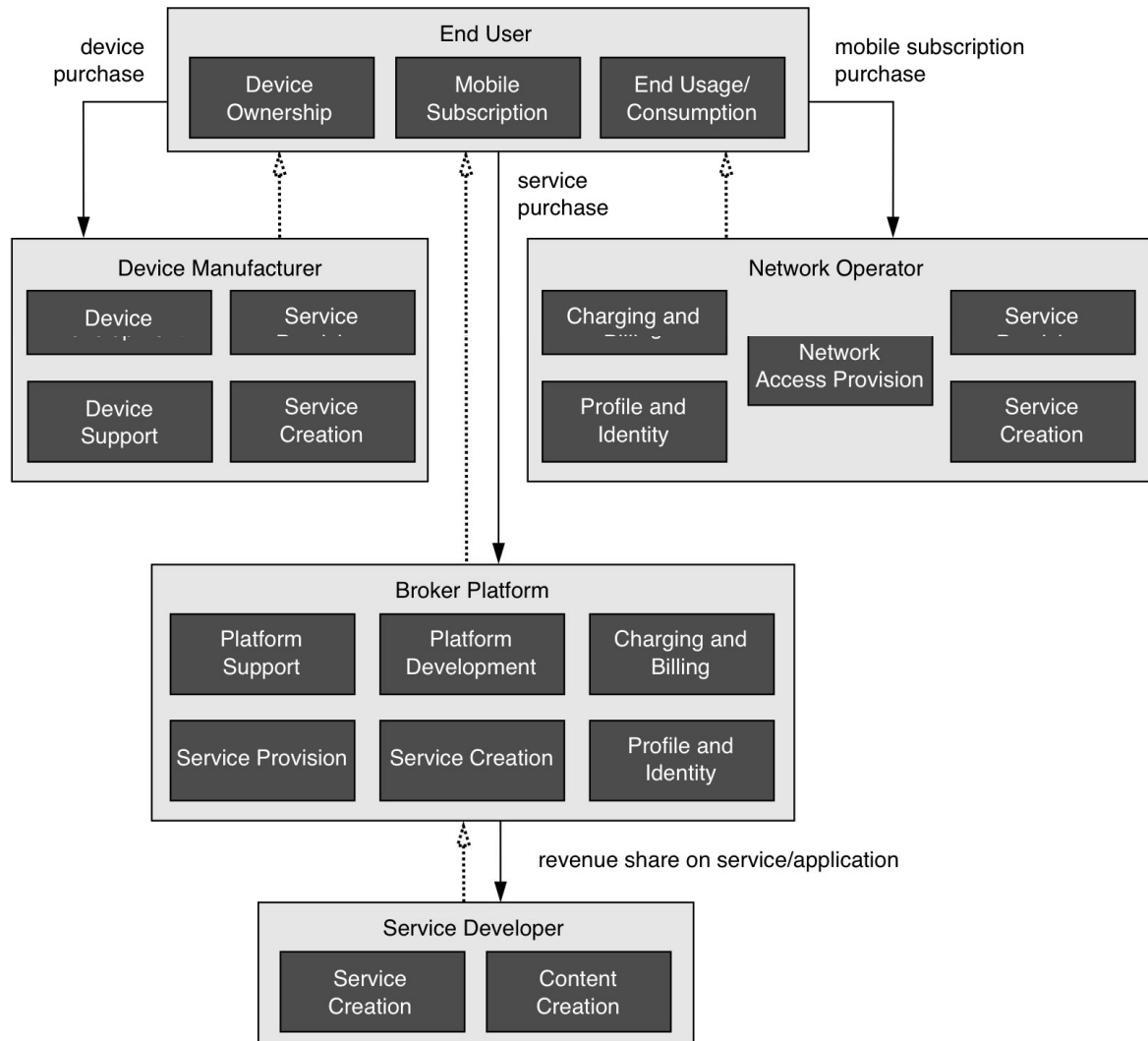


Figure 14 Broker Platform Model (Vânia, et al., 2010)

The core competencies here are again related to attracting both developers and end-user by providing developers incentives to publish on the platform e.g. an attractive revenue sharing scheme and tools to track statistical information on their products. Focus on user experience and provide a user-friendly environment with competitive prices is a further advantage.

Pieter in (Pieter, et al., 2006)(Pieter, et al., 2008) covers classifies the platform environment due to their ownership model depending on the role of the actors. These models are discussed in short in the next chapters.

6.3.5 Telco centric model

The Telco centric model places the majority of roles within the domain of a single real-life stakeholder in this case the telecommunication network operator, which acts as portal provider, service aggregator, network operator and platform operator.

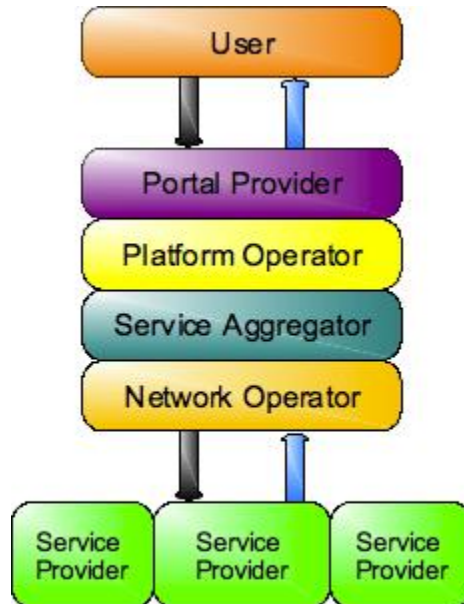


Figure 15 Telco centric model (Pieter, et al., 2008)

6.3.6 Device centric model

According to Pieter in (Pieter, et al., 2008) *the device centric model is a model where the main service platform is incorporated in, or tied together with the mobile device. A real-life example is the Apple iPhone.*

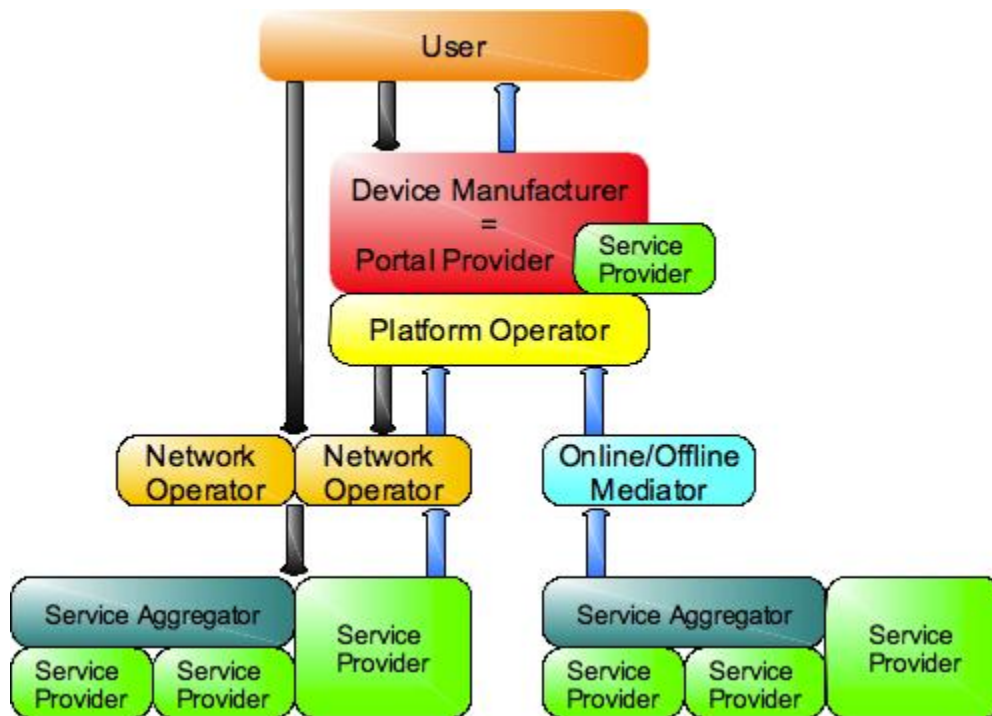


Figure 16 Device centric model (Pieter, et al., 2008)

In this model, several services are offered as an integral part of the device. The user gets access to a number of services embedded into the device. In this model, the device manufacturer functions as portal provider, choosing and controlling which services are made available to consumers. This actor also defines what specific platform is used to provide services to the user. Most of the platform operator activity is performed by the device manufacturer, as almost all information and tools needed to develop services and applications for the device are internal to the company. The manufacturer can also decide to provide tools and resources in the form of a Service Development Kit (SDK) to service providers for the development of new services (Pieter, et al., 2008).

6.3.7 Aggregator centric model

In this model the function of portal provider is taken over by the service aggregator. Real life examples are the Nokia developed Widsets client, which is available for different mobile operating systems, and social networking site Facebook.

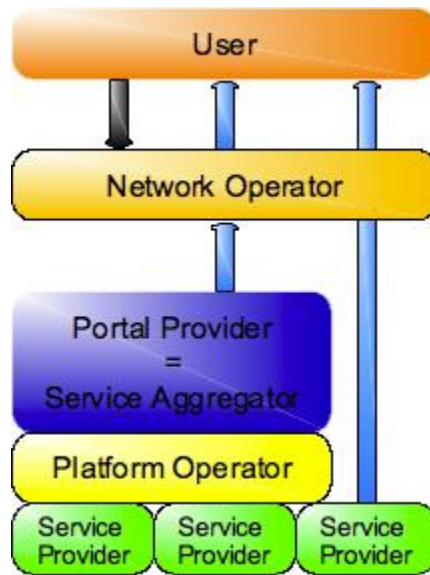


Figure 17 Aggregator Centric model (Pieter, et al., 2008)

In this scenario the service aggregator actually becomes the portal provider. It serves as the portal to the user, who can choose to install several smaller applications that coincide with his interests and preferences. The user pays the network operator for access to the network and gains access to the portal (Pieter, et al., 2008).

6.4 Distribution characteristics

In this section I analyzed the current state of the mobile application distribution market as well as trends discussed in (Holzer, et al., 2010) with respect to the three components of the application distribution process model presented in figure 8. I examine the approaches used by platform providers towards development tools, portals and devices.

6.4.1 Mobile application development tools

According to (Allan, et al., 2010) central to every development platform, software development kits (SDK) enable third-party developers to build applications running for the platform. These kits usually include libraries, debuggers, and handset emulators, among other useful development tools. Existing platforms have taken different approaches when sharing their SDK with developers. Some have chosen to restrict access as much as possible; this is referred as “closed technology”, like

Apple where all strategic decisions about the platform are controlled by Apple. Whereas others have chosen to disclose the entire source code of their SDK and OS, named “open technology” – this approach is used by Google as well as WAC where the source code is accessible by the developer community. The shift of major players towards openness had a significant impact. The market moved from a majority of closed systems to a small predominance of devices running open-source systems.

6.4.2 Mobile application portals

In the distribution process, an application is developed and made available to customers through an application portal. The mobile application portal is an essential component in the mobile application distribution process. Portals play the role of intermediary between developers and consumers(Anar, et al., 2010). Several platforms use a centralized single point of sale strategy like Apple and Google, when one portal is proposed as the main portal on which all applications are published. While others use a decentralized multiple points of sale strategy which means that developers can freely upload and distribute their applications on any third-party portal the WAC is following this approach.

Following Apple’s lead, traditional platforms like Nokia, RIM and Microsoft are moving in this direction. This approach makes it more difficult for developers and also customers to decide what platform and device to go for.

6.4.3 Device set

The device used by customers is more than ever of central importance. New technical features enhanced the development of more advanced mobile applications. Platforms can have different approaches. A platform could dedicate itself to one type of device like Apple’s iOS series the so called “device uniformity” or a set of varied devices like Android or WAC labelled “device variety”. When looking at the set of targeted devices, commercial platforms were traditionally targeting a variety of devices. Apple, RIM and Google both began by targeting uniform devices. However, Google shifted its approach to target a plethora of different devices and manufacturers. Apparently issues with device compatibilities arise across different manufacturer models which lead to an extra layer of

complexness and resource intensity when developing for a platform(Anar, et al., 2010).

6.4.4 Platform integration

Some platforms focus on their core business, which is to provide an OS with programming support for developers, whereas others integrate some or all elements of the distribution process. Holzer describes as well the case of no integration whereas this is not applicable in the current market environment. Platforms with a full integration have a strict control over every step of the distribution process from device manufacturing to application publishing. A fully integrated platform can take advantage of the two-sided market in the following way. Reducing the price of one element, such as the mobile device will attract more customers which will then attract more developers. Platforms with portal integration focus on application development and application sale by integrating a portal. Google provides such integration with its Android Market. Contrary to Apple, Google does not manufacture mobile phones on which its OS runs, the Google G and Nexus series are manufactured by companies like HTC and Samsung. In the device integration model, platforms also manufacture devices but are not in the application portal business. There are only some handset manufacturers in the market that do not own their own application store like Motorola, NEC, Panasonic (Anar, et al., 2010)(Allan, et al., 2010).

6.5 Aggregated Distribution Model – Concept Criteria Analysis

In the following chapter the aggregated model is discussed. First I compiled based on the results and learning of the previous parts the requirements of the stakeholders in the value chain. Then I proposed aggregated concept criteria and evaluated each of the three application store environments against the aggregated concept criteria as a compliance analysis at the status at the time of this thesis. The evaluation was done by giving the marks 3 GOOD, 2 NEUTRAL, 1 BAD. This table shows the evaluation with the marks per application store against the aggregated concept model as well as the description for the value. I've clustered the criteria according to the overarching business processes that have been

identified for the distribution of mobile applications in the modelling process. First I analysed the three application stores in terms of their development environment setup:

Table 4 Analysis Development Environment Setup

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Development Environment Setup	Workstation	3	1	3	Support of WIN, MAC, Linux workstations	WAC, Android full support, Apple Appstore needs Mac Workstation
	User Manual	1	3	3	High Quality of description of user guides, details of user guides	Apple & Android show great detail in their user guides and videos. WAC lacks that support
	Single SDK	2	1	2	Single SDK to support multiple platforms necessary	One SDK per platform supplied,
	Multi Platform Compatibility	3	1	2	All major platforms should be supported with a single development effort	Apple only iOS, WAC & Android don't support iOS
	Unified Development Environment	3	1	2	Standard programming languages should be supported, no special knowledge needed	WAC builds on HTML, Java, CSS - Apple on Objective C; Android Java
	Software development tools	2	3	2	All software tools should be supplied with SDK	WAC and Android require further software and setup - not out of the box
	Average	2.33	1.67	2.33		

The calculation of the average shows that the WAC and Android environment have equal values due to the fact that Android is near a multi-platform distribution environment due to the nature of the OS and both can be setup on the dominant workstation environments. As the iOS setup is just working on Intel MAC workstations and only supports iOS devices the result is lower. Apple's strong point is the user manuals provided as well as a complete SDK means everything that a developer needs is automatically provided and installed through a single SDK file.

Despite the application development is out of scope of the later business process analysis, it definitely plays a major role in multi-platform compatibility of mobile applications. Therefore I've compared the three application store environments in terms of their application development under the aspect of cross-platform compatibility reflecting important factors that developers consider when developing currently applications.

Table 5 Analysis Application Development

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Application Development	Current Customer Base	1	3	3	Highest volume of customer base	Appstore and Android market with biggest customer bases
	Potential Customer Base	3	2	3	Highest volume of customer base	WAC and Android have potential to lead the market due the openness of the platform and multi-device capabilities
	Monetization Potential	1	3	2	Amount of paid apps	Apple is leading the monetization with 3 times more available paid apps that android
	Programming Languages	3	1	2	Standardized languages attract most developers and don't require further resources	WAC leads this space now with W3C Standards
	Operating Territories	1	3	2	Availability of Store worldwide	Apple is leading this criteria with full working billing in stores in most countries
	Multi Platform Compatibility	3	1	2	One software build compatible on all platforms	Apple only iOS; WAC: Proprietary & Android; Android only Android
	Development Costs	3	1	2	Costs for acquiring special knowledge for the development of that platform	WAC builds on HTML, Java, CSS - Apple on C++ + Special Code; Android Java + Special Code
	Unified device capabilities and APIs	3	3	2	Device standards and APIs are unified	WAC uses only standards created for multiple devices; Apple iOS runs on all iOS derivatives due to unified device capabilities

	Device Variety	3	1	3	Platform openness leads to a higher device variety and customer base	high to low end devices supported; difficulty in testing and porting if device standards are not implemented
	Available Standards	3	2	2	Standardisation of development languages and packaging necessary	WAC uses W3C standards
	Device Standardization to reduce porting effort	2	3	2	One software build compatible on all platforms	iOS porting effort minimal; Android porting for different software versions necessary
	Unified network APIs	3	1	1	Unified network APIs for improved app capability	Android & iOS use only network for data traffic; WAC enables network specific APIs
	Test Environment	1	3	1	Complete Test environment comes with SDK	Apple support with full testing software and adhoc deliver; WAC runtime still limited;
	Average	2.31	2.08	2.08		

WAC is slightly ahead of the other environments just due to the fact that its application development is targeting cross platform distribution and the others not. Clearly the difference is currently not as huge as initially thought but this is due to the fact that WAC is currently in its infancy and the customer bases and developer reach is not yet as matured as for example the Apple AppStore environment.

The next table analyses the registration process of the environments looking predominantly at the capability to reach as many different even potential store environments.

Table 6 Analysis Registration

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Registration	Unified Signup	3	1	1	single signup for multiple platform	WAC acts as meta platform; Apple and Android require registration separately
	Free of Charge	3	1	1	registration should be free of charge for the developer	WAC is free of charge; Apple and Android charge fees

	Easy to manage	3	2	2	Single sign-on for multiple appstores and management area	WAC has one management area for all connected Appstores; Apple, Android individually
	Security	2	2	1	Secure management of developer specific data - payment, application data	All appstores have secure account management environments but piracy of intellectual property is existent in all environments; android has the biggest piracy problem currently
	Average	2.75	1.50	1.25		

WAC is in reference to the selected criteria clearly ahead of the other store environments due to the facts that the unified and free registration will attract developers more easily once the distribution reach is given. In terms of security the Android Marketplace has the biggest problem in terms of piracy and infringement of intellectual property as no monitoring of applications is currently conducted.

The application upload process is the next criteria analysed in the distribution process.

Table 7 Analysis Application Upload

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Application Upload	Single point of content submission	2	1	1	1 content submission interface and process for multiple stores	Apple and Android currently support only their store; WAC opens for more but doesn't include android and apple
	Interface for submission and backfilling	1	3	3	Accessibility and easy to use interface	WAC just only recently launched, Android and Apple optimised their interface over the years

	Application signing process	2	3	1	Application signing supports the usability of the application on the phone to avoid device security warnings	Apple no separate signing necessary due to development process; Android doesn't provide signing; WAC includes a certification process
	Piracy protection	3	3	1	Protection against file copy and distribution	Marketplace has no piracy protection; Apple doesn't allow pirated files; WAC has certification
	Approval Process	2	2	1	Approval process should guarantee quality of content, in short timeframe but with limited restrictions	Android has no approval process, anything can go live; Apple has a detailed process that takes quite long, and is restrictive; WAC's approval runs through partner stores - no detailed info
	Average	2.00	2.40	1.40		

The Apple Appstore upload is by far the most matured solution in the market currently thus the highest average score. The tools provided as well as the certification of the application happens in one process and the piracy protection within the Apple Appstore is maintained especially through the controversial approval process. Only through jail breaking³ of the iOS devices are pirated files even installable.

The next table covers the application distribution looking into the reach, monetization potential, multiple platform distribution and supported business models.

Table 8 Analysis Application Distribution

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Application Distribution	Current Customer Base	2	3	3	Highest volume of customer base	Appstore and Android market with biggest customer bases

³ Removal of limitations in the operating system i.e. installation of third party applications on iOS

	Potential Customer Base	3	2	3	Highest volume of customer base	WAC and Android have potential to lead the market due the openness of the platform and multi-device capabilities
	Monetization Potential	1	3	2	Amount of paid apps	Apple is leading the monetization with 3 times more available paid apps than android
	Multiple Platform Distribution	3	1	1	Distribution of application on multiple platform stores	WAC is setup to do that currently; Appstore and marketplace not supported
	Cost Control	3	1	1	no overhead due to management of multiple upload processes	WAC is setup to do that currently; Appstore and marketplace not supported
	Unified Content Management	3	2	2	Single content management tool for multiple stores and countries	WAC is setup to do that currently; Appstore and marketplace support multiple countries
	WIFI compatibility	2	3	3	User should be able to download through any data connection	Credit card billing enables this for Apple and Android; WAC depends on partner stores
	Localization Support	2	3	2	Localization Support for multiple countries	Apple offers most detailed localization support
	Supported Business Models	2	3	2	Variety of supported business models are crucial for monetization	Apple is leading the offer of different business models
	Pay Per Download	3	3	3	Pay Per Download Model	Standard model supported
	Advertising	2	3	3	Advertising Funded Model	established on apple and marketplace - not yet in WAC
	Freemium Model	2	3	3	Freemium Model (Free download and In game Purchases)	established on apple and marketplace - not yet in WAC
	Subscription	1	3	1	Subscription (Recurring billing model)	Apple supports this only
	Billing Methods	3	1	2	Credit Card, Mobile Phone Bill, Premium SMS, PayPal etc.	WAC is capable of supporting the most billing options due to the integration with MNO's
	Average	2.29	2.43	2.21		

Despite Apple supports only one storefront for its iOS the average score is pushed by its support in business models and monetization potential, which make this environment currently the developer's first choice for distribution. WAC clearly has

the advantage of upload once and distribute in multiple storefronts but this is yet to be proven especially in terms of the monetization potential.

The next criteria analysed is the application delivery and purchase by the end user.

Table 9 Analysis Application Delivery / Purchase

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Application Delivery / Purchase	Application Discovery	2	3	2	Application discovery through storefront, on-store merchandising features like search, recommendations	Apple is the most advanced store offering full feature set
	Quality of Service	1	3	2	Maintenance and Service Levels	WAC is not yet fully established; Apple leads the quality field through tight control of every aspect of the store
	Easy to use user interface	1	3	2	Usability of the storefront Interface to browse	Appstore is the most advanced store; Android is improving; WAC storefronts in early stages
	Attractive Content and Applications	1	3	2	Most versatile content offer in range and quality	Apple is leading this criteria with most available applications, followed by Android despite quality issue; WAC limited offer
	Application Merchandising	2	3	2	Merchandising features like search, recommendations, bundling	Apple is the most advanced store offering full feature set
	Average	1.40	3.00	2.00		

As before the Apple Appstore is ahead of the other application environments due to the centralized platform development that has only one device set as end user. Therefore this environment is highly optimised for this specific device in terms of discovery, quality of service and usability. Obviously the success of the Apple Appstore in terms of monetization makes it the first choice of developers thus has the most attractive and highest volume of content available. Android is catching up

with Apple but still has a load of free applications due to the missing business and billing options. WAC's performance in terms of storefronts differs according to the plethora of connected storefronts and providers offering this content.

Storefront reporting & settlement under the light of multiple storefront distributions is analysed in the next two tables.

Table 10 Analysis Storefront Reporting & Settlement

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Storefront Reporting & Settlement	Centralised Reporting	3	1	1	Reporting from all storefronts in one unified tool	WAC offers this in the My Account page; A & A report only for their store
	Centralised Settlement	3	1	1	Settlement and Payments from all storefronts in one unified process	WAC offers this ; Appstore and Marketplace settle only for their store
	Average	3.00	1.00	1.00		

WAC will be the only environment that offers the distribution of applications through multiple connected storefronts by different providers as a Meta platform. Therefore the settlement and reporting through a centralized platform gives WAC a clear advantage over the other environments.

Table 11 Analysis Multiple Storefront Settlement

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
----------	--	-----	-----------------	----------------------	--------------------------	---------------------

Multiple Storefront Settlement	Accurate Settlement	3	1	1	Accurate settlement process across all storefronts are necessary	WAC offers this ; Appstore and Marketplace settle only for their store
	Payment of Royalties	3	1	1	Payment of royalties need to be unified and accurate	WAC offers this ; Appstore and Marketplace settle only for their store
	Average	3.00	1.00	1.00		

The payment of the monetization through the selling of applications through the storefronts, so called royalty payments is looked at in the next table.

Table 12 Analysis Royalty Payments

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
Royalty Payments	Accurate Settlement	3	3	3	Accurate settlement process across all storefronts are necessary	Assumed accuracy of settlement standards
	Payment of Royalties	3	3	3	Payment of royalties need to be unified and accurate	Assumed accuracy of settlement standards
	Commercially Attractive business model	2	2	2	Business Model needs to attract developers (industry standard 70/30 share)	All stores meet his criteria
	Average	2.67	2.67	2.67		

All examined environments are equally considered when it comes to the payment of royalties as well as their business model. Settlement and payments are happening due to the number of registered developers in an automated process therefore the accuracy is assumed to be good. In terms of the offered business models i.e. revenue share towards the developer, all application stores offer a 70 % revenue share.

The last criteria cluster analysed is the IT infrastructure with the goal of multiple platform and storefront distribution.

Table 13 Analysis IT Infrastructure

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
IT Infrastructure	Multiple Platform Distribution	3	1	1	IT infrastructure needs setup to support connection of multiple storefronts for the distribution of content	WAC's setup is designed with the WAC client interface to act as meta platform
	Multiple Storefront Settlement	3	1	1	IT infrastructure needs setup to support settlement of multiple storefronts	WAC's setup is designed with the WAC client interface to act as meta platform for settlement information
	Application Delivery / Purchase	2	2	2	Application delivery and purchase infrastructure needs to support browsing and merchandising features	all stores offer this similarly; WAC offers operator billing; Appstore and market place have established the credit card billing model
	Application Upload	3	2	2	1 content submission interface and process for multiple stores	WAC's setup is designed with the WAC client interface to act as meta platform
	Registration	3	2	2	Secure, easy to use, single signup for multiple platform	WAC's setup is designed to act as meta platform for a single registration platform
	Average	2.80	1.60	1.60		

WAC as such is the environment out of the three that offers an IT infrastructure that is designed to connect multiple storefronts to its eco system to offer cross platform distribution. Across all major processes for distributing the application WAC is the one with the highest score, due to the focus of integrating more than one environment to distribute applications especially to the upload and distribution process as well as the settlement across multiple storefronts.

Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments
	Overall Average	2.45	1.93	1.75		

When calculated the overall average from above analysed criteria, the WAC application environment comes out as the highest scored eco system, when the focus lies on the distribution and development across multiple platforms. However as already discussed is the WAC environment in its infancy when it comes to supported devices, connected storefronts and customer base.

6.6 Aggregated Distribution Model - Business Process Analysis

In this chapter the aggregated distribution model will be discussed in terms of the business processes from a developers perspective. The applied methodology was for the processes from “Development environment setup” to “Application Delivery / Purchase” a hands-on approach by myself and in depth analysis and comparison of the modelled business processes of the WAC, Apple Appstore and Android Marketplace which can be found in Appendix 2. By comparing the business processes that have been identified during the modelling process the following processes are forming up the aggregated model:

Development Environment Setup

Application Development

Registration

Application Upload

Application Distribution

Application Delivery / Purchase

Storefront Reporting & Settlement

Multiple Storefront Settlement

Royalty Payment

IT Infrastructure

The results of this analysis are summarized in the following table by giving the marks 3 GOOD, 2 NEUTRAL, 1 BAD. This table shows the evaluation with the marks per application store against the aggregated concept model as well as the description for the value.

Table 14 Aggregated Distribution Model Evaluation Matrix

Business Process	WAC	Apple AppStore	Android Marketplace	Aggregated Model Concept	Comments
Development Environment Setup	2	2	2	One multiplatform compatible SDK download containing all development tools should be available free of charge on a developer page, supporting WIN, MAC, Linux workstations with high quality user manual and guide; no registration should be necessary to obtain the SDK to attract developers	WAC supplies SDK for multiple platforms, nevertheless it's in its early stages and not all platform are yet supported; further WAC SDK does not offer all tools and runtimes in the SDK; Apple's SDK contains all necessary tools and runtimes to start developing, however it runs only on Intel MAC computers and supports only iOS devices; prior registration is also necessary to get access; Android's SDK doesn't contain all tools and supports only single platform
Application Development				Business Process for Application Development is not in scope	Business Process for Application Development is not in scope
Registration	2	1	1	Single Sign on for multiple platforms and storefronts; Free Signup ; No Extra Certification; Payment Details for Royalty payments should be optional if offered free or paid for applications; payment details should be obtained directly in one process;	Apple and Google require each 3 different signup processes to be registered as full developer who can sell apps which are not directly connected; both are not free of charge; not multi storefront; WAC is only multi storefront - leaves out Apple and Android; WAC further certification needed

Application Upload	2	1	2	Combined Process of application upload and application details specification; web browser interface for multi-workstation use; device specification interface; localization option; storefront management option; definition of price per storefront and territory; private signature	WAC supports multiple storefronts, simple web interface for marketing setup, but has certification process through external party; Apple requires an extra application for the upload and application registration and upload are separate processes, further the application approval is a time consuming process, single storefront multiple territories; Android has streamlined process but only single platform but multiple territories
Application Distribution	2	2	2	Distribution into multiple storefronts through single distribution process; Content should be tested or automatically screened to obey to policies of the Appstore; but should not hinder the distribution process; application store should offer automatic categorisation and manual content management;	WAC supports multiple storefronts but verification and distribution is upon the individual storefronts - developer has to rely on each storefront to publish the app; Apple has complicated and sometimes difficult approval process, not multiple storefronts; Android has no verification process, all applications go directly to the store which increases the risk of piracy and malware
Application Delivery / Purchase	2	3	2	No Security Warnings due to signing and compatibility of application, easy to use payment flow; automatic download, save and installation	WAC process can vary between the storefronts as well as the usability; Apple has a user-friendly download process for applications which just require password entry for payment; Android delivers security warnings before installing, could cause cancellation by user;
Storefront Reporting & Settlement	2	1	1	Multiple storefront reporting and settlement in one interface, with filtering possibilities per product, territory and storefront	WAC is the only meta platform that can deliver this, accuracy and process are not testable at the time of this thesis
Multiple Storefront Settlement	2	1	1	Multiple storefront reporting and settlement in one interface, with filtering possibilities per product, territory and storefront; accurate and automated in real time settlement across all platforms and storefronts required	WAC is the only meta platform that can deliver this, accuracy and process are not testable at the time of this thesis; Apple and Marketplace do not support this

Royalty Payments	2	1	1	Multiple storefront royalty payments, per product, territory and storefront	WAC is the only meta platform that can deliver this, accuracy and process are not testable at the time of this thesis
IT Infrastructure	2	1	1	IT infrastructure needs to allow connection of multiple storefronts through a module for all necessary processes of the application distribution	WAC is the only meta platform that can deliver this through the WAC client that connects the storefront with the meta platform, storefront back ends vary per provider; Apple and Marketplace do not support this compatibility
Average	2	1.375	1.375		

Aggregated Application Distribution 1.0

The aggregated application distribution process has the idea of WAC which employs a meta platform as basis. This enables the developer to use a single interface to develop, sign up, upload, and distribute his application and reach so multiple storefronts that run content for different platforms. The business process of distribution of mobile applications in the aggregated model consists of the following business processes that have been the outcome of the prior analysis of the application store processes and modelled:

Development Environment Setup

Registration Process

Application Upload

Application Distribution

Application Delivery / Purchase

Storefront Settlement

Meta Platform Settlement

Royalty Payment

These processes are discussed in the following part of this thesis.

The business processes “application development” and the ones that involve any 3rd party stores are not in scope of this thesis.

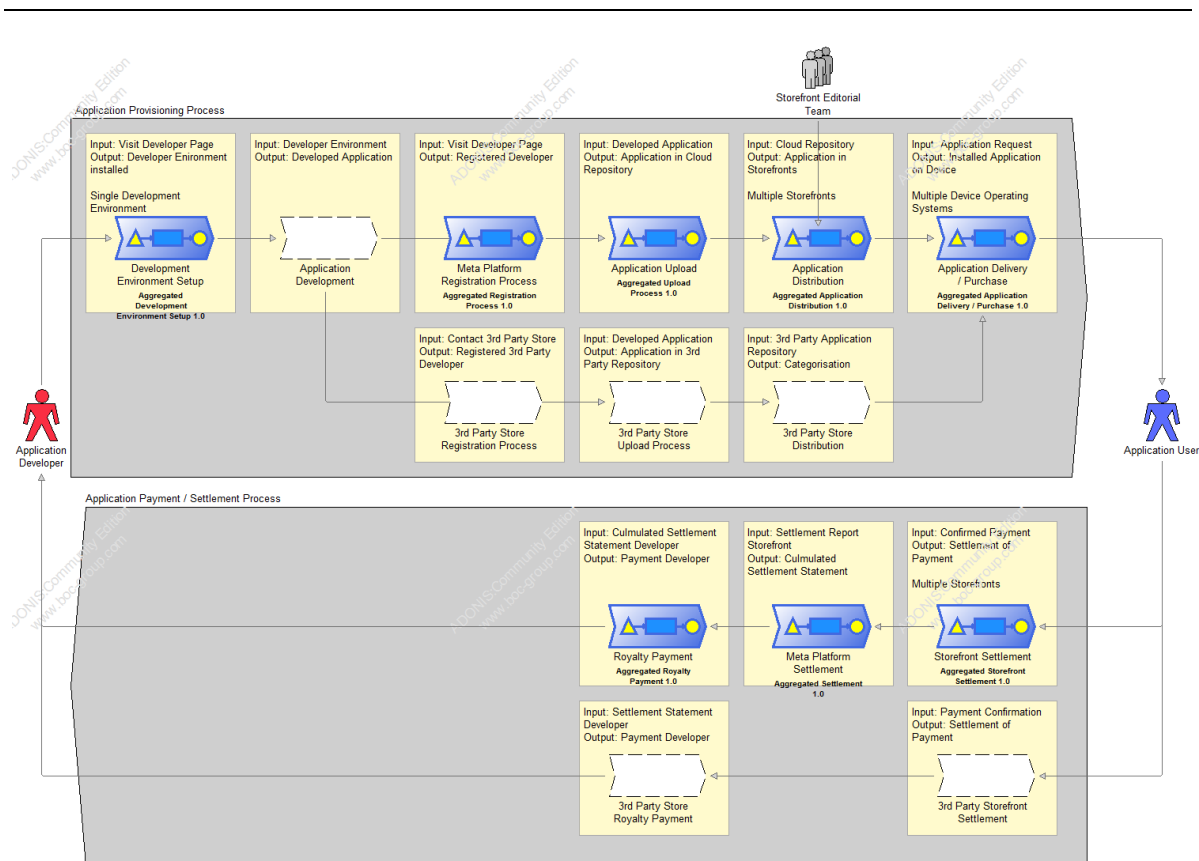


Figure 18 Aggregated Application Distribution 1.0

As the WAC model embraces the Meta platform model, in the aggregated model in Figure 18 the Meta platform is used for the provision of the development environment, the registration as well as the upload for the developer. The application distribution modelled in “Aggregated Application Distribution 1.0” is managed through an API driven client module that connects the partner storefront to the Meta platform environment. Through this client the interaction between the Meta platform and the partner storefronts are managed including the application portfolio, reporting and settlement. The Meta platform itself does not provide directly the application to purchase and acts only as a mediator between the developer and the storefront eco system. Therefore the storefront and its management is entirely in the control of the storefront and therefore might differ in its management, appearance and usability on a per storefront basis. The reporting and settlement accumulation will run as well through the Meta platform client into the Meta platform and will be processed accordingly towards the developer, so that the developer will receive a single report, statement and payment from the Meta platform.

Aggregated Roles 1.0

The following figure shows the interacting roles within the aggregated storefront model. The application developer who is the driving role in the development and upload of the application. The application user on the other end of the process who finally purchases and downloads the application to his device. The storefront editorial team, which is optional depending on the management and approval process setup of the partner store.



Figure 19 Aggregated Roles 1.0

Aggregated Development Environment Setup 1.0

The criteria for the aggregated development environment setup process should consist of one multiplatform compatible SDK download containing all development tools which should be available free of charge on a developer page. This SDK should be supporting WIN, MAC, Linux workstations with high quality user manual and guide. Furthermore extra no registration should be necessary to obtain the SDK to attract developers and don't oppose any entry barriers. The SDK should contain tools that use existing standardized programming languages so that developers do not need extra training. During the comparison of the existing eco systems the outcome was the following:

WAC:

WAC supplies SDK for multiple platforms, nevertheless its in its early stages and not all platform are yet supported. Further WAC SDK does not offer all tools and runtimes in the SDK whereas

Apple iOS:

Apple's SDK contains all necessary tools and runtimes to start developing after the installation. However it runs only on Intel MAC computers and supports only iOS devices. Prior registration is also necessary to get access to the SDK.

Android:

Android's SDK doesn't contain all tools, so sideloading of further runtime etc. is necessary and further it supports only single platform despite multiple different devices.

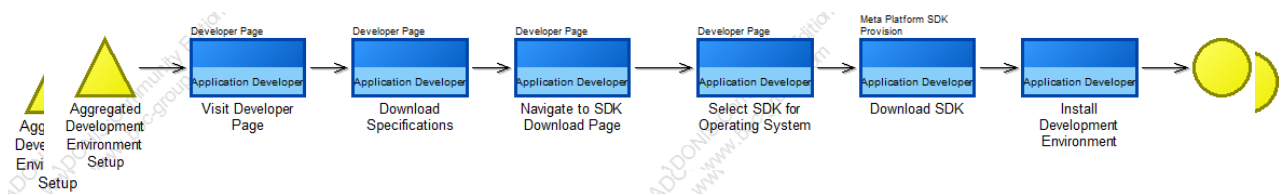


Figure 20 Aggregated Development Environment Setup 1.0

Aggregated Registration Process 1.0

The registration process for the developer in the aggregated process should run through the Meta platform developer portal using a single sign on method for all partner platforms included consisting of a single process for the registration as a developer offering to register with the personal or company details, the type of certificate either, company or private, as well as the payment details which should be optional if only free applications are offered. The registration should be free and no extra certification should be needed.

Apple and Google require each 3 different signup processes to be registered as full developer who can sell apps which are not directly connected. Both developer programs are not free of charge yet it can be argued that this prevents scammers etc. from not signing on to the platform. Android and Apple do not support not multi storefront registration processes. Whereas the Apple environment is an enclosed ecosystem, the Android platform is an open one, which enables 3rd party providers to open storefronts that support Android applications on their platform. Latest example is the Amazon application store. WAC is the only multi storefront environment but currently it doesn't include Apple and Android.

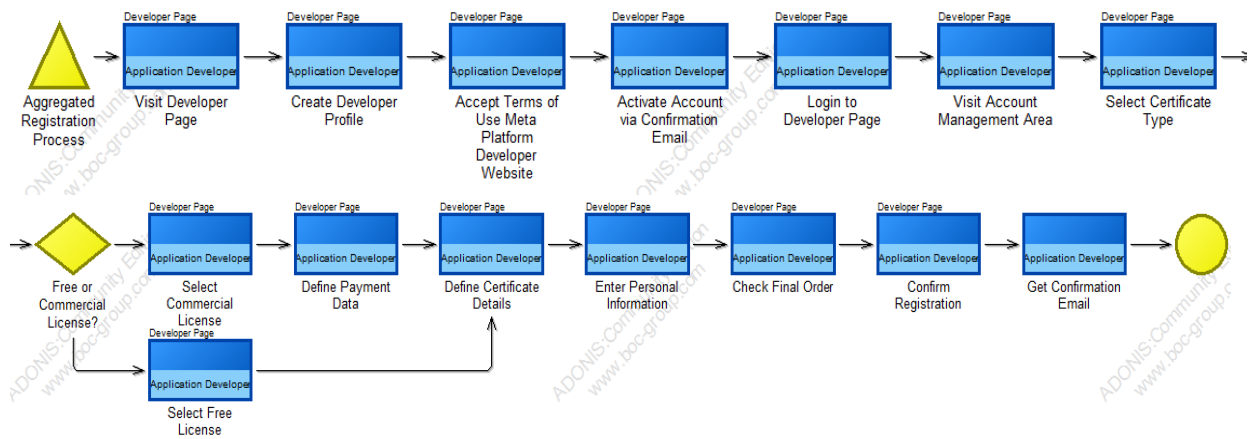


Figure 21 Aggregated Registration Process 1.0

Aggregated Upload Process 1.0

The aggregated upload process should be a combined Process of application upload and application details specification for the distribution of the content. To maximise the compatibility it should be a browser based webinterface for multiworkstation use. It should offer a device specification interface to define on which devices the application is tested and works as well as a localization option to identify the market territories. Further the Meta platform should offer a storefront management option for the definition of price per storefront and territory. The Meta platform should only require a private signature issued by the developer to overcome a complicated certification process. Certification should happend within the development process by the supplied development tools. Following observations have been made during the analysis of the existing application environments:

WAC supports multiple storefronts through the simple webinterface for marketing setup, but has certification process through external party. Apple requires an external application which is delivered with the SDK for the upload of the mobile application. Further Apple requires an application registration prior to the upload of the application therefore these are two separate processes. The application

registration is followed by the application approval which can be a time consuming process. Apple supports despite a single storefront multiple territories. Android has streamlined upload process for its application on the Android Marketplace but supports only single platform but multiple territories.

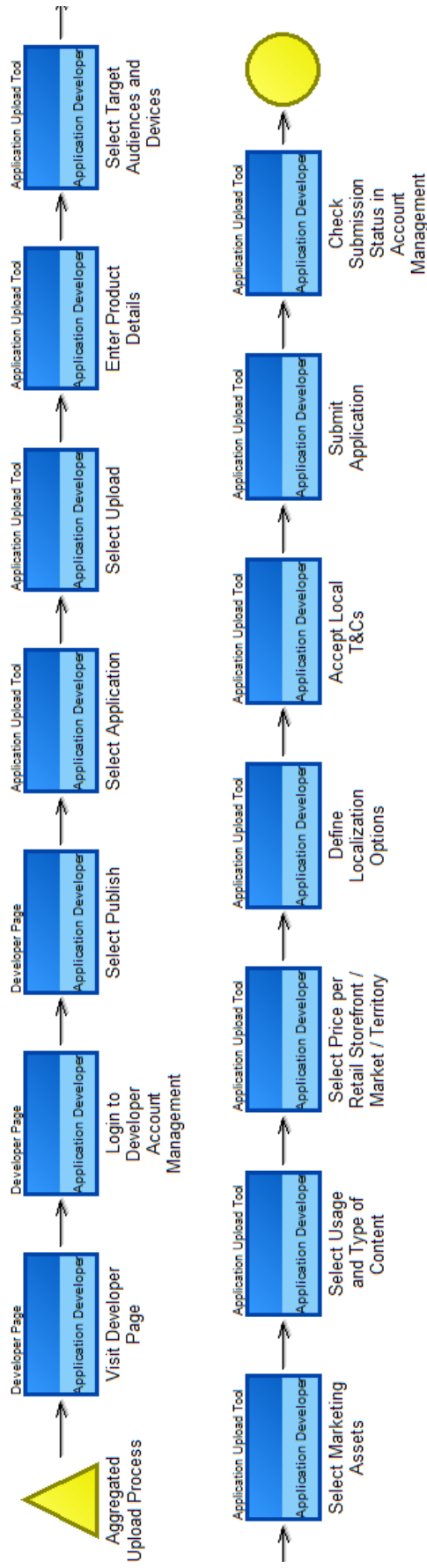


Figure 22 Aggregated Upload Process 1.0

Aggregated Application Distribution 1.0

The aggregated application distribution process should enable the developer to distribute into multiple storefronts through single distribution process. The content should be tested or automatically screened to obey to policies of the appstore and to secure a standard of quality for the end user. This process should not hinder the distribution process in terms of time and resources. The application store should offer automatic categorisation and manual content management by the storefront and developer.

The comparison shows that WAC supports multiple storefronts but verification and distribution is upon the individual storefronts therefore the developer has to rely on each storefront to publish and manage the the application. Apple has a complicated and sometimes difficult approval process that prevents certain applications to be distributed when they breach the policies of the AppStore. Further the Apple environment is an enclosed system where no partner storefronts are able to participate. Android has no verification process within its distribution chain, there all applications go directly to the Android Marketplace which increases the risk of piracy and malware to the enduser. Further monetization is a problem on the Marketplace as copyright protected content is offered free of charge in pirated versions over the original versions which are paid for.

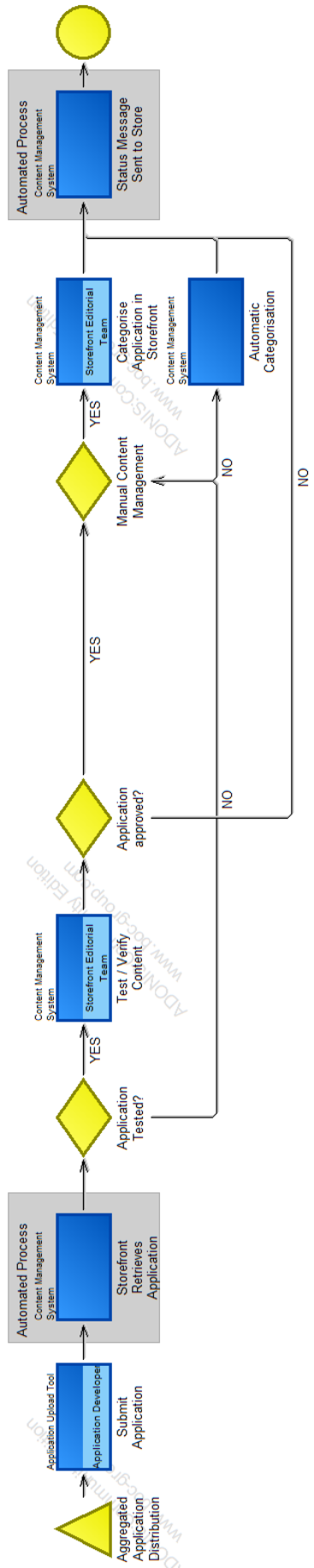


Figure 23 Aggregated Application Distribution 1.0

Aggregated Application Delivery / Purchase 1.0

The delivery in the aggregated model should prevent the popups of security warnings due to the signing in the development process and compatibility of application with the end users device. Further an easy to use payment flow should be standardized over all partner stores to prevent the drop off rate of customers. In addition an automatic process from download to the saving and installation of the application should be achieved to make the process as user friendly as possible to increase adoption of application downloads.

If compared to the existing environments, the WAC process can vary between the storefronts as well as the usability. Apple has developed a userfriendly download process for applications which just require password entry for payment if the customer is a registered iTunes user. Android delivers security warnings before the installation which could cause cancellation by user.

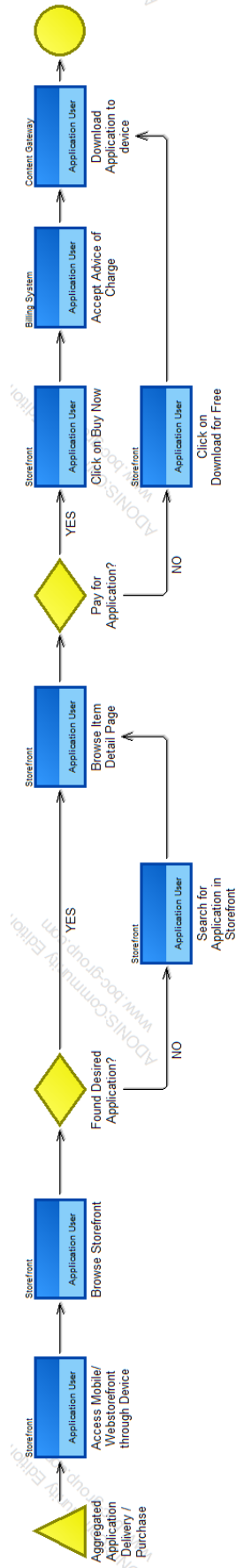


Figure 24 Aggregated Application Delivery / Purchase 1.0

Aggregated Storefront Settlement 1.0

The aggregated storefront settlement process should comprise a solution for multiple storefront reporting and settlement using the Meta platform client which retrieves the accumulated data and payments automatically from the partner storefront and combines these within the Meta platform

By comparing the existing storefronts it showed that WAC is the only Meta platform that can deliver a multiple storefront settlement process. However the accuracy and the process itself are not testable at the time of this thesis

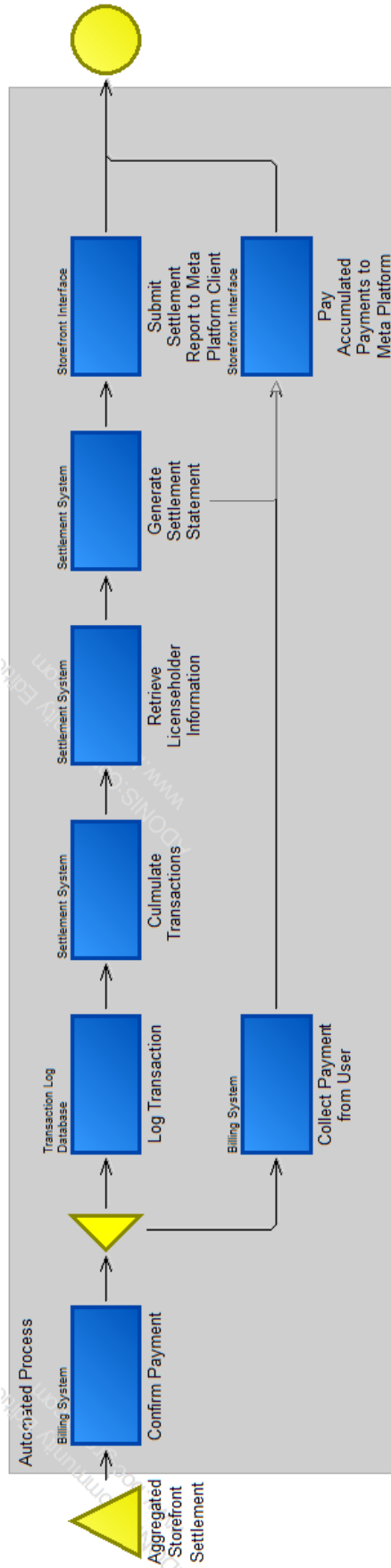


Figure 25 Aggregated Storefront Settlement 1.0

Aggregated Settlement 1.0

The aggregated settlement process should be using the Meta platform which retrieves the accumulated data and payments automatically from the partner storefront and combines these within the Meta platform for accumulated reporting to the developer.

By comparing the existing storefronts it showed that WAC is the only Meta platform that can deliver a multiple storefront settlement process. However the accuracy and the process itself are not testable at the time of this thesis

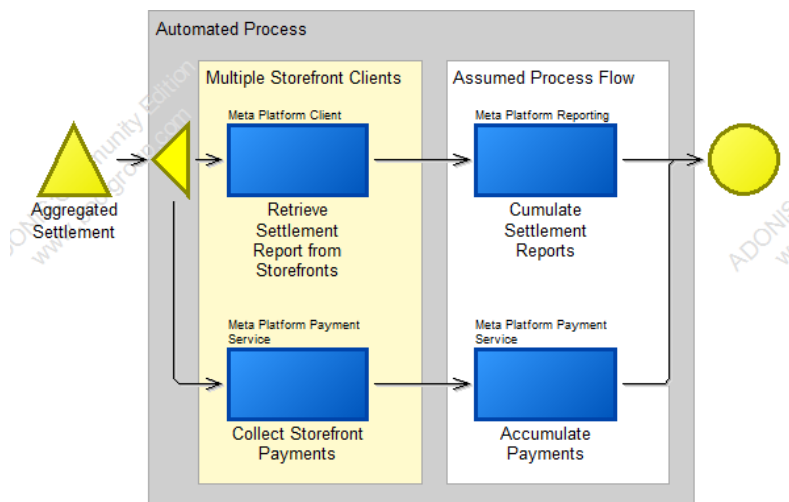


Figure 26 Aggregated Settlement 1.0

Aggregated Royalty Payment 1.0

The aggregated royalty payment should enable payments in a single process to the developer using the accumulated data and payments from multiple partner storefronts royalty payments. Reporting of this data and the settlement statements should be generated automatically per product, territory and partner storefront to give the developer visibility of the performance of his application.

By comparing the existing storefronts it showed that WAC is the only Meta platform that can deliver a multiple storefront settlement process. However the accuracy and the process itself are not testable at the time of this thesis

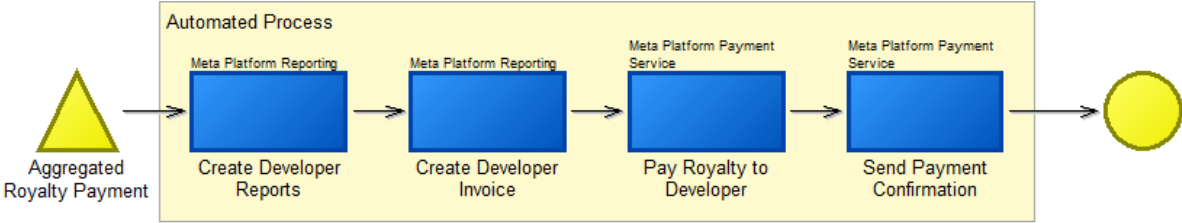


Figure 27 Aggregated Royalty Payment 1.0

Aggregated IT Environment 1.0

The aggregated IT environment should be designed to act as single interface between the developer on one end and the partner store on the other hands. The IT infrastructure needs to allow connection of multiple partner storefronts through a module for all necessary processes of the application distribution including the application portfolio, reporting, payments, settlement and management of the application. Despite the various storefronts there should be a standardization in place to enable an homogenous customer experience and application offer.

The comparison of the existing application stores shows that the WAC is the only meta platform that can deliver this through the WAC client that connects the partner storefronts with the WAC platform. Currently the partner storefront backends vary per provider as no standardization is in place. Apple AppStore and Android Marketplace do not support this compatibility at the time of this thesis.

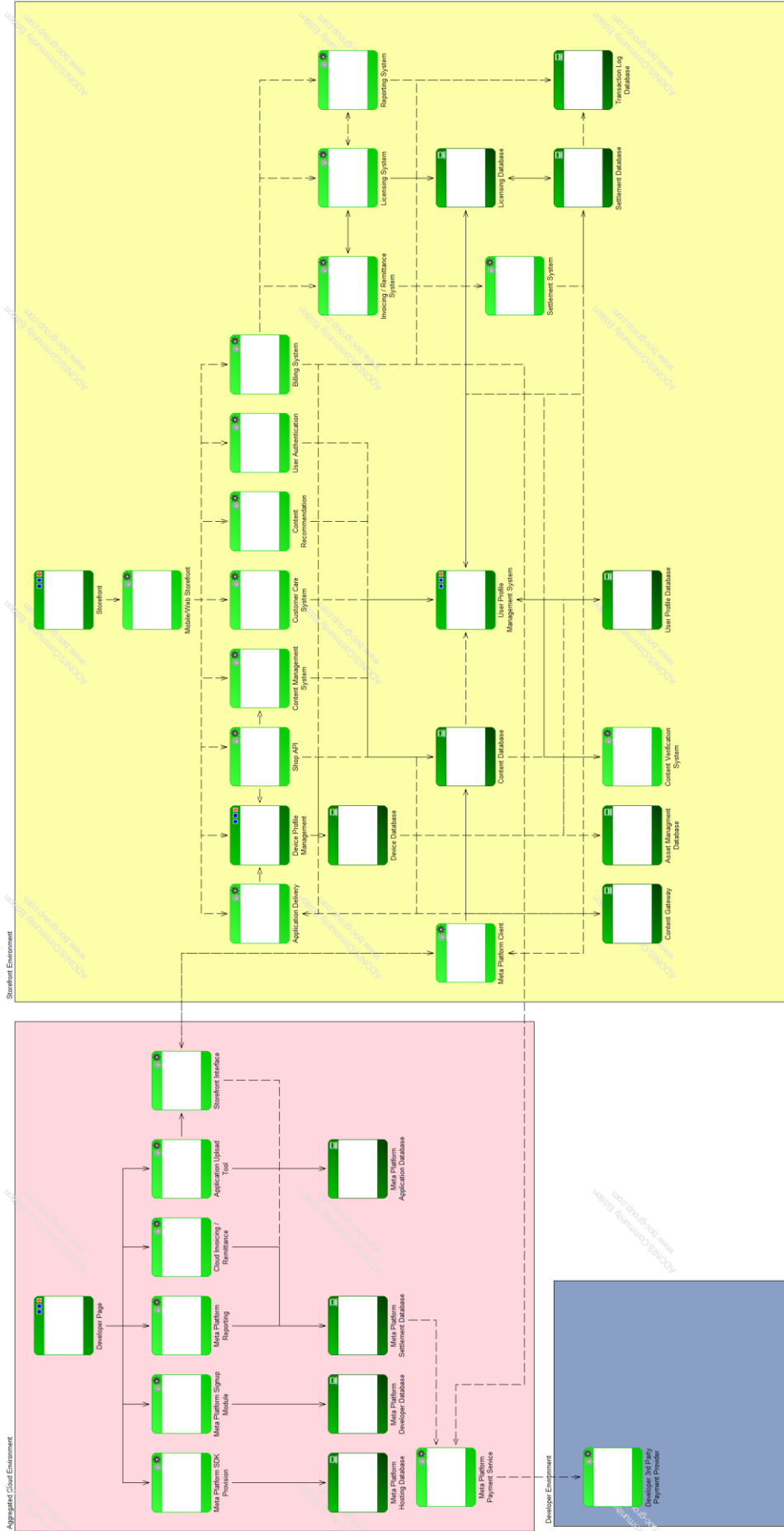


Figure 28 Aggregated IT Environment 1.0

6.7 Infrastructure Analysis

6.7.1 Approach

The research methodology employed is based on a synthesis of literature and case studies related to content distribution, application development and distribution, complemented by a number of interviews with business and system architects of content developers, portal and mobile operators as well as derivation and assumptions based on the business processes modelled in chapter 6. First I looked at open infrastructure concepts found in the literature, to find key characteristics of a multi-platform distribution environment. In a second step I derived from the application store IT models in chapter 6 the infrastructure models which I generated with ADOit® with the ICT infrastructure model type. The comparison between the literature concepts and the derived models should give a final criteria map for the generation of an aggregated infrastructure model.

6.7.2 Open Model Infrastructure Concepts

During my literature research I've found a number of open infrastructure concepts mobile application distribution which are discussed in the following part to derive aggregated concept criteria.

Service Storm

Service Storm is an infrastructure concept discussed in (Yu Chen, et al., 2010). Despite the main focus of Service Storm is to provide web service like applications, this concept could be seen as a possible solution for mobile application distribution by hosting the applications in a cloud like environment that could support multiple platforms and devices. Further it provides an automatic deployment mechanism to support rapid and flexible deployment and scalability adjustment.

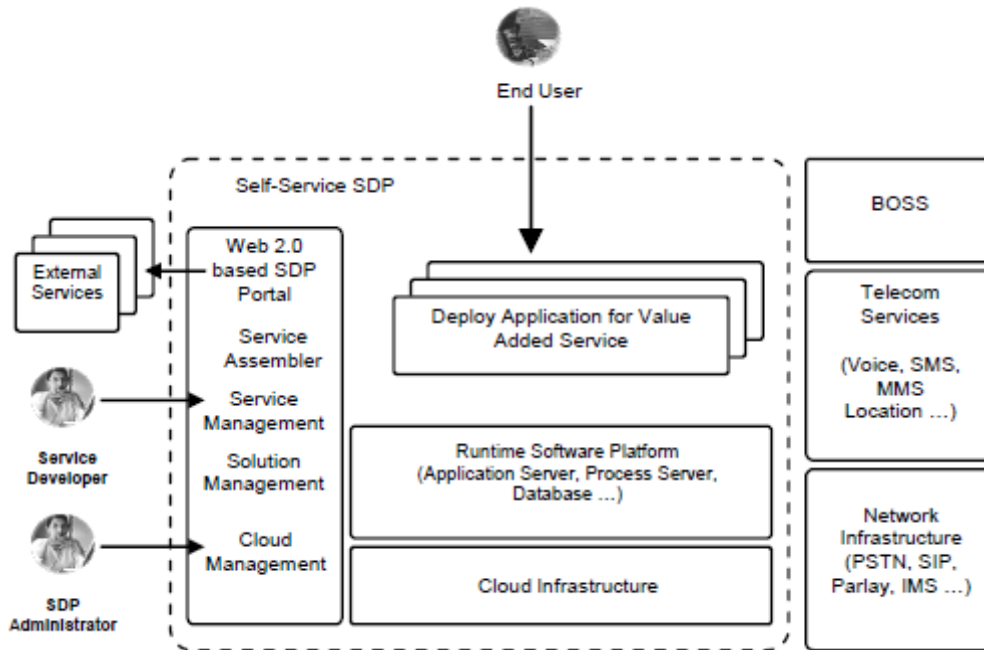


Figure 29 Service Storm Architecture (Yu Chen, et al., 2010)

The architecture of Service Storm show in the above figure consists of these main component parts(Yu Chen, et al., 2010):

Service Assembler: It is a visualized tool for the developer to define the application logic in a drag and drop manner, and integrate the telecommunication services.

Management Components:

Service Management for managing the telecommunication services exposed in SDP, external third party services registered in SDP, and the newly generated applications.

Solution Management for managing the solution implementing the services and deployed in cloud environment.

Cloud Management for managing the computing resources as cloud infrastructure

Runtime Software Platform: Built on cloud environment, it provides the runtime environment to host the applications for value added services. Normally it consists of application servers, process servers, and other runtime environment focusing on handling business rules, events, and operating status.

Cloud Infrastructure: Cloud environment provides a centralized management on virtualized computing resources to provide the underlying capabilities needed by solution deployment

Platform as a Service (PaaS) concept

The Platform-as-a-Service (PaaS) model is an approach for software suppliers that want to focus primarily on the software development cycle and the monetization of new applications, thus bypassing the investment in and maintenance of the underlying infrastructure/services for application design, development, testing, deployment and hosting. PaaS creates a virtual platform for application development and deployment. In PaaS, the system's provider makes most of the choices that determine how the application infrastructure operates. Users build their applications with the provider's on-demand tools and collaborative development environment. PaaS enables centralized cloud computing model by which different roles in the ecosystem are magnetized around value added services, including telecommunication operator, partner for development of value added services, enterprises using telecommunication related application, and individual customers, into a centralized hosting environment (Mitchell, 2008)

Mobile Application Discovery and Acquisition Framework

The framework presented in (Qusay H., et al., 2010) touches upon the areas of mobile application discovery, distribution, and acquisition. The framework discusses marketing across various mobile application distribution channels through the use of a mobile application description schema. The figure below shows the architecture for the proposed mobile application discovery and distribution framework.

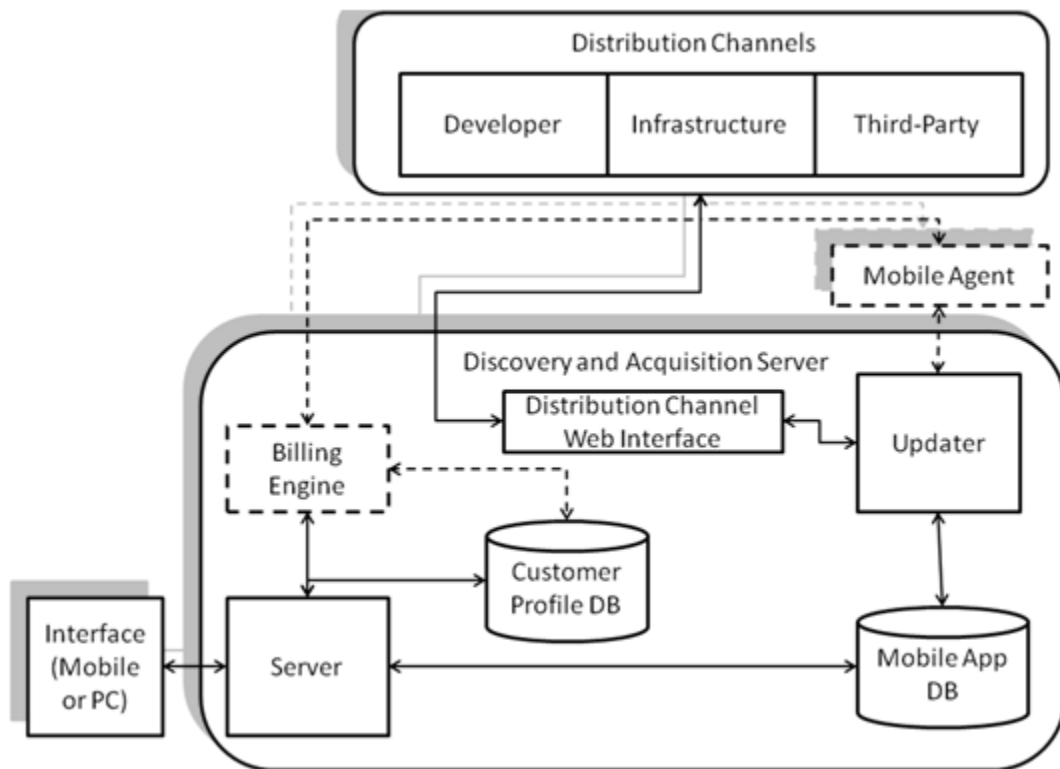


Figure 30 Mobile application discovery and acquisition framework (Qusay H., et al., 2010)

The proposed framework is comprised of several components and processes which are: the billing engine to facilitate payments for mobile applications; the customer profile database to store customer information; the database to store specific information so as to identify the mobile device and consumer allowing him or her to make purchases in the framework; the distribution channel web interface to be used by the distribution channels of mobile applications to provide the framework with the mobile application description schemas as well as the source files for the mobile applications in the case where the framework would be hosting the application; the mobile agent to traverse the multiple mobile application distribution channels and retrieving the information stored in the mobile application description schemas; the mobile application database would store information regarding all the mobile applications available to the framework. Mobile applications could either be stored completely within the framework or could be reference to an external location in case where the mobile agent retrieves the information. Qusay suggests that the mobile application database would contain copies of the mobile application description schemas which would enable an

adaptation to multiple platforms if the key components can be deployed towards the specifics of each operating system supported. The server handles the interaction and requests between the interface and the other components in the framework. The server has direct interaction with the billing engine, mobile application database, and the customer profile database. The updater is responsible for populating the mobile application database in the framework. The updater retrieves the mobile application description schemas and files created by either the mobile agent or the distribution channel web interface and would submit that information into the mobile application database (Qusay H., et al., 2010).

Generic Content Delivery System

The generic content delivery system proposed in (Meng, et al., 2008) is an enterprise server application developed in J2EE. The basic architecture of the System follows a three-tier model which provides a distinct separation between presentation (web tier), business logic (business tier), and data storage (data tier).

The web tier is the presentation layer of the system. The web tier interacts with the web browsers and mobile device micro-browsers through HTTP and WAP gateways. Content providers and subscribers sign into the system through an Internet connection (with appropriate security). Customer care agents and administrators typically access the Content Delivery System through an intranet connection, providing increased security. JSP and servlet containers reside on servers external to the system core and host web content for the Content Provider, Subscriber, and Customer Care web sites. A load balancer controls the traffic on redundant web servers for these sites. In contrast, the Administrator web site is hosted directly on the main servers, using a web server that is tightly integrated with application server.

The business tier implements the business logic of the system and provides a uniform method for business service lookup and creation. The system's core consists of an application server, in which the installation populates the web container with Java Server Pages and servlets. The Enterprise Java Bean container incorporates the generic content delivery system operating and business logic and handles application provisioning and downloads. The application server provides a number of services, such as Java Database Connectivity and Simple

Network Management Protocol. The system core also includes the console for administration. Tightly integrated within the application server, a web server manages HTTP requests and responses. This web server also contains the web content for the Administrator web site. The adapter servers are provided for the integration of the generic content delivery system with other existing systems. The billing adapter and the push services adapter are custom modules that reside on one or more of these adapter servers. The billing adapter integrates the system with your billing applications, while the push services adapter integrates generic content delivery system with push gateway, queries, and similar database operations. The system's database schema is normalizing, minimizing data redundancy and making the overall table structure easier to understand and expand. The database design allows creating highly optimized data indices and retrieval methods through a properly normalized schema.

The generic content delivery system provides four web sites as user interfaces, each for a user role. These four web sites are administration site for carriers, providing site for content providers, personalization site for the subscriber and customer care site for the service representatives.

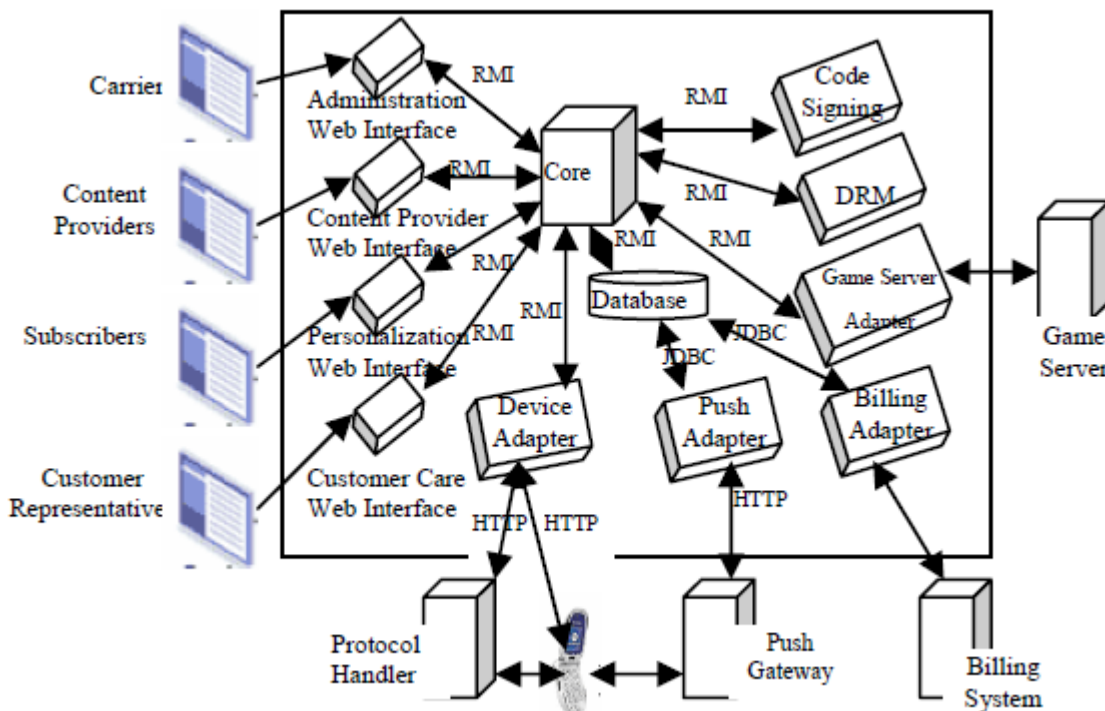


Figure 31 Generic Content Delivery System Architecture (Meng, et al., 2008)

Another component is the WAP portal, which is also one of the most important user interfaces. Mobile users browse the personalized portal, and send requests to system core through WAP gateway and device adapters. The device adapter will take charge of the protocol handling and device specific content or display format adapting. When the media contents and right objects are found in the database, they will be push to the mobile through push adapter and push gateway. The system contains billing adapter and game server adapter, which is used to connect the carriers' billing system and the external game server. The billing adapter will provide enough information for the carriers' billing system to create CDR files. DRM and code signing are also two essential components in the system, which implement the OMA DRM specification and code signing functions(Meng, et al., 2008).

The above discussed distribution concept found in the literature are quite specific in terms of the distribution concept. In all models the content provider or developer is delivering his content directly to the store environment through an interface. This is basically the problem that we are seeing today with the application store distribution. By extracting the main parts of these distribution environments, and introducing a Meta platform like concept into the IT infrastructure the aggregated model should be derived. This is discussed in the next part.

6.7.3 Aggregated Infrastructure Model

In this chapter the aggregated distribution infrastructure model will be discussed from a developers perspective. The applied methodology was a derivation from the analysis and comparison of the modelled infrastructures of the WAC, Apple Appstore and Android Marketplace which can be found in Appendix 2 with the infrastructure concepts in the earlier part. By comparing the infrastructure models in the light of a multiple platform distribution environment I derived concept criteria for the aggregated model. Then I created an evaluation matrix with the findings during the the analysis. The results of this analysis are summarized in the following table by giving the marks 3 GOOD, 2 NEUTRAL, 1 BAD. This table shows the evaluation with the marks per application store against the aggregated concept model as well as the description for the value.

Table 15 Aggregated IT Infrastructure Evaluation Matrix

		Legend	Bad	Neutral	Good		
			1	2	3		
Criteria		WAC	Apple App Store	Android Market place	Aggregated Model Concept	Compliance Comments	
Runtime	Runtime Support in the market	1	2	2	Cross platform support of runtime, and compatibility in existing and future devices	WAC runtimes are not yet finalised and the market penetration is very low; Apple doesn't support cross platform; Android's runtime has highest penetration in the market and runs on multiple device types but only one platform	
Scalability	Modular connection of Storefronts	3	1	2	The distribution to multiple stores and platforms should come from a Meta platform like concept that can connect in a modular way further stores for the distribution	WAC is the only Meta platform like concept in the market with the according infrastructure; Apple's environment doesn't support any other content store than iTunes/ Appstore; Android due to its open system is compatible on other stores but the environment itself is not designed to connect anything else then the Android Marketplace	
Automatic Deployment Mechanism	Compatibility of application source to multiple devices	3	1	2	The IT infrastructure should support multiple OS and multiple devices through the device rendering and detection, as well as supporting file download in standardised source codes.	The WAC infrastructure is device in depended, and the supported file format is base on standardised languages; Apple is only setup to support the iOS devices and platform; Android support multiple devices but only one platform	
Cross Platform Distribution	Distribution flexibility to multiple storefronts	3	1	2	The infrastructure should support a distribution on multiple platforms through a single entry point for the developer	WAC is designed to enable single sign on and multiple distribution, despite currently connected only to operator stores: Apple only allows to distribute through the Appstore; Android Marketplace is supporting only Android; but any other store can make Android files available	

Support of Multiple Operating Systems	OS Compatibility	3	1	2	The infrastructure should be setup in a way that multiple OS types are able to be served with the right content through device detection, rendering and support of standardised content	The WAC infrastructure is device in depended, and the supported file format is base on standardised languages; Apple is only setup to support the iOS platform; Android Marketplace support only one platform, but Android files can be offered on any application store
	Average	2.60	1.20	2.00		

The aggregated IT infrastructure should support multiple OS and multiple devices through the device rendering and detection, as well as supporting file downloads in standardised source codes. The distribution to multiple stores and platforms should come from a Meta platform or cloud like concept that can connect in a modular way further stores for the distribution. Further the infrastructure should support a distribution on multiple platforms through a single entry point for the developer in which he can register, upload and manage the applications, as well as see the reporting and financials across all connected application stores. The next figure shows the aggregated IT infrastructure derived from the extracted application store models as well as the distribution concepts.

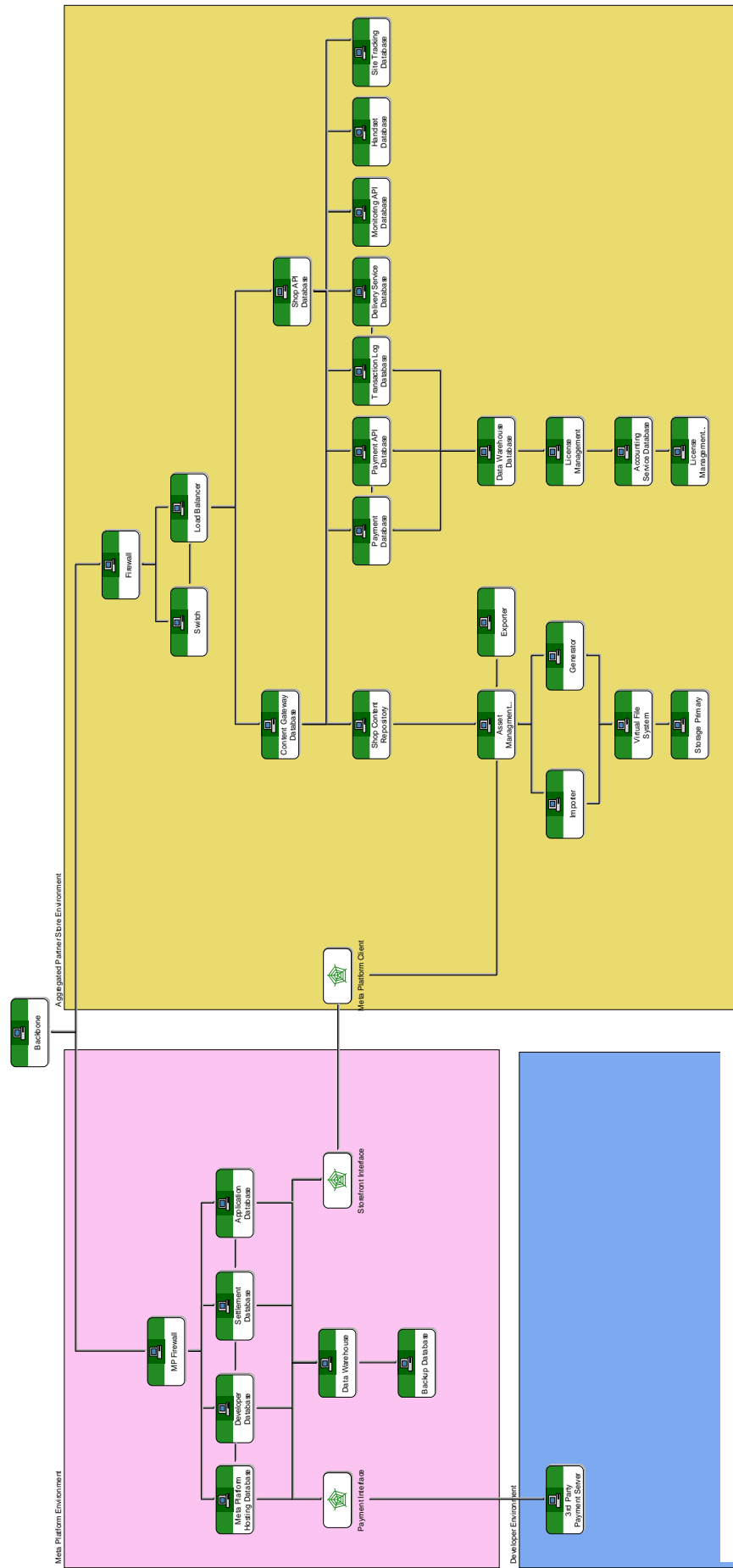


Figure 32 Aggregated Infrastructure Model

7 Multiple Platform Application Development

In this chapter the focus is on the development of applications for multiple platforms. As discussed in earlier chapter a single solution for the development and the distribution of mobile applications is not yet established in terms of the coverage of multiple market relevant platforms. Developers have to decide which platforms to develop for and distribute on. Software companies have seen the need for solutions in that sector therefore have or are developing cross platform application layers, virtual machines, middleware solutions or extended browsers.

In the following I will give an overview of solutions and different tools to use to develop native or cross platform applications

7.1 Platform Approach

This is an overview according to (Baxter-Reynolds, 2011) of development tools for developing native applications for the distribution in only one environment or platform:

<u>Platform</u>	<u>Development Environment</u>
Android	<i>Eclipse, available on Mac, Windows, or Linux with the “Android ADT” plug-in providing extra functionality within Eclipse – included in the Android SDK</i>
iOS	<i>Xcode, available only on Mac</i>
Windows Phone	<i>Visual Studio 2010, available only on Windows</i>
Windows Mobile	<i>Visual Studio 2008, available only on Windows</i>
BlackBerry	<i>Eclipse, available on Mac, Windows, or Linux with the “BlackBerry Java Plug-in for Eclipse” providing extra functionality within Eclipse</i>
HTML via ASP.NET	<i>web site ASP.NET via Visual Studio</i>

7.2 SWOT Analysis of native vs. alternative application development

As described above the alternative to a native mobile application development approach is to design the mobile applications so as to act as a web browser or as a widget. Based on this approach, the mobile applications obtain information and services from a server and relay them to the users by acting as a web browser. It is, however, important to note that a native development approach specific to device/OS would still be better than a web browser-based approach for applications with specific requirement for maximum performance, such as graphic-intensive games and image/audio processing applications. In this regard, it will be important to decide at the beginning of the project whether a native application is an absolute necessity (Anar, et al., 2010).

The next paragraph indicates individual pros and cons of native and web approach and attempts to keep the position of reasonable centre:

Application user friendliness – *web applications can never be as user-friendly as classic native applications. In its beginnings, the HTML code was not designed for these purposes. This insufficiency is now being partially compensated by massive use of the JavaScript technology, which, however, has excessive performance requirements on many technologies. Programming in native code enables the usage of specific features for user interfaces (Wha111).* The design and behaviour of components is thus comprehensible and predictable for the end user and fits to general experience of the device.

Usage of device specific features– *web applications have major limitations in terms of access to specific HW features of mobile devices. Some issues are partially supported by specialised JavaScript libraries, but developers often have no options for using the extending features of a device (Wha111).* Native code and also API standardization enable support for advanced features, such as GPS, integrated cameras etc., integrated directly at its core.

Compatibility between device versions– *native mobile applications are often incompatible among various types and versions of mobile devices. Multiple*

versions of the same applications must be developed and tested to be supported by a wider range of devices. The logic of web languages used for the development of applications enables compatibility for multiple mobile platforms that support the specific runtimes (Wha111).

Developer friendliness – *programming and testing of a native application for mobile devices places high requirements on the expertise of the entire development team, incl. knowledge of the mobile platform itself, the relevant programming code and development environment. Testing a native application is also a rather demanding process, as final tests have to be carried out on the specific mobile device. Development of a mobile application with web technologies (ASP.Net, Java and PHP) doesn't require the need to learn new languages and development teams can use existing resources (Wha111). The web solutions allow as well testing the final application by the developer through a standard web browser.*

Simplicity of app version changing and editing – *distribution to customer end devices and consequent reinstallations of new versions, after logic adjustments, represent a major pitfall of native mobile applications. If there are hundreds of mobile devices operating in the field by the customer, new version update requirements are indeed high. Web technologies are currently improved at a rapid pace. Impressive animations, playing audio and video files, accessing databases etc. is pretty easy to implement (Wha111). Problems facing developers on different platforms*

In the following I will give an overview of solutions and different tools to use to develop native or cross platform applications

7.2.1 Software development platforms

The following software platforms will run on mobile device from different mobile handset manufacturers:

Airplay

Airplay SDK provides the facility to build applications as an entirely OS-agnostic binary file that contains native CPU instructions with the standard languages C/C++, including all features of the language, and unrestricted use of C and C++ standard libraries and STL. It supports deployment to all of the following operating systems: iPhone OS, Android, Samsung Bada, Symbian, Windows Mobile, BREW, Palm/HP webOS, Maemo, and the Khronos OpenKODE Core APIs for OS abstraction (Air11).

Android

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Android includes a set of C/C++ libraries used by various components of the Android system. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management(Android1).

Java 2 Micro Edition

Java 2 Micro Edition (J2ME) is the newest and smallest addition to the Java family. The other members of the Java family are the Java 2 Standard Edition (J2SE) and the Java 2 Enterprise Edition (J2EE). The former is intended for conventional desktop applications development, while the latter one is specifically intended for building distributed applications with emphasis on the server side development and web applications. J2ME is intended to build applications running on mobiles and other embedded devices (Holzer, et al., 2010)

Python Mobile

Python is an ideal prototyping language since it is easy to learn and program and it is possible to save considerable time during program development. Different Python versions exist depending on the mobile OS. The one that we will concentrate on in this paper is PyS60 running on Symbian. Usually Python scripts

are much shorter than the equivalent of C, C++ and Java programs due to several reasons (Holzer, et al., 2010)

Qt

Qt is a cross-platform application framework that runs on desktop OS as Windows, Linux and Mac. On mobile phones Qt run on Symbian and MAEMO. Qt was originally created by Norwegian company Trolltech and acquired by Nokia in June 2008 as stated earlier in this report. Qt provides an intuitive C++ class library with a rich set of application building blocks for C++ development. Qt goes beyond C++ in the areas of inter-object communication and flexibility for advanced GUI development. Since 2005, Qt had a fast development phase which makes it one of the important mobile program languages of nowadays (Holzer, et al., 2010).

Symbian

Symbian is an open source operating system (OS) and software platform designed for smart phones and maintained by Nokia. The Symbian platform is the successor to Symbian OS and Nokia Series 60. The latest OS version is Symbian 3 which includes the Qt framework, which is now the recommended user interface toolkit for new applications. Qt can also be installed on older Symbian devices (Holzer, et al., 2010).

HTML 5

The evolution of the HTML 5 specification will also bring portability of mobile web applications that rival the power of mobile applications. Unfortunately, unless the HTML 5 specification allows web-based applications to leverage advanced Smartphone capabilities, it will lack impact.

The following software platforms will only run on a hardware platform from a specific manufacturer:

BlackBerry

The BlackBerry OS in the majority of devices built-in QWERTY keyboard and supports push e-mail, mobile telephone, text messaging, internet faxing, web browsing and other wireless information services.

iOS

The iPhone, iPod Touch, iPad SDK uses Objective-C, based on the C programming language.

7.2.2 Portable Applications

A burgeoning vendor market is emerging that addresses the challenge of creating cross-platform mobile applications. These solutions provide development tools that allow developers to create a single edition of an application that can be compiled to a native application that targets a specific mobile application platform. The following gives an overview of companies and their tools.

RhoMobile

The tagline “one codebase, every smart phone” pretty much says it all. RhoMobile offers Rhodes, an open source, Ruby-based framework that allows for development of native apps for a wide range of smart phone devices and operating systems. OSes covered include iPhone, Android, Windows Mobile, RIM and Symbian. The framework lets you write your code once and use it to quickly build apps for every major Smartphone. Native apps are said to take full advantage of available hardware, including GPS and camera, as well as location data. In addition to Rhodes, currently in its 2.0 iteration, RhoMobile offers RhoHub, a hosted development environment, and RhoSync, a standalone server that keeps app data current on users’ mobile devices (Jolie O’Dell, 2011).

Appcelerator

Appcelerator’s Titanium Development Platform allows for the development of native mobile, tablet and desktop applications through typical web dev languages such as JavaScript, PHP, Python, Ruby and HTML. Titanium also gives its users access to more than 300 social and other APIs and location information.

Appcelerator's offerings also include customizable metrics for actions and events. App data can be stored in the cloud or on the device, and apps can take full advantage of hardware, particularly camera and video camera capability (Jolie O'Dell, 2011).

WidgetPad

WidgetPad is a collaborative, open-source mobile development environment for creating smart phone apps using standard web technologies, including CSS3, HTML5 and JavaScript. This platform includes project management, source code editing, debugging, collaboration, versioning and distribution. It can be used to create apps for OSes such as iOS, Android and WebOS. (Jolie O'Dell, 2011).

PhoneGap

PhoneGap is an open source development framework for building cross-platform mobile apps for iPhone, iPod, iPad, Android, Palm, Symbian and BlackBerry devices using web development languages such as JavaScript and HTML. It also allows for access to hardware features including GPS/location data, accelerometer, camera, sound and more. The company offers a cross-platform simulator (an Adobe AIR app), as well as online training sessions to help access native APIs and build functioning mobile apps on the PhoneGap platform (Jolie O'Dell, 2011).

MoSync

MoSync is another FOSS cross-platform mobile application development SDK based on common programming standards. The SDK includes tightly integrated compilers, runtimes, libraries, device profiles, tools and utilities. MoSync features an Eclipse-based IDE for C/C++ programming. Support for JavaScript, Ruby, PHP, Python and other languages are planned. The framework supports a large number of OSes, including Android, Symbian, Windows Mobile and even Moblin, a mobile Linux distro. Support for iPhone and Blackberry is in development (Jolie O'Dell, 2011).

Whoop

The Whoop Creative Studio is a WYSIWYG web editor that allows dragging and dropping mobile app elements. Once done, export an app in formats for several

devices and operating systems, including iPhone, Android, RIM, Windows Mobile and other OSes (Jolie O'Dell, 2011).

iPFaces

iPFaces is the framework for simple creation of native, form-oriented network applications for mobile devices. The aim of the solution is to screen the programmer completely out from the mobile platform itself, and transfer the entire application logic to central application server level.

Each iPFaces application consists of two main parts:

- *Thin iPFaces client for mobile devices*
- *The server part with application logic and definition of iPFaces views (Wha111)*

Unify

Unify allows the programming of applications for smart phones, tablets, desktops and other web enabled devices with a single technology stack. The main claim of Unify is to allow solutions that users can not differentiate from natively programmed applications. Unify makes use of several existing frameworks and technologies. At its core it is based on web technologies. HTML5 and CSS 3 are two of the essential ingredients. The whole UI is generated by JavaScript. Unify makes use of the qooxdoo framework for professional JavaScript development. Supports the iOS and Android and support for BlackBerry OS 6.0, Windows Phone and Nokia's operating systems is planned for 2011 (Nat11). PhoneGap is an open source software used to publish applications in the Apple AppStore and Android Market. It is currently the only framework that supports iOS and the Android Market and more interesting due to the acceptance in the market for application developers than initiatives like WAC.

Grapple

Grapple has created a development environment for building native mobile phone applications with standard web technologies - HTML, CSS and Javascript (Gra11).

MotherApp

Enables web developers to create native mobile apps on iPhone, Android and Blackberry using HTML instead of the mobile SDKs. The MotherApp Engine, converts HTML with special mark-ups, based on MotherApp HTML, a subset of HTML, into native apps for every major mobile platform (App11).

8 Perfect Egg - Application implementation

In this chapter of the thesis the development of an example application will be done using the WAC SDK to proof cross platform compatibility incorporating findings of the prior concept extractions. As described in the earlier chapters is the WAC environment at the time of this thesis the only cross platform layer that supports distribution through a meta-platform to multiple storefronts hosted in various locations and from different companies. However it is in its early stage an full extend is yet to be demonstrated

8.1 Installation – Developer Environment

This section provides an overview of WAC's mobile widget development environment. It describes how to use the SDK to create mobile widgets that conform to WAC's specification, which are composed of HTML, CSS and JavaScript files.

8.2 Platform requirements

This part should give an overview on what is necessary in terms of workstation setup to develop with the WAC SDK.

The WAC SDK supports the following Operating Systems: Windows, MAC, Linux

Furthermore the SDK doesn't come with all necessary tools included therefore it is necessary to download and install the following development environments:

- Eclipse 3.5.1 (Galileo)
- Eclipse WTP 3.2.0 plug-in (included in most Web tools packages)
- JDK 5 (JRE alone is not sufficient)

8.2.1 WAC SDK

WAC's mobile widget SDK offers an integrated environment which allows developing, debugging and deploying your widgets. It is based on the industry standard Eclipse Software Development Environment. The editors in the SDK are customized to facilitate the code writing, with a handy debug tool. The WAC SDK contains a handset emulator that helps to test and verify the widget. The SDK file is an all in one package which contains documents, sample widget, emulator, and eclipse IDE with widget development plug-ins. Only the Java Development Kit (JDK) is needed as well which includes the Java Runtime Environment (JRE).

8.2.2 Eclipse

Eclipse Release 3.5.2

Most of the Eclipse SDK is "pure" Java code and has no direct dependence on the underlying operating system. The chief dependence is therefore on the Java Platform itself. Portions are targeted to specific classes of operating environments, requiring their source code to only reference facilities available in particular class libraries (e.g. J2ME Foundation 1.0, J2SE 1.3 and 1.4, etc.).

In general, the 3.5 release of the Eclipse Project is developed on a mix of Java 1.4, Java 5 and Java 6 VMs. As such, the Eclipse SDK as a whole is targeted at all modern, desktop Java VMs. Full functionality is available for 1.4 level development everywhere, and extended development capabilities are made available on the VMs that support them (Eclipse, 2010).

Running Eclipse

After installing the Eclipse SDK in a directory, you can start the Workbench by running the Eclipse executable included with the release (you also need a 1.4.2 JRE, not included with the Eclipse SDK). On Windows, the executable file is called `eclipse.exe`, and is located in the `eclipse` sub-directory of the install. If installed at `c:\eclipse-SDK-3.5-win32`, the executable is `c:\eclipse-SDK-3.5-win32\eclipse\eclipse.exe` (Eclipse, 2010).

8.2.3 Typical widget structure

A widget package is a zip-compressed file with a “.wgt” extension. The archived widget consists of the following key files (WAC10):

- *config.xml*, which provides the widget’s essential meta-data, including name, ID, icon, size, required handset APIs, etc.
- *author-signature.xml* when signing is required, which contains the digital signature identifying the widget as trusted.
- *index.html*, the default start index file that is to be rendered when the widget is activated. Note that an alternate file name can be specified.
- JavaScript files (optional), containing scripts that can optionally be executed by the widget.
- CSS files (optional), cascading style sheet files defining visual display attributes for the widget.
- Local directories (optional), specially named directories that can contain local-specific config.xml and content files, including the index start file.

The config.xml file follow the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<widget xmlns="http://www.w3.org/ns/widgets"
xmlns:JIL="http://www.jil.org/ns/widgets1.2"
id="http://jil.org/wid/$(echo -n YOURNICKNAME | tr '[:upper:]'
'[:lower:]' | sha256sum)/ForExample"
version="1.0">
<name>Example widget</name>
<JIL:access network="true"/>
</widget>
```

8.2.4 ID Attribute

The Widget tag’s “id” attributes is optional according to the W3C specification. However in practice, the WAC portal requires the attribute to be present in any widgets that it ingests. Therefore, the “id” attribute should be treated as mandatory. The WAC SDK typically assigns the value (WAC10).

The id is a URI that includes the user’s portal nickname encrypted using SHA256 as well as the URL-encoded widget’s name. For example, if the developer’s nickname is ‘Pete69’ and the widget name is ‘WidgetPerfectEgg’ the widget ID is the following:

8.2.5 Perfect Egg – An application description

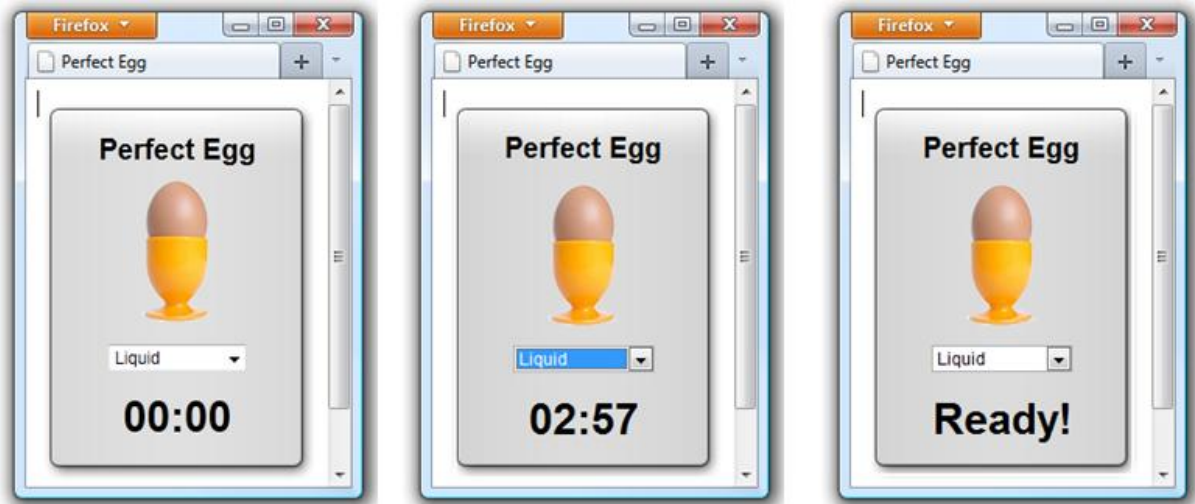
The application development in this work is predominantly used to analyse the compatibility through multiple operating systems with one single built and the distribution onto multiple application stores for the download. Therefore the emphasis is on the distribution rather than the development of the application. I kept the development and complexity of the application simple without using the WAC APIs such as accelerometer or messaging by developing a utility application that stops the cooking time designed to cook eggs in your favourite way.

The basic application should be able to:

- display text
- display input fields
- use of a count-down timer

The application's functionality is to give the user the choice of three different cooking times for the egg – liquid at 3 minutes, soft at 4 minutes and hard at 8 minutes. The cooking times are set values within the application but are not scientifically researched. When the preferred cooking time is chosen out the three values the countdown timer starts to count down in second steps and when the countdown is over the message "Ready" is displayed.

Below is a visualisation of the final application work flow emulated in Mozilla Firefox 4.0.1:



8.2.6 Development and source code

In this chapter the file structure and source code of the application is displayed.

File structure

Name	Type
.settings	File Folder
bin	File Folder
css	File Folder
img	File Folder
js	File Folder
.project	PROJECT File
config.xml	XML Document
index.html	HTML File
WrapperIncludes.js	JS File

Source Code of the application

Config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<widget xmlns="http://www.w3.org/ns/widgets"
  xmlns:jil="http://www.jil.org/ns/widgets1.2"
  xmlns:its="http://www.w3.org/2005/11/its"

id="http://jil.org/wid/c6e675421d8a8d1102c4065d3c99df1adba041ce9ddb38dfd9
058285b9cca54d/Perfect%20Egg"
  version="1.0"
  height="315"
  width="220">
  <icon src="img/icon.jpg" width="0" height="0"/>
  <content src="index.html"/>
  <jil:access network="true"/>
```

```

<name>Perfect Egg</name>
<description>How to make your perfect Egg!</description>
<author
    href="peter.bacher@aon.at">Peter Bacher</author>
</widget>

```

Index.html

```

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>
      Perfect Egg
    </title>
    <link type="text/css" href="css/style.css" rel="stylesheet" /><script
xml:space="preserve"></script><script
type="text/javascript" src="WrapperIncludes.js"
xml:space="preserve"></script><script
type="text/javascript" src="js/script.js"
xml:space="preserve"></script>
  </head>
  <body onload="onload()">
    <div id="mainView">
      <div id="mainEditableContent">
        <table cellpadding="2" border="0" cellspacing="0" width="100%">
          <tr>
            <td valign="middle" align="center">
              <div id="lblUeberschrift">
                Perfect Egg
              </div>
            </td>
          </tr>
          <tr>
            <td valign="middle" height="120" align="center">
              
            </td>
          </tr>
          <tr>
            <td valign="middle" align="center">
              <select id="lstAuswahl" onchange="lstAuswahl_onchange()"
name="lstAuswahl">
                <option value="-1">
                  Please select
                </option>
                <option value="180">
                  Liquid
                </option>
                <option value="240">
                  Soft
                </option>
                <option value="480">
                  Hard
              </option></select>
            </td>
          </tr>
          <tr>
            <td valign="middle" height="60" align="center">
              <div id="lblZeit">

```

```
        00:00
    </div>
</td>
</tr>
</table>
</div>
</div>
</body>
</html>
```

Style.css

```
/* Basic styles for the front and back of the widget */
#mainView {
    display: block;
    width: 200px;
    height: 300px;
    background: transparent url(../img/200_300_milk.png) 0px 0px no-
repeat;
}
#mainEditableContent {
    position: absolute;
    top: 35px;
    left: 7px;
    background: transparent;
    height: 300px;
    width: 200px;
}
#mainView #mainEditableContent #lblZeit {
    font-family: "Arial";
    font-size: 30px;
    color: #000000;
    font-weight: bold;
}
#mainView #mainEditableContent #lblUeberschrift {
    font-family: "Arial";
    font-size: 20px;
    color: #000000;
    font-weight: bold;
}
#mainView #mainEditableContent #lstAuswahl{
    font-family: "Arial";
    font-size: 12px;
    color: #000000;
}
}
```

Java script – script.js

```
var zeit = -1;

function onload(){
    //window.resizeTo(400, 400);
    window.setInterval("ausgabe()", 1000);
}

function lstAuswahl_onchange() {
    var lstAuswahl = document.getElementById("lstAuswahl");
```

```

    wert = lstAuswahl.options[lstAuswahl.selectedIndex].value;
    zeit = wert;
}

function ausgabe() {
    var lblZeit = document.getElementById("lblZeit");

    if (zeit < 0) {
        lblZeit.innerHTML = "00:00";
    }
    else if (zeit == 0) {
        lblZeit.innerHTML = "Ready!";
    }
    else {
        minuten = parseInt(zeit / 60);
        sekunden = zeit % 60;

        lblZeit.innerHTML = format(minuten) + ":" + format(sekunden);

        zeit--;
    }
}

function format(zahl) {
    if (zahl < 10) {
        return ("0" + zahl);
    }
    else {
        return (zahl);
    }
}

```

Distribution Test

At the time of this thesis the commercial launch of the WAC environment has been achieved, nevertheless the support in devices and application stores is not yet given to test the distribution mechanism of the WAC environment.

Device compatibility test

During the time of this thesis the first prototypes of runtimes have been developed and provided by WAC to its developers for testing purposes. The runtimes are developed by the companies Borqs, Obigo, and Opera and do mainly support the device OS Android 2.2+ and Android 2.1+, but only with the limited compliant devices: Samsung Galaxy S and GT-I7680; Motorola MT716, MT810 and MT 820; SonyEricsson A8i; Dell Mini 3v, HTC Desire, HTC Nexus One.

WAC certifies compliance for a WAC runtime on a device, so runtime vendors and OEMs have to submit a compliance test report for every device they want certified with a runtime. The list of Compliant Devices typically reflects devices used during development of the runtime, and is then extended as mobile network operators and device manufacturers request additional devices.

By using the Opera widget emulator and runtime the compatibility was tested for Android was successfully tested:



Figure 33 Perfect Egg Compatibility Test Opera Widget Emulator

9 Conclusion

Developers still have to prioritize their resources by deciding for which platform to develop for and manage the distribution in a silo approach per application store. Attempts of developer communities like WAC to use and convert existing web standards do seem like to have a huge potential but aren't to the time of this thesis ready for the market nor cover an acceptable amount of platforms. From the developers' point of view, having diverse actors in the mobile market will boost the need for standardization of mobile hardware and software protocols further. Both software developers and mobile manufacturers will benefit from this process. While software companies will develop cross platform application layers, virtual machines, middleware solutions or extended browsers, mobile device manufacturers will try to facilitate the development of applications for their own operating systems. This can be achieved by first resolving any compatibility issues between different OS versions and also by facilitating developers' effort to develop and deploy their applications. Furthermore, open source operating systems such as Android may also affect the strategies of the existing platforms to be more open and offer richer tools/services to facilitate application development. The discussed Meta platform approach could be a solution to overcome the resource issue of the development and management of the distribution where the developer uses a specified SDK to program the application, publishes the application once to the Meta platform. Application stores that are connected to the Meta platform can distribute this application in their environment to their users. Compatibility across multiple platforms could be achieved by using standardized technologies across the value chain. However, the setup and commercialization of such an approach is a huge task and needs the involvement of all actors in the application eco system and the requirements and processes discussed in this paper could be used to give a foundation to such a project. Furthermore, the evolution of the HTML 5 specification will also bring portability of mobile web applications that rival the power of mobile applications if the Smartphone APIs and network APIs can be made available with this environment the distribution runs then directly from the

developer to the customer excluding the application stores at all. This is certainly content for further research on this topic.

10 List of Figures

Figure 1: Common characteristics of application distribution	13
Figure 2: Structural approach of the thesis	14
Figure 3: Historical development of cell phones (Speckmann, 2008).	17
Figure 4 Mobile Application Ecosystem (compare: (Kirk Knoernschild, 2010)	26
Figure 5 Widget Landscape W3C (Marcos Caceres, 2008).....	29
Figure 6: Numeric overview of application stores	38
Figure 7 E-BPMS Metamodel – Conceptual View (Kühn, et al., 2001)	50
Figure 8 Meta Model Application Distribution adapted from (Holzer, et al., 2010) .	52
Figure 9 Current Cross Platform Distribution Scenario	53
Figure 10 Cross Platform Distribution Scenario with Meta Platform.....	54
Figure 11 Enabler Platform Model (Vânia, et al., 2010)	55
Figure 12 System Integrator Platform Model (Vânia, et al., 2010)	56
Figure 13 Neutral Platform Model (Vânia, et al., 2010).....	57
Figure 14 Broker Platform Model (Vânia, et al., 2010).....	58
Figure 15 Telco centric model (Pieter, et al., 2008)	59
Figure 16 Device centric model (Pieter, et al., 2008)	60
Figure 17 Aggregator Centric model (Pieter, et al., 2008).....	61
Figure 18 Aggregated Application Distribution 1.0	79
Figure 19 Aggregated Roles 1.0	80
Figure 20 Aggregated Development Environment Setup 1.0.....	81
Figure 21 Aggregated Registration Process 1.0	82
Figure 22 Aggregated Upload Process 1.0	84
Figure 23 Aggregated Application Distribution 1.0.....	86
Figure 24 Aggregated Application Delivery / Purchase 1.0.....	88

Figure 25 Aggregated Storefront Settlement 1.0	90
Figure 26 Aggregated Settlement 1.0	91
Figure 27 Aggregated Royalty Payment 1.0	92
Figure 28 Aggregated IT Environment 1.0	93
Figure 29 Service Storm Architecture (Yu Chen, et al., 2010)	95
Figure 30 Mobile application discovery and acquisition framework (Qusay H., et al., 2010).....	97
Figure 31 Generic Content Delivery System Architecture (Meng, et al., 2008).....	99
Figure 32 Aggregated Infrastructure Model	103
Figure 33 Perfect Egg Compatibility Test Opera Widget Emulator	121
Figure 34 Apple Appstore Application Distribution 1.0	138
Figure 35 Apple Appstore Roles 1.0	139
Figure 36 Apple Appstore Documents 1.0	140
Figure 37 Apple Appstore Development Environment Setup 1.0	140
Figure 39 iOS Developer Program Registration 1.0.....	141
Figure 38 Apple Developer Registration 1.0	141
Figure 40 Apple Appstore Upload Process 1.0	142
Figure 41 Apple Appstore Upload Preparation Process 1.0.....	142
Figure 42 Apple Appstore Application Loader Upload Process 1.0	143
Figure 43 Apple Appstore Application Distribution 1.0	143
Figure 44 Apple Appstore Application Delivery / Purchase 1.0	144
Figure 45 Apple Appstore Settlement 1.0	144
Figure 46 Apple Appstore Royalty Payment 1.0	145
Figure 47 Apple Appstore IT Environment 1.0	146
Figure 48 Apple Appstore IT Infrastructure	147
Figure 49 Android Marketplace Application Distribution 1.0.....	148

Figure 50 Android Marketplace Roles 1.0.....	149
Figure 51 Android Marketplace Documents 1.0.....	149
Figure 52 Android Marketplace Development Environment Setup 1.0.....	150
Figure 53 Android Marketplace Registration Process 1.0.....	152
Figure 54 Android Marketplace Upload Process 1.0.....	152
Figure 55 Android Marketplace Application Distribution 1.0.....	152
Figure 56 Android Marketplace Application Delivery / Purchase 1.0	153
Figure 57 Android Marketplace Settlement 1.0.....	154
Figure 58 Android Marketplace Royalty Payment 1.0	154
Figure 59 Android Marketplace IT Environment 1.0.....	155
Figure 60 Android Marketplace IT Infrastructure.....	156
Figure 61 WAC Application Distribution 1.0	157
Figure 62 WAC Roles 1.0	158
Figure 63 WAC Documents 1.0	158
Figure 64 WAC Development Environment Setup 1.0.....	159
Figure 65 WAC Registration Process 1.0	159
Figure 66 WAC Upload Process 1.0	160
Figure 67 WAC Application Distribution 1.0	161
Figure 68 WAC Application Delivery / Purchase 1.0.....	162
Figure 69 WAC Storefront Settlement 1.0.....	162
Figure 70 WAC Settlement 1.0	163
Figure 71 WAC Royalty Payment 1.0	163
Figure 72 WAC IT Environment 1.0	164
Figure 73 WAC IT Infrastructure	165

„Ich habe mich bemüht, sämtliche Inhaber der Bildrechte ausfindig zu machen und ihre Zustimmung zur Verwendung der Bilder in dieser Arbeit eingeholt. Sollte

dennoch eine Urheberrechtsverletzung bekannt werden, ersuche ich um Meldung bei mir.“

11 List of Tables

Table 1 Smartphone Operating Systems and Development Environments (Kirk Knoernschild, 2010)	25
Table 2 Comparison Table Mobile Widgets, Browser Applications, Native Applications	28
Table 3 Classification of Mobile Applications (Anar, et al., 2010)	31
Table 4 Analysis Development Environment Setup	64
Table 5 Analysis Application Development	65
Table 6 Analysis Registration	66
Table 7 Analysis Application Upload.....	67
Table 8 Analysis Application Distribution	68
Table 9 Analysis Application Delivery / Purchase	70
Table 10 Analysis Storefront Reporting & Settlement	71
Table 11 Analysis Multiple Storefront Settlement.....	71
Table 12 Analysis Royalty Payments.....	72
Table 13 Analysis IT Infrastructure	73
Table 14 Aggregated Distribution Model Evaluation Matrix	75
Table 15 Aggregated IT Infrastructure Evaluation Matrix.....	101

12 Bibliography

3G - Encyclopedia [Online] // experiencefestival.com. - 07 04 2011. - <http://www.experiencefestival.com/a/3G/id/1901152>.

ABiresearch www.abiresearch.com [Online] // www.abiresearch.com. - ABiresearch, 03 March 2011. - 04 04 2011. - [http://www.abiresearch.com/research/1007219-](http://www.abiresearch.com/research/1007219-Mobile+Data+Services+More+Valuable+than+Mobile+Voice+Services+for+40%25+of+Mobile+Business+Customers)

Mobile+Data+Services+More+Valuable+than+Mobile+Voice+Services+for+40%25+of+Mobile+Business+Customers.

Agar Jon Constant Touch - A Global History of the mobile phone [Book]. - [s.l.] : Icon, 2004.

Airplay [Online] // Airplay SDK. - Airplay. - 07 04 2011. - www.airplaysdk.com.

Allan Hammershoj, Antonio Sapuppo and Reza Tadayoni Challenges for Mobile Application Development [Report]. - Copenhagen, Denmark : Center for Communication, Media and Information Technologies (CMI) Aalborg University, 2010.

Anar Gasimov [et al.] Visiting Mobile Application Development: What, How and Where [Conference] // 2010 Ninth International Conference on Mobile Business / 2010 Ninth Global Mobility Roundtable. - [s.l.] : IEEE, 2010. - Vols. DOI 10.1109/ICMB-GMR.2010.20.

Android Market [Online] // Android. - Google Inc.. - 2010 йил 13-12. - <http://www.android.com/market/#app=basesign.alltie>.

Apple [Online] // iOS Overview. - Apple Inc.. - 2010 йил 12-12. - <http://developer.apple.com/>.

Apple Inc. Apple's iOS 4.2 Available Today for iPad, iPhone & iPod touch [Online] // Apple.com. - Apple Inc, 2010 йил 22-11. - 2010 йил 16-12. - <http://www.apple.com/pr/library/2010/11/22ios.html>.

Apps that build your business [Online] // Motherapp. - Motherapp. - 07 04 2011. - <http://www.motherapp.com/>.

Baxter-Reynolds Matthew Cracking Windows Phone and BlackBerry Native Development: Cross-Platform [Book]. - [s.l.] : apress, 2011.

Bayer Franz [et al.] E-BPMS: Ein Modellierungs-Framework für E-Business-Anwendungen [Conference] // GI-Jahrestagung "Informatik 2001". - Vienna : [s.n.], 2001.

Bill Ray theregister [Online] // theregister. - theregister, 09 Feb 2009. - 04 04 2011. - http://www.theregister.co.uk/2009/02/09/omtp_bondi/.

Cáceres Marcos w3.org [Online] // <http://www.w3.org/TR/widgets-reqs/>. - W3C, 2010 йил. - 2010 йил 15-12. - <http://www.w3.org/TR/widgets-reqs/>.

Canalys [Online] // Canalys.com. - Canalys. - 2010 йил 12-12. - <http://www.canalys.com/pr/2010/r2010111.html>.

Copeland Rebecca Telco App Stores - Friend or Foe? [Journal]. - [s.l.] : Core Viewpoint Limited / IEEE, 2010 йил.

Daum Adam Canalys [Online] // canalys.com. - 2010 йил. - 2010 йил 12-12. - <http://www.canalys.com/pr/2010/p2010091.html>.

Dunnewijk Theo et al. A Brief History of Mobile Telecommunication in Europe [Journal]. - Maastricht, The Netherlands : United Nations University - Maastricht Economic and social Research and training centre on Innovation and Technology, 2006 йил. - #2006-034.

Eclipse Eclipse Project Release Notes [Online] // eclipse.org. - Eclipse, 07 04 2010. - 07 04 2011. - http://www.eclipse.org/eclipse/development/readme_eclipse_3.5.1.html.

gizmodo [Online]. - gizmodo. - 2010 йил 15-12. - <http://gizmodo.com/5199933/giz-explains-all-the-smartphone-mobile-app-stores>.

Google [Online] // **Android Developers.** - Google Inc.. - 2010 йил 12-12. - <http://developer.android.com/guide/basics/what-is-android.html>.

Google Inc. Android 2.3 Platform Highlights [Online] // Android Developers. - Google Inc., 2010 йил. - 2010 йил 16-12. - <http://developer.android.com/sdk/android-2.3-highlights.html>.

Grapple Technology [Online] // Grapplemobile. - Grapplemobile. - 07 04 2011. - <http://www.grapplemobile.com/team/>.

gsma.securespsite.com [Online] // GSMA. - GSMA. - 04 04 2011. - <https://gsma.securespsite.com/access/default.aspx>.

Hall Curtis and Harmon Paul The 2005 Enterprise Architecture, Process Modeling & Simulation Tools Report [Report]. - [s.l.] : BUSINESS PROCESS TRENDS, November, 2005.

Harmon Paul The BPTrends 2010 BPM Software Tools Report on BOC's Adonis Version 4.0 [Report]. - [s.l.] : BPTrends, 2010.

Holzer Adrian and Ondrus Jan Mobile application market: A developer's perspective [Journal]. - Lausanne, Switzerland : Elsevier: Telematics and Informatics, 2010 йил. - Vol. 28.

Jolie O'Dell 5 Cross-Platform Mobile Development Tools You Should Try [Online] // dumaslab.com. - dumaslab.com, 2011. - 07 04 2011. - <http://dumaslab.com/2011/09/5-cross-platform-mobile-development-tools/>.

Kaar Christian An introduction to Widgets with particular emphasis on Mobile Widgets [Report]. - Hagenberg, Austria : University of Applied Sciences, 2007.

Kirk Knoernschild Market Profile: Rich Mobile Application Platforms for the Smartphone 2010 [Report]. - [s.l.] : Burton Group, 2010.

Korhonen Juha Introduction to 3G mobile communications [Book]. - Norwood : ARtech House, Inc., 2003. - Vol. 2nd ed..

Kühn H., Herbst J. and Schwarz S. Managing Complexity in E-Business" in Baake, U. F [Conference] // Proceedings of the 8th European Concurrent Engineering Conference 2001. - Valencia, Spain : Society of Computer Simulation, 2001. - Vols. pp. 6-11, April 18-20, .

Marcos Caceres Widgets 1.0: The Widget Landscape [Online] // w3.org. - w3.org, 14 04 2008. - 07 04 2011. - <http://www.w3.org/TR/widgets-land/>.

Meng He, Zhao Zheng and Geng Xiaoqing Generic Content Delivery System Research [Conference]. - Tianjin : IEEE, 2008.

Michael Wei China issues 3G licences to main carriers [Online] // uk.reuters.com. - uk.reuters.com, 2009. - 02 April 2011. - <http://uk.reuters.com/article/2009/01/07/oukin-uk-telecoms-china-idUKTRE5061KP20090107>.

MING-CHUN CHENG and SHYAN-MING YUAN An Adaptive and Unified Mobile Application Development Framework for Java [Journal]. - Taiwan : JOURNAL OF INFORMATION SCIENCE AND ENGINEERING, 2007. - 23.

Mitchell D. Defining Platform-As-A-Service, or PaaS [Online] // blogs.bungeeconnect.com. - 2008. - June 2011. - <http://bungeeconnect.wordpress.com/2008/02/18/defining-platform-as-a-service-or-paas/>.

Mobile Application Architecture Guide [Online]. - 2009 йил. - 2010 йил 12-12. - <http://apparch.codeplex.com/releases/view/19798>.

Morisson Ben The Symbian OS Architecture Sourcebook: Design and Evolution of a Mobile Phone OS [Book]. - [s.l.] : John Wiley & Sons, 2007.

Native-like applications for smartphones, tablets and desktops [Online] // Unify. - Unify. - 07 04 2011. - <http://unify-training.com/unify/>.

OMTP [Online] // BONDI. - OMTP. - 04 04 2011. - <http://bondi.omtp.org/whatisbondi/WHAT%20WE%20DO/Home.aspx>.

Peter Crocker and Matt Lewis Mobile Widget Platform Market Analysis: Understanding the Business Case and ROI [Online] // smithspointanalytics. - ARCchart, March 2010. - 11 04 2011. - <http://www.smithspointanalytics.com/MobileWidgetPlatformMarketAnalysis.pdf>.

Pieter Ballon and Michaël Van Bossuyt Comparing business models for multimedia content distribution platforms [Journal]. - Brussels, Belgium : SMIT-IBBT, Vrije Universiteit Brussel, 2006 йил.

Pieter Ballon and Nils Walravens Competing Platform Models for Mobile Service Delivery: the Importance of Gatekeeper Roles [Conference] // 7th International Conference on Mobile Business. - Brussel, Belgium : IEEE, 2008.

Qusay H. Mahmoud and Pawel Popowicz Toward a Framework for the Discovery and Acquisition of Mobile Applications [Conference] // 2010 Ninth

International Conference on Mobile Business / 2010 Ninth Global Mobility Roundtable. - Guelph, Ontario, Canada : Centre for Mobile Education and Research Department of Computing and Information Science University of Guelph, 2010.

Rupp Christian Business Development Deutsche Telekom [Interview]. - Cologne : [s.n.], 2010 йил 15-12.

Sarah Allen, Vidal Graupera and Lee Lundrigan Pro Smartphone Cross-Platform Development [Book]. - New York : apress, 2010. - Vols. 978-1-4302-2869-1.

Speckmann Benjamin Benjamin Speckmann [Book]. - Michigan : [s.n.], 2008.

Stanley Morgan The Mobile Internet Report [Online] // Morgan Stanley. - Morgan Stanley, 2009 йил 15-12. - 2010 йил 15-12. - www.ms.com/techresearch.

Stephen Pritchard www.itpro.co.uk [Online] // www.itpro.co.uk. - itpro, 10 Feb 2011. - 04 04 2011. - <http://www.itpro.co.uk/630913/data-not-voice-is-now-driving-telecoms>.

Utz Wilfried and Karagiannis Dimitris Towards Business and IT Alignment in the Future Internet - Managing Complexity in E-Business [Report]. - Vienna : [s.n.], 2009.

Vânia Gonçalves, Nils Walravens and Pieter Ballon “How about an App Store?” Enablers and Constraints in Platform Strategies for Mobile Network Operators [Conference] // 2010 Ninth International Conference on Mobile Business / 2010 Ninth Global Mobility Roundtable. - Brussels, Belgium : IEEE, 2010.

WAC Developer Site [Online] // WAC. - WAC. - 2010 йил 12-12. - <http://www.jil.org/web/jil/about>.

WAC Membership [Online] // Wholesale Application Community. - WAC. - 2010 йил 19-12. - <http://www.jil.org/web/jil/membership>.

What is 4G [Online] // gsmserver.com. - GSM Server. - 12 04 2011. - <http://gsmserver.com/articles/4g.php>.

What is iPFaces? [Online] // IPFaces. - IPFaces. - 07 04 2011. - <http://www.ipfaces.org/content/developer>.

Wikipedia [Online] // http://en.wikipedia.org/wiki/History_of_mobile_phones. - 2010 йил 12-12. - http://en.wikipedia.org/wiki/History_of_mobile_phones.

Wikipedia [Online] // Wikipedia - List of digital distribution platforms for mobile devices. - 2010 йил 13-12. - http://en.wikipedia.org/wiki/List_of_digital_distribution_platforms_for_mobile_devices.

Wikipedia [Online] // <http://en.wikipedia.org>. - Wikipedia. - 02 April 2011. - http://en.wikipedia.org/wiki/Comparison_of_Android_devices.

Windows Marketplace for Mobile [Online] // Windows Marketplace. - Microsoft Inc.. - 2010 йил 13-12. - <http://marketplace.windowsphone.com/Default.aspx>.

Windows Phone Marketplace [Online] // [wikipedia.org](http://en.wikipedia.org/wiki/Windows_Phone_Marketplace). - [wikipedia.org](http://en.wikipedia.org/wiki/Windows_Phone_Marketplace). - 2010 йил 16-12. - http://en.wikipedia.org/wiki/Windows_Phone_Marketplace.

Yabusaki Masami Next Mobile Network Architecture [Journal]. - Munich, Germany : DOCOMO Communications Laboratories Europe GmbH, 2010 йил.

Yonghong Wu, Jianchao Luo and Lei Luo Porting mobile web application engine to the Android platform [Journal]. - School of Computer Science and Engineering, University of Electronic Science and Technology : [s.n.], 2010 йил. - Chengdu 610054 : Vol. 2010 10th IEEE International Conference on Computer and Information Technology (CIT 2010).

Yu Chen Zhou [et al.] Service Storm: A Self-Service Telecommunication Service Delivery Platform with Platform-as-a-Service Technology [Conference] // 2010 IEEE 6th World Congress on Services. - Beijing, P.R. China : IEEE, 2010.

13 Appendix

13.1 Appendix 1: Mobile Operating Systems Detail

13.1.1 Android

Google acquired the small mobile software developer company, Android, Inc. in July 2005. This was obviously a move from Google to extend their successful business on the Internet to also include the mobile market. The Android OS is a result of the Open Handset Alliance (OHA) with Google as one of the very active partners in the implementation. Other notable partners in, OHA being handset manufacturers, are HTC, LG, Samsung, Sony Ericsson, Motorola and NTT Docomo - not Nokia, Palm and Apple (Google Inc., 2010). It is currently in version 2.3 (Gingerbread). Android OS isn't made in Java, but the application development for Android is in Java. However, C/C++ and ARM Assembly can also be used when using a Native Development Kit. Android is based on a Linux kernel with the user space and the JVM for Android (Dalvik) being written in C. The OS is fully open source; however Android applications created by Google to access existing Google web services are not open source and not allowed to be distributed without permission from Google (Allan, et al., 2010). 2010 was a strong year for Android, the latest and most popular Android devices on the market are HTC's Desire Range (HD and Z), Samsung's Galaxy Range (S and Tab), Google's new released Nexus S (manufactured by Samsung) and Sony Ericsson's Xperia series. Furthermore latest developments have shown that the Android OS is versatile and can be used not only in mobile phone derivatives but also in hardware like notebooks, netbooks, eReaders and TV sets, as open source OS the implementation capabilities are likely to be further exploited in terms of its hardware usage (Wiki3).

13.1.2 Blackberry OS

The Blackberry OS and development platform is developed by the Canadian company Research-In-Motion (RIM), and was released in its latest version 5 in

October 2009. The OS is providing a platform for doing application development supporting solely J2ME. The Blackberry Java Virtual Machine (JVM) is based on Sun's implementation of the J2ME being written partly in C, C++ and assembler. It is a native implementation located in the actual firmware of the device, making it very hard to hack or in any way alter. The two greatest advantages of this are that: 1) the OS doesn't have to be compiled to the CPU type of the device, and at the same time 2) it provides a hardware abstraction layer to other hardware functionalities of the device like button control, sound, radio communication etc. On paper this gives a better device performance eliminating many bottlenecks in hardware access (Allan, et al., 2010).

13.1.3 iPhone OS

The iPhone OS is developed by Apple. It is currently in version 4.2 and is based on a variant of Mac OS X and available for iPhone, iPad and iPod devices. (Apple Inc., 2010). The OS is capable of supporting bundled and future applications from Apple, as well as from third-party developers. Applications development for the iOS is mainly done using Objective-C, but C/C++ development is also possible. Effort has been put in supporting Web Runtime (WRT) based services as well. These services are written in JavaScript, CSS and HTML that are supported on most smart phones and web browsers in general. This is due to the fact that most available browsers (except Firefox and Internet Explorer) are based on WebKit. WebKit is an open source web browser engine used in Apple's Safari, KDE's Konqueror, Nokia's S60 browser, Google's chrome and more. It has also been ported to Qt. Development of the iOS is controlled by Apple in all aspects. (Allan, et al., 2010).

13.1.4 Symbian

The Symbian OS is the most popular mobile OS in the world. Nokia has been the main actor using Symbian for their OS implementation called the S-series. Other handset manufacturers are using Symbian as well, but the future of Symbian has heavily been influenced by the decisions of Nokia. The most advanced of the S-series OS was the S60 that from version 5 supported touch screens. None of the S-series OSs from Nokia have been open source so far. The S60 version 5 was

designed on top of the Symbian v9.4. This implementation has been renamed to Symbian^1. The next version is the first open source version of Symbian called Symbian^2, Symbian^3 will focus on the graphical experience and better support for streaming video, and from Symbian^4 a complete new user interface with all graphical components and standard application development based on Qt. Qt is a cross-platform application framework that runs on desktop OS as Windows, Linux and Mac. On mobile phones Qt run on Symbian and MAEMO. Qt was originally created by Norwegian company Trolltech and acquired by Nokia in June 2008. Qt provides an intuitive C++ class library with a rich set of application building blocks for C++ development. Qt goes beyond C++ in the areas of inter-object communication and flexibility for advanced GUI development. Since 2005, Qt had a fast development phase which makes it one of the important mobile program languages of nowadays (Allan, et al., 2010).

13.1.5 Windows Phone 7

Windows Phone 7 is a mobile operating system developed by Microsoft, and is the successor to their Windows Mobile platform. It launched in a staggered approach across the world between 2010 and 2011, therefore it is a relatively new OS to the market. With Windows Phone 7, Microsoft offers a new user interface with their design language named Metro, integrates the operating system with 3rd party and other Microsoft services, and plans to strictly control which hardware it runs on. The final SDK was made available on September 16, 2010. Microsoft created a web application, App Hub, for Windows Phone 7 and Xbox LIVE application developers to register, submit and manage their third party applications for the platforms. *The App Hub provides development tools and support for third-party application developers. Windows Phone 7 application development is based on Silverlight, XNA, and the .NET Compact Framework 4 only. The Silverlight version will be based on Silverlight 3, with some elements back ported from Silverlight 4. The main tools used for development will be Microsoft's Visual Studio 2010 and Expression Blend (Allan, et al., 2010).*

13.2 Appendix 2: Business Process Models

In the following chapters the reference models analysed within this paper are listed. The application store environments and business processes from a developer perspective of the Apple Appstore, Android Marketplace and the Wholesale Application Community are modelled.

13.2.1 Apple Appstore

Apple Appstore Application Distribution 1.0

This company process map was created by analysing the Apple developer program environment with a hands-on approach from a developer's perspective by running through the distribution process.

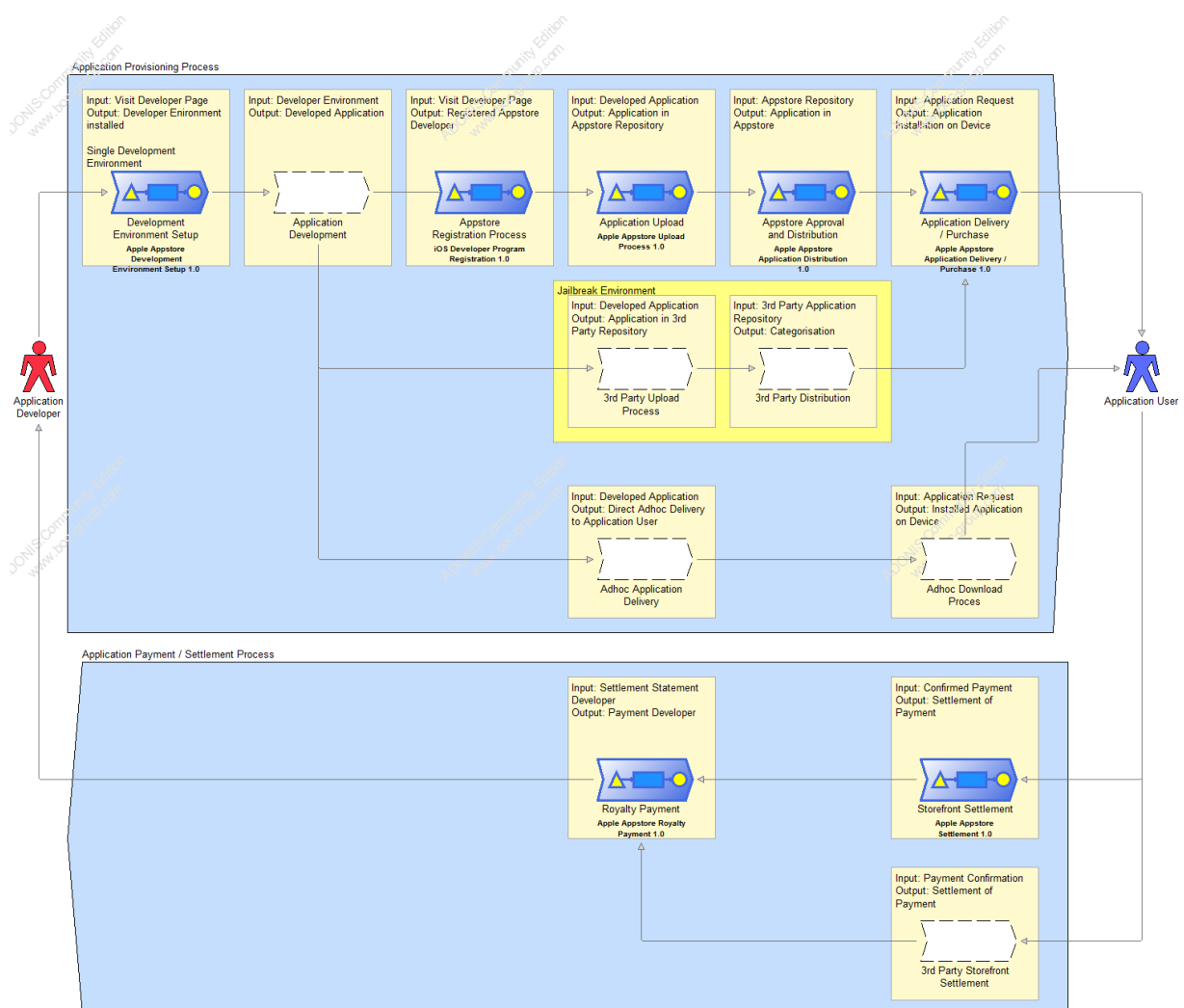


Figure 34 Apple Appstore Application Distribution 1.0

The application distribution process consists of the following processes: Development environment setup, application registration process, application upload, Appstore approval and distribution, application delivery / purchase, storefront settlement and royalty payment. Processes like application development, the jailbreak environment, and adhoc delivery are shown on the process map for completeness of the environment, but are not discussed in detail as they are out of scope for this thesis.

Apple Appstore Roles 1.0

The Appstore roles consist of the key players in the business processes – these are the application developer, the application user, and the Apple Application review team.



Figure 35 Apple Appstore Roles 1.0

Apple Appstore Documents 1.0

These are the documents found within the process analysis to enable the development and registration as Apple developer.

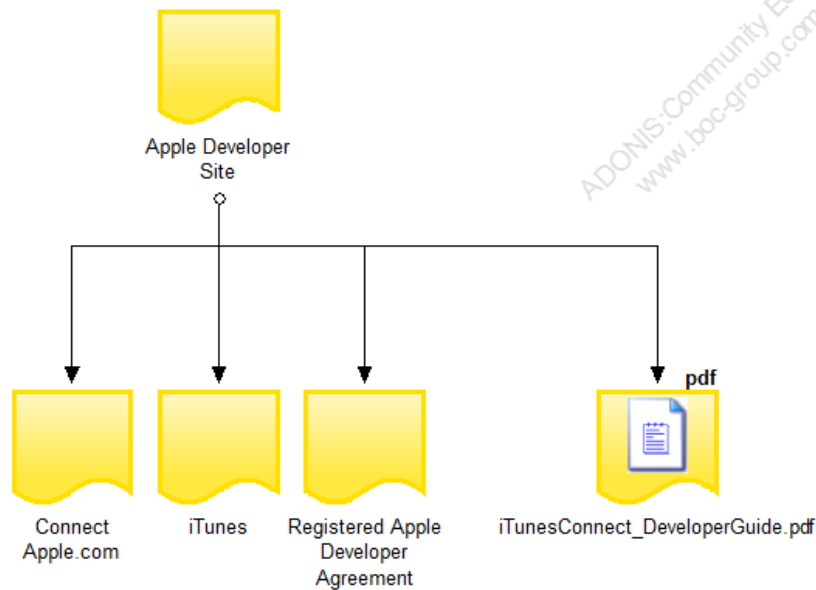


Figure 36 Apple Appstore Documents 1.0

Apple Appstore Development Environment Setup 1.0

The development environment setup is the first process in the process chain, which starts with the developer registration sub process, to enable the download and installation of the SDK through the iOS Dev Center.

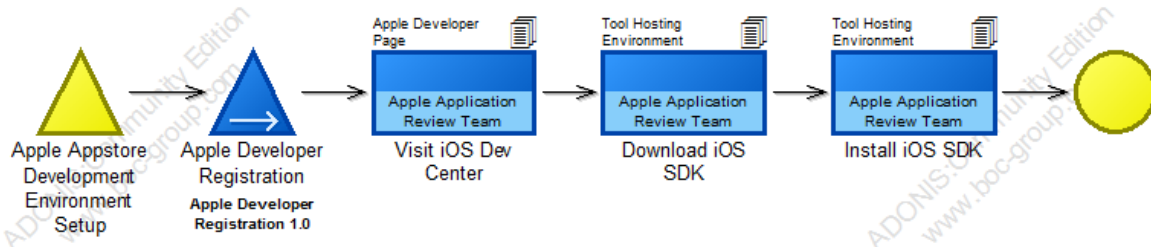


Figure 37 Apple Appstore Development Environment Setup 1.0

Apple Developer Registration 1.0

The sub process of the developer registration requires the download of iTunes and the setup of an iTunes ID, and then the developer registration can be done on the developer site. This follows then a standard procedure of entering personal details, accepting legal terms and ends with an email verification code challenge.

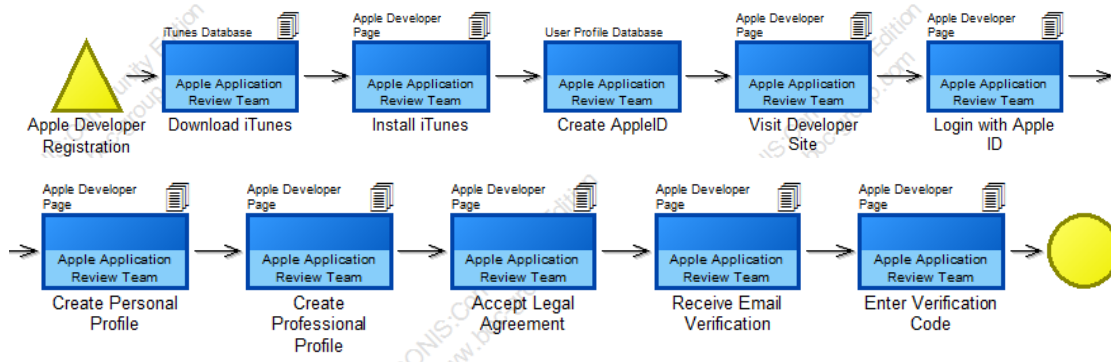


Figure 38 Apple Developer Registration 1.0

iOS Developer Program Registration 1.0

Apple requires an extra registration process if a developer wants to develop and use the iOS environment. Here the billing and payment information is retrieved as well as the purchase of the developer license happens through a link to the apple online store.

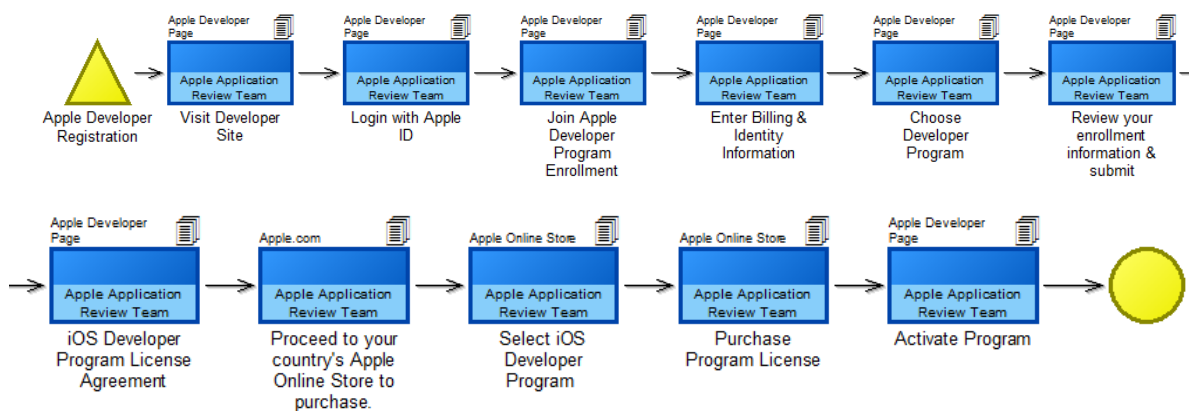


Figure 39 iOS Developer Program Registration 1.0

Apple Appstore Upload Process 1.0

The upload process with the Appstore requires an upload preparation process, after completing this and the Apple approval the application file can be uploaded through the Loader application provided with the SDK.

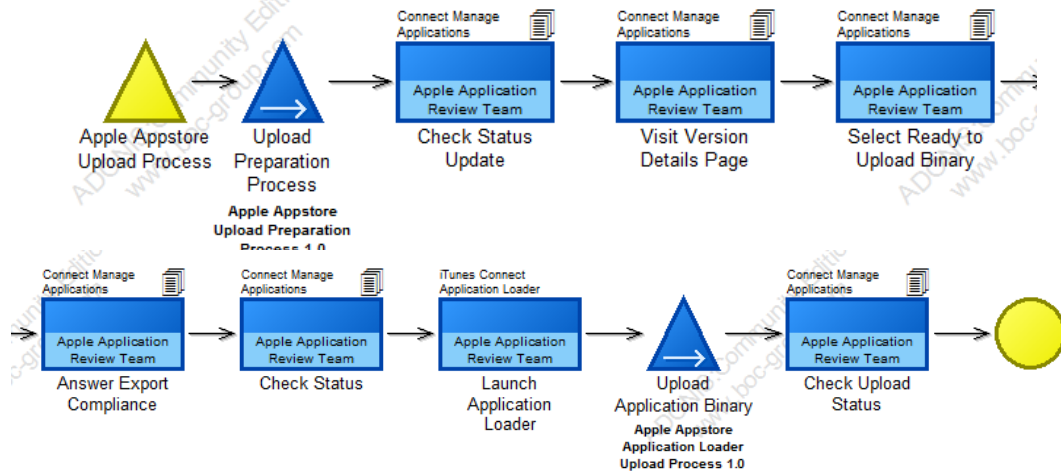


Figure 40 Apple Appstore Upload Process 1.0

Apple Appstore Upload Preparation Process 1.0

The upload preparation sub process runs through the developer site. Here the main details of the application as well as a description must be provided. This enables the Apple team to verify and approve the application.

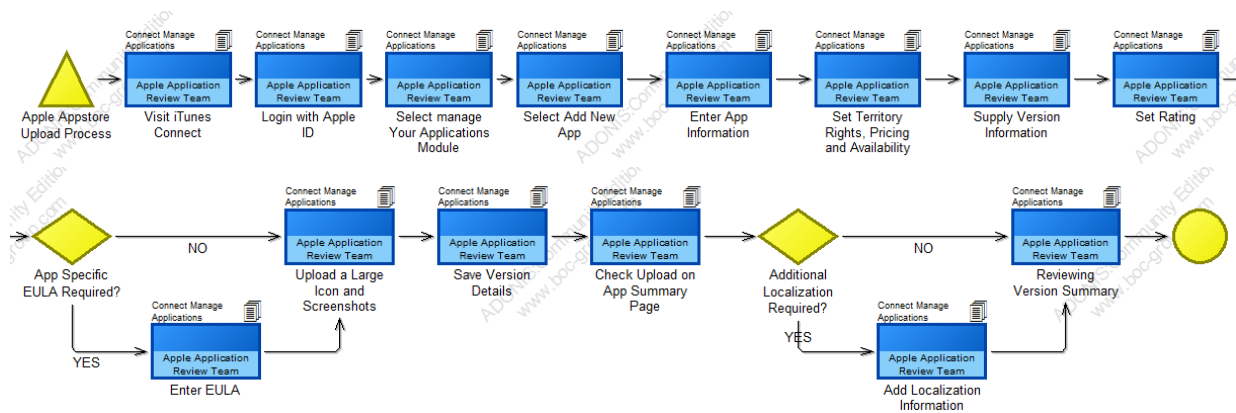


Figure 41 Apple Appstore Upload Preparation Process 1.0

Apple Appstore Application Loader Upload Process 1.0

In the application Uploader process the application is delivered to the Appstore environment by the Uploader software delivered in the SDK. This is the only way now to deliver to the application store and an Intel MAC workstation is needed.

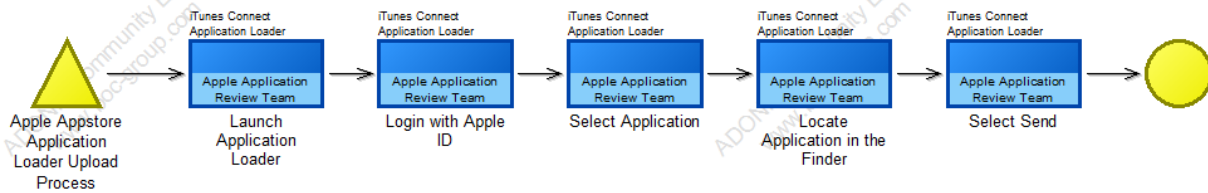


Figure 42 Apple Appstore Application Loader Upload Process 1.0

Apple Appstore Application Distribution 1.0

The application distribution process starts with the approval by Apple. Once approved further export authorization might be required by Apple to enable the distribution. Apple gives the possibility to set a live date for the application, so that it doesn't go immediately live after the publish process.

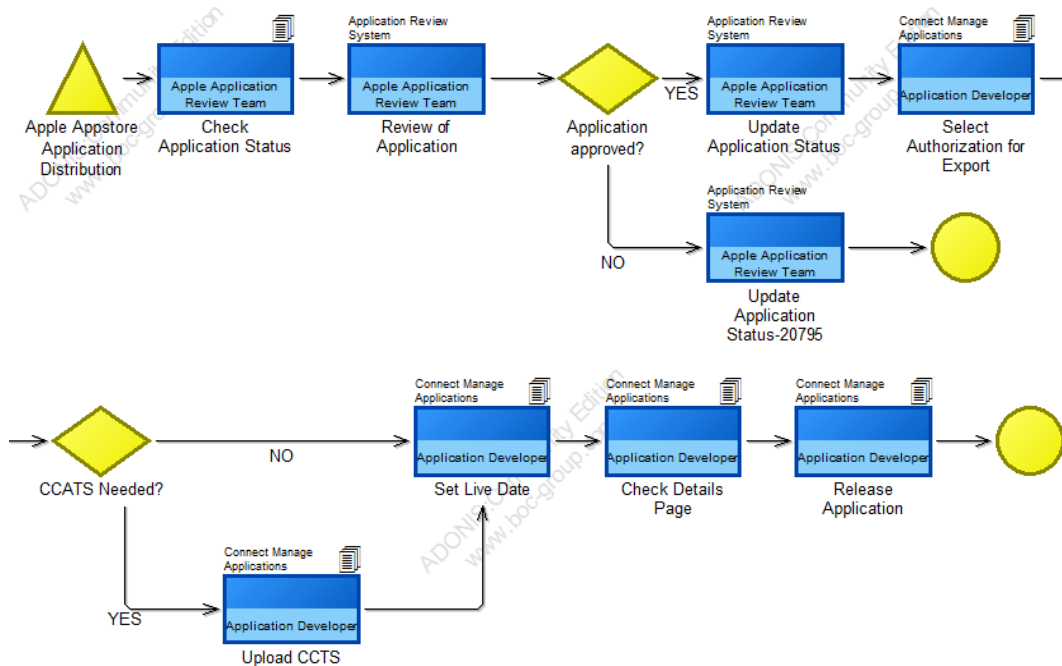


Figure 43 Apple Appstore Application Distribution 1.0

Apple Appstore Application Delivery / Purchase 1.0

The application delivery and purchase process is, once the customer has set up the iTunes environment, a quick and easy process that is very customer friendly and requires only a password to authorize the payment. Purchase, download and installation is without security warnings and a streamlined flow.

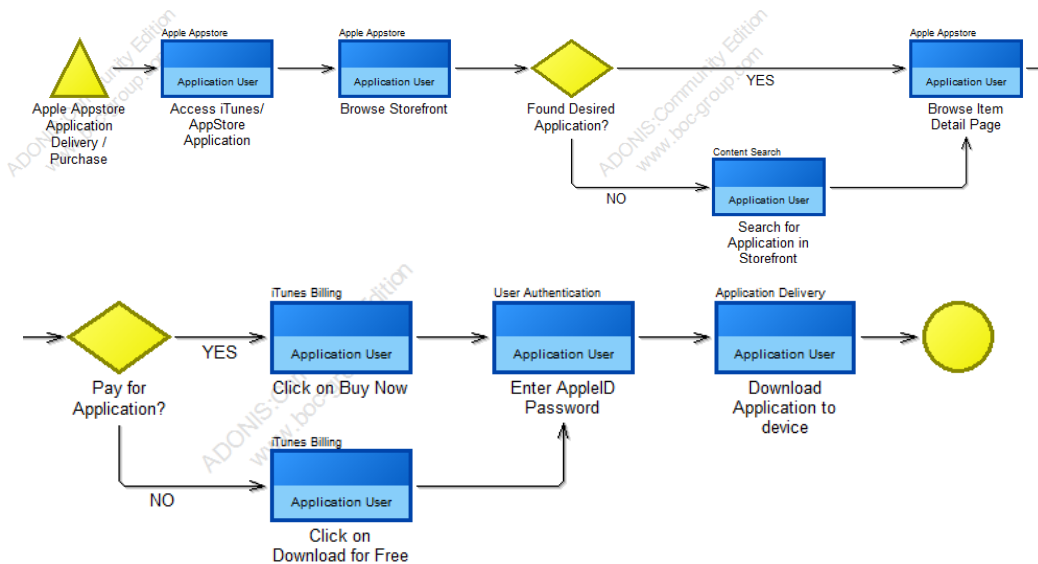


Figure 44 Apple Appstore Application Delivery / Purchase 1.0

Apple Appstore Settlement 1.0

The Appstore settlement process is an automated process due to the vast amount of publishers / developers on the system. This is the assumed process flow which includes the payment confirmation, and the automated email confirmation for the customer. As well as the transaction accumulation by the settlement system which translates to a report for the developer.

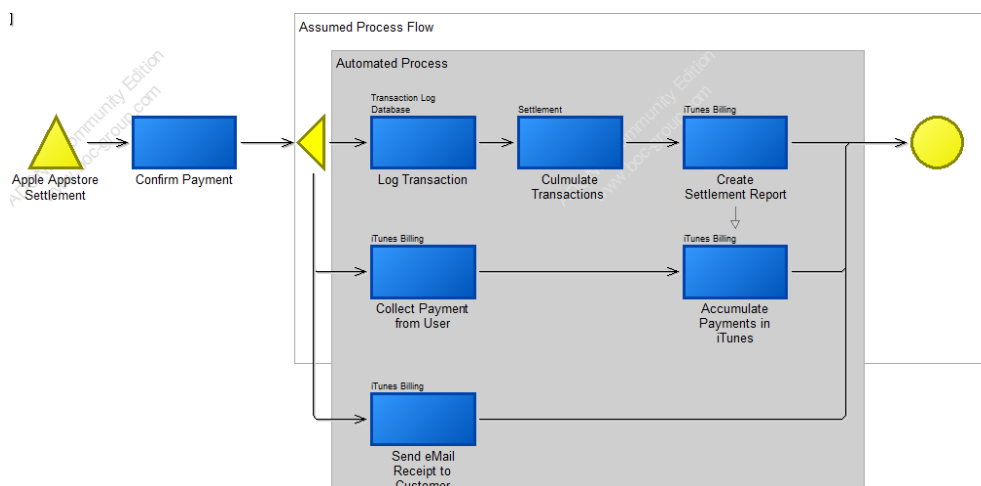


Figure 45 Apple Appstore Settlement 1.0

Apple Appstore Royalty Payment 1.0

The royalty payment is as well an automated process with regards to the vast amount of publishers involved. The reports, invoices and payment confirmations can be viewed in the iOS developer environment.

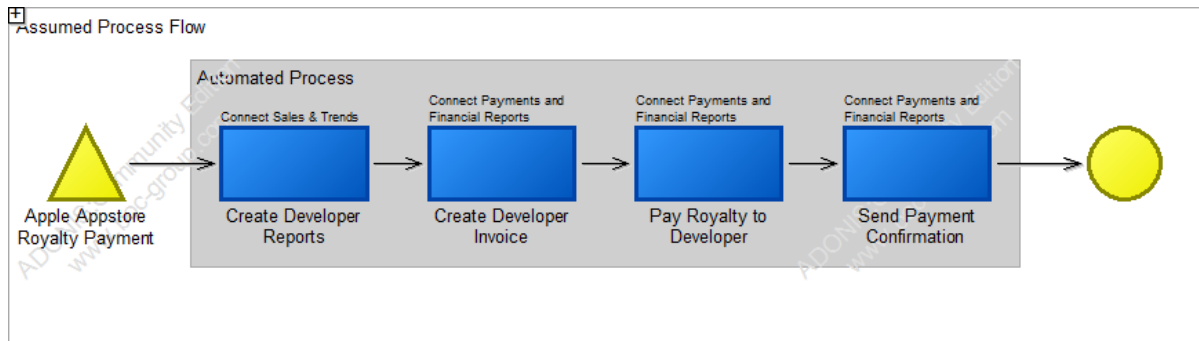


Figure 46 Apple Appstore Royalty Payment 1.0

Apple Appstore IT Environment 1.0

The Appstore environment is a complex system with several elements like iTunes, Appstore, Apple Online Store, and smaller components that are interrelated. Despite the highly advanced environment it is only compatible to iOS devices and Intel MAC workstations which limit the compatibility for a multi store and platform concept.

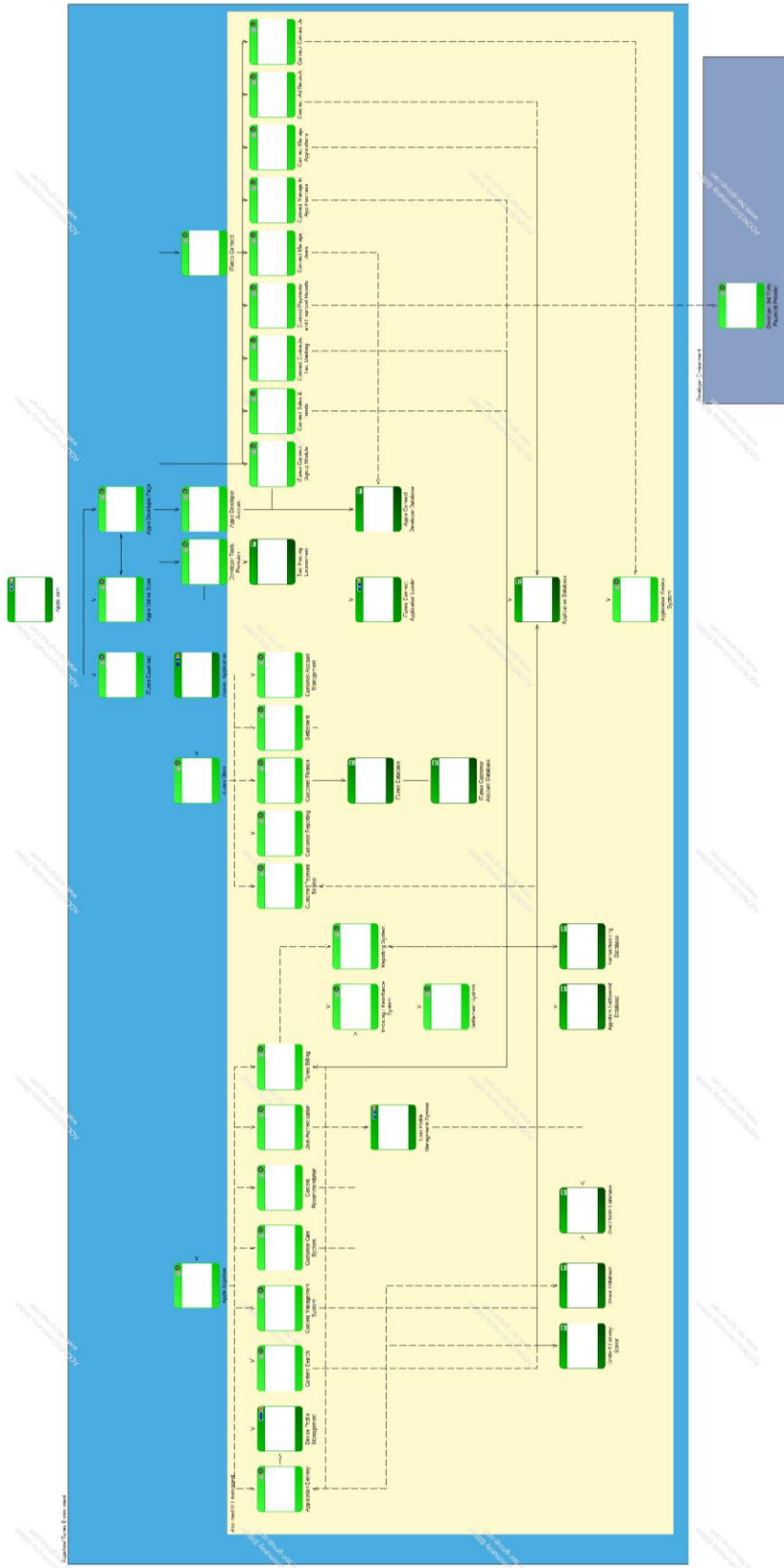


Figure 47 Apple Appstore IT Environment 1.0

Apple Appstore IT Infrastructure

This is the assumed Apple Appstore IT infrastructure derived from the business process models discussed in the earlier chapters.

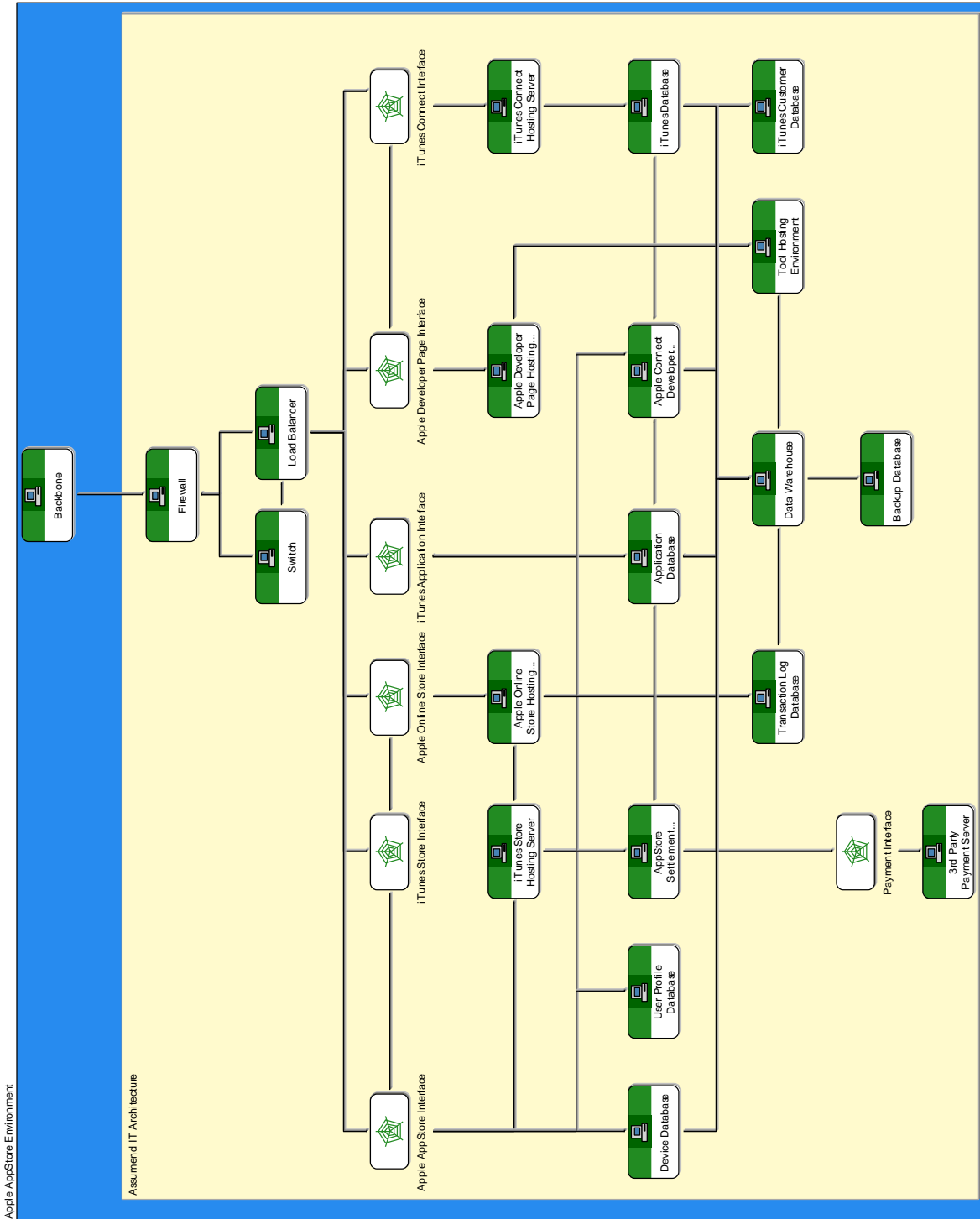


Figure 48 Apple Appstore IT Infrastructure

13.2.2 Android Marketplace

Android Marketplace Application Distribution 1.0

This company process map was created by analysing the Android developer program environment with an hands-on approach from a developers perspective by running through the distribution process. The application distribution process consists of the following processes: Development environment setup, Android Marketplace registration process, Application upload, Application delivery / purchase, Marketplace settlement and royalty payment. Processes like application development, the 3rd party environment, and adhoc delivery are shown on the process map for completeness of the environment, but are not discussed in detail as they are out of scope for this thesis.

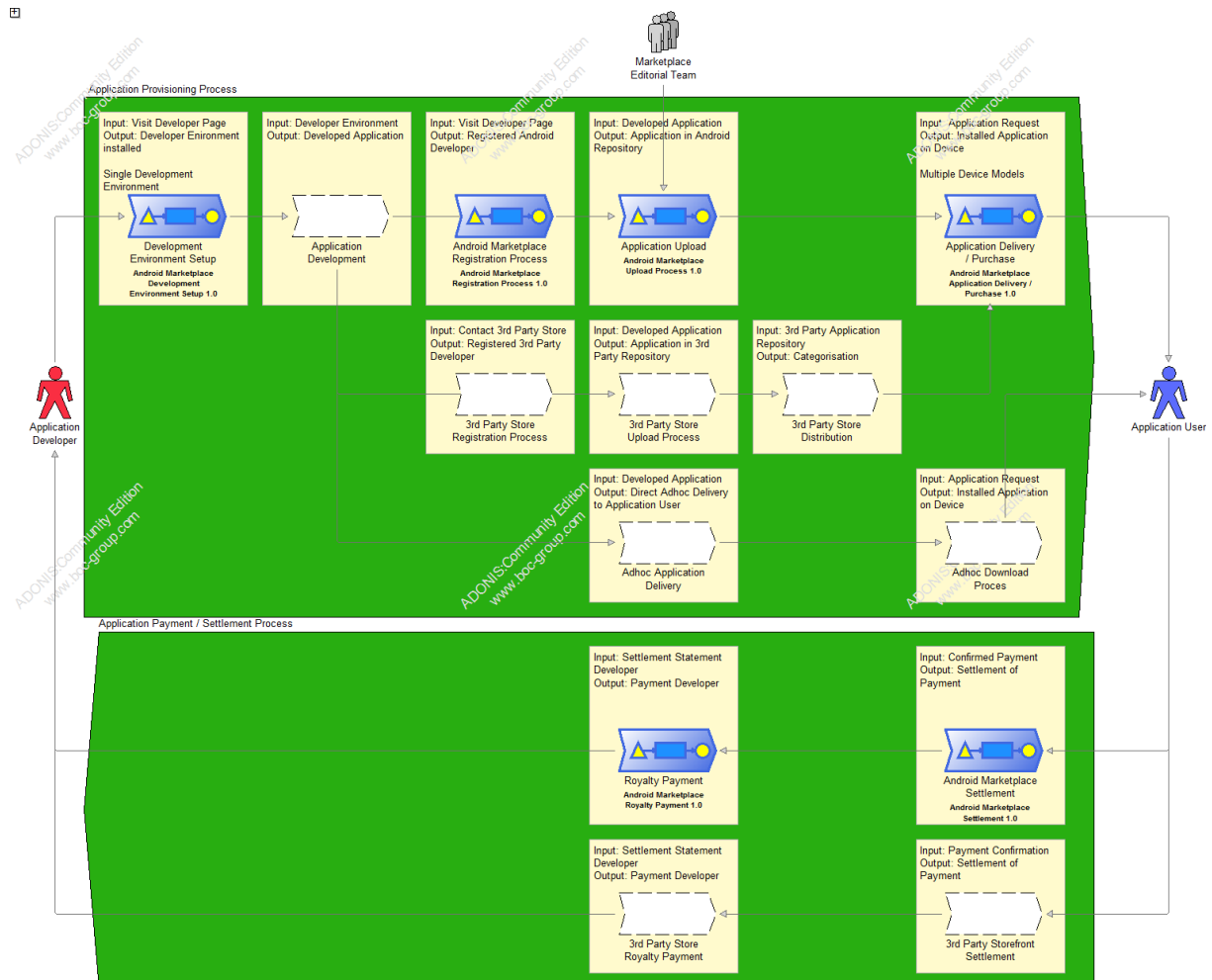


Figure 49 Android Marketplace Application Distribution 1.0

Android Marketplace Roles 1.0

The Android Marketplace roles consist of the key players in the business processes – these are the application developer, the application user, and the Marketplace Editorial team.

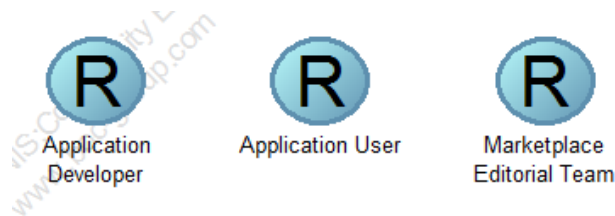


Figure 50 Android Marketplace Roles 1.0

Android Marketplace Documents 1.0

These are the documents found within the process analysis to enable the development and registration as Android developer.

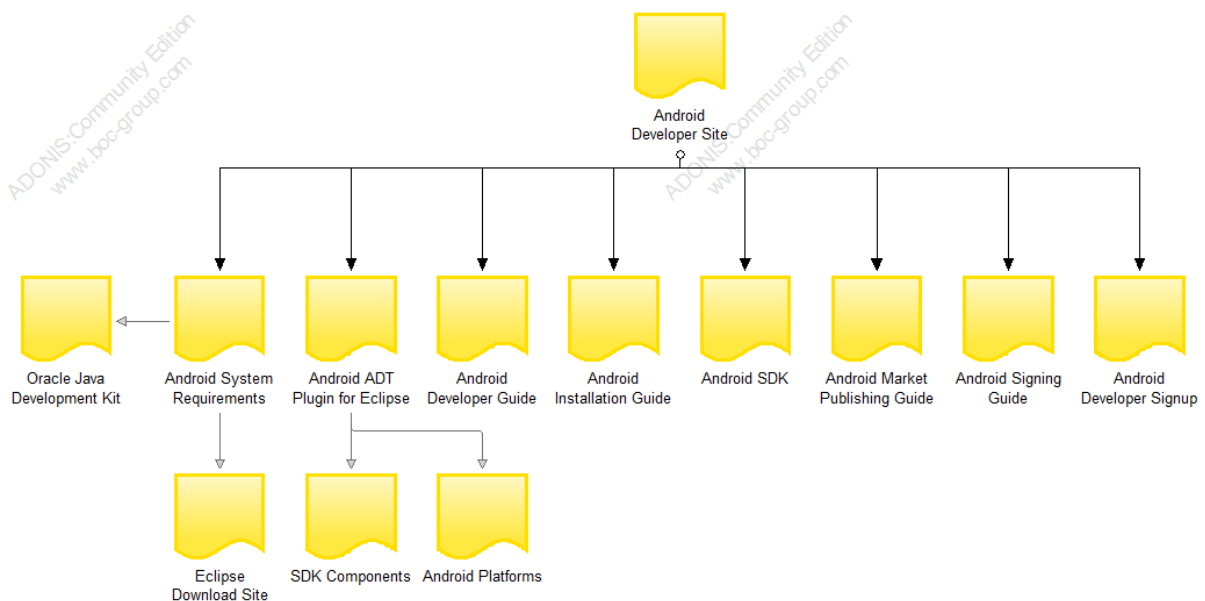


Figure 51 Android Marketplace Documents 1.0

Android Marketplace Development Environment Setup 1.0

The development environment setup is the first process in the process chain, which starts with the download and installation of the SDK and necessary environments like JDK, eclipse and ADT plug-in that are not part of the Android

SDK. The download does not require a prior registration to the Android developer site.

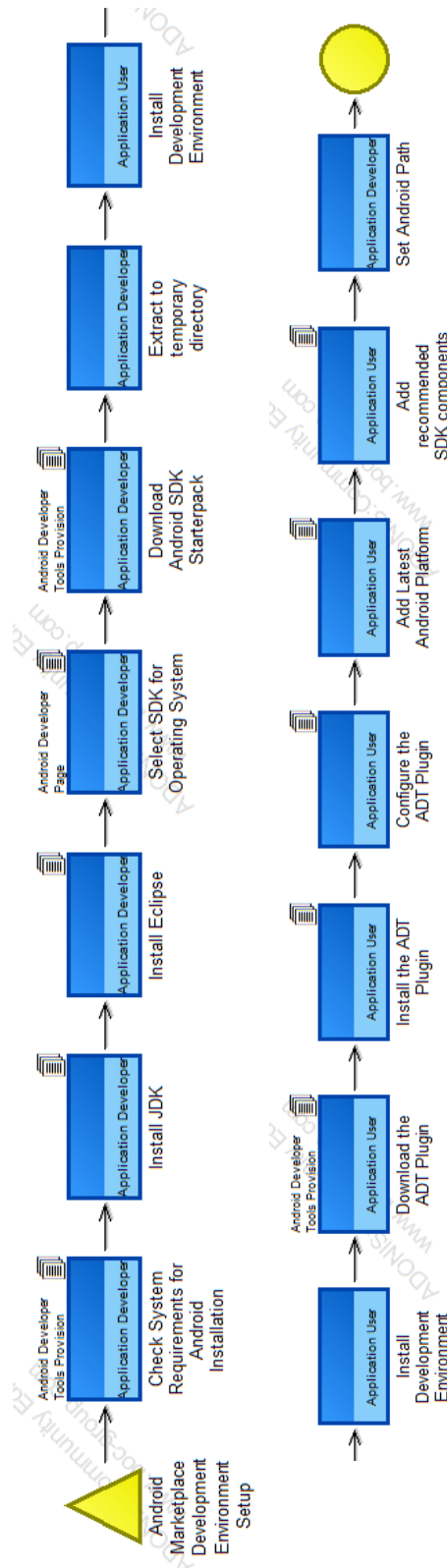


Figure 52 Android Marketplace Development Environment Setup 1.0

Android Marketplace Registration Process 1.0

The registration with the Android developer program requires first the registration of a Google account. Once this is done a Google checkout account needs to be created as the registration as developer requires a payment. After that a Google Merchant account needs to be created if the developer wants to sell the applications.

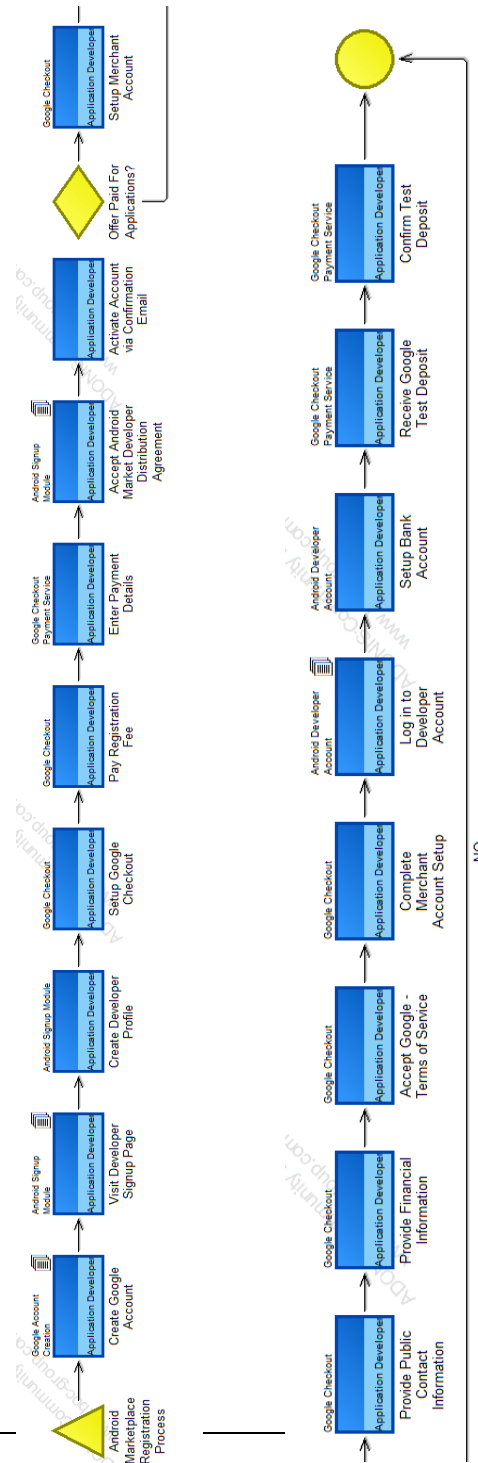


Figure 53 Android Marketplace Registration Process 1.0

Android Marketplace Upload Process 1.0

The application upload process is started through the Google developer publish page. In one process and on one web page, all information is given and has to be provided by the developer around the application. After all application details are entered the submission is checked if it is a valid apk file. Error messages are given if the system detects any abnormalities with the submitted file.

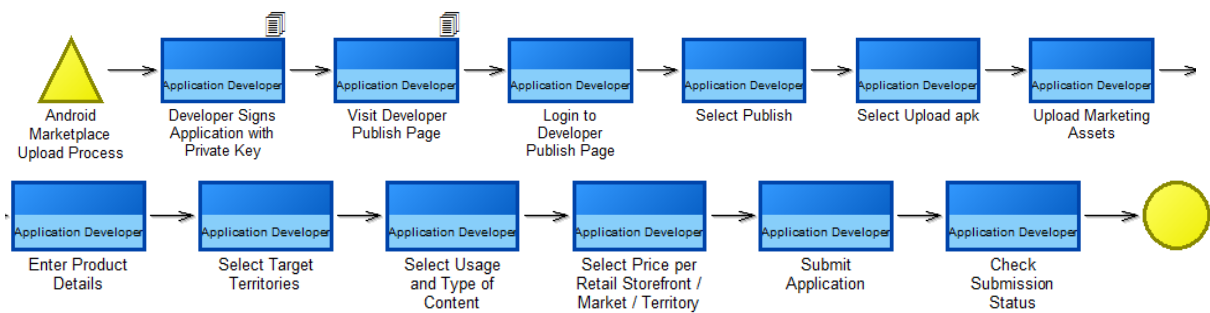


Figure 54 Android Marketplace Upload Process 1.0

Android Marketplace Application Distribution 1.0

Once the application is submitted and successfully uploaded, the application is instantly in the storefront managed through the automatic categorisation. Further the android marketplace employs a storefront editorial team that handpicks and manages the landing pages of the storefront.

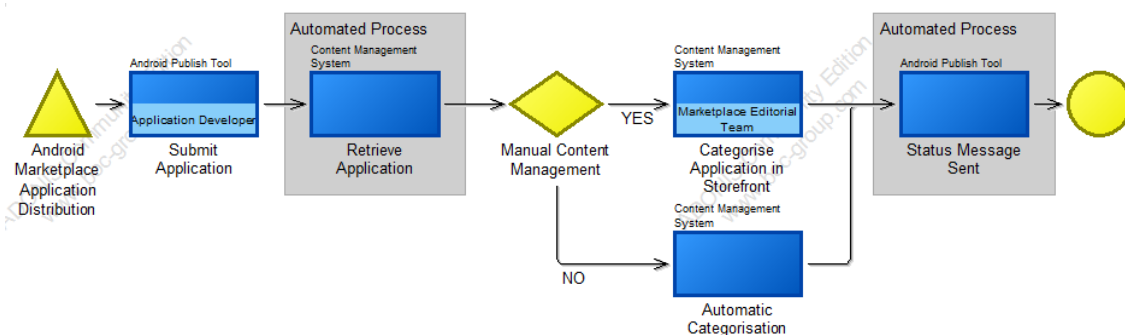


Figure 55 Android Marketplace Application Distribution 1.0

Android Marketplace Application Delivery / Purchase 1.0

The application delivery is relatively easy and the storefront provides all necessary tools to find an application like search, recommendations, and top lists. However the download process is not as streamlined as the Apple store as separate steps are necessary to get to the application. The purchase requires a setup with a payment provider prior to accept the purchase. After that an extra advice of charge is displayed, after accepting this, the application is downloading, then a security warning is shown that gives a list of APIs that the application is using in the phone. Then the user has to install the application in a final step.

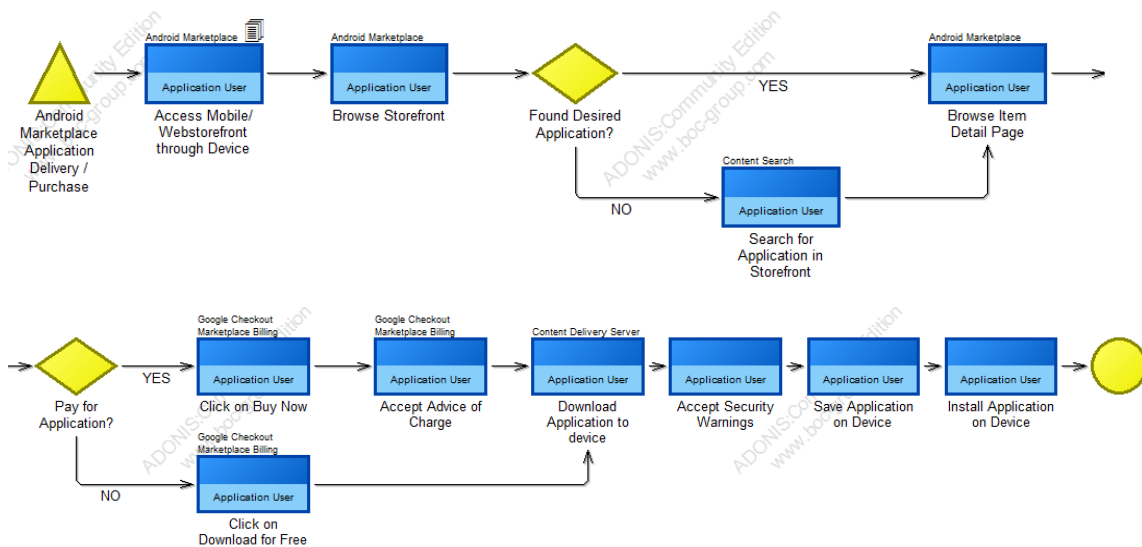


Figure 56 Android Marketplace Application Delivery / Purchase 1.0

Android Marketplace Settlement 1.0

The Android settlement process is an automated process due to the vast amount of publishers / developers on the system. This is the assumed process flow which includes the payment confirmation, and the automated email confirmation for the customer. As well as the transaction accumulation by the settlement system this translates to a report for the developer.

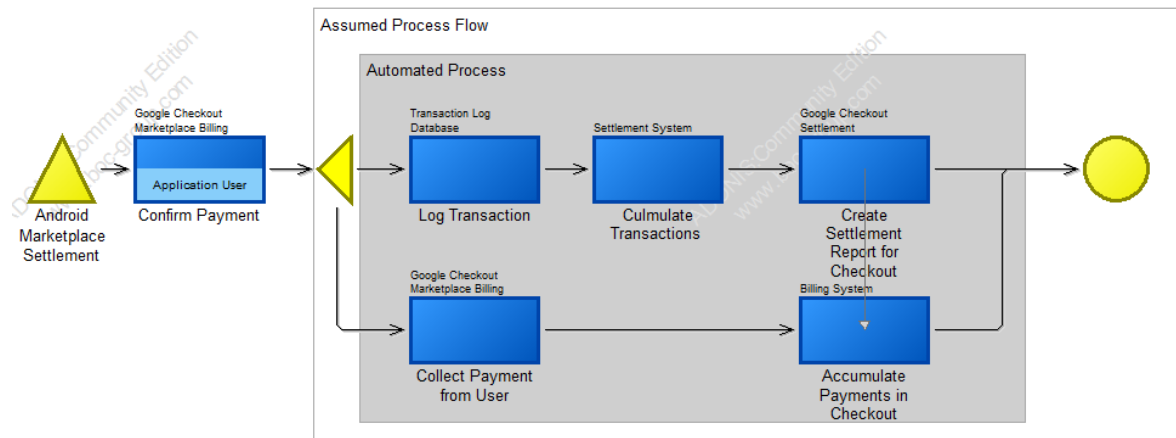


Figure 57 Android Marketplace Settlement 1.0

Android Marketplace Royalty Payment 1.0

The royalty payment is as well an automated process with regards to the vast amount of publishers involved. The reports, invoices and payment confirmations can be viewed in the Android developer environment.

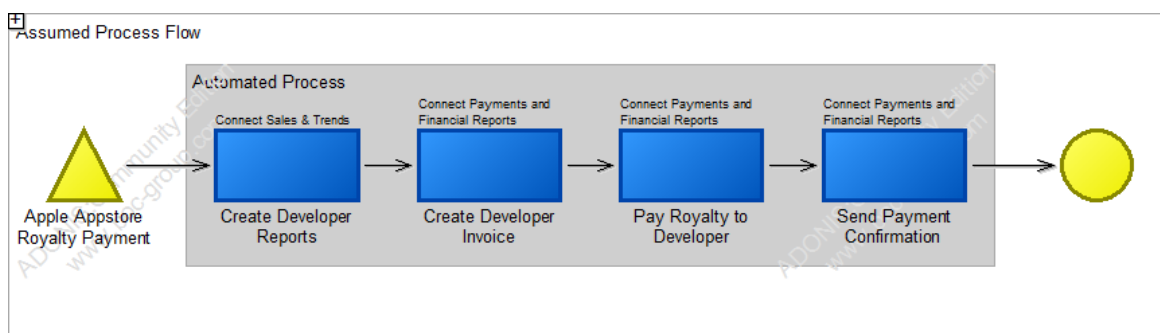


Figure 58 Android Marketplace Royalty Payment 1.0

Android Marketplace IT Environment 1.0

The Android Marketplace environment is a complex system with several elements like Android Marketplace, Google Webpage, Android Developer Site and smaller components that are interrelated. Despite the highly advanced environment it is only compatible to Android devices which limit the compatibility for a multi platform concept. However as the Android OS is an open source OS several 3rd party environments are able to host Android applications in addition to other platforms. Therefore the Android Marketplace environment is a single OS platform but Android can be used in other platforms that support multiple OS file types.

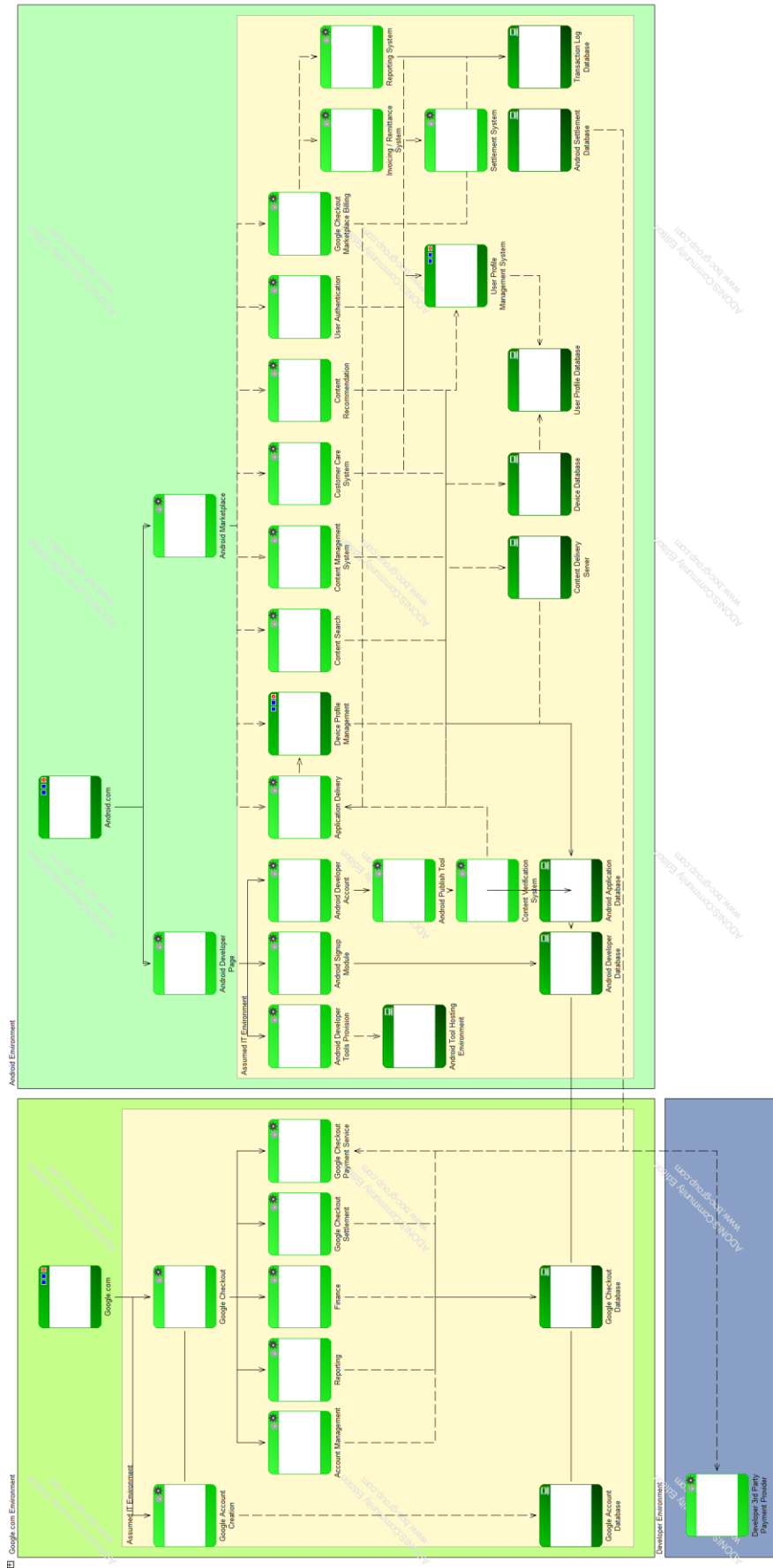


Figure 59 Android Marketplace IT Environment 1.0

Android Marketplace IT Infrastructure

This is the assumed Android Marketplace IT infrastructure derived from the business process models discussed in the earlier chapters.

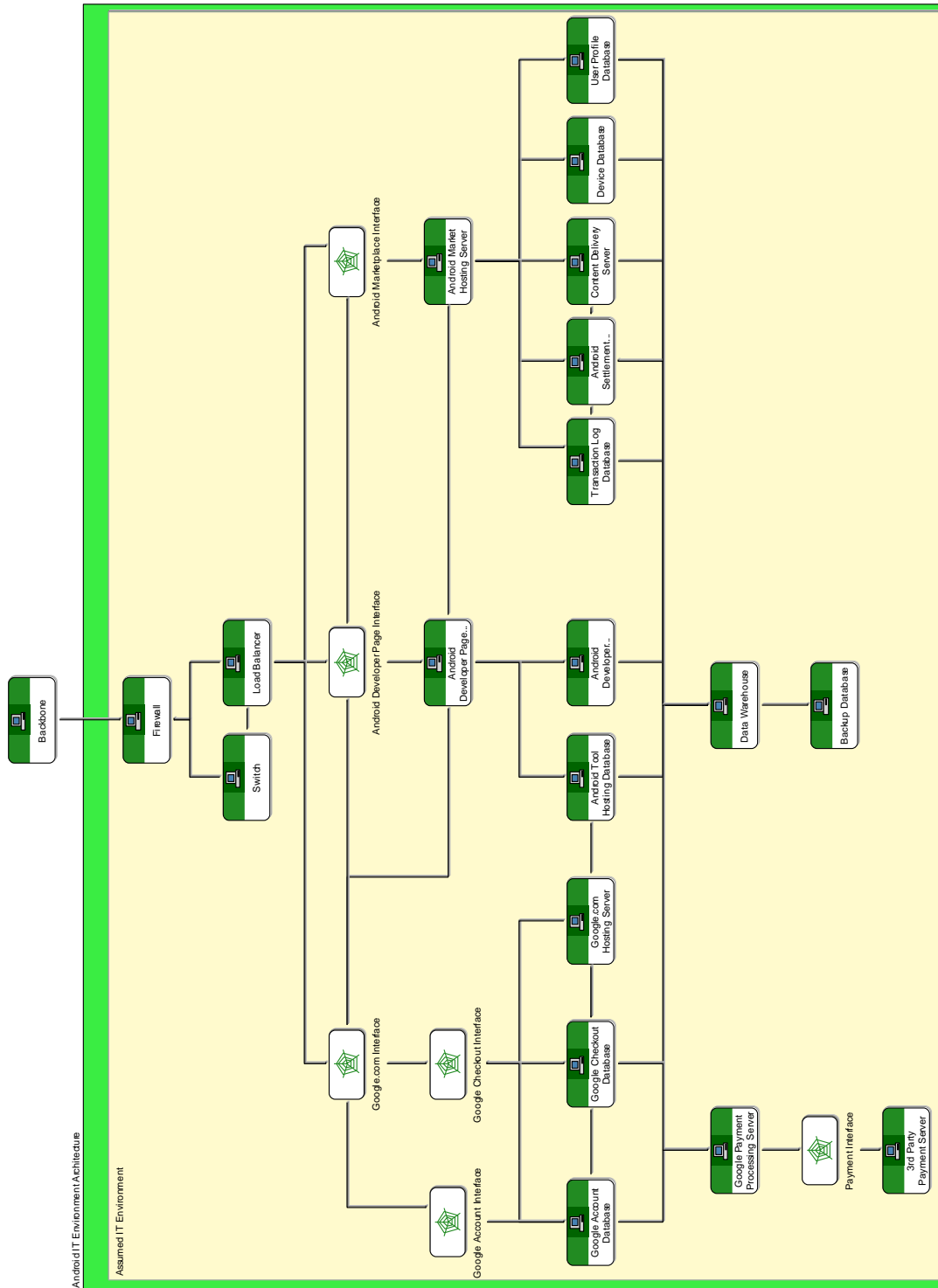


Figure 60 Android Marketplace IT Infrastructure

13.2.3 Wholesale Application Community

WAC Application Distribution 1.0

This company process map was created by analysing the WAC developer program environment with a hands-on approach from a developer's perspective by running through the distribution process.

The application distribution process consists of the following processes: Development environment setup, WAC registration process, application upload, WAC application distribution, application delivery / purchase, storefront settlement, WAC settlement and royalty payment. Processes like application development and 3rd party distribution are shown on the process map for completeness of the environment, but are not discussed in detail as they are out of scope for this thesis.

The WAC environment is the only environment that should enable a modular connection and the distribution through a network of partner storefronts by the WAC client application.

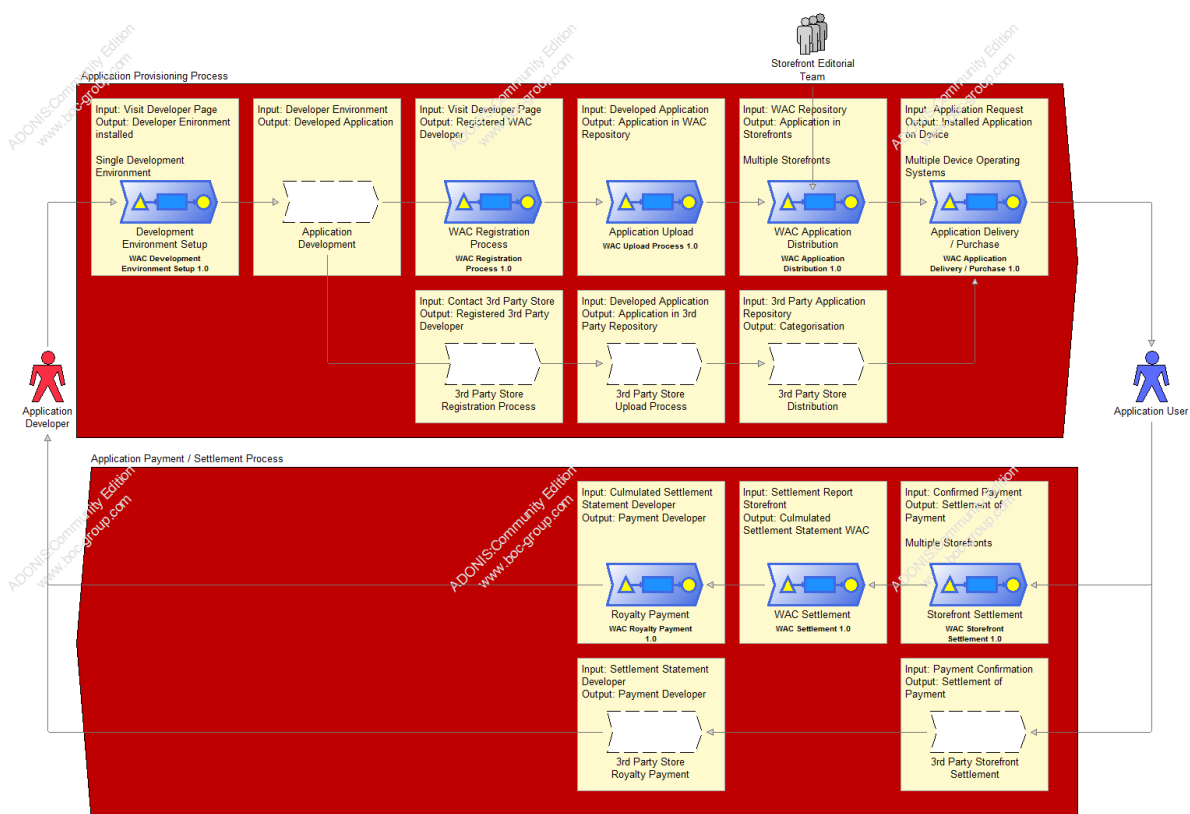


Figure 61 WAC Application Distribution 1.0

WAC Roles 1.0

The WAC roles consist of the key players in the business processes – these are the application developer, the application user, and the storefront editorial team which is optional depending on the partner storefront requirements.



Figure 62 WAC Roles 1.0

WAC Documents 1.0

These are the documents found within the process analysis to enable the development and registration as WAC developer.

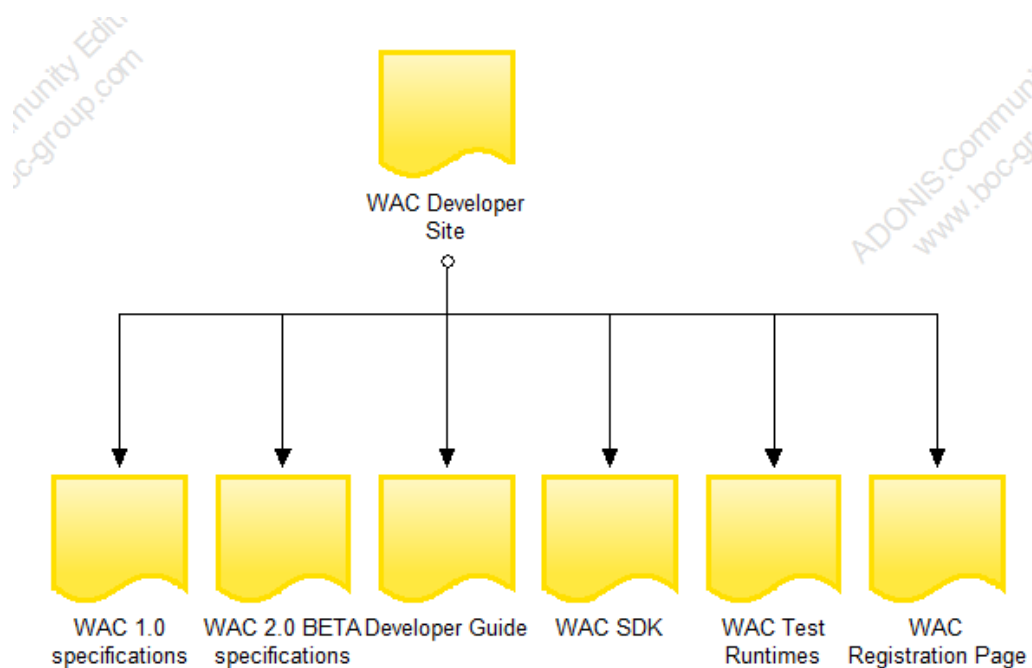


Figure 63 WAC Documents 1.0

WAC Development Environment Setup 1.0

The development environment setup is the first process in the process chain, which starts with the download and installation of the WAC SDK and necessary

environments like JDK, eclipse and widget emulator that are not part of the WAC SDK. The download does not require a prior registration to the WAC developer site.

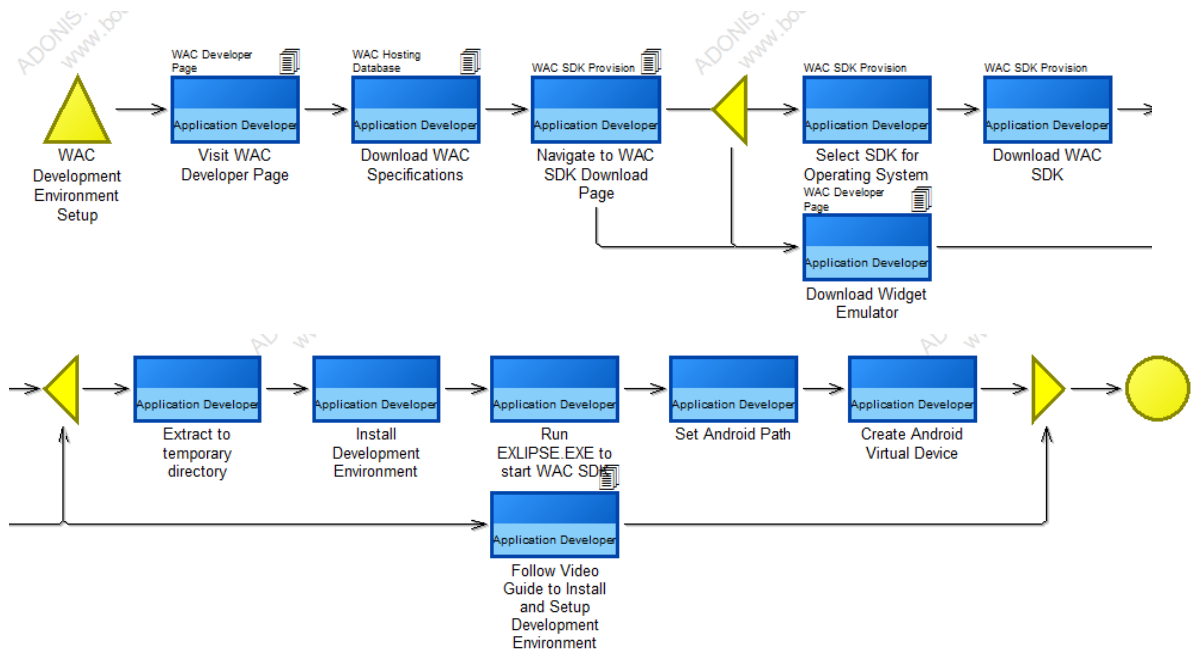


Figure 64 WAC Development Environment Setup 1.0

WAC Registration Process 1.0

The registration with the WAC developer program requires only one registration with the WAC developer site for all partner storefronts that are connected. However a certificate needs to be obtained by a 3rd party certification organisation to identify the developer. This certificate needs to be installed in a separate step with a Mozilla Firefox browser to be able to test and upload the applications.

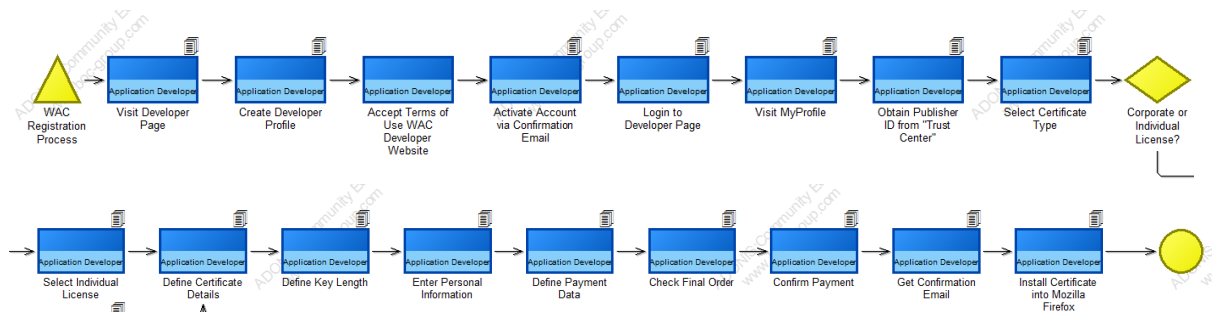


Figure 65 WAC Registration Process 1.0

WAC Upload Process 1.0

The application upload process is started through the WAC developer publish page. One web page, all information is given and has to be provided by the developer around the application. After the application is uploaded the widget has to be signed in the Widget signing process, thereafter the details around the application are entered, then the local storefront T&Cs are accepted and the submission is finalised. In the MyWidgets area the status of the submission / distribution can be checked by storefront.

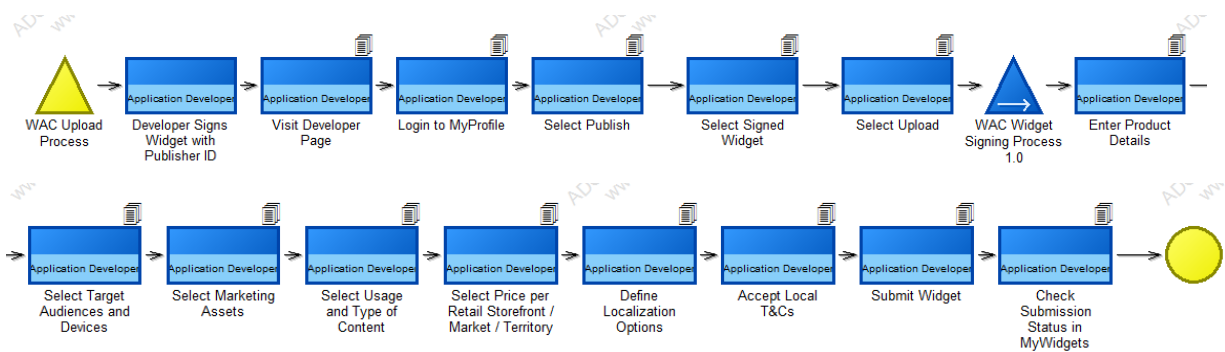
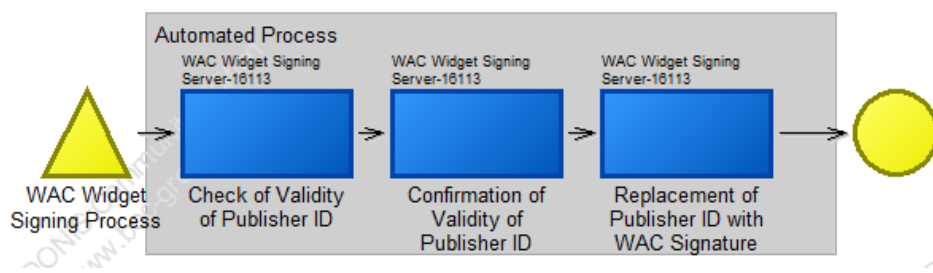


Figure 66 WAC Upload Process 1.0

WAC Widget Signing Process 1.0

The WAC widget signing process is a sub process of the upload process. In this step the publisher ID is automatically checked and after approval replaced with a WAC signature, this is a unique key per application.



WAC Application Distribution 1.0

The WAC application distribution use a WAC client / storefront interface to connect the WAC application repository with the multiple storefront environments that the WAC will be connected with. Through this interface the application is automatically retrieved by the storefront. Depending on the storefront the approval processes can vary from very strict and manual to loose and automatic, depending on the

existence of an editorial team. After successful categorisation in the storefront the WAC client notifies the developer through a status message.

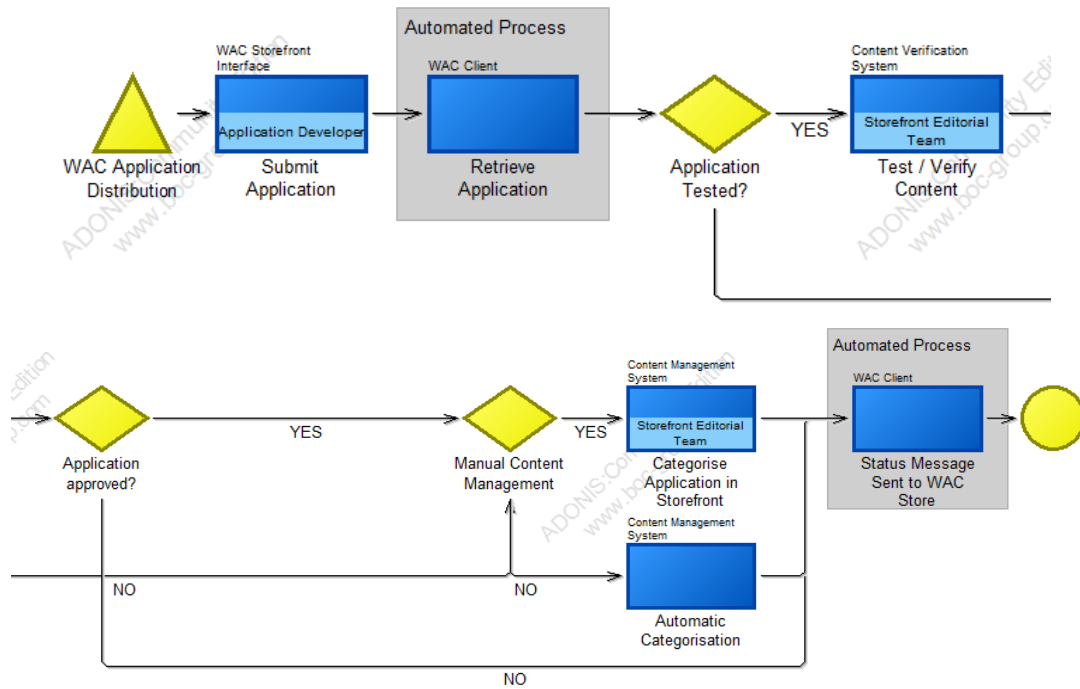


Figure 67 WAC Application Distribution 1.0

WAC Application Delivery / Purchase 1.0

The delivery and purchase process can vary in the WAC environment per storefront.

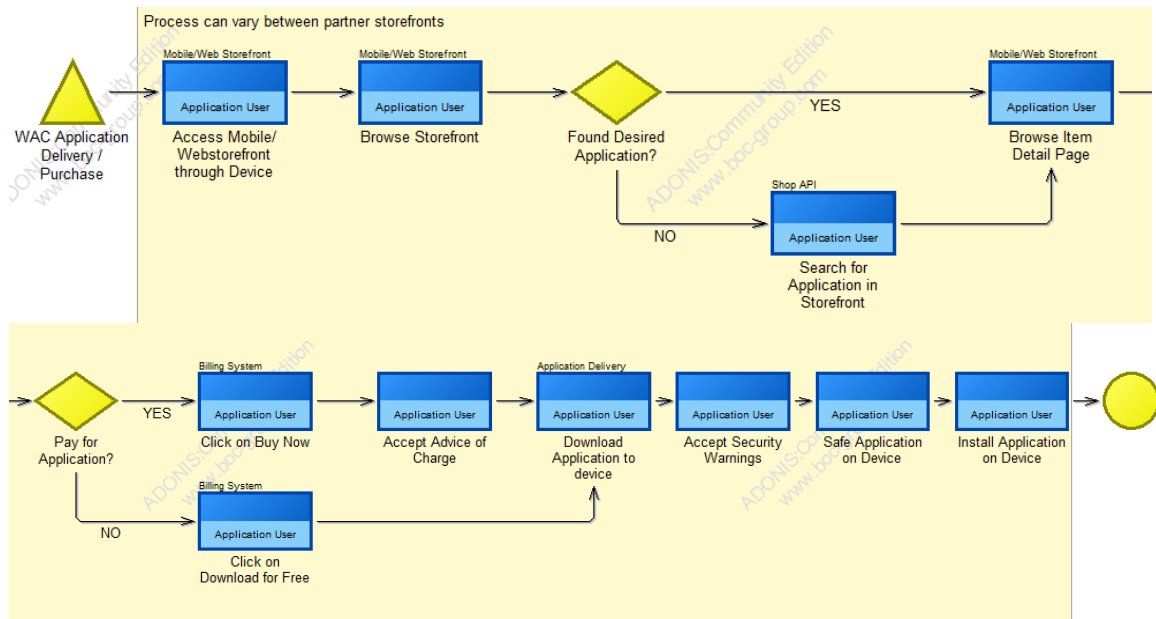


Figure 68 WAC Application Delivery / Purchase 1.0

WAC Storefront Settlement 1.0

The WAC storefront settlement process captures the payment and transaction logs per partner storefront and sends this through the WAC client in an accumulated form to the WAC platform.

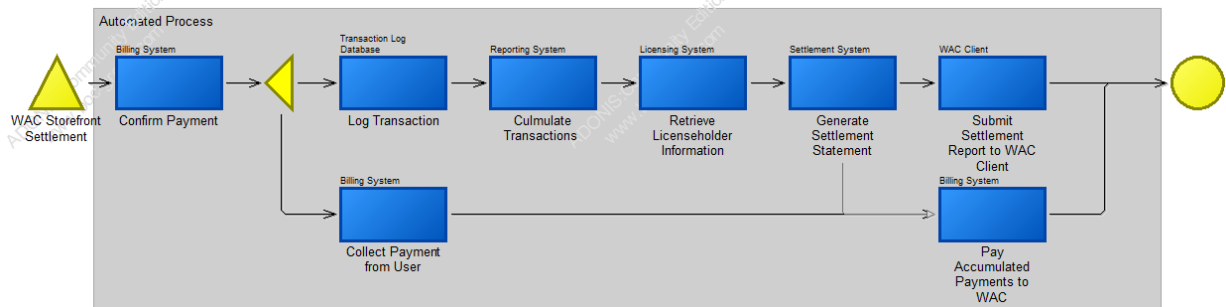


Figure 69 WAC Storefront Settlement 1.0

WAC Settlement 1.0

The WAC settlement process is an automated one that retrieves all cumulated storefront reports and payments and combines them to a single developer report and settlement statement.

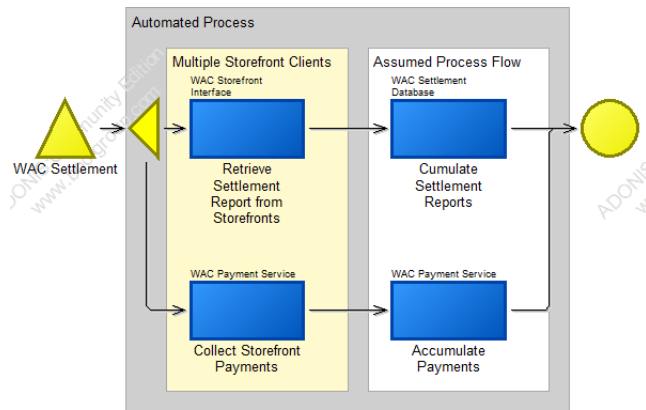


Figure 70 WAC Settlement 1.0

WAC Royalty Payment 1.0

The royalty payment is as well an automated process with regards to the vast amount of publishers involved. The reports, invoices and payment confirmations can be viewed in the WAC developer environment. The developer should receive a single statement and payment from the WAC platform for all partner storefronts.

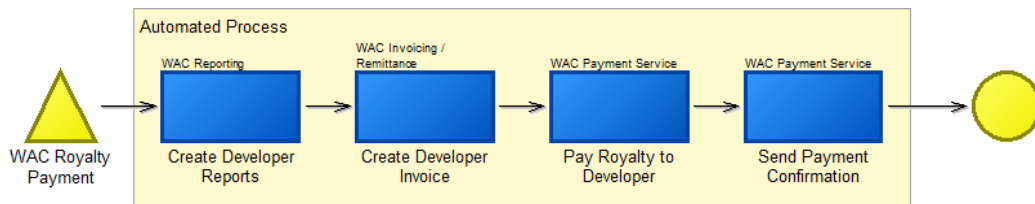


Figure 71 WAC Royalty Payment 1.0

WAC IT Environment 1.0

The WAC environment is a complex system with several elements to make the WAC platform a meta platform like environment as a single interface towards the developer to enable multi platform distribution through a single deployment. WAC is responsible for the developer relationship, the SDK provide, the application distribution as well as the commercial aspects of the distribution eco system towards the developer like reporting, settlement and payments. The WAC client and storefront interface acts as single pipe to the storefronts. These storefronts can vary in appearance, setup and management. The partner storefront environment modelled here is an example storefront setup to serve mobile applications.

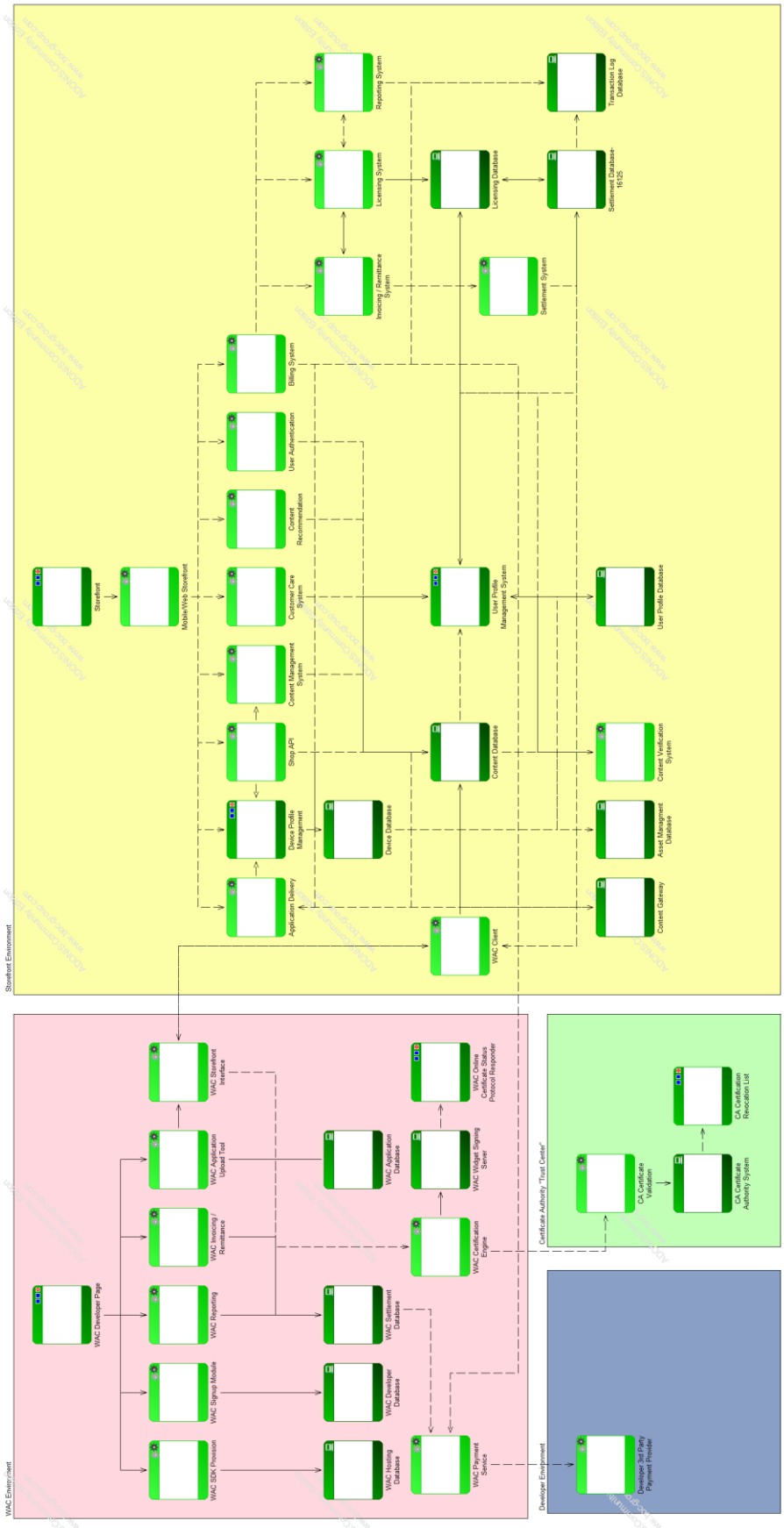


Figure 72 WAC IT Environment 1.0

WAC IT Infrastructure 1.0

This is the assumed Android Marketplace IT infrastructure derived from the business process models discussed in the earlier chapters.

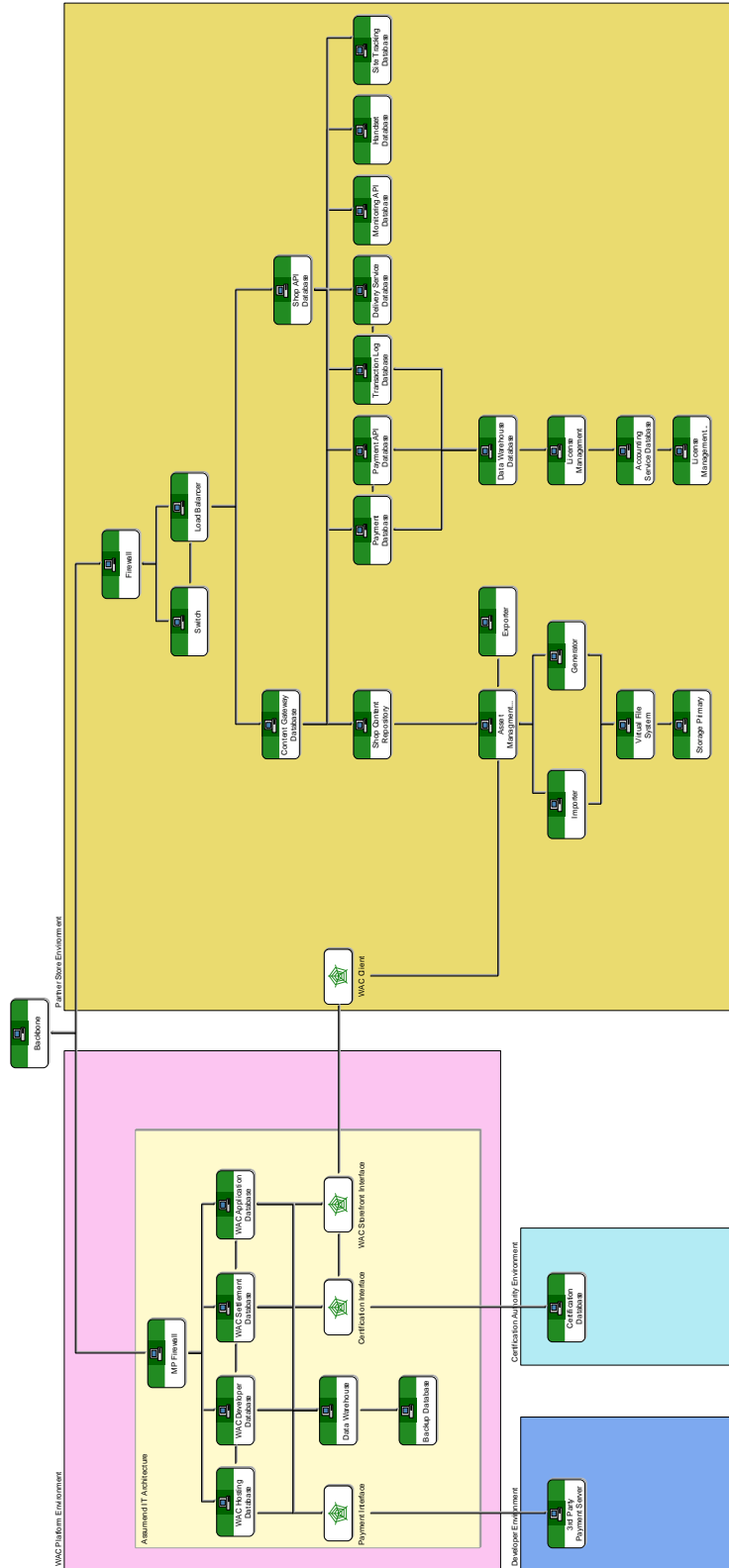


Figure 73 WAC IT Infrastructure

13.3 Appendix 3

13.3.1 Abstract

The mobile application market continues to grow drastically due to the explosion in the sales of mobile device. One of the drivers behind that increase is the development and penetration of application stores provided by different stakeholders in the mobile space especially handset manufacturers, operating system developers and network operators. Therefore handsets nowadays contain competing operating systems, development platforms and physical characteristics.

This diversity leads to a large degree of uncertainty in the mobile space on a strategic, technological, and demand level for mobile application developers. Currently developers need to decide which platform to develop and distribute for. Decision factors include among others the target market, compatibility issue, development time, hardware requirements and scalability.

There is a large literature on architectures and tools that propose to solve the challenges of mobile application development like the cross-platform compatibility. However, the subject of cross-platform distribution is still in development stage and presents an opportunity for further research to limit the resource effort in the development stage and publishing of applications.

This work provides an overview of the existing mobile application and app store market, investigating in business models, processes and infrastructures to develop and distribute mobile applications across multiple platforms.

As the goal is to find an aggregated model for the distribution of cross-platform applications I will start with a top-down approach to identify the existing distribution and infrastructure landscape, therefore I will conduct a research of the literature, internet i.e. Application store developer sites, specialized press and expert talks. The modelling of the business processes will be done with ADONIS® Business Process Management Toolkit and the modelling of infrastructures with ADOit® IT Architecture- & Service Management Toolkit. The final part of the thesis describes the development of a sample application using the WAC environment and the compatibility of on different platforms will be tested.

The discussed Meta platform approach could be a solution to overcome the resource issue of the development and management of the distribution where the developer uses a specified SDK to program the application, publishes the application once to the Meta platform. Application stores that are connected to the Meta platform can distribute this application in their environment to their users. Compatibility across multiple platforms could be achieved by using standardized technologies across the value chain. However, the setup and commercialization of such an approach is a huge task and needs the involvement of all actors in the application eco system and the requirements and processes discussed in this paper could be used to give a foundation to such a project. Furthermore, the evolution of the HTML 5 specification will also bring portability of mobile web applications that rival the power of mobile applications if the Smartphone APIs and network APIs can be made available with this environment the distribution runs then directly from the developer to the customer excluding the application stores at all. This is certainly content for further research on this topic.

13.3.2 Zusammenfassung

Der Markt fuer mobile Applikationen ist in den letzten Jahre drastisch gewachsen, vorallem durch die staendige steigende Zahl and Mobiltelefonen. Gruende fuer den raschen Anstieg sind unter anderem die steigende Anzahl an Applikationsportalen von Endgeraeteherstellern sowie Telekomunternehmen. Durch die Vielzahl an unterschiedlichen Endgeraeten mit konkurrierenden Betriebssystemen, Entwicklungsplattformen, physische Charactersistika sowie Netzwerk Infrastrukturen ist ein in sich komplexes Oekosystem entstanden.

Durch die Unterschiede der Systeme ist vorallem auf Seiten der Applikationsentwickler ein hoher Grad an Unsicherheit in Bezug auf die Entwicklungsstrategie entstanden was die Technologie und vorallem auch die Nachfrage betrifft. Die Frage stellt sich fuer welche Plattform entwickelt und die Anwendungen distribuiert werden sollen. Einflussfaktoren sind vorallem die Groesse des Zielmarktes, Kompatibilitaet, Entwicklungszeit, Hardware Spezifika und das gewuenschte Level an Skalierbarkeit.

Es gibt ein Vielzahl an Literatur zu den Themen Anwendungs Architekturen und Werkzeuge die den Aspekt der Entwicklung von mobilen Applikations und deren Kompatibilitaet ueber multipler Plattformen ermoeglichen soll. Jedoch das Thema der Distribution von mobilen Anwendungen ueber mehrere Plattformen ist derzeit in der Entwicklungsstufe und gibt eine Moeglichkeit fuer weitere Untersuchungen um eine Einsparung und Komplexitaet bei der Entwicklung und auch beim Vertrieb der Anwendungen erreicht werden kann.

Diese Arbeit gibt anfaenglich einen Ueberblick ueber den bestehenden mobilen Anwendungs und Plattform Markt, wobei im besonderen Geschaeftsmodelle, Prozesse und die Infrastrukturen im Bezug auf die Entwicklung und Distribution dieser Applikationen ueber multiple Plattformen, betrachtet werden.

Das Ziel dieser Arbeit ist es ein "aggregiertes" Modell fuer die Distribution von Applikationen ueber mehrere Plattformen zu finden. Im ersten Schritt wird die Analyse der existierenden Literatur in der Fachpresse, Internetquellen und Experteninterviews zum Thema Distributions- und Infrastrukturlandschaft in Form eines „Top-Down“ Ansatzes durchgefuehrt um eine Vergleichsbasis aufzubauen und eine Bewertung durchfuehren zu koennen. Im Folgenden wird die

Modellierung der analysierten Geschäftsprozesse mit dem ADONIS® Business Process Management Toolkit durchgeführt sowie für die Erstellung der Infrastrukturmodelle ADOit® IT Architecture- & Service Management Toolkit verwendet. Die daraus resultierenden Ergebnisse werden analysiert und gegen die „ideal“ Charakteristika verglichen und ein aggregiertes Modell erstellt. Im Anschluss wird der Ansatz eines aggregierten Modells in Form der Meta Platform WAC getestet indem deren Entwicklungsumgebung für die Erstellung einer Beispielsapplikation verwendet und die Kompatibilität auf verschiedenen Plattformen getestet wird.

Der diskutierte Meta Platform Ansatz könnte eine mögliche Lösung für das Ressourcen Problem sein im Bezug auf das Entwickeln und das Management der Distribution indem der Entwickler nur ein SDK für die Programmierung der Applikation verwendet und diese dann über die Meta Platform distribuiert. Applikationsplattformen die an diese Meta Platform angeschlossen sind können die Anwendungen innerhalb ihrer Umgebung an deren Kunden verbreiten. Das Kompatibilitätsproblem könnte durch die Standardisierung der Technologien über die gesamte Wertschöpfungskette erreicht werden. Jedoch ist die Erstellung und Kommerzialisierung eines solchen Ansatzes eine riesige Unternehmung und Bedarf die Unterstützung aller Beteiligten im Applikations-Oekosystem. Die Beduerfnisse und Prozesse die in dieser Arbeit erstellt wurden können als Grundlage eines solchen Projektes verwendet werden.

Darueberhinaus wird die Entwicklung der HTML 5 Spezifikation einen weiteren Schritt in Richtung Portabilität von mobilen Web Anwendungen ermöglichen die eine Alternative zu nativen mobilen Anwendungen darstellt, wenn Smartphone APIs und Netzwerk APIs in dieser Umgebung verfügbar gemacht werden können und ein direkter Vertrieb an den Endkunden durch den Entwickler ermöglicht wird. Hierbei könnten die Applikationsplattformen vollkommen umgangen werden. Dieses Gebiet bietet definitiv Inhalt für weitere Untersuchungen.

13.3.3 Lebenslauf

Beruflicher Werdegang

Senior International Localization Manager Entertainment <i>Deutsche Telekom UK Ltd, Product & Innovation, Hatfield, UK</i>	Nov. 2010 to date
Senior Content Commercial Manager <i>T-Mobile International UK Ltd, Hatfield, UK</i>	Jan. 2010 - Nov. 2010
Content Manager Entertainment <i>T-Mobile International UK Ltd, Hatfield, UK</i>	Jun. 2008 - Jan. 2010
Marketing Manager Mobile Games, Sports, Movies <i>T-Mobile International Austria GmbH, Vienna, Austria</i>	Sep. 2006 – May 2008
Portal Manager <i>T-Mobile International Austria GmbH, Vienna, Austria</i>	Feb. 2005 – Jan. 2008
Associate <i>Friedrich Kriegl GmbH, Vienna, Austria</i>	Jan. 2005 to date
Internship Portal & Content <i>T-Mobile International Austria GmbH, Vienna, Austria</i>	May – Dec. 2004
Key Account Market Research <i>Interconnection Consulting, Vienna, Austria</i>	Feb. – Jul. 2003

Ausbildung

International Economics <i>Centre of business economics, University of Vienna, Vienna, Austria</i> Areas of Concentration: International Marketing & Economical Informatics	1999 – to date
Diploma in Tourism Management & A-levels Hotel- and Tourism school MODUL of the Chamber of Commerce, Vienna, Austria	1994 – 1999
Realgymnasium Oedenburger Strasse 1210 Wien	1990 – 1994