“Gap Analysis”

Deliverable number D3.2

D3.2_Mobile Game Arch_Gap Analysis-V.1.0

Version: 1.0

Last Update: 02/04/2013

Distribution Level: PU

- ***Distribution level***
  PU = Public,
  RE = Restricted to a group of the specified Consortium,
  PP = Restricted to other program participants (including Commission Services),
  CO= Confidential, only for members of the Mobile GameArch Consortium (including the Commission Services)

Partner Name	Short Name	Country
JCP-CONSULT	JCP	FR
European Game Developers Federation	EGDF	SW
NCC SARL	NCC	FR
NORDIC GAME RESOURCES AB	NGR	SW

Abstract:
This document seeks to identify the Gaps in the European mobile games content industry, in view to use these findings in the Recommendations paper, to be published in the last months of this project (June 2013).

"The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 288632"

Document Identity

Title:	Gap Analysis
Subject:	Report
Number:	
File name:	D3.2_Mobile Game Arch_Gap Analysis-v.1.0
Registration Date:	2013.04.02
Last Update:	2013.04.02

Revision History

No.	Version	Edition	Author(s)	Date
1	0	0	Erik Robertson (NGR)	27.02.2013
	Comments:	Initial version		
2	0	2	Kristaps Dobrajs (JCP-C)	29.03.2013
	Comments:	Formatting and editing		
3	1	0	Kristaps Dobrajs, Jean-Charles Point (JCP-C), and Erik Robertson (NGR)	30.03.2013
	Comments:	Formatting and editing		
4		3	Kristaps Dobrajs	02.04.2013
	Comments:	Referencing and formatting		

Executive Summary

The Support to Mobile Game requirements and Architecture project (Mobile GameArch) is an EU FP7 funded Support Action project, gathering 4 different SME's from France (JCP-Consult and NCC) and Sweden (Nordic Game Resources and European Games Developer Federation), with a 369.000 EUR contribution from the EU, running over a period of two years.

The Mobile GameArch project support European actors from both the content side and technology side of mobile industry to join forces to tackle the barrier of growth caused by a lack of standards (e.g. for mobile Application Programming Interfaces (API's)) in the mobile content industry. By making significant contributions to global standards the project intends to create the base for the new generation of mobile technology that will allow the European content the flourish globally and narrow the digital gap in Europe in addition to reinforcing the positioning of SME's working in the European ICT and digital media industry and increasing the accessibility of digital media/service platforms aggregators.

The main objectives of the Mobile GameArch project are to:

- Identify the current growth barriers in the mobile games industry
- Identify a roadmap for growth of the mobile gaming industry and identify specific European barriers
- Support actions to provide recommendations on standards, specifications, actions, practices in the area of mobile technology.

As the result, the Mobile GameArch project will produce a roadmap and set of technological, commercial and political recommendations. It may be assumed that Mobile GameArch solutions to mobile technology standards, commercial recommendations and political initiatives will be applicable to other mobile applications as well.

In view of these objectives, the Mobile GameArch project has created this Gap Analysis, which lays the foundation for the Roadmap to be released at the end of the projects' lifespan, using the identified gaps in mobile game creation processes, namely the preconditions such as EU policy, the process of creation, such as API's, monetizing, such as vision and marketing, and receiving payment, such as payment processes and its gaps.

Contents

List of Figures and Tables

1. 1. Introduction

1.1 PURPOSE OF THIS GAP ANALYSIS

1.1.1 Mobile GameArch Objectives and this Paper

Part of the wider objectives of the Mobile GameArch project are the following, which were also the most immediately relevant objectives for our State of the Art document, which preceded and served as the main departure point for this paper.

- Identify existing and new specific challenges in terms of **standards and specifications that can contribute to defragmentation** and a better functioning of the mobile gaming architecture.
- Identify **challenges in innovation** and R&D (augmented reality, geo-localisation, context awareness, privacy/data-mining, piracy, transmedia, user behaviour, etc...)
- Identify challenges in the fields of **distribution, publishing, new emerging business models, and possible future barriers** to market.

This paper documents the current situation for European mobile games developers in the above main areas. The emphasis has naturally shifted as the work has proceeded, in response to the findings. The State of the Art document and its verification process in co-operation with core stakeholder groups led to in-depth studies of Cloud technology and the potential impact of HTML5 on games, studies which also serve as part of the foundation for the present document. The documentation for this can be found in Annex 5 and 6, respectively.

In addition to this, this paper presents visions of desirable near-future situations, per area, thus indicating the present gaps between what is, and what should be, from the perspective of the European mobile games developers. It is important to note that it is this perspective, looking from the typical micro-SME viewpoint that is chosen for this paper, rather than a top-down industry-as-a-whole view, or as seen through the individual creators' and engineers' eyes. Our main reason for this is that the situation, the access to information, technology, and marketplaces is primarily related to the situation for each company as an independent business entity. What is available of such resources for a large game development studio is not at all certain to be available for a small, but talented, start-up. So, seeing the gaps between what is and what would be desirable must be regarded in relative, quality or volume terms, rather than either/or absolutes.

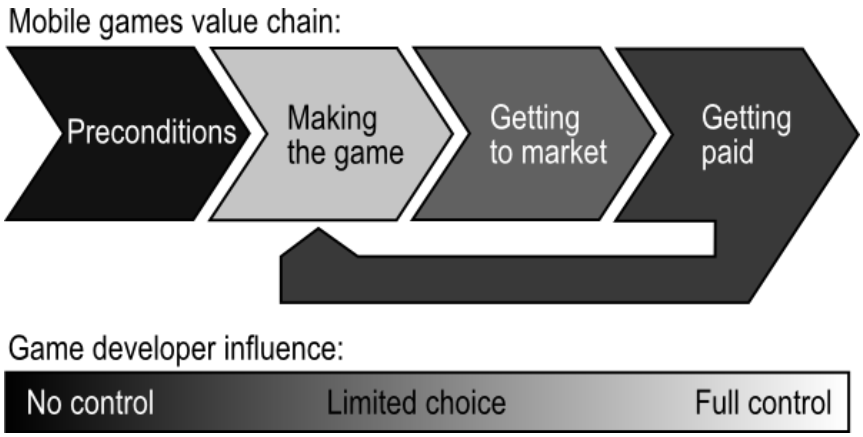
1.1.2 A Vision: The Impact of the Mobile GameArch Project

The potential importance of the Mobile GameArch project as a sign of recognition for game developers as European hopes for the future cannot be overstated. The existence of this project in itself is an indicator of this recognition, and this fact should be continuously communicated. Put differently, this is *the* gap this project is all about; the position of European game developers in society and their own perception of this position. In the following paper, however, somewhat more practical gaps are addressed.

1.1.3 Gaps in the Mobile Games Value Chain

To illustrate and position the gaps for the European mobile games industry, we have chosen the value chain model as our primary structure. We feel that it is better suited to broaden the understanding and seems to have more explanatory value for the situation of this industry, rather than stakeholder, financial, geographical market access, or other models.

Figure 1: A value chain model adapted for mobile games (Source: Own.)



The simplified value chain model, as shown above, for our purposes encompasses the sections of preconditions for the game, making of the game, bringing the game to market, and getting paid for the game. It also indicates the level of influence wielded by the game developers, as documented in our various studies, including this Gap Analysis.

Preconditions are here seen as a wide set of factors, ranging from public policy and applicable law, standards, infrastructure, platforms, operating systems, APIs, middleware, and tools, and the developers awareness and access to information about them. These factors are as a rule not under any influence from the game development company. The developers do have a varying degree of choice when it comes to APIs, middleware, and tools, although some of these are specific for the hardware and OS targeted. Certainly the conditions in the market, with user preferences, competition etc, are not controlled by the developer either, but the process of getting to market we see as an iterative process where the developer's choices will have a direct impact, even if marginal. The developer may choose payment systems and structures, but, again as a rule, have no influence over them as such.

It can be claimed that the mobile games industry is evolving so fast that new gaps are appearing the whole time. Notably in the last year or two, mobile game development has literally exploded. In events like Game Connection, Game Developers Conference or Nordic Game conference the presented projects in development for the mobile platforms easily outnumber all other platforms in 2012-2013.

Hopefully, a general model with the above approach may still serve well for addressing new gaps. This will be put to the test during the remainder of the Mobile Game Arch project, especially bringing the gap analysis over into the Roadmap work – our coming proposals for practical measures to close the documented gaps.

2. Value Chain gaps I – Preconditions

2.1 Public Policy and Applicable Law

2.1.1 Vision for Policy

A non-fragmented European digital single market area is one of the key requirements for the economic growth and new jobs generated by the mobile content creators. That means that we need more middleware, but not necessarily more regulation. However, it could be that a legally harmonized European digital single market area has to be created to not endanger the growth of the European mobile games industry.

2.1.2 Policy Situation

Currently the main political barriers fragmenting the European digital single market area for mobile content are: upcoming VAT regulation and the challenges related to the copyright legislation. Concerning the regulation for protection of minors the situation is more complicated.

There is no single age rating system for digital mobile content. Instead each mobile platform (App Store, Google Play etc.) has their own age rating system for the applications published in the platforms. As there is no European mobile platform on the market anymore, these age rating systems are not based on European standards. The Commission tried to solve this problem by forming a CEO coalition on industrial self-regulation in this area, but almost no European game content creators were included in the process. As a result it must be stated, that bad regulation is even worse than no regulation. Therefore the current anarchic state has some positive aspects.

From the beginning of the year 2015, VAT will be paid in the European digital single market area based on the location of the consumer instead on the location of the company. According to the new regulation, the mobile platform holders, unless otherwise stated, should be responsible for paying the VAT. This will significantly strengthen the position of non-European platform holders in the mobile value chain.

At the moment, no cultural heritage institution in Europe focuses on preserving mobile games and applications partly due to strict patent and copyright legislation. Consequently, a large part of the early European digital cultural heritage might be lost during upcoming years. This might seriously hinder the understanding of the historic trends in the industry.

The current copyright legislation also seriously hinders the use of European music in European mobile content. It is often easier for European mobile content creators to access European music through an American licensing operator or by composing it by themselves than contacting each national collection society. Further from a general point of view the European games industry appreciated the initiative of the commission to develop a unified European license system.

2.1.3 Gaps in Regard to Policy

Firstly, in the area of protection of minors, there is clear gap between the European actors creating the mobile content and mainly non-European actors running the protection of minors system. Consequently, European Commission should launch a parallel initiative to the current CEO coalition that focuses on engaging the European creators of mobile content to self-regulatory actions in the area of protection of minors. This initiative could also reach into the field of technical standards.

Secondly, as the implementation of the new VAT regulation is likely to strengthen the dominant market position of mobile platform holders, the Commission should secure that the way the regulation is implemented does not lead to double taxation and still makes it possible for European actors to bill their content directly through mobile internet as a web service, instead of using the closed mobile ecosystems.

Thirdly, the European copyright framework should be rewritten in a way that it clearly allows breaking the patents and anti-piracy measures for research and preservation purposes. This is the only way to secure that we will not lose early European digital cultural heritage published through mobile stores. Secondly the pan-European licensing for music should be made much easier and flexible, so that more European music could be used in European mobile content.

2.2 STANDARDS

2.2.1 Vision for Standards

Standardisation has always been required in the network domain to ensure product interoperability. This is difficult to achieve in the middleware / API domain where consensus , or harmonization, is always almost impossible to reach as different incompatible implementations could coexist.

This has resulted for instance in the fragmentation of game engines and required many different implementations of the same game. Android and Apple application stores have temporarily improved the situation but fragmentation starts again to appear according to different Android implementations. HTML5 could be a solution which would improve the situation, but it is not suited for AAA or mobile games yet.

Standardisation "ad minimum" is going to be more and more necessary with the deployment of mobile cloud gaming architectures. This because Cloud architectures must evolve and be linked with network resources to ensure QoE and latency for gaming. Therefore common semantics, at least, are required to interface cloud to networks. Additionally, game engines must be adapted to cloud, among other using common assets in a cloud server, or being able to partition an application.

In a more positive scenario, game developers are not only aware of, but consulted with and directly involved in standardisation processes. They are also able to influence harmonization processes by establishing benchmarks, such as for the implementation of HTML5 on mobile devices. Such benchmarks may not only serve as references for game developers on what may be done, but eventually also for browser vendors when they realize that the developers and the users of their games know specifically what to expect and what to demand.

Furthermore, as a technical question, standardised ways of integrating a “panic button” for immediate user reporting of things like cyber bullying, harmful content, sexual behaviour, or grooming, in mobile apps should be explored.

2.2.2 Standards Situation

In the mobile ecosystem, the leading standardisation bodies are World Wide Web consortium (W3C), European Telecommunication Standards Institute (ETSI), Open Mobile Alliance (OMA), Khronos Group, GSM Association (GSMA). It can basically be taken as a given that SMEs, and especially mobile game or application developer studios, have no discernible influence on standards. What is even worse, is that their limited resources for business intelligence makes standards, whether they are of the stipulated or more of the ad hoc harmonization kind, not an opportunity for excellence, but rather something that just happens to these small companies, with varying foreknowledge or early warning.

As has been documented in our HTML5 paper (please see Annex 6), many developers feel that HTML5 could alleviate the fragmented situation somewhat. However, it is quite obvious that HTML5, especially the implementation thereof, still has a way to go. The obvious practical gaps for game developers are today in the differences among web browsers, especially in their implementations of Canvas, for graphics drawing, and in Audio. (For details of practical workshop findings, please refer to Annex 2.)

Cloud is emerging rapidly for mobile gaming applications, but today the cloud infrastructure is not able to meet the constraints introduced by mobile games. Cloud must be decentralised and linked with QoS provided by the network to fulfil the requirements of gaming [2.3]. In additional to current cloud standard efforts, an additional set of standards is required to support this evolution of cloud.

At the same time, there is emerging and increasing demand for non-technological standards especially in the area of payment solutions and protection of minors solutions. In both of these issues there is an increasing demand for standardized procedures and requirements in addition to the harmonization of technological solutions.

2.2.3 Gaps in Regards to Standards

Regarding cloud, the main specific issues which have been identified for standardisation can be listed as:

- Define a framework for resource identification, discovery, authentication; as available resources can appear and vary dynamically
- Define a framework for application partitioning and scalability, like it was done for multimedia content. This concerns of course game engine, but also code scalability, which can be linked to the criterion above (CPU resource…)
- Define a framework for management of offloading application components, including surrogate discovery, handover, and interfacing with the cloud.
- Define pricing/billing interface to allow billing between the different actors in the chain
- Define general interfaces between the different actors (network operators, cloud providers {IaaS/SaaS providers}, application providers)
- Interoperate with cloud at different level (but this is a general issue in cloud already)
- Define security interfaces and framework

Unlike in classical cloud where standardisation is not crucial, but interoperability between different systems is desirable, decentralising the cloud definitely requires standardisation (at least between layers) as it involves the network layer.

On the audio-visual and multimedia aspects, MPEG4 has already proposed a set of tools for media (text, image, video, audio, 2D/3D graphics) and scene representation and MPEG-V (for sensors and actuators). Additionally ARAF is building now on top of MPEG4 with Augmented reality application format. However these solutions have to compete with de-facto proprietary solutions and the success of this set of standards will depend on industry support.

HTML5 could be a viable standard, and thus platform, for mobile games. It is highly unclear for the developers when this may happen, on what hardware, what OSs, and which browsers. The developers need to be better informed on and more involved in the HTML5 implementation processes.

In the area of payment solutions the specific gaps are related to the procedure according to which payments are processed and what information is required for those processes. Furthermore, there is a demand for technological standards on how different payment solutions can be integrated in a mobile app.

On protection of minors the standardisation questions are related more to creating common procedures for in-game micro-transactions for children playing free-to-play games.

2.3 INFRASTRUCTURE

2.3.1 Vision for Infrastructure

Cloud gaming is considered as a promising solution because it addresses several recent problems. With gaming no longer restricted to game console and PC, but expanding to non-game-specific platforms such as TV, tablets and smartphones, users are faced with the issue of running computationally expensive games on devices with no sufficient processing power. This is a relatively old problem, which seems to come back. Mobile and TV manufacturers choose in fact hardware configurations with relatively low computing and storage capabilities in order to address battery issues and to maintain low prices. By performing the intensive computation on the cloud, cloud gaming allows games that require heavy computation to be played on any device. In reality this means, that from a game developer perspective the device and its specifications become less important compared to the content on the cloud.

From the game developers' point of view, the proliferation of devices has made the creation of cross platform games even more complex, increasing costs and reducing control on the software. By being hosted on servers in the cloud, games only need to be developed for the cloud eliminating the need of developing platform specific versions. As a consequence, developers can benefit from increased control over the software, which, differently from now, can be updated whenever required.

2.3.2 Infrastructure Situation

Current cloud gaming solutions, which we refer to as the first-generation cloud gaming systems, suffer from critical weaknesses:

Today's game engines are not designed for cloud gaming. Game engines aim at fully utilizing computing resources of one computer to render the game. The engines are not designed for serving remote clients and hosting multiple game sessions on one server. These software architectural decisions prevent servers to simultaneously serve multiple clients with short processing lag.

Today's cloud infrastructure does not match requirements of cloud gaming. In particular, most cloud providers do not offer specialized hardware resources such as GPU and fast memory. Moreover, servers are located in a relatively small number of data-centers, whose locations are chosen to minimize cooling and electricity costs, rather than to minimize latency to end-users. These infrastructure architectural decisions are in conflict with the needs of cloud gaming, which are interactive (hence highly latency-sensitive), and require specialized hardware resources.

Today's cloud delivery systems are not designed for cloud gaming. Cloud computing centres are centralized (or sub-sharded) systems that generally have less-than-ideal global access. Technologies such as CDNs are not immediately applicable in this field, as the cloud gaming initiative is a specialized subset of the more general cloud computing services. This results in a large-enough latency for the game experience to suffer. At its base, some genres/types of games cannot even be played (action-oriented zero-latency games such as first-person-shooters) while others suffer. In its cradle, the technology can today in practice only be utilized for projects specifically constructed to work around the latency problem and still deliver a captivating gaming experience.

2.3.3 Gaps in Regard to Infrastructure

In general, decentralizing the cloud for mobile, and optimizing the QoE and energy parameters of the mobile terminals requires the definition of a framework, something that we could call "Resource Delivery network", by analogy to CDN, where application components are hosted dynamically in an optimal way, according to the terminal requirements and location in real time.

As detailed in our cloud report (please refer to Annex 6) this includes the definition of the following features:

- Management of application components in the cloudlet: the offloading partitions applications from mobile devices and delivers some of them to run in the cloud, which helps to improve the application performance on the resource constrained mobile devices. However, there is still much work to do on the optimal strategy or algorithm on partitioning and offloading. Moreover, offloading also cannot directly transplant the services from PCs in the cloud to mobile devices because of the different features between PCs and smartphones. This requires the method on how to provide suitable and friendly interactive services for smartphone.
- Surrogate discovery: there are some open issues in surrogate discovery for further investigation and research. First, the service discovery protocols require adapting context-awareness mechanisms. Second, interoperability of service discovery is also a major issue that requires a ubiquitous and pervasive nature. Another major problem is to develop a universal evaluation framework to allow fair comparisons among different protocols.
- Security: in addition to classical cloud security issues, all the requirements which correspond to decentralising the cloud have to be added like application authentication, linking application security and network security, i.e. securing

APIs, protection about service hijacking, ensuring terminal / remote servers authentication, and more generally an overall AAA framework linking network layer and upper layers**.**

2.4 ECOSYSTEMS

It is highly important to note the growing importance of ecosystems in the mobile application space. For our purposes, a definition of an ecosystem is something that comprises a hardware platform, an operating system, and a distribution channel or platform.

The more of these components that are proprietary, in that they cannot be replaced with a parallel product or service, the more there is vertical integration from hardware to software application distribution to the end user, the more of a closed ecosystem is the case. The more that are readily available, as open source, or basically equivalent and interchangeable products from competing manufacturers, the more open is an ecosystem.

Choosing to work in a more closed ecosystem may mean a total absence of choice in the software components, tools, or marketplaces that may be utilised.

The list below is categorised by first-party mobile operating system (OS)/ /hardware/distribution platforms, third-party hardware/distribution platforms and third-party app channels. All first-party OS platforms provide a first-party hardware and distribution platform solution. In some cases (notably Android and Windows), first-party authorised (licensed) hardware and distribution platforms also exist.

As a special case, Google's Android is considered an open-source OS (licensed free-of-charge) which means that several mobile device manufactures (ex. Samsung) have created their own proprietary variations of the Android OS as well as their own hardware and distribution platforms.

Several OS are also served by cross-platform distribution platforms and other independent third-party app channels (some official, some not - as in the case distribution platforms for 'jail-broken' (unlocked from a specific operator & region) iOS and rooted/unlocked Android devices). The list also covers only major/active ecosystems and distributors (For example, Motorola’s app channel is being discontinued).

We find that ecosystems come in two types, those owning OS and those licensing OS. Some comparable distribution platforms are not ecosystems, as they do not have hardware. Some distribution platforms even require a compromised OS. (For more information, please refer to Annex 4.)

Table 1: A typology of mobile games ecosystems and distributors. (Source: own.)

Type	*Characteristics/Brand*	*Distribution platform/-s*	*Remarks*
A	**First-Party (Hardware, OS owned, Marketplace) Ecosystems:**		
	Apple	Apple App Store (https://itunes.apple.com)	
	BlackBerry	BlackBerry World (http://appworld.blackberry.com)	
	Google	Google Play (https://play.google.com)	
	Microsoft	Windows 8 App Store (www.windows8appstore.com)	
	NOKIA	NOKIA Ovi Store (http://store.ovi.com)	Previous generation smartphones
B	**Third-Party (Hardware, OS licensed, Marketplace) Ecosystems:**		
	Amazon	Amazon (http://www.amazon.com) and Google Play (Android)	
	Barnes & Noble	Barnes & Noble (http://www.barnesandnoble.com) and Google Play (Android)	
	HTC	HTC (http://htcapps.com), Google Play (Android) and Windows 8 App Store	
	LG	LG Smart World (http://us.lgworld.com), Google Play (Android) and Windows 8 App Store	
	Ouya	Ouya and Google Play (Android) (TBA)	
	NOKIA	NOKIA Ovi Store (http://store.ovi.com) and Windows 8 App Store	Current generation smartphones
	NVIDIA	NVIDIA, Google Play (Android) and Windows 8 App Store (TBA)	
	Samsung	Samsung (http://apps.samsung.com), Google Play (Android) and Windows 8 App Store	
C	**Third-Party (Independent) Distribution:**		
	Dell	Dell Mobile App Store (http://dellmobileappstore.com)	
	Getjar	Getjar (http://www.getjar.com)	
	Handango	Handango (http://handango.com)	
	Intel	Intel AppUp (available under Windows 8 App Store)	
	Mobango	Mobango (http://www.mobango.com/)	
	OpenAppMkt	OpenAppMkt (http://www.openappmkt.com)	HTML5 based
	Softonic	Softonic (available under Windows 8 App Store)	
D	**Third-Party (Unlicenced) Distribution:**		
	AppsApk	AppsApk (http://www.appsapk.com)	Android ('rooted'/unlocked)
	Cydia	Cydia (http://cydia.saurik.com)	Apple iOS ('jailbroken'/unlocked)

2.5 HARDWARE

2.5.1 Vision for Hardware

From a game developer point of view, hardware manufacturers should agree on a certain number of basic standards, which would benefit gamers worldwide. An intelligent way to differentiate hardware, without creating unnecessary development work.

The Khronos group is doing a lot of work in that direction with their open standards. A second, equally important issue, is that the large majority of game developers have no idea what hardware manufacturers are planning for their next generation of smartphones. They instead need to wait until the phone is brought to market so they can buy one to find out for themselves.

In conclusion, an agreement between hardware manufacturers which could consist of a smarter way to differentiate, without creating problems for game and app developers could be beneficial for everybody, including consumers.

Secondly, a broad dialogue between developers and hardware manufacturers about future devices and the games that could run on them can lead to better products and thus more value for costumers.

2.5.2 Hardware Situation

Playstation, Nintendo and X-Box are computers specifically designed for games. These systems are closed and developers are required to go through admission policies of the console manufacturers in order to publish their games for these consoles. The console manufacturers keep a tight control on the complete ecosystem.

Smartphones have developed into small powerful computers with a wide range of technical features including a large glass touch screen, accelerometers, high resolution

video and photo camera, speakers, microphones, GPS and other localization technologies, NFC, connectivity via Bluetooth, infrared, Internet, digital storage capacity and much more.

Next to these high-end smart phones a mid- and low end type of phones are sold with limited functionalities and a much lower price. These phones are called feature phones.
Games have grown into the bestselling application for smartphones and to a certain extent for feature phones as well. Because of the exponential growth of the smartphone penetration worldwide, the addressable market for games is bigger than any other market for entertainment, including television and internet.

Hardware manufacturers are trying to compete with each other by offering larger screens, high-performance chipsets, lighter phones and other special hardware features that could make their smartphones more unique and more competitive.

However, this differentiation of devices, one of the drivers of the growth of the market, has always resulted in fragmentation: developers need to address different screen-sizes, a variety of GPU's and CPU's super high-resolution cameras and all the other distinctive features. They have to pay the price for this fragmentation, since every different phone requires a dedicated version of the game.

The development of a game often starts before the release of the hardware, basically operating in the dark, without knowing the actual specifics of that device. Thus can be especially dramatic if the hardware introduces totally new features, such as accelerometers or multi-touch interfaces.

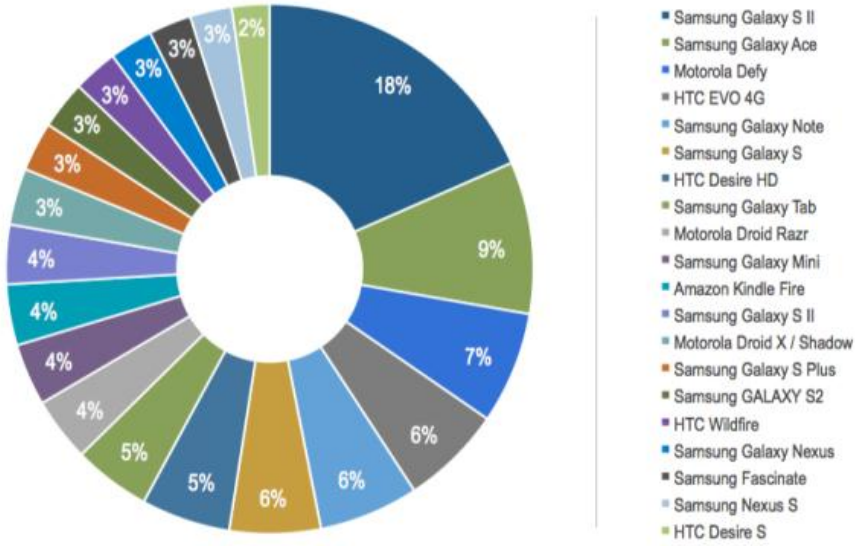
Android divides the range of actual screen sizes and densities into a set of four generalized sizes: small, normal, large, and xlarge, and a set of four generalized densities: ldpi (low), mdpi (medium), hdpi (high), and xhdpi (extra high). As can be seen in the table below, ten of these were in use in accessing Google Play from September 25 to October 1. Two of these amount to over 75% of the total and three over 86%. The 80/20 rule seems to apply.

Table 2: Distribution of device screen sizes in accessing Google Play.
(Source: http://developer.android.com/about/dashboards/index.html as accessed on 2013-03-18.)

	ldpi	mdpi	hdpi	xhdpi
small	1.7%		1.0%	
normal	0.4%	11%	50.1%	25.1%
large	0.1%	2.4%		3.6%
xlarge		4.6%		

The hardware fragmentation still poses high requirements on game developers. To support 80% of the documented user base, it seems that 13 of the 20 devices, or 65%, in the sample below, from May 2012, should be supported. It is not known to us how large a share of the entire distribution that these 20 devices accounted for.

Figure 2: Distribution of user sessions by top 20 Android devices.
(Source: Flurry Analytics, http://blog.flurry.com/?month=6&year=2012 on 2013-03-18.)



It is not only Android that suffers from fragmentation At the time of writing, Apple has released six major versions of the iPhone smartphone hardware, most or all of which are still in use.

2.5.3 Gaps in Regard to Hardware

The diversity and fragmentation of devices is not something that is desirable for mobile games developers. Evolution and introduction of new features enable new, creative games and new user experiences. But this is something that developers should be closely involved in, as they need to be better informed on and more influential in the hardware development processes. We do not even know to what extent different manufacturers consult with game developers, if at all.

2.6 OPERATING SYSTEMS

2.6.1 Vision for Operating Systems

It would be highly beneficial if it was easy for developers to follow operating systems (OS) developments, from features to timelines and market shares, in a unified environment, with notes, tips, peer reviews, comparison tables, ratings, discussions and the like.

With clear benchmarks accessible to developers and users alike, force would be applied simultaneously from both content providers and end customers, a force that would hinder or at least put the light on, negative fragmentation.

2.6.2 Operating Systems Situation

In the early 1980-ies, Sony's Betamax was a more technologically advanced alternative to VHS, but the latter became the choice of the public. When relatively identical

technologies are at war, the one with the best marketing wins: HD DVD vs. Blu-Ray and USB vs. Firewire.

At the moment Android seems to become the predominant Operating System (OS) for Smart Devices and Apple number two. Microsoft and BlackBerry (previously Research in Motion - RIM) follows in a currently undecided third and fourth place, while the newly announced Tizen of Samsung could become a fifth player and potentially a threat to Microsoft's and BlackBerry's positions.

However, in terms of revenue for game developers, iOS is by far number one, leaving Android and the other OS's behind.

Given the current legal battles between Apple and Samsung, Microsoft and Apple against Google, Oracle against Google and Nokia against Apple, Apple against Nokia, etc., it is clear that the real battle is about stopping competitors from catching up with your advanced technologies.

Needless to say, these battles have little or no real meaning for consumers or developers, but for manufacturers a lot is at stake. What does interest developers is how the OS gatekeepers manage the fragmentation inside their operating system.

In the case of the iOS, the number of supported devices is relatively small, but growing with each new release. In the case of Android the variation is much higher and reminds of the times of J2ME. Every handset manufacturer adds proprietary features on top of the Android OS, notably Samsung and Sony, and many App Stores do the same.

The more fragmentation is allowed inside an OS, the higher the costs for developers to reach consumers. For consumers it is quite frustrating to buy a device that allegedly supports Android applications and games, but doesn't work properly in a variety of cases. It can be claimed that Android and Blackberry 10 OS are more or less open, whereas iOS and Microsoft are closed, not open to modification. We currently consider the fragmentation within Android as one of the biggest headaches for mobile game developers in the context of this discussion on operating systems.

Another issue that is relevant for game developers is the insight they may or may not have in the roadmap of the operating system. In general terms, this roadmap is more accessible in open platforms such as Android than iOS.

The market research firm Strategy Analytics reported in January 2013 that smartphone sales grew 38 per cent last quarter 2012 to reach 217 million units worldwide, and over 700 million units for the entire year [reference]. Of those 700 million-plus smartphones sold, 68.4% ran Android as the operating system, while 19.4 per cent ran iOS. Thus, a developer seems to be able to reach around, or more than, 80% of devices by supporting these two operating systems.

Below is shown data collected from February 19 to March 4, 2013 on devices accessing Google Play. It seems that a developer supporting two out of ten OS versions would cover 72.5% of the used devices and with three versions reach 87.4%.

Table 3: Android version distribution for devices accessing Google Play
(Source: http://developer.android.com/about/dashboards/index.html as accessed on 2013-03-18.)

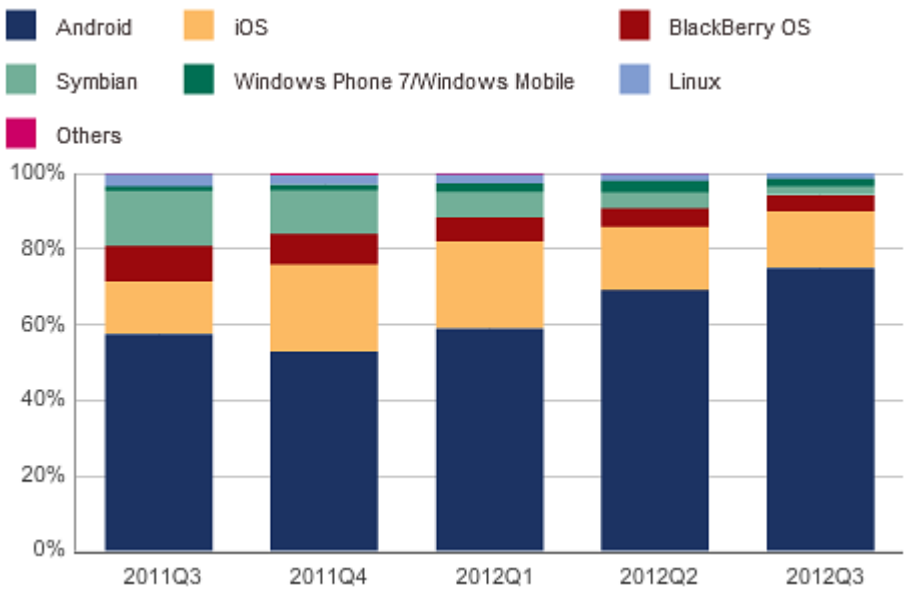
Version	*Codename*	*Distribution (%)*
1.6	Donut	0.2
2.1	Eclair	1.9
2.2	Froyo	7.5
2.3 - 2.3.2	Gingerbread	0.2
2.3.3 - 2.3.7	Gingerbread	43.9
3.1	Honeycomb	0.3
3.2	Honeycomb	0.9
4.0.3 - 4.0.4	Ice Cream Sandwich	28.6
4.1	Jelly Bean	14.9
4.2	Jelly Bean	1.6
		100.0

Still, a developer needs to be aware of the developments in the operating system market, and to track the progress of at least the following eight OS's that seem to be serious contenders in 2013 and 2014, as well as their predecessors:

- Android (Google, US)
- BlackBerry 10 and earlier (BlackBerry, previously RIM/Research in Motion, CA)
- Firefox OS (Mozilla, US)
- iOS (Apple,US)
- Sailfish (Jolla,FI)
- Tizen (Samsung, KR)
- Ubuntu (Canonical, UK)
- Windows Phone 8 (Microsoft, US)

There are of course highly differing opinions of the future potential and impact of the above. Not all predictions, positive as well as negative, will become reality. Just looking at recent history, the market shares of operating systems can change quite rapidly, as the chart below shows.

Figure 3: Worldwide smartphone OS sales from 2011Q3 to 2012Q3 (Source: IDC Worldwide Mobile Phone Tracker, November 1, 2012.)



2.6.3 Gaps in Regard to Operating Systems

The more fragmentation is allowed, the higher the costs for developers to reach consumers, and the more frustrating for consumers that expect to, but do not get reached.

Mobile game developers need insight into, and potential influence over, the development plans for operating systems. We need to document to what extent OS manufacturers consult with game developers on issues regarding OS development.

3. Value Chain gaps II – Making the Game

3.1 APIs

3.1.1 Vision for APIs

It would be highly beneficial if it was easy for developers to look up APIs - application programming interfaces - in a unified environment, with notes, tips, peer reviews, comparison tables, ratings, discussions and the like.

This would enable mobile game developers to make more informed choices, and thus improving both their production efficiency and the end user's benefit.

3.1.2 API Situation

An API allows a developer to access services provided by the operating system or other components running on the system. The API exposes these to the application, but of course requires the developers' knowledge of the location, names, functions and methods to call, constants and variables to pass to these, and the results to expect. Using APIs may make applications both more conforming in appearance and more efficient to make, but also enable functionality that could be impossible for the developer to re-create in application code, as it requires access to system and/or hardware resources. As a rule, APIs pose no restriction on the programming language or development environment used by the developer.

APIs come documented with SDKs - software development kits - from OS vendors, but also from other sources, for exposing components but also as other means of accessing OS services. Thus, there may be alternative APIs providing the same functionality. APIs are OS- and generally often also OS version-specific, and may very well also be tied to a single mobile operator's services, such as location information provisioning.

Something of a special case within APIs are analytics tools. Including such allows the developer to track the usage and get a lot of feedback on how their game is used, by whom, on what OS versions and handsets. Especially what drives, or hinders, monetization within the game is of crucial importance to the developer. As a rule it is gathered by the analytics provider, which also has their own uses for the gathered data. Notable providers are AppAnnie, Distimo and Flurry.

3.1.3 Gaps in Regard to APIs

Many APIs exist, and for example the Wireless Industry Partnership lists some 185 entries. ("WIP Connector API Catalog" http://www.wipconnector.com/apis as accessed on 2013-03-16.) There seems to be no obvious, central and trusted resources that provide an overview and can support game developers in evaluating APIs to use for their projects.

Developers, as a rule, lack insight into how analytics providers process and retain their data. Your analytics provider being acquired by your competitor is probably not an ideal situation.

3.2 MIDDLEWARE

3.2.1 Vision for Middleware

It would be highly beneficial if it was easy for developers to look up middleware and game engines in a unified environment, with notes, tips, peer reviews, comparison tables, ratings, discussions and the like.

A very important aspect of middleware is limitations. Aggressive marketing obscures facts from the developers and hides the weaknesses of a system. A middleware solution might promise code to be run-able on a wide spectra of platforms and devices, but in reality is a very constricted platform when the developer aims for such broad market availability. True write-once-run-everywhere systems do not exist and the complexity of development and difference between platforms (both technical and ideological) makes this a very difficult goal to achieve.

An ideal information resource for developers would shine a light on both advantages but also drawbacks of different middleware, thus enlightening the developer to and enabling informed choices, thus ensuring a higher-quality end-product to the consumer.

3.2.2 Middleware Situation

Middleware refers to a professionally produced product used in development. Today, companies thrive in software development by just producing middleware for other companies to consume. This practice is vigorously pursued in game development, as games by nature are complex and combine enough components to make middleware a lucrative solution to cut production time and costs. It is hard to find a product today that ships without any form of middleware, as many components are regarded as *de facto* standards and used for any triple-A production.

The rise of middleware as an important factor in game development comes from the equally expansive rise in game development complexity. With new systems, consumers' technological demands and market segregations, the costs of pure development skyrocket. Developers look for ways to cut costs and middleware solutions are thus built around this premise.

Over the last few years, development in cross-platform game engine middleware has expanded fiercely. Developers are today offered several different solutions to cut costs and deploy on many platforms at once, with the write-once-run-everywhere paradigm. As mentioned several times in this document, segregation of the market and walled-garden architectures drive developers towards these solutions. A large problem, that is in detail discussed in the gap subsection, is that being locked in to a different platform still is problematic, even if this platform is less restricted than the previous.

Sometimes the distinction between middleware and game engines is made, on the account that game engines offer a complete solution for an entire production, while middleware is regarded as having a specialist role covering a single system, or group of

related subsystems. We will in the coming sections regard game engines as middleware as well.

Middleware provides a richer functionality than that of single APIs. As a rule, middleware enables both easy access to lower-level systems services, often replacing - and abstracting - a number of APIs, but typically also the same functionality across different platforms and OSs, allowing the developer to code once, but deploy in many configurations. Middleware may very well encompass several to basically all the fields of user-entry processing, sound playback, 2D and 3D graphics, memory handling, physics and AI processing, including higher-level scripting. The middleware suites providing the most of these are called “game engines”. The choice of game engine as a rule comes with a requirement for the use of a specific programming language, as the ready-made modules and templates that are important parts of a game engine are in a single language.

Below is a table listing a number of middleware solutions and their features:

Table 4: An overview of 35 mobile games middleware engines listed as covering both iPhone and Android (Source: http://www.mobilegameengines.com as accessed on 2013-03-16)

Product	*2D*	*3D*	*Language*	*Licence*	*Reviews*	*Games*
Antiryad Gx	x	x	C++	proprietary	0	0
App Game Kit	x		C++	proprietary	0	4
BatteryTech Engine	x	x	C++	proprietary	0	5
Cocos2d-x	x		C++	open source	1	5
Corona SDK	x	x	Lua	proprietary	0	11
Cuttlefish Engine	x		Cuttlescript	proprietary	0	0
EDGELIB	x	x	C++	proprietary	0	4
Emo-Framework	x		C, C++, Objective-C	open source	1	0
Esenthel Engine	x	x	C++	proprietary	0	0
GameKit		x	C++	open source	0	0
GameMaker	x		None (graphical)	proprietary	0	0
GamePlay		x	C++	open source	0	0
GameSalad	x		None (graphical)	proprietary	0	0
Gideros Mobile	x		Lua	proprietary	2	2
Ignifuga Game Engine	x		Python, Cython	open source	0	0
LiveCode	x		LiveCode	proprietary	0	0
Marmalade	x	x	C++	proprietary	1	1
Moai	x		C++	open source	0	6
MonoGame	x		C#	open source	0	0
MoSync Mobile	x	x	C, C++	open source	0	0
NME	x	x	haXe	open source	0	0
openFrameworks	x	x	C++	open source	0	0
Orx	x	x	C, C++, Objective-C	open source	0	0
Papaya Social Game Engine	x		ActionScript	proprietary	0	0
PlayN	x		Java	open source	0	0
Proton SDK	x	x	C++	open source	0	0
QuickTiGame2d	x		Javascript	open source	0	0
RapaNui	x		Lua	open source	0	0
ShiVa3D Game Engine		x	Lua	proprietary	6	6
SIO2 Engine	x	x	C++	proprietary	0	7
SpacePort	x		Javascript	proprietary	0	0
Starling Framework	x		ActionScript 3	open source	0	0
Stencyl	x		Graphical, Action Script 3	proprietary	0	5
UNIGINE Engine		x	C++	proprietary	0	1
Unity3D		x	.NET (Mono)	proprietary	2	14

Referring again to the Mobile Game Engines website (http://mobilegameengines.com as accessed on 2013-03-18), we find that of the 62 game engines for iPhone, 31 also

support Android, four support both Android and HTML5, and two support iPhone and HTML5 but not Android. This site seems to provide quite a lot of what is expressed in our vision, but the number of reviews and references are limited - more than half of the engines currently have neither.

3.2.3 Gaps in Regard to Middleware

Today, a developer can find middleware for any part of production, making gaps in the traditional (segregational) sense virtually non-existent. However we can also discuss gaps between middleware. As boldly stated before, there is no perfect middleware solution, meaning that the choice of middleware is always with its drawbacks. Some platforms might not be supported by your choice of middleware, or specific functionality on a specific platform might have been ignored to supply broad-spectra support.

For developers, this means that their product may have to co-exists in several middleware systems (and even in several game engines). Since most systems use different paradigms for development, switching from one to the other can in many cases be an outright painful process. These types of gaps are very hard to work around or reduce, as it is the middleware manufacturers right to create a system catering to their development ideology and choice of language, platform, feature-set, etc. Regardless, it still is a gap for developers to overcome.

It should also be noted that along with all the advantages that middleware and game engines bring, particularly as efficient and fragmentation-addressing parts of the development process, comes also the risks of lock-in and lock-out. Committing to a certain game engine that is subsequently not maintained and developed by the producer is certainly a disadvantage. There are also cases of commercially available game engines, in the wider game industry, being acquired by publishers and then made not only proprietary, but internal and private, closing off other users.

3.3 TOOLS

Like any industry, game development uses a series of tools to facilitate production. By tools we generally mean ‘’productional’’ software but it is extended to hardware as well in some cases, such as motion-capture equipment, cameras, audio-recording, etc.

The two main categories of tools are Integrated Development Environments (IDEs) and graphical tools such as 3D-modelling software (Maya, 3D Studio Max, Blender) and 2D-graphical software (Photoshop, Gimp). These are used in a majority of game development scenarios and are thus very important to a developer.

3.3.1 Vision for Tools

It would be highly beneficial if it was easy for developers to look up mobile game development tools in a unified environment, with notes, tips, peer reviews, comparison tables, ratings, discussions and the like. This would simplify the process of tool selection but also open up multi-platform development by choosing the right - non-platform-locking - tools.

3.3.2 Tools Situation

The choice of toolchain can either be a preference of the studio itself (which is often the case in regards to graphical tools) or a restriction enforced by the platform (which is usually the case for IDEs). For instance, developing for iOS (or any Apple-developed

hardware platform) must be done in XCode, an IDE provided by Apple (though there are some exceptions to this rule).

Beyond the two main toolchains there is a subset of smaller tools with the purpose of creating anything from a bitmap font to physical models. Many of these tools are legacy-ware that is unsupported, or locked to inconvenient platforms, but are still in use for lack of better alternatives. This is why in-house-developed tools are very popular in game development. Sometimes it is preferable to reinvent the wheel in order to get a tool that snugly fits in one's production pipeline, rather than trying to crowbar in a bulky piece of software that only works half the time, so to speak.

Larger game development studios create entire multi-purpose toolchains that are proprietary but can be shared with other studios owned by the same publisher. Some studios also create full-fledged game engines, with much of the production pipeline and toolchain in place, for use by other developers at a considerable license fee.

3.3.3 Gaps in Regard to Tools

Tools today affect game development in a technological gap. The choice of tool might enforce a certain paradigm or technological choice that cannot be reverted (unless production itself is).

Many tools have implementations and connected APIs/libraries are restricted to a single platform, making multi-platform development much harder and might force smaller studios to disregard multiple platforms entirely, at a cost for both the studio and the unknowing consumer.

Finally, for proprietary tools there is no way for game developers to be active in the development of the tools themselves, other than plain suggestions. There are some initiatives in the tools sector aiming to streamline and unify different productional formats so that tools and APIs can be synchronized over a myriad of platforms, but still there is much to be done. Involving more of the game development community in a community-based process can speed up the relatively slow standards process and also highlight actual needs of actual game developers.

4. Value Chain gaps III – Getting the Game to Market

4.1 VISION FOR MARKETING

It would be highly beneficial if it was easy for developers to look up market channels in a unified environment, with market penetration, business models and terms, free-to paid conversion rates, notes, tips, peer reviews, comparison tables, ratings, discussions and the like. It would be desirable if something corresponding also was available from the end users' perspective, highlighting European mobile games and their producers.

This would enable mobile game developers to make more informed choices on what market segments to address with what products. This would not only improve their production efficiency, but also the end user experience.

4.2 MARKETING SITUATION

In the first 10 years of the mobile games history, mobile operators were the main gatekeepers of the industry and sometimes handset manufacturers allowed game developers or publishers to pre-load their games on certain devices.

Today every developer can become a publisher on iOS, Android, Microsoft, Nokia or RIM or the stores of GetJar, Amazon, eBay and other stores.

However, this new situation has led to an extremely competitive industry, where those who have the means to boost their visibility and use the available tools to optimize the monetization of their games can reach unprecedented high returns on investment.

Traditional marketing, such as advertising and PR is not really efficient in the app store world. Talking to journalists from Pocketgamer, Slide to Play, Touch Arcade, IGN and many others can increase the discoverability of an app and raise the chances for being featured in an app store.

Understanding social media has become a key asset in promoting a game. First of all, the promotion of a game cannot be done without a good social media campaign, which involves the use of Twitter and Facebook of as many people as possible who worked on the game. This would give players the possibility to build a deep relationship with the game and the company who made it.

Secondly, in the game the presence of Twitter, Facebook, YouTube buttons or those of other networks will allow players to identify themselves and to post their achievements immediately.

TapJoy, Chartboost and many other services allow publishers and self-publishing developers to buy users and downloads. The more money they spend, the higher they will be in the charts. Success on the app stores can now be bought to a certain extent, since a game which is listed in the top ten gets noticed by a large number of consumers. And this fact alone generates more downloads and in time revenues.

Costumer care. Responding quickly and adequately to questions or remarks from players create a lot of goodwill and allows the company to learn from its clients.
Analytics. Advanced in-game data mining allows game developers to improve their game and to adjust difficult levels or raise the price of popular virtual items and/or power-ups.
External analytics allow companies to understand their competitive environment better and to find opportunities for to-be-launched games.

Localisation. The mobile games market allows you to publish games in over 50 countries worldwide. Localisation of games is important and can prove to be elaborate. Not only language, but also color and virtual items, pricing, characters may need to be adapted to local taste.

App Store specifics. Every large app store has its specific rules and gatekeepers and therefore different times to market. Most of them want to test the application and check if there is any content they do not allow, such as sex, violence, politically or ethically sensitive content.

Piracy. Pirates are hacking every operating system, but are exceptionally active in Android. This is making Android one of the worst monetizing OS. Nothing substantial is currently done against Android piracy.

Discoverability. With an estimated 200 new games being launched on iOS, the discoverability of apps becomes more difficult every day and ends the dream of developers who think they can launch their game without any form of marketing.

Most developers are now self-publishing and decide themselves what to deliver and what not to do. For most of them the decisions regarding the production of additional builds of a game, adapted for varying hardware devices, different OSs, and separate market channels or business models, are based on the expected return on investment.

Questions the developer needs to answer regarding costs are such as: “How much does it cost to port my game, originally developed for iOS, to Android?” “Which Android stores does it make sense to try to make distribution agreements with?” “How much will a third or fourth platform, such as BlackBerry or Windows, cost?”

Regarding revenues, at the very least the following questions must be answered:
“What kind of revenues can I expect from Android and from each of the different stores?”
“What kind of revenues can be expected from a third or fourth platform?”

Visibility, or discoverability, is a main issue for developers. The competition is fierce, and the ad-campaigns known from the online games space are entering the mobile field, which becomes problematic for the independent developer.

“[...] Flurry calculates that the difference in revenue generated per active user is still 4 times greater on iOS than Android. [...] At the end of the day, developers run businesses,

and businesses seek out markets where revenue opportunities are highest and the cost of building and distributing is lowest. In short, Android delivers less gain and more pain than iOS, which we believe is the key reason seven out of every ten apps built in the new economy are for iOS instead of Android." (Source: "App Developers Signal Apple Allegiance Ahead of WWDC and Google I/O, Posted by Peter Farago on Thu, Jun 07, 2012", http://blog.flurry.com/default.aspx?month=6&year=2012 as accessed on 2013-03-16)

From a game developers point of view only one thing really counts: Return on Investment. Any such ROI calculation must take the following issues in consideration:

- Revenue share for the developer. This is mostly 70% for the developer.
- Installed base and fragmentation of installed base. The more fragmented, the higher the development costs.
- Piracy and what does the OS owner do against it?
- Average pricing and consumer behavior on that specific OS.
- Competition: On iOS only, about 200 games are launched each day.
- Publishing strategy. Does the developer or publisher want to be present on all platforms and reach as many players as possible?
- Time to market: how long does it take to get your game through the selection procedures?
- Any censorship for specific games: violence, pornography, etc.

To make this calculation, and then basing an informed choice on this, it is of course essential to have unambiguous information on all the issues in question.

4.3 GAPS IN REGARD TO MARKETING

The current, to a large extent lottery-like system, is not a lasting solution, though it will most likely not change without outside stimuli. The storefront suppliers' sales are not hurt by congestion of high-quality content - quite the opposite. For this to be a healthy ecosystem also for small to medium-sized developers changes are needed. Since these systems are not open, but proprietary, the changes must probably come from the outside, possibly through consumer-enlightenment programs and initiatives.

Unfortunately the same small and medium-sized developers do not possess the means of pursuing such programs and thus fall behind their larger (often American or Asian) competitors due to lack of marketing power. A system that ensures the European mobile consumer's awareness of the problem would benefit both the consumer (as high-quality content can instead be browsed on an interest basis, rather than luck basis) while developers can focus on creating games that target niche markets with quality productions, instead of droves of lottery-win-hopefuls of under produced general whole-market content.

In general, the markets are too diverse, and are changing too fast, even for our highly creative and technologically skilled micro-SMEs. They just cannot make informed decisions on what markets to address and how to reach them. It is simply far, far beyond their means to gather and continuously analyse the necessary data.

5. Value Chain gaps IV – Getting Paid for the Game

5.1 Vision for Payments

For game developers it is crucial that their customers, the players, can make payment for and within their games with the payment solutions they prefer to use. As games are sold globally there should be a universal joint payment solution uniting existing payment solutions in one system that could be used in different platforms.

Game developers, if they had a joint payments solution, would be able to sell everything to all platforms under one and the same banner, and keep one and the same revenue also share. Of course the store owners are able to stop this the same moment it measurably hurts their business. This means that any such strategy needs its next moves well planned in advance.

Today, it would be beneficial if it was easy for developers to look up payment solutions in a unified environment, with market penetration, terms, comparison tables, ratings, discussions and the like.

5.2 Payments Situation

Today, there exists a wide diversity of payment systems in the mobile games field. For an overview, please refer to Annex 3.

Currently, mobile distribution platform holders are taking stronger and stronger control over what payment solutions can be used in their systems. From the beginning Apple has limited the payment solutions that can be used in Appstore to credit cards. While Google has opened its system for carrier billing, besides that it only accepts payment solutions provided by major credit card companies.

Consequently non-European credit card companies are taking over the mobile payment markets, working with non-European mobile platform holders. This means that European payment solutions like operator billing and direct debit payments are facing new challenges in the mobile field.

One of the main challenges hindering the integration of European payment systems to the mobile platforms is the fact that there is no standardized way to do that. Each payment solution requires different information and often also uses different technological solutions for transferring the payments.

5.3 Gaps in Regard to Payments

There is a need for single solution uniting the different payment solutions in a way that they can be easily integrated to different mobile platform. Furthermore, a more unified

European pressure is needed towards non-European mobile platform holders forcing them to integrate the European payment solutions in their systems.

6. Coming Roadmap Work

The roadmap, our next step in the Mobile GameArch project, is meant to point out how the gaps identified here, between what the situation is and what would be desirable, may be be closed.

Such recommendations must take into account both the severity and prevalence of the problems posed by the gaps. In weighing that against the realism, economy and effectiveness of possible solutions, the ability of the European mobile game developers, as individual businesses and as a collective, we should be able to suggest some priorities. The priorities should take into account not only possible relationships and influences between the different measures, of course aiming for harmony and maximizing synergies, but also consider external relationships with strategic partners, such as stakeholders within or outside the mobile games value chain.

For now, we stop at summarizing our findings, grouping them by the value chain links, in the table below.

Table 5: An overview of identified gaps

I – Preconditions

Policy	European creators of mobile content should be encouraged to self-regulatory actions in protection of minors.
	VAT regulation must not lead to double taxation.
	European actors should be able to bill for their content as a web service, not be forced to use Non-European closed ecosystems for payments.
	The European copyright framework should secure that we do not lose early European digital cultural heritage published through mobile stores.
	The Pan-European licensing for music should make it easier to use European music for European mobile content.
Standards	Decentralising the cloud definitely requires standardisation as it involves the network layer.
	Developers need to be better informed on and more involved in HTML5 implementation.
	Standards on payment solutions integration are needed.
	Common, good practice for in-game micro-transactions for children is needed.
Infrastructure	Mobile cloud-based games need definition of management of application components, surrogate discovery, and security.

Hardware	Developers need to be better informed on and more influential in hardware development processes. Knowledge of hardware manufacturers' relations with game developers is needed.
OS	Developers need to be better informed on and more influential in development paths for operating systems. Knowledge of OS manufacturers' relations with game developers is needed.
II - Making	Overviews for evaluating APIs are needed. Developers lack insight into how analytics providers process and retain their data. Middleware and game engines brings the risks of lock-in and lock-out. Unified production formats are needed for tools and APIs to be synchronized across platforms.
III- Marketing	The European mobile consumer's awareness of available games is low. Developers cannot make informed decisions on what markets to address and how to reach them.
IV - Payments	A single payment solution does not exist. Non-European mobile ecosystems and distributors should integrate European payment solutions.

7. References

Blog.flurry.com (2012) *June, 2012 | The Flurry Blog - Mobile Application Analytics | iPhone Analytics | Android Analytics*. [online] Available at: http://blog.flurry.com/?month=6&year=2012 [Accessed: 18 March 2013].

Blog.flurry.com (2012) *June, 2012 | The Flurry Blog - Mobile Application Analytics | iPhone Analytics | Android Analytics*. [online] Available at: http://blog.flurry.com/default.aspx?month=6&year=2012 [Accessed: 18 March 2013].

Developer.android.com (2013) *Dashboards | Android Developers*. [online] Available at: http://developer.android.com/about/dashboards/index.html [Accessed: 18 March 2013].

IDC (2012) *Worldwide Mobile Phone Tracker*. [report].

Mobilegameengines.com (2012) *Mobile Game Engines: find the perfect engine for your mobile game*. [online] Available at: http://www.mobilegameengines.com [Accessed: 18 March 2013].

8. ANNEX 1: Methodology – Mapping the gaps

As can be seen from the table below, the gaps analysed in this document are a combination of cross-cutting themes mapped during the State of the Arts analysis. They were first identified in the workshop organized in Paris and they were analysed further in the extensive State of the Arts document.

After the State of the Arts document the consortium identified the development of HTML5 and cloud technology to be the key mega trends reshaping the themes in the near future. Consequently the potential gaps were analysed further through these themes. As an outcome of this process the gaming workshop in Helsinki and smaller workshops in Aalborg and Malmö were organised and two separate papers were finalised analysing these megatrends in detail.

The outcomes of all the aforementioned documents are now united in this document and developed further to define the precise gaps in the mobile game ecosystem. Based on this document a road map document will be built to find the ways to overcome these challenges hindering the economic growth and generation of new jobs in the European digital single market area.

Beside the official workshops organised by the project, the analysis is also based on the wide interaction the consortium partners have with different stakeholder groups in their daily work. For example, public Policy and Applicable Law is also based on the feedback EGDF collects from European game developer associations while building up its positions on the different policy issues.

Table 6: How the gaps were identified

	D.2.2. Workshop in Paris	D3.1. SoA	D.2.3. Workshop in Helsinki	Cloud paper	HTML5 paper
Public Policy and Application Law	2.1 Is there a real opportunity for a European mobile games industry...	3.6 The support and regulatory landscape	2.3. Investment in the mobile gaming industry?	7. Technology issues related to decentralised cloud	7. European policy implications
Standards	2.2. The mobile games tech talk...	4.2 Anti-Fragmentation initiatives	2.1 HTML5, the solution to fragmentation?	5. Architecture	4. Technical challenges for HTML5...
Infrastructure	2.2. The mobile games tech talk...	4.9 The networks	2.2. Mobile cloud reconnecting telcos to mobile value chain?	5. Architecture	
Ecosystems	2.3. The business of mobile games...	3.2 Roles in value chain	2.2. Mobile cloud reconnecting telcos to mobile value chain?	2. requirements from game developers 3. requirements from operators	3. Commercial challenges for HTML5...
Hardware	2.2. The mobile games tech talk...	4.4 The evolution of handsets			4. Technical challenges for HTML5...
Operating systems	2.2. The mobile games tech talk...	4.5 Evolution of software and platforms	2.1 HTML5, the solution to fragmentation?		4. Technical challenges for HTML5...

API's	2.2. The mobile games tech talk...	4.1 Addressing fragmentation	2.1 HTML5, the solution to fragmentation? 2.2. Mobile cloud reconnecting telcos to mobile value chain?	7. Technology issues related to decentralised cloud	4. Technical challenges for HTML5...
Middleware	2.3. The business of mobile games...	4.6. Cross-platform development frameworks and 4.7 integrated development frameworks	2.1 HTML5, the solution to fragmentation?		4. Technical challenges for HTML5...
Tools	2.3. The business of mobile games...	Same as above			4. Technical challenges for HTML5...
Getting the game to Market	2.1 Is there a real opportunity for a European mobile games industry...	3.3. Getting to Market	2.3. Investment in the mobile gaming industry?	7. Technology issues related to decentralised cloud	3. Commercial challenges for HTML5...
Getting paid for the game	2.1 Is there a real opportunity for a European mobile games industry...	3.4. Generating revenue	2.2. Mobile cloud reconnecting telcos to mobile value chain? 2.3. Investment in the mobile gaming industry?	6. Pricing of cloud gaming	3. Commercial challenges for HTML5...

8.1 MAPPING THE STATE OF THE ARTS: WORKSHOP IN PARIS

The first workshop was held in Paris on December, 6th 2011 during the Game Connection Europe and was the occasion to gather experts and representatives of the video game industry and mobile game developers to discuss about the opportunities of the European market in terms of investments and standards, and the role operators could have in the value chain to struggle against payment fragmentation; they gave a relevant feedback about the original State of the Art document. The "Mobile Game Arch Workshop Minutes" reflects the main discussion of each three sessions as the conclusion of the workshop and gives a picture of the European mobile games market from professionals in the field.

Main outcomes:

- The map of the emerging trends in the mobile ecosystem: new business models, types of games and development methods
- A list of main technological challenges: changing hardware and operating systems, standardisation, latency, fragmentation, HTML5, lack of tools
- A list of main economic challenges: access to markets, access to funding, access to users and access to visibility

8.2 ANALYSING THE STATE OF THE ARTS: SOA DOCUMENT

The State of the Art of the European Mobile Games Industry is an overview of existing and new challenges of the current European market. The latest version is based on the remarks and comments from the leading industry representatives in the European mobile ecosystem. These contributions allowed strengthening the information and analyses of the initial report and making it even more relevant for the reader.
With the aim of participating by its content to the market and technologies defragmentation and to identify the possible obstacles to its development, this document covers the mobile video games market evolution that occurred since 2007 and determine the present challenges in terms of standardisation and technical innovation, as the modifications which took place in the business model.

Main outcomes, the detailed analysis of:

- Development of mobile ecosystem in Europe

- Business related issues in the mobile value chain: market fragmentation, roles in value chain, access to market, generating revenue,
- Technology related issues: anti-fragmentation imitatives, handsets (hardware), software and platforms, development environments and frameworks (tools and middleware), testing and deployment, the network (infrastructure)

8.3 MAPPING THE POSSIBILITIES OF MOBILE CLOUD: PRE WORKSHOP IN AALBORG

The workshop focused on mapping the different perspectives mobile game developers and telcos could co-operate in the area of mobile cloud.

Main outcomes:

- Mobile game developers and telcos see cloud technology from different perspectives and sometimes also understand it differently
- Identification of technological challenges: traffic within the hosted virtual network, access to cloud services via different infrastructures, latency, architectures.

8.4 MAPPING THE HTML5, AND MOBILE CLOUD AND UPDATING THE MARKET ANALYSIS: WORKSHOP IN HELSINKI

With the aim of getting contributions to the Gap Analysis document and updating the State of the Art document, the second workshop was held in Helsinki in co-operation with

the Mobile Monday network and the Aalto Centre of Entrepreneurship on August, 27th 2012. The three sessions of the workshop gathered industry experts to discuss about the possibilities defragmentation using HMTL5, the re-introduction of operators in the value chain via mobile cloud and the funding issues of mobile game development for private investors. The "Summary Report on Second Workshop and Impact Assessment" document summarizes these discussions in coordination with the keynotes and the experts present during the meeting.

Main outcomes:

- An overview of the state of the art of standardisation in the mobile web environment and issues related to it (HTML5)
- A detailed understanding of the role of the telcos in the mobile value chain and technological issues and business opportunities related to the development of mobile cloud
- An investor perspective on the challenges related to access to markets, access to funding

8.5 BRINGING GAME DEVELOPERS AND STANDARDISATION BODIES TOGETHER: THE WEBCONFERENCE

Mobile GameArch's Web conference in November 27, 2012 was intended to discuss standardization requirements from the point of view of different stakeholders: mainly from the game developer point of view, but also from the operator perspective. ETSI and MPEG representatives participated to the workshop by informing about their activity and the status of their standards.

Main outcomes:

- Mapping the perspectives on standardisation: the standardisation process, good and bad sides of HTML5, other ways to address fragmentation
- Mapping the issues related to cloud computing: good and bad sides of mobile cloud solutions, roles and positions in the value chain, technical requirements for cloud infrastructure

8.6 MAPPING THE POSSIBILITIES OF HTML5: HTML5 WORKSHOP IN MALMÖ

Our HTML5 workshop, with the headline "What are the gaps and how do we close them", took place on January 22 between 17.30-21.30. We had 20 attendees from the games industry, academia and the software development industry. They were all skilled HTML5 developers. The invitation went out to a majority of HTML5 developers in our region (Southern Sweden).

The chosen workshop methodology was based on group assignment taken from a priority list of important HTML5 features - to be enhanced or created. Each group was

thoughtfully assembled with 4-5 people with different but complementary backgrounds. The priority list was also created during the workshop.

Main outcomes:

- A list prioritized list of open issues related to HTML5

8.7 ANALYSING THE CLOUD: CLOUD PAPER

Cloud. A word everybody hears about but creates confusion, as it is interpreted differently even by stakeholders. To clarify the understanding of cloud technology, the "Cloud report" gives the requirements from game developers and operators and their impacts on game development. After a review of the architectures available for the Cloud, the report gives the benefits and costs of this technology and the issues related to a decentralised

Main outcomes:

- A list of main open issues for standardisation
- An analysis possibilities of managing mobile applications in a cloud and greater scalability and portioning
- An overview of available architectures as well as pricing, security and privacy issues related to cloud solutions
- An analysis of impact of cloud computing on game development
- Summaries of mobile game developer and operators perspectives on cloud

8.8 ANALYSING THE HTML 5: HTML5 PAPER

From the perspective of the European mobile game developers, the "HTML5 report" aims to clarify the problems and the potential of a HTML5 platform for mobile games. This report identifies commercial issues and technical challenges to be responded and the policy implications of such possibilities from a European position.

Main outcomes:

- An analysis of open commercial issues: Augmented reality, Geo-localisation, User Privacy, Data-Mining, transmedia solutions, OS fragmentation and payment systems
- An analysis of technological challenges: Augmented reality, Geo-localisation, User Privacy, Data-Mining, transmedia solutions, OS fragmentation and payment systems
- Policy implications caused by these issues

9. ANNEX 2: Practical Gaps in HTML5 Implementations for Mobile Games

Rendering Performance (“Seriousness”: 1 on a 1-5 scale)
Problem: There seem to be a general bad implementation of Canvas for every web browser.
At least not a unified standard performance.
"Solution": Set a standard, this applies for more than just Canvas. For example W3C having
a physical computer with attached demands on different things the web browser should be able to do. Or much better benchmarks created by professionals and closely monitored, so output does not result in bad code/micro-optimizations, that browser manufacturers can then use in their process.

Audio (“Seriousness”: 1 on a 1-5 scale)
Problem: The biggest problem is only being able to play one sound on mobile platforms.
This is too much of a constraint when developing mobile games. The current way to play sound via
Html5 is too simple, would be nice to do more complex mixing etc.
"Solution": A better standard that supports more options than just playing, pausing and stopping a sound. Perhaps a full sound API, or a webAL (OpenAL for the web).
Webbrowsers (“Seriousness”: 1 on a 1-5 scale)

Problem: Browsers are too different, they should support the more common extensions.
"Solution": Maybe some of the extensions should be part of the standard instead. Testing community for web
developers where they can submit and get feedback on problems they have encountered.
An easy way to report the differences between browsers. Get browser vendor support/participation.
Better standard.
Loose points not discussed during webbrowsers: Unified input interface(mouse + touch = same),
local storage clear requirements, resolution - scale/aspect, support for Endianness, android fragmentation.
User experience (“Seriousness”: 2 on a 1-5 scale)

Problem: There are a lot of smaller communities which make it harder for popular games reaching more people than it does today.
"Solution": A hope that there will be some sort of community like Steam has but for web-games,
with the possibility to rate and search for top games.
Purchases (“Seriousness”: 2 on a 1-5 scale)

Problem: Now you as a developer have to fix all support for payment, so every developer does this job.
"Solution": API that makes it possible for you as a developer to just call a function for payment
and the browser does rest of the job.

JavaScript Performance ("Seriousness": 2 on a 1-5 scale)

Problem: JavaScript is not made to support high performance because the browsers can't support it anyway.
"Solution": Once again demanding better standards and better implementations with respect to high-performance scenarios.
Low-level ("Seriousness": 2 on a 1-5 scale)

Problem: Too much high level when developing for browsers today.
"Solution": Browser API should give access to more low level functionality, such as vector math (SSE/MMX, etc).
High performance timer ("Seriousness": 2 on a 1-5 scale)

Problem: The timers you can set now aren't precise enough.
"Solution": Make it possible to set timers at a higher resolution.
WebGL ("Seriousness": 2 on a 1-5 scale)

Problem: All browsers do not support WebGL.
"Solution": Standardize implementation of WebGL for all browsers, enforce browser vendors to support it.
Privacy ("Seriousness": 3 on a 1-5 scale)

Problem: panopticlick.eff.org – easy to identify a person given enough data. With full specs of computer (including CPU make/model, memory make/model/amount, etc, it is easy to track someone)
"Solution": Features need to be approved by users of browsers on a app-per-app basis.
Security ("Seriousness": 3 on a 1-5 scale)

Problem: Hard to communicate safe with javascript as of now.
"Solution": Some sort of built in security protocols for communication.
General performance ("Seriousness": 3 on a 1-5 scale)

Problem: No requirements are set for how low performance a browser can have.
"Solution": A lowest requirement made by W3C, also there should be good benchmarks made by professionals that you can measure against.
Bureaucracy ("Seriousness": 3 on a 1-5 scale)

List of things brought up on the workshop:
- Supported and improved by major players
- Game dev. work group
- Lobbying towards manufacturers
- Better way to handle patents
- Less people without know-how who slows the process down, like the EU!?
- Low-level standardization

Problem:
"Solution":
WebCL (("Seriousness": 4 on a 1-5 scale)

Problem: This does not exist.
"Solution": Not really that important, but a nice feature. The far more important issue is enabling parallel development in JS (as it is effectively single-threaded).

General loose points not discussed during the workshop:
- Device access
- Marketing(as in "selling" html5, replacing older tech like flash)
- Browser-specific optimizations
- Documentation
- Hardware information(APD"not sure they wrote APD or something else")
- Less frameworks(combine)
- Knowing how a device will perform

10. ANNEX 3: Major Payment Models Used in the Games Industry

The following payment methods are just simplifications summarizing the most important actors in the value chain. Quite often it is possible that the value chains are even longer than demonstrated elsewhere.

1. Prepaid cards
Used especially in Europe and Africa (e.g. Neosurf and Paysafe) and North-America (e.g. Zeevex, Ultimate Game CArd and Rixty). Payment service providers include PlaySpan, for example.

2. Premium SMS
Used especially in Europe, Asia, Latin America and Africa). Examples of aggregators include Allopass, Playspan, Boku.

3. Payment by phone
Used especially in Europe, this involves calling a premium-toll number for a one-time or per-minute charge.

4. Payment by ISP billing
Used especially in Europe, this adds the cost to your monthly bill from the operator.

5. Payment by cash invoices
Used especially in Latin America, but also in Portugal, examples include Dineromail and Western Union Quick Pay

6. Payment by debit/credit card
Used all around the world, especially in the USA and Europe, examples include VISA, Mastercard, Visa electron, Maestro. It is worth noting that services like MB NET by Multibanco [1]or Paypal have been introduced to secure that no information associated with the consumer's credit card is transferred to a service provider.

7. Payment by direct debit / electronic fund transfer
Used in Europe, examples include SEPA payments

8. Payment by a digital wallet
Used especially in Europe and Latin America, examples include Skrill, Hipay and Dineromail

[1] http://www.multibanco.pt/pt/

11. ANNEX 4: Mobile games ecologies

A. FIRST-PARTY (OS/hardware/distribution platform) ECOSYSTEMS

Apple
OS
iOS 4.x/5.x/6.x
Hardware
Smartphones (Apple iPhone 3GS/4/4S/5, tablet (Apple iPad 2/3/4/mini) and media player (Apple iPod Touch 3/4/5)
Distribution platform
Apple App Store (https://itunes.apple.com)

Blackberry
OS
Blackberry 4.5.x/5-9/Blackberry 10 (BB10)
Hardware
Smartphones (BlackBerry Torch/Style/Bold/Storm/Storm2/Tour/Pearl/Pearl Flip/Curve/8800/Z10/Q10) and tablet (BlackBerry PlayBook)
Distribution platform
BlackBerry World (http://appworld.blackberry.com)

Google
OS
Android Gingerbread (2.3.x)/Honeycomb (3.x)/Ice Cream Sandwich (4.0.x)/Jelly Bean (4.1.x/4.2.x)
Hardware
Smartphone (Google (LG) Nexus 4), tablet (Google (Asus/Samsung) Nexus 7/10 and Google TV
Distribution platform
Google Play (https://play.google.com)

Microsoft
OS
Windows 8 Pro/RT & Windows Phone 7.5/7.8/8
Hardware
Tablet (Windows Surface RT/Pro)
Distribution platform
Windows 8 App Store (www.windows8appstore.com)

NOKIA (previous generation smartphones)
OS
Nokia Symbian OS (to be discontinued)
Hardware
NOKIA Symbian smartphones
Distribution platform
NOKIA Ovi Store (http://store.ovi.com)

B. THIRD-PARTY (licensed OS/hardware/distribution platform) ECOSYSTEMS

Samsung
First-party ecosystems
Samsung version of Google (Android OS), Microsoft (Windows 8)
Hardware
Samsung (Android) smartphones (ex. Galaxy SIII/IV), tablets (Android/Windows 8) and laptop PCs (Windows 8)
Distribution platforms
Samsung (http://apps.samsung.com), Google Play (Android) and Windows 8 App Store

HTC
First-party ecosystems

HTC version of Google (Android OS), Microsoft (Windows 8)
Hardware
HTC smartphones (ex. HTC One and HTC 8S/X)
Distribution platforms
HTC (http://htcapps.com), Google Play (Android) and Windows 8 App Store

NOKIA (current generation smartphones)
First-party ecosystem
NOKIA version of Microsoft (Windows 8), Nokia Symbian OS (to be discontinued)
Hardware
NOKIA smartphones (ex. Nokia Lumina 920/820)
Distribution platforms
NOKIA Ovi Store (http://store.ovi.com) and Windows 8 App Store

LG
First-party ecosystem
LG version of Google (Android OS), Microsoft (Windows 8)
Hardware
LG smartphones (for Android OS and Windows 8)
Distribution platforms
LG Smart World (http://us.lgworld.com), Google Play (Android) and Windows 8 App Store

Amazon
First-party ecosystem
Amazon (Kindle) version of Google (Android OS)
Hardware
Amazon tablet (Amazon Kindle Fire)
Distribution platform
Amazon (http://www.amazon.com) and Google Play (Android)

Barnes & Noble
First-party ecosystem
Barnes & Noble (Nook) version of Google (Android OS)
Hardware
Barnes & Noble tablet (Barnes & Noble Nook)
Distribution platform
Barnes & Noble (http://www.barnesandnoble.com) and Google Play (Android)

Ouya
First-party ecosystem
Ouya version of Google (Android OS)
Hardware
Ouya game console
Distribution platform
Ouya and Google Play (Android) (TBA)

NVIDIA
First-party ecosystems
NVIDIA (Project SHIELD) version of Google (Android OS), Microsoft Windows
Hardware
NVIDIA game console (NVIDIA Project SHIELD)
Distribution platform
NVIDIA, Google Play (Android) and Windows 8 App Store (TBA)

C. THIRD-PARTY (single & cross-platform app channels) ECOSYSTEMS

OpenAppMkt (HTML5 based)
First-party ecosystem
Apple (unofficial)
Distribution platform
OpenAppMkt (http://www.openappmkt.com)

Getjar
First-party ecosystems
Android
Distribution platform
Getjar (http://www.getjar.com)

Intel
First-party ecosystem
Windows 8
Distribution platform
Intel AppUp (available under Windows 8 App Store)

Softonic
First-party ecosystem
Windows 8
Distribution platform
Softonic (available under Windows 8 App Store)

Mobango
First-party ecosystems
BlackBerry (unofficial), Android, Windows Phone, Symbian
Distribution platform
Mobango (http://www.mobango.com/)

Handango
First-party ecosystems
Apple iOS & BlackBerry (unofficial), Android, Windows Phone, Symbian
Distribution platform
Handango (http://handango.com)

Dell
First-party ecosystems
BlackBerry, Android, Windows Phone, Symbian
Distribution platform
Dell Mobile App Store (http://dellmobileappstore.com)

D. THIRD-PARTY (jailbroken' iOS and 'rooted' Android) ECOSYSTEMS

Cydia
First-party ecosystem
Apple iOS ('jailbroken'/unlocked)
Distribution platform
Cydia (http://cydia.saurik.com)

AppsApk
First-party ecosystem
Android (unlocked)
Distribution platform
AppsApk (http://www.appsapk.com)

12. ANNEX 5: Cloud report

1. Introduction

Cloud computing is the buzz word of the year 2012 in the ICT context. Many stakeholders understand it differently. The purpose of this paper is to clarify the understanding of cloud technology, its context and its expectations from the mobile game developer point of view.

The definition of cloud gaming varies. Within the game industry cloud gaming reflects on a cloud based multi device strategy where the content is independent from the device "in the cloud". More generally cloud computing is the use of computing resources that are delivered as a service over a network. There are many types of cloud computing, the most common being Infrastructure as a service (IaaS), Platform as a service (PaaS) and Software as a service (SaaS).

These standard cloud architectures are now well defined. Analysis of the issues to solve as well as the needed standardisation, is well addressed and understood by many projects. As such, the goal of this document is not to develop these topics, but instead, the purpose is to improve the understanding of what are the opportunities and specific issues of cloud related requirements for mobile games.
Mobile games have latency, QoS and bandwidth requirements which cannot be fulfilled by classical cloud. More decentralised cloud allows addressing these issues, and the purpose of this paper is to investigate these new paradigms.

2. Requirements from Game developers

2.1 CLOUD COMPUTING GAME DEVELOPMENT CHALLENGES

Below follow the issues assigned to us among the challenges to game development posed by cloud computing. We have tried to follow the ambition of 1-2 pages for each issue, so as to have a fairly in-depth coverage.

The presentation below is based on our own in-house experience, unless where external sources are explicitly referenced.

2.2 CLOUD COMPUTING; THE REQUIREMENTS OF GAME DEVELOPERS

When discussing the opportunities and challenges for game development as presented by cloud computing, it may be useful to view the Cloud as just, in its simplest implementation, as just any client-server architecture, especially when adding load-balancing and the like, that

allow for fairly seamless scaling of a client-server based application.

What may be of specific interest to game developers, is the value that may be added by the cloud service providers. Supporting caching, device/session transparency, usage analytics, and quality-of-service especially when including connection drop handling and resume, cloud services may provide essential enhancement or crucial cost/efficiency improvements that are unique or very clearly much more important for game applications than for other categories of media provisioning and data processing.

Quality of service is not traditionally in the domain of cloud providers. However, making the choice between the cloud and a traditional distribution model, developers' requirements regarding quality may force them to forego the cloud. If cloud centers could provide basic quality of service guarantees, the cloud itself would be a much more interesting place for developers to place their applications on. Without this, it is simply a cheap storage alternative with virtualization.

Virtualization is another very important topic. By virtualization we mean seamless scaling of servers. A dedicated gaming server for a first person shooter, may spawn several instances of itself across multiple machines, when the load (on a sunday afternoon) increases drastically. However, it is still not practical to spawn servers on the go. Instead developers must do this preemptively. If latencies regarding server seeding would be reduced the use of the cloud would be much more attractive.

The use of networks dematerializes over time as it becomes more and more part of our every day life. Mobile phones develop into computers. In this context it seems to be less relevant what device you have. In fact content business in a dematerialized cloud will become more of a pure content vision accessible with any possible device, the console, the PC, the TV and the mobile phone. From a business perspective this opens new possibilities, but also from a content view, this will be actually the pathway for the long time saught for convergence between linear and interactive. In an decentralized cloud environment the resources of bandwith are used in a more efficient way, leading to better and more likeable content solutions with more graphical debth.

3. Requirements from operators

Mobile telephone operators are putting their emphasis on the networks being one of the main factors for game performance. Gaming clouds will have to address issues of low latency and deployment and access, and the resulting quality-of-service at the edge of the clouds.

Operators are looking at developing standard APIs to deploy games on top of global networks. For example Telefonica [1] suggests that their APIs could be used to deploy games on top of a Global Gaming Cloud. Telco APIs could, in this scenario, help the gamer choose the desired level of network quality given the acceptable price.

Mobile telephone operators see content routing as eventually evolving into ‘content execution’ where it is possible to inject code to be executed in coordinated nodes. It would

thus be possible to design games that execute at the edge, store state in an inner cloud and dynamically choose links, execution nodes and storage based on network state. This could help ensuring bounded latency, granted functional scaling in both dataset and user base.

The changes needed may be interpreted from the following illustrations. As an example, the current France Telecom network [2] in Brittany (Bretagne) is tied to a single center (PoP) in Rennes, France (as seen in Figure 5 below). Developed cloud services, solving latency and other game-critical issues would require an evolved structure, with local processing and data centers (as illustrated in Figure 5 below).

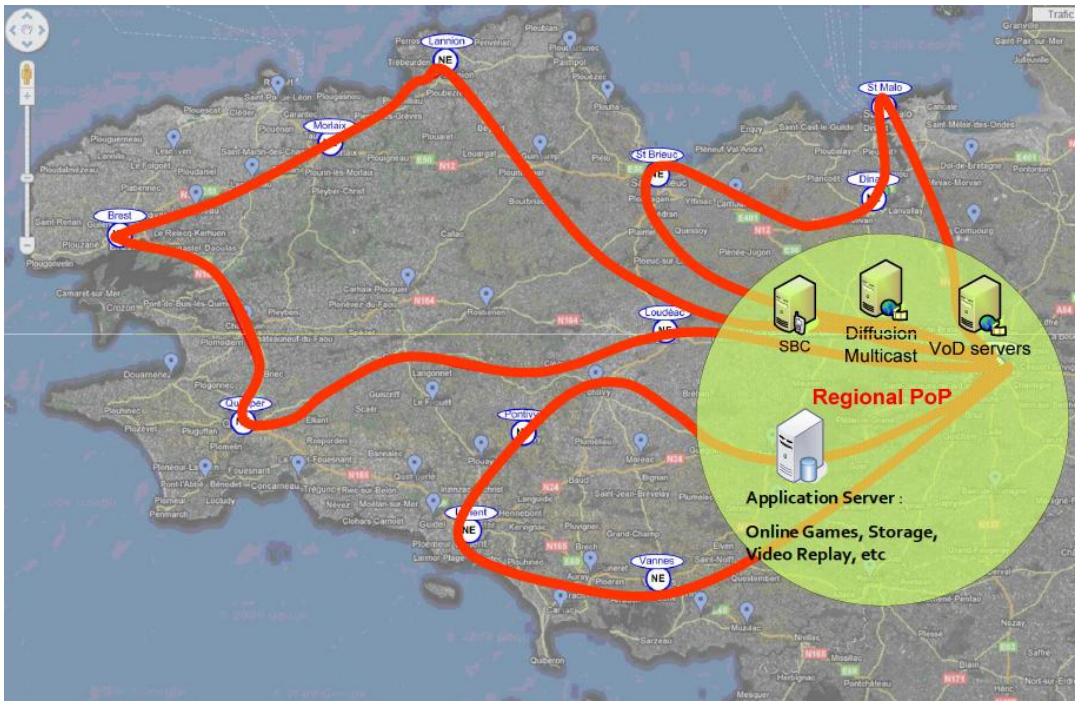


Figure 4 Current France Telecom network architecture

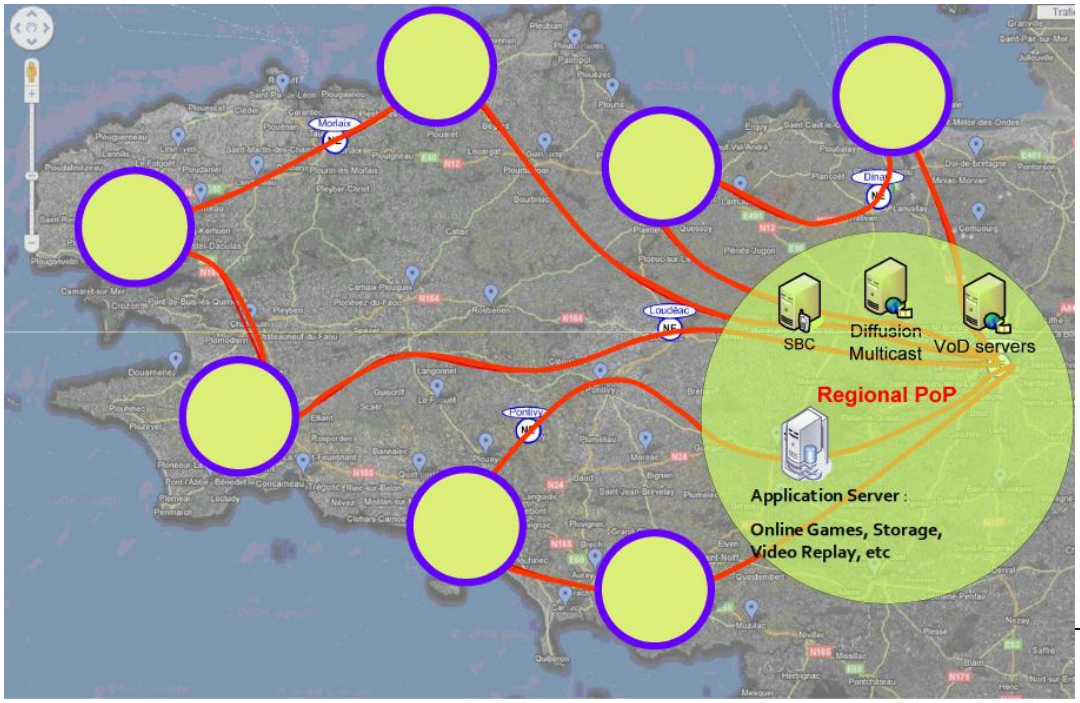


Figure 5: a first example of cloud decentralization

[1] http://www.mobilegamearch.eu/wp-content/uploads/2012/05/Telefonica-presentation.pdf

[2] http://www.mobilegamearch.eu/wp-content/uploads/2012/05/Orange-presentation.pdf

4. Impact of Cloud on game development

The upcoming of cloud technology is reversing a general trend of developing more and more technology in the individual sphere. The home computer or smart phones are just two examples of trends which have been going into this direction. It seems as if the human character tends to wish to "own" the data carriers and administration machines and to have it at his physical disposal and control. We know this trend from movies and games, which are often owned stored on DVD's as humans wanted to own them physically. However with the increasing omnipresence of the internet in our society, the natural resistances to lose physical control over the data and its processing seem to fade away. Two main obstacles can be identified – functionality (what do I do, when it does not work?) and security (who is using my data?). These obstacles are mainly psychological from an end-user point of view.

Online games are – at least in part – run on a remote server. Mobile games increasingly require a steady connection to a central server. If we try to define "cloud" from a game developer point of view, we therefore need to assure, that cloud means the same thing here than in the general ICT world. From our practical experience this is not necessarily the case. To a certain degree every browser based online game is cloud technology. Game developers tend to differentiate online games in client based (where you have a piece of software in your individual PC – the client) and browser based (where you do not need any software any more in your computer). In the case of browser based games it seems obvious, that they are basically always fulfilling the requirements of cloud based technology in general ICT. There are many browser games existing and some of them use high data streams (e.g. Battle Star Galactica from Norway published by Bigpoint). This development is now expanding to the mobile sector.

We therefore have to realise the fact that cloud technologies are already widely applied in the game industry. They are also appreciated by the end user. When the gameplay is entirely on the cloud (in the case of browser games) the European end user is more willing to accept it, than, if there is a mixture (client based game). Indeed security considerations do play an important role from the gamer's perception in this context. Considerations of functionality do not play such an important role any more. But we have to see that once again game developers and gamers are at the forefront of innovation.

This is specifically true for the monetization. Browser based games and increasingly mobile games are nowadays developed in the freemium model. For the end user the games are initially free to play. However within the game virtual items and gameplay advantages can be purchased with virtual currency. This virtual currency needs to be bought with real money. Game developers talk about the convergence of not paying users to paying users. This

convergence can be measured. The higher the convergence rate is, the more profitable is the game. For the payment, game companies use as many different payment channels as possible.

Consequently, the impact of the cloud is not limited only to the games themselves. The payment channels used in the mobile game industry form a complex value chain, where cloud services will be increasingly used to provide different kinds of payment, metrics and aggregator services. The various information and money transfers are demonstrated in the picture below. The different payment models used are summarized in Annex I.

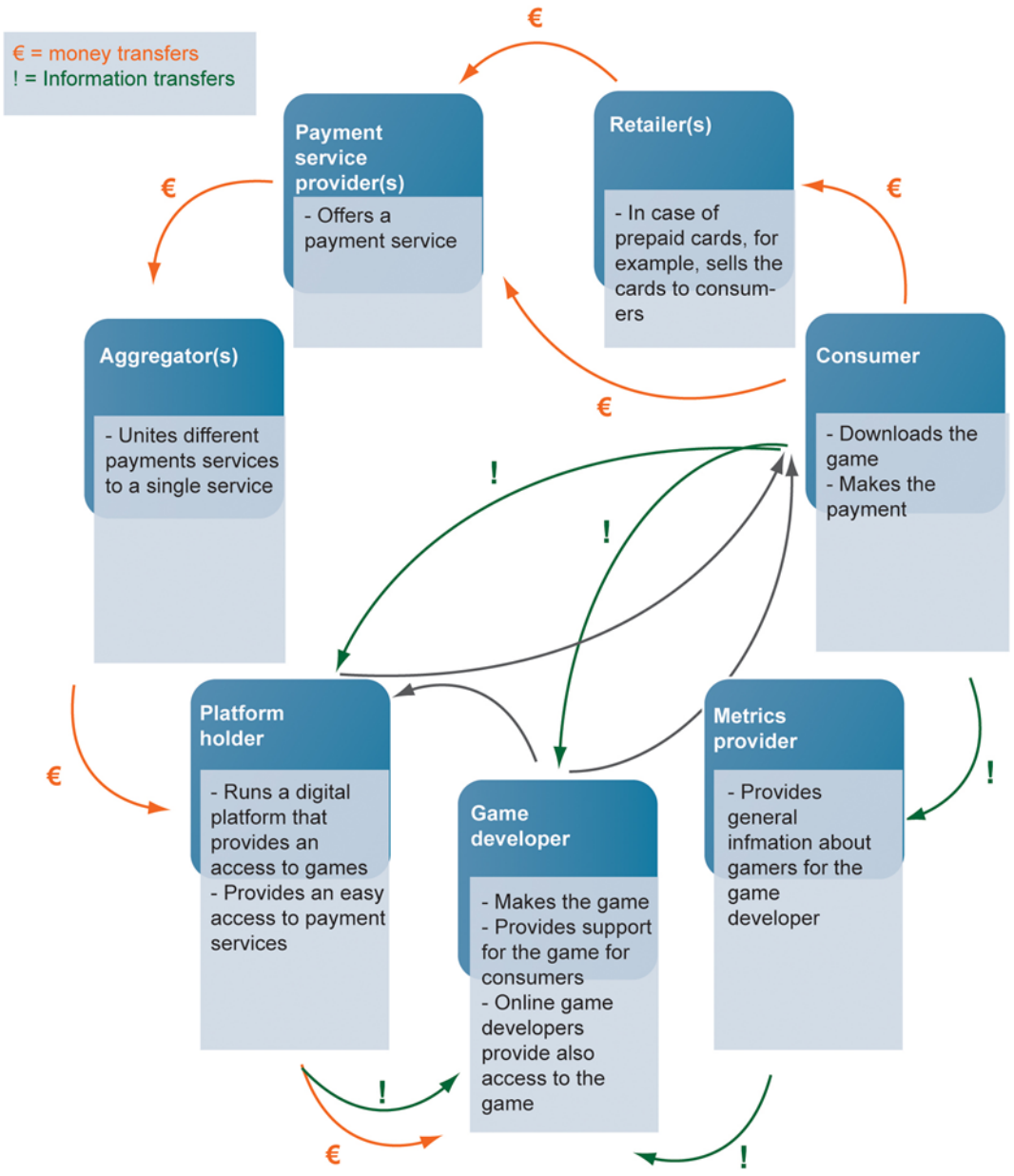


Figure 6: transaction models for game

It is important to note, that in the experience of the game developers, the multitude of different payment methods has increased the turnover and eventually revenue of games. The simple rule is the more payment solution a game has the more revenue it can generate. It is therefore an error to try to standardize payment solutions (as it is sometimes suggested by the ICT research community). Due to heavy advertising the player price however has risen

considerably in the last years in Europe. Therefore advertisement driven business models become more important again (like free TV in gaming).

The massive success of virtual item based gaming on the browser is showing well the potential of cloud based gaming in Europe. The link is specifically interesting in the context of piracy. An online game, which runs on a server, cannot be pirated in the same way a stand alone game on an individual computer can be pirated. The piracy proof business model of online gaming is therefore probably the biggest advantage. However it needs to be said, that pirates also innovate (unfortunately) and that fake keys and identities as well as whole servers are pirated too.

	Bigpoint (DE)	**Gameforge (DE)**	**Ankama (FR)**	**Jagex (GB)**	**Sulake (FI)**
Founded	2001	2003	2001	2001	2000
Employees	about 700	about 450	about 460	about 375	about 270
Turnover €	140 million (2010)	Over 100 million (2009)	40 million (2010)	40 million (2010)	56,2 million (2010)
Registered users	194 million	200 million	30 million	170 million	210 million

Source: *ICO Partners 2011*

In the mobile gaming sector this is just happening now. Especially in Asia, where the piracy problem is even more imminent mobile games require increasingly business models on a micro transaction base. They are – besides advertisement driven models – the most important revenue stream in China.

Beyond mobile and PC-online the next upcoming console generation will also embrace the online user. Today, the new Xbox-360 has no DVD drive any more. Massive multiplayer solutions will be embraced. Korean MMOG like Lineage or US based worlds like WOW will be accessible through the mobile phone, the PC as well as the new consoles. The user will be able to steer the same avatar with several devices. Whether the new business models will prevail or whether a new quality driven paid content level will emerge is still undecided.

The question of energy consumption is rarely discussed among game developers. However in the mobile space, battery is a valuable factor and well retained by the end user. It must be said that the main motivations for cloud gaming do not lie in energy saving, but in piracy and multi-device strategy, especially also in cooperation with the film industry.

5. Architectures

A number if issues are still to be solved in classical cloud architectures, to ensure interoperability between cloud providers, and portability between applications:

- Common models for Virtual Machines.
- Common models for networks.
- Common models for storage.
- Other issues like common architecture for monitoring, authentication, security, authorisation, and accounting.

These apply to current centralised cloud architectures, where QoS and latency consideration are not considered as critical and can be dealt with by providing sufficient network bandwidth. The purpose of this document is not to develop these issues as they are identified as generic issues for cloud interoperability.

Mobile applications and games in particular bring additional requirements to the cloud:

- Latency: latencies in the ranges of 10 ms can be required by some games.
- Throughput: game download requirement can be heavy, as well as information sent back to the network.
- Interaction is also an important requirement in mobile games.
- Openness: the user in mobility requires seamless transition between different terminals and therefore full interoperability of the cloud to enable this functionality.

In addition to these criteria, the cost of bandwidth becomes an important parameter: indeed, whereas the cost of bandwidth for fixed network will be reduced considerably with FTTX access, mobile network bandwidth cost will still remain significant as far as games requiring complex graphics are concerned (AAA games?).

All these considerations make new decentralised cloud architectures a solution matching well the needs of mobile gaming applications (references [3], [4], [5], [6]).

Two levels of decentralisation can be proposed (see Figure 7):

- An extension of the concept of CDN to the concept of "CDN cloud": in CDN, content is distributed intelligently to servers located nearby the users, in order to respect the QOE. CDN are usually managed by operators. In the concept of "CDN cloud" both content and applications (or part of applications) are distributed, using the same concept as CDN. In extreme cases where low latency has to be provided, CDN can be extended to Edge servers like depicted in Figure 8. However these edge servers don't allow to reduce the 3G/4G wireless network bandwidth consumption.
- A new model (see Figure 9) can be proposed to cope with optimisation of the local wireless network bandwidth. In that case content and applications can be stored also on the local network or home network devices. These devices can be game consoles, PCs or mobile tablets or phones in an extreme case. Reference [26] for instance proposes a framework based on the concept of cloudlet. The principle of this architecture is to run part of the application on the mobile device, and offload

applications components to the cloud according to their latency, CPU and bandwidth constraints: a component having low latency constraints will run in the Local network, close to the terminal; whereas a component with high CPU constraint will run in a more distant cloud.

Figure 7: cloud architecture

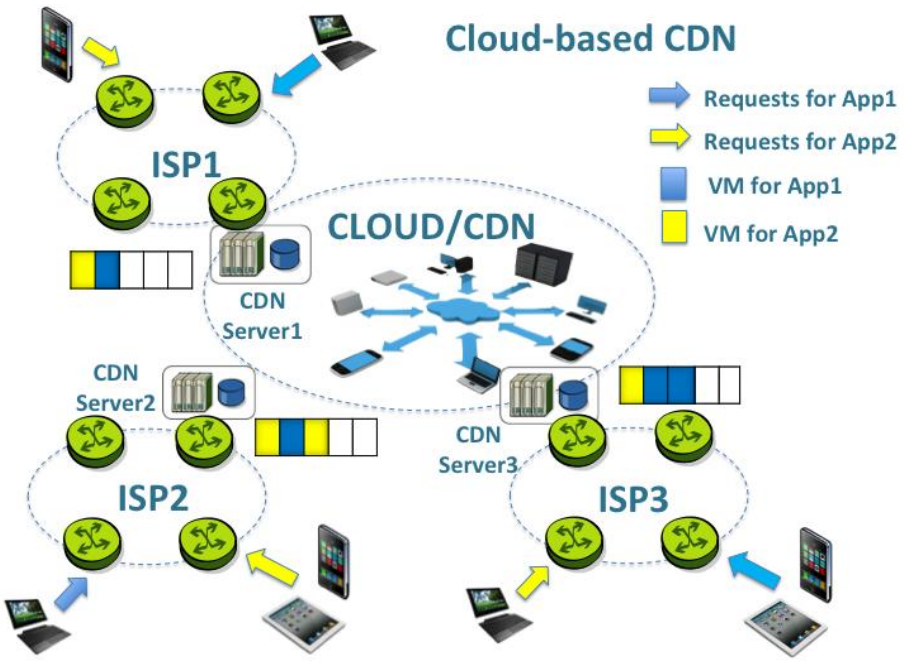


Figure 8: "CDN cloud"

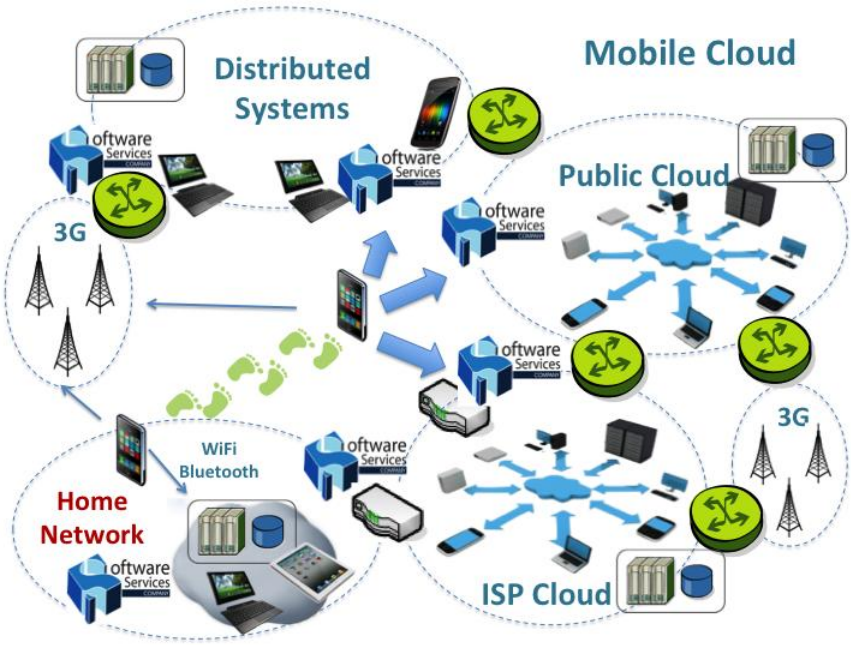


Figure 9: decentralized cloud

[3] Eduardo Cuervo†, Aruna Balasubramanian‡, Dae-ki Cho∗, Alec Wolman§, Stefan Saroiu§, Ranveer Chandra, Paramvir Bahl : « MAUI: Making Smartphones Last Longer with Code Offload » ; Mobysis 2010

[4] Hai Anh TRAN, Abdelhamid MELLOUK, Said HOCEINI QoE « Content Distribution Network for Cloud Architecture » ; 2011 First International Symposium on Network Cloud Computing and Applications

[5] Chia-Feng Lin, Muh-Chyi Leu, Chih-Wei Chang, Shyan-Ming Yuan »The Study and Methods for Cloud based CDN » ; 2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery

[6] Morten V. Pederse, and Frank H. P. Fitzek ; « Mobile Clouds: The New Content Distribution Platform » ; Proceedings of the IEEE | Vol. 100, May 13th, 2012

6. Pricing of cloud gaming

6.1 BENEFITS OF CLOUD GAMING

Cloud gaming is offering a cost saving alternative for the traditional games distribution model.

The impact cloud gaming potentially has for the triple A Console gaming market is extensive, as it could potentially replace the costs for dedicated hardware (consoles), replace physical distribution in stores, packaging, shipping and, last but not least, wipe out the huge license

fees for Microsoft (Xbox), Nintendo (3DS, Wii) and Sony (Playstation, PST): end users no longer need a console, they can use their PC's or tablets.

For mobile games, cloud gaming can potentially reduce the costs for developers and publishers related to fragmentation of Operating Systems and handsets and allow cross-platform gaming.

Next to direct cost reduction and extended market reach, other benefits for game developers and publishers can be identified:

- Immediate access to sales results
- Immediate access to analytics and to implement changes accordingly
- Better cost management, lower risk

6.2 COSTS OF CLOUD GAMING

Some of the larger publishers have their own cloud gaming service. For example, Sony recently acquired the cloud gaming service Gakai for 380 million. Gakai was offering white label cloud gaming services to many publishers worldwide.

OnLive, which only one year ago was valued at 1 billion US$, has recently applied the Californian bankruptcy law (the ABC law), which meant that all original investors lost their investments and a new company could acquire the assets and continue the business. With the sales price the company could pay off its creditors.
Onlive has 2,5 million subscribers and 1,5 million regular users.

The price of cloud services can be determined easily by connecting to the Amazon service "Simple Monthly Calculator" of the Amazon Web Services site.

However, the costs of turnkey cloud gaming solutions, such as Onlive are far less standardized. Some services offer a revenue share, others a mix of revenue share and start up costs. All turnkey solutions are looking for exclusive, high profile games, which enable their user base to grow.

But the costs of cloud gaming are not limited to the price paid for external services. A game publisher should also bear the costs of adapting the games for the service, which includes integrating SDK's and integrating monitoring services or other additional technologies in the workflow.

Additional costs may be expected when they want to change to another provider or when the service provider ceases to exist.
More additional costs may become necessary if the existing game on the cloud evolves.

In conclusion, we may state that every studio needs to make an estimation of the return on investment and the risks related to the move towards cloud gaming. The technology and the business are still young, so we can expect many unknown threats and opportunities.

6.3 SECURITY AND PRIVACY ISSUES

Just as any data processing and transfer application may fall under the European Data Directive as it is applied to the handling of personal information, so do games. This is not unique as it applies to many sectors. When contracting and implementing cloud services a further layer of complexity arises.

Where, in our case, a European game developer may operate a server in a co-location environment, the responsibility for negligence rests quite obviously with the developer, and is governed by national law, which should reflect and implement the European Data Directive. Common sense may also provide some assistance for the developer in avoiding mistakes in a field that they may not be very familiar with: law and specifically as applied to responsibility for security and integrity of customers' personal data. However, when utilizing cloud services, the game developer has little influence over the actual geographical location of storage and handling of data. Culture, tradition, and law in this particular field is for example quite different in the USA, home to many cloud service providers. Any mis-handling of EU citizen data that takes place in the USA on behalf of a EU company is the responsibility of the EU company. A lot of routines, quality and security measures and explicit legal agreements between several layers of parties is necessary to conform to EU legal requirements while storing and processing data off-shore, as in the USA. Agreements are in place between the USA and EU to provide a ”safe haven”, and it is far from impossible to check whether a US cloud provider fulfills EU standards before using their services.

The uniqueness in this field lies not in particular security and privacy issues that apply only to games. Rather, we would, if somewhat speculatively, claim that it is the level of knowledge and experience as it comes to privacy legal issues that is uniquely low among game developers in relation to the technical advancement of their applications. Cloud services provide many important opportunities for advancement of European game developers, but also a new level of legal, and thus economical, risk exposure that a worrying proportion of the developers may be unaware of.

An imagined technology, hypothetically enabling the game developers to monitor in real-time and assure the compliance of external, European and non-EU service providers, and possibly also exposing this status to the end-users for their own privacy and security monitoring, could be a big help for the small, independent European developer. It would not only minimize their legal exposure and risks, it could actually turn into a competitive advantage.

7. Technology issues related to a decentralised cloud

The decentralized approach for “mobile game cloud” add some technology issues to classical clouds, like particular constraints on application partitioning, management of application components in the decentralized cloud, surrogate discovery, security and privacy, which are summarized in the following paragraphs.

7.1 APPLICATION PARTITIONING AND SCALABILITY

One of the principal advantages of off-loading application components in mobile gaming is to optimize the usage of available resources, such as power consumption and bandwidth [7]. In this case, the partitioning of an application is driven by optimally using available resources.

There are two popular approaches for application partitioning:

The first requires the developer of the application to define the components of the application that can be considered for off-loading by annotating the source code. The energy reduction achieved using this approach is often considerable since the partitioning can be fine-grained and is optimized for the particular application and use-case [8]. However, the disadvantage is that it requires modifications to the source code, which is not always practically feasible and it complicates maintenance. Moreover, since the partitioning is manually defined, it may be suboptimal [9].

The second approach uses virtualization of hardware resources, which are dynamically augmented when the need arises [10]. This approach does not have the drawback associated with manually portioning the application, but the application partitioning is realized on the level of individual processes, which is often not sufficiently fine grained for optimal resource conservation. The MAUI (memory arithmetic unit and interface) system attempts to take the beneficial features of both approaches by allowing for fine-grained energy-aware offloading while minimizing the changes necessary to the application [11]. In contrast to partitioning the application manually during development (approach 1) MAUI partitions the application at a runtime based on the costs of network communication and CPU on the mobile device to maximize energy savings given network connectivity. The experimental results show that MAUI saves 27% of energy usage for a latency sensitive arcade game and 45% for a turn-based chess game. Besides energy conservation, MAUI also improves the performance of the arcade game (i.e., the game's refresh rate increases from 6 to 13 frames per second). Similarly, cloud-based mobile games use and adaptive rendering technique to dynamically adjust the game rendering parameters according to communication constraints and gamers' demands. The adaptation technique reduces the complexity of a scene by removing decorative objects that are not necessary for playing the game in order to improve interactivity and increase user experience given the communications and computing costs [13].

Besides optimizing resource usage, off-loading application components can also be used as an enabler for mobile applications that require extensive computational resources far superior than can be achieved by the native hardware of the mobile device. In this case the application partitioning is driven by resource demands. An example of such application is the "Wolfenstein: Ray traced project" from Intel Corporation, which off-loads the highly complex and computationally intensive ray-tracing to a cloud infrastructure. The result is a high definition photo realistic 3D arcade game that can be run on a mobile device [12]. This type of approach is also known as a thin-client approach, which attempts to minimize processing on the client device by off–loading nearly all computational tasks to a server. Onlive [14], Ubitus GameCloud [16] and Gaikai [15] are commercial offerings of a thin-client based approach to play sophisticated games on a mobile device. Although thin-client approaches reduce CPU usage on the client device, the costs of the network communication are considerable, both from an energy and bandwidth consumption perspective. As a result, resource usage is often suboptimal [17].

[7] Z. Li, C. Wang, and R. Xu, "Computation offloading to save energy on handheld devices: a partition scheme," in Proceedings of the 2001 international conference on Compilers, architecture, and synthesis for embedded systems (CASES), pp. 238 - 246, November 2001

[8] R. Balan, J. Flinn, M. Satyanarayanan, S. Sinnamohideen, and H.-I. Yang. The Case for Cyber Foraging. In 10th ACM SIGOPS European Workshop, Saint-Emilion, France, September 2002

[9] F. Douglis and J. Ousterhout. Transparent Process Migration: Design Alternatives and the Sprite Implementation. Software - Practice and Experience, 21(8):757–785, August 1991.

[10] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The Case for VM-based Cloudlets in Mobile Computing. IEEE Pervasive Computing, 8(4), 2009.

[11] E. Cuervo, A. Balasubramanian, Dae-ki Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: Making Smartphones Last Longer with Code offload," in Proceedings of the 8th International Conference on Mobile systems, applications, and services, pp. 49-62, June 2010.

[12] Daniel Pohl, "Tracing Rays Through the Cloud", http://software.intel.com/file/41894, March 1, 2012

[13] S. Wang and S. Dey, "Rendering Adaptation to Address Communication and Computation Constraints in Cloud Mobile Gaming," in IEEE Global Telecommunications Conference (GLOBECOM), pp. 1-6, January 2011.

[14] http://www.onlive.com/about

[15] http://www.gaikai.com/

[16] http://www.ubitus.net/release/2012_release_20120718_english.html

[17] A. Rudenko, P. Reiher, G. J. Popek, and G. H. Kuenning, "Saving portable computer battery power through remote process execution," Journal of ACM SIGMOBILE on Mobile Computing and Communications Review, vol. 2, no. 1, January 1998.

[18] K. Kumar and Y. Lu,"Cloud Computing for Mobile Users: Can Offloading Computation Save Energy," IEEE Computer Society, vol. 43, no. 4, April 2010.

7.2 MANAGEMENT OF OFF LOADING APP. COMPONENTS

7.2.1 Management of app components in the cloudlet

Mobile cloud users need access to servers in the cloud to use cloud services and cloud resources. However, mobile users usually suffer from insufficient network bandwidth, network disconnection and signal attenuation due to mobility feature. A computation offloading technique is proposed to migrate the large computations and complex processing from resource-limited devices (i.e., mobile devices) to resourceful machines (i.e., servers in clouds). Offloading the multimedia code can save energy for mobile devices and improve the performance of applications. For example, using memory arithmetic unit and interface (MAUI) to migrate mobile game components [19] to cloud can save 27% of energy for video

games and 45% for the chess game. However, experiments show that offloading is not always the effective way to save energy. For example, for a compilation of a small size of source codes (such as 500KB), offloading might consume more energy than that of local processing on mobile devices. Moreover, offloading to the cloud is not always a good solution, since it brings high latency which challenges applications with real-time constrains. Hence, it requires a sophisticated design whether to offload and which portions of applications' codes need to be offloaded.

***The State of the Art*:**
Many offloading solutions have been designed based on communication cost (the size of transmitted data and the network bandwidth) and computation cost (computation time). CloneCloud first introduced offloading execution from the mobile device to cloud. The idea is to clone the entire set of data and applications from the smartphone into the cloud, and selectively execute some of operations on the clones, and reintegrate the results back to the smartphone. The clone actually is a mirror image of a smartphone running on a virtual machine. However, CloneCloud is limited in handover delay and bandwidth, since the end user has to communicate with remote cloud. To resolve these issues, the authors in [27] propose an elastic application programming model by extending the mobile terminals to cloud through a distributed framework. A single application is divided into weblets, a range of elasticity patterns, and dynamically configured to run on internet-based cloud and mobile devices.

In mobile gaming [19] [20], MAUI [19] enables fine-grained energy-aware offloading of mobile codes to a cloud. MAUI partitions the application codes at a run time based on the costs of network communication and CPU on the mobile device. A rendering adaptation technique is proposed [20] to reduce the number of objects created by the game engine in the playing list.

A Cloudlet is a trusted and resource-rich computer (or a cluster of computers) that shares its resources with nearby mobile devices. Normally, a mobile user uses a nearby local Cloudlet as a proxy to access to the remote cloud center. When a mobile user does not want to offload to the remote cloud (probably due to delay, cost or low bandwidth), he could offload to a nearby Cloudlet. The objective is to achieve a low latency, one hop transmission and high bandwidth. For example, a typical home network is deployed by high bandwidth wireless connectivity and available computing resources on PCs, game consoles, TVs or entertainment units.

These available devices could be configured by mobile users as a Cloudlet. The authors in [24] propose VM-based Cloudlets architecture where mobile users can instantiate customized service software on a nearby cloudlet and then use this service via a wireless LAN. The cloudlet infrastructure possesses base VMs. A mobile device delivers a VM overlay to cloudlet infrastructure, and then the infrastructure creates a launch VM for the user from via applying the VM overlay to the base VM.

The paper [26] claims that the VM-based cloudlet approach still has some issues. First, applications based on VM-based cloudlet are dependent on SPs to deploy a cloudlet infrastructure in a LAN. Instead of offloading the complete VM from the cloud to the cloudlet, the paper [26] proposes a component-based offloading to allow all available devices in a LAN to cooperate in the cloudlet. The applications in this cloudlet architecture are managed on component level such that the application components can be distributed among the

cloudlets. There are two kinds of cloudlets: the ad-hoc cloudlet including dynamically discovered nodes in the LAN, and the elastic cloudlet running on a virtualized infrastructure.
Game players usually appreciate a larger screen; however, this could not be satisfied by a smartphone in mobile game. Y. Liu [28] proposes a solution called Virtualized Screen, to move screen rendering from the smartphone to the cloud. A thin-client, remote-computing system with virtualized screen has been developed to leverage interactive screen-remoting technologies. This scalable screen virtualization solution allows users to render screen partially in the cloud and partially at the clients, depending on several facts such as local processing power, bandwidth and delay, data dependency and data traffic, and display resolution,

Issues and Trends:
The offloading partitions applications from mobile devices and delivers some of them to run in the cloud, which helps to improve the application performance on the resource constrained mobile devices. However, there is still much work to do on the optimal strategy or algorithm on partitioning and offloading. Moreover, offloading also cannot directly transplant the services from PCs in the cloud to mobile devices because of the different features between PCs and smartphones. This requires the method on how to provide suitable and friendly interactive services for smartphone.

References:

[19] E. Cuervo, A. Balasubramanian, Dae-ki Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: Making Smartphones Last Longer with Code offload," in Proceedings of the 8th International Conference on Mobile systems, applications, and services, pp. 49-62, June 2010.

[20] S. Wang and S. Dey, "Rendering Adaptation to Address Communication and Computation Constraints in Cloud Mobile Gaming," in IEEE Global Telecommunications Conference (GLOBECOM), pp. 1-6, January 2011.

[21] K. Kumar and Y. Lu,"Cloud Computing for Mobile Users: Can Offloading Computation Save Energy," IEEE Computer Society, vol. 43, no. 4, April 2010.

[22] S. Ou, K. Yang, A. Liotta, and L. Hu. "Performance Analysis of Offloading Systems in Mobile Wireless Environments," in Proceedings of the IEEE International Conference on Communications (ICC), pp. 1821, August 2007.

[23] B-G. Chun and P. Maniatis, "Dynamically partitioning applications between weak devices and clouds," in Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond (MCS), no. 7, June 2010.

[24] Mahadev Satyanarayanan, Paramvir Bahl, Ramon Caceres, and Nigel Davies, "The Case for VM-based Cloudlets in Mobile Computing", IEEE Pervasive Computing, 2009.

[25] Roelof Kemp, Nicholas Palmer, Thilo Kielmann, and Henri Bal, "The Smartphone and the Cloud: Power to the User". MobiCloud 2010, Santa Clara, CA, USA, October 2010.

[26] Tim Verbelen, Pieter Simoens, Filip De Turck, and Bart Dhoedt , "Cloudlets: bringing the cloud to the mobile user", the 3rd ACM workshop on Mobile cloud computing and services, pages 29-36, NY, USA, 2012.

[27] Xinwen Zhang, Sangoh Jeong, Anugeetha Kunjithapatham and Simon Gibbs, "Towards an Elastic Application Model for Augmenting the Computing Capabilities of Mobile Devices with Cloud Computing", Mobile networks and applications, vol. 16, no. 3. 2011.

[28] Yan Lu, Shipeng Li and Huifeng Shen, "Virtualized Screen: A Third Element for Cloud—Mobile Convergence", IEEE Multimedia, Vol. 18, No. 2, pp. 4-11, April 2011.

7.2.2 Surrogate discovery

To augment the capabilities of smartphones, one approach is to duplicate the runtime environment of smartphone in the cloud and run the application either on the smartphone or in the cloud. This off-device runtime environment has been called a "surrogate" in Cyber Foraging, a "clone" in CloneCloud and a "cloudlet".

In a Cyber Foraging approach, the first step is to discover available surrogates [29] [30]. Considering service information dissemination, there are three basic architectures for service discovery: directory-based architecture, directory-less architecture and hybrid architecture. In a directory-based architecture, a mobile node could be a server, a client or a service directory. Service providers register services to the service directories and service consumers are informed about available services through these directory nodes. A directory-less architecture is much simpler from directory-based architecture, since there is no service directory between service providers and consumers. The service providers broadcast available services and service consumers broadcast their service requests. In a hybrid architecture, service providers register available services, either to service directories if they locate any in their vicinity or via broadcast by themselves. Service consumers send their demands, either to the service directories if they could find or via broadcast. The evaluation of the effectiveness for these three architectures includes service availability, messaging overhead and latency.

In contract to application layer based service discovery, cross layer based techniques have been studied such as routing layer based service discovery. The idea [31] is from that the updating routing messages can help to provide service discovery at the same time. Another benefit from cross layer based approaches is to allow routing information to restore service sessions, or make handovers from different providers [32].

A model at link layer has been proposed in [33], which integrates service information from an application layer service discovery protocol, as well as routing information from a network layer protocol. In this way, there is no need to extend or modify routing protocols.

There are some open issues in surrogate discovery for further investigation and research. First, the service discovery protocols require adapting context-awareness mechanisms. Second, interoperability of service discovery is also a major issue that requires a ubiquitous and pervasive nature. Another major problem is to develop a universal evaluation framework to allow fair comparisons among different protocols.

References:

[29] C. N. Ververidis and G. C. Polyzos, "Service discovery for mobile ad hoc networks: A survey of issues and techniques," IEEE Commun. Surveys Tutorials, vol. 10, no. 3, pp. 30–45, 2008.

[30] M. D. Kristensen, "Scavenger - mobile remote execution," University of Aarhus, Technical Report DAIMI PB-587, 2008.

[31] R. Koodli and C. E. Perkins, "Service discovery in on-demand ad hoc networks," IETF Internet Draft, draft-koodli-manet-servicediscovery- 00.txt, October 2002.

[32] D. Chakraborty, A. Joshi and Y. Yesha, "Integrating service discovery with routing and session management for ad hoc networks," Ad Hoc Networks, Volume 4, Issue 2, March 2006, pp. 204-224.

[33] V. Atanasovski and L. Gavrilovska, "Efficient Service Discovery Schemes in Wireless Ad Hoc Networks Implementing Cross-Layer System Design," Proc. 27th International Conference on Information Technology Interfaces (ITI 2005), Cavtat, Croatia, June 20-23, 2005.

7.3 SECURITY RISKS

Besides Steve Wozniak's assessment on cloud security (http://www.theregister.co.uk/2012/08/06/wozniak_cloud_will_be_horrendous/ published 2012-08-06, accessed on 2012-09-30), the Cloud Security Alliance has published a study identifying user-percieved risks with cloud-computing, some of them which can be transferred to game developers as well. The study reveals that cloud users in 50 countries were least confident about the following issues (ranked from least confident to most confident):

- Government regulations keeping pace with the market (1.80)
- Exit strategies (1.88)
- International data privacy (1.90)
- Legal issues (2.15)
- Contract lock in (2.18)
- Data ownership and custodian responsibilities (2.18)
- Longevity of suppliers (2.20)
- Integration of cloud with internal systems (2.23)
- Credibility of suppliers (2.30)
- Testing and assurance (2.30)

(https://cloudsecurityalliance.org/csa-news/cloud-maturity-study-reveals-top-issues/ published on 2012-09-27, accessed on 2012-09-30).

Furthermore, the Cloud Security alliance has earlier identified seven top threats to cloud computing. In our view, the following three are the most immediately relevant for game developers. (Main source for the following section: "Top Threats to Cloud Computing V1.0. Prepared by the Cloud Security Alliance, March 2010").

7.3.1 Insecure Interfaces and APIs

Beyond control of the developer are security issues such as anonymous access and/or reusable tokens or passwords, clear-text authentication or transmission of content, inflexible access controls or improper authorizations, limited monitoring and logging capabilities, and unknown service or API dependencies.

Remediating measures include, but are not limited to, analyzing the security model of cloud provider interfaces. Ensuring strong authentication and access controls are implemented in concert with encrypted transmission. And, as a basis, understanding the dependency chain associated with the API.

7.3.2 Data Loss or Leakage

Data loss or unauthorized access may result from insufficient authentication, authorization, and audit (AAA) controls; inconsistent use of encryption and software keys; operational failures;
persistence and remanence challenges: disposal challenges; risk of association; jurisdiction and political issues; data center reliability; and disaster recovery.

Remediating measures include, but are not limited to, implementing strong API access control, encrypting and protecting integrity of data in transit, analyzing data protection at both design and run time, implementing strong key generation, storage and management, and destruction practices.
Furthermore, contractual demands on providers to wipe persistent media before it is released into the pool, and contractually specifying provider backup and retention strategies are important.

7.3.3 Account or Service Hijacking

Unauthorized access is a classical problem, with both technological and social connotations. However, shared functions and linkages between services based in the cloud, account access security and API implementations may mean that the breach of a single user account results in total identity theft or data destruction over a wide variety of providers and services. (Source for the above paragraph: own.)

Remediating measures include, but are not limited to, prohibiting the sharing of account credentials between users and services, leveraging strong two-factor authentication techniques where possible, employing proactive monitoring to detect unauthorized activity, and should be based on a thorough understanding of cloud provider security policies and SLAs.
http://vmetc.com/2009/03/12/virtual-machine-sniffer-on-esxhosts

8. Open issues for standardisation

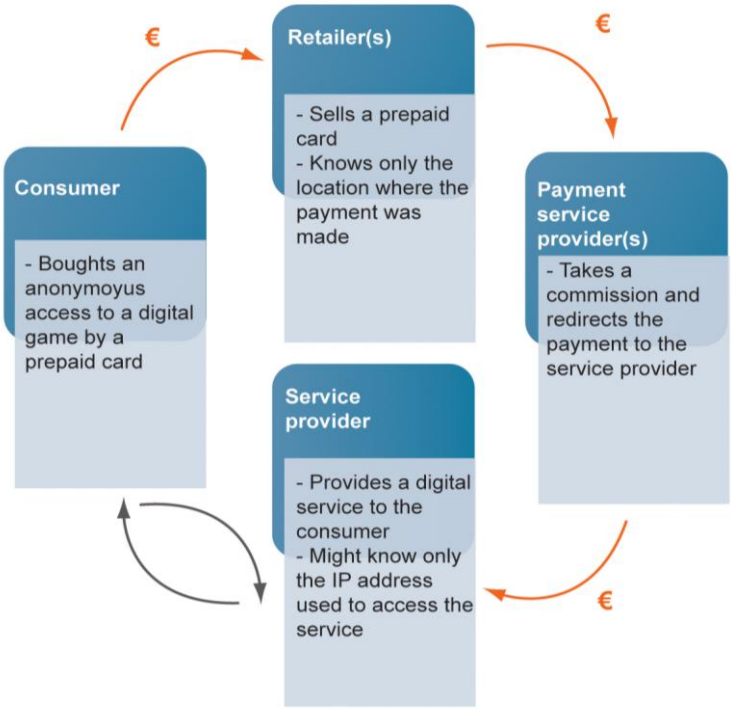
Besides usual issues related to cloud, this paper highlights the interest of cloud for mobile gaming, and more particularly the interest of decentralised cloud to fulfil the bandwidth and latency constraints of mobile gaming. Moreover criteria like CPU resource availability, bandwidth cost and consumption have to be involved in the available parameters, which will allow to take the right offloading decisions.

The main issue which have been identified for standardisation can be listed as:

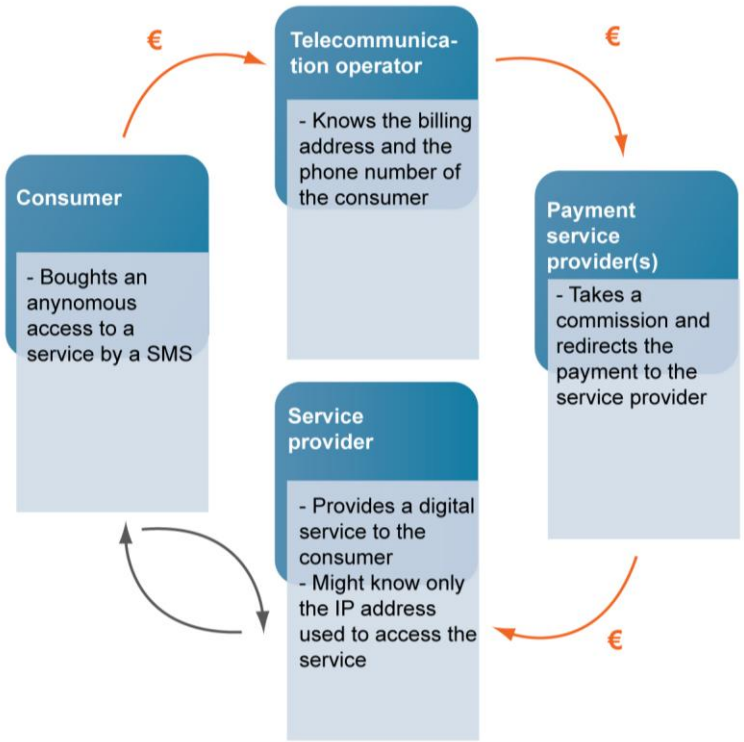
- Define a framework for resource identification, discovery, authentication; as available resources can appear and vary dynamically
- Define a framework for application partitioning and scalability, like it was done for multimedia content. This concerns of course game engine, but also code scalability, which can be linked to the criterion above (CPU resource…)
- Define a framework for management of offloading application components, including surrogate discovery, handover, and interfacing with the cloud.
- Define pricing/ billing interface to allow billing between the different actors in the chain
- Define more generally interfaces between the different actors (network operators, cloud providers (IaaS/SaaS providers), application providers,..)
- Interoperate with cloud at different level (but this is a general issue in cloud already)
- Define security interfaces and framework

ANNEX V-I: Major payment models used in games industry

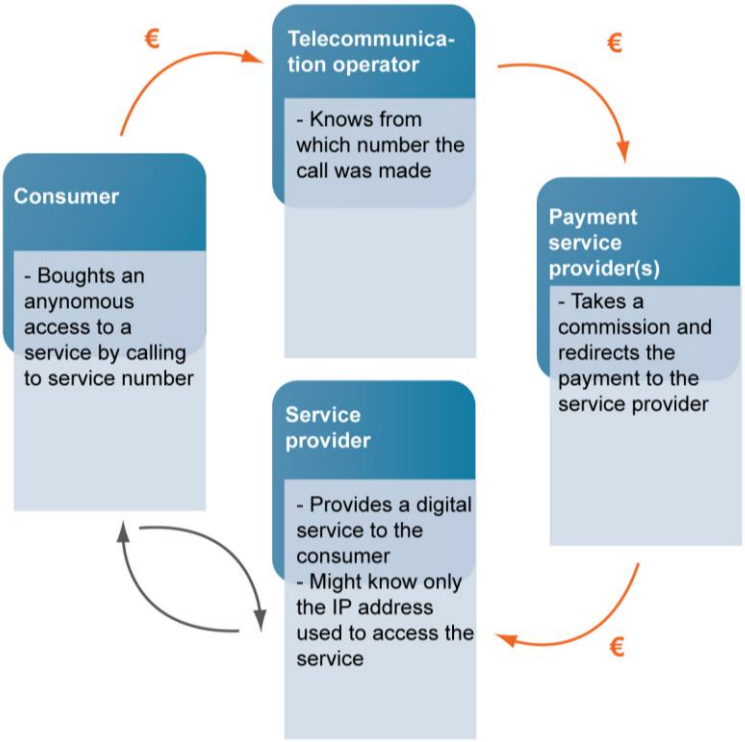
The following payment methods are just simplifications summarizing the most important actors in the value chain. Quite often it is possible that the value chains are even longer like demonstrated in Figure 6.



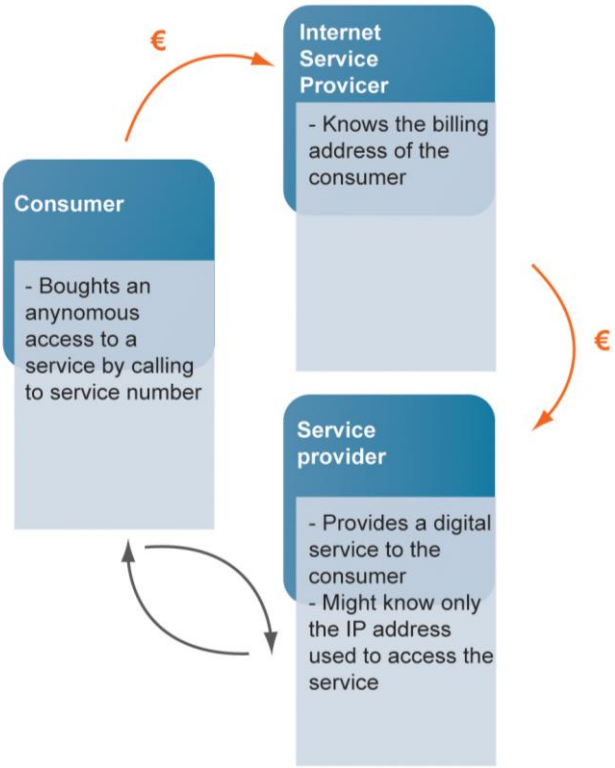
1. Prepaid cards
Used especially in Europe and Africa (e.g. Neosurf and Paysafe) and North-America (e.g. Zeevex, Ultimate Game CArd and Rixty). Payment service providers include PlaySpan, for example.



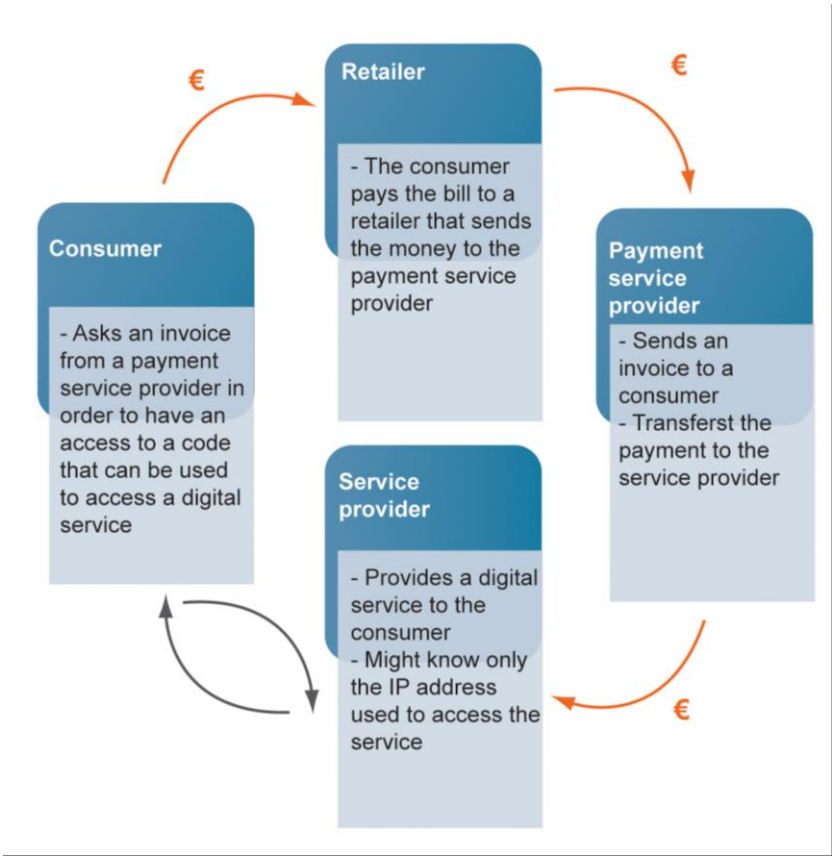
2. Premium SMS
Used especially in Europe, Asia, Latin America and Africa). Examples of aggregators include Allopass, Playspan, Boku.



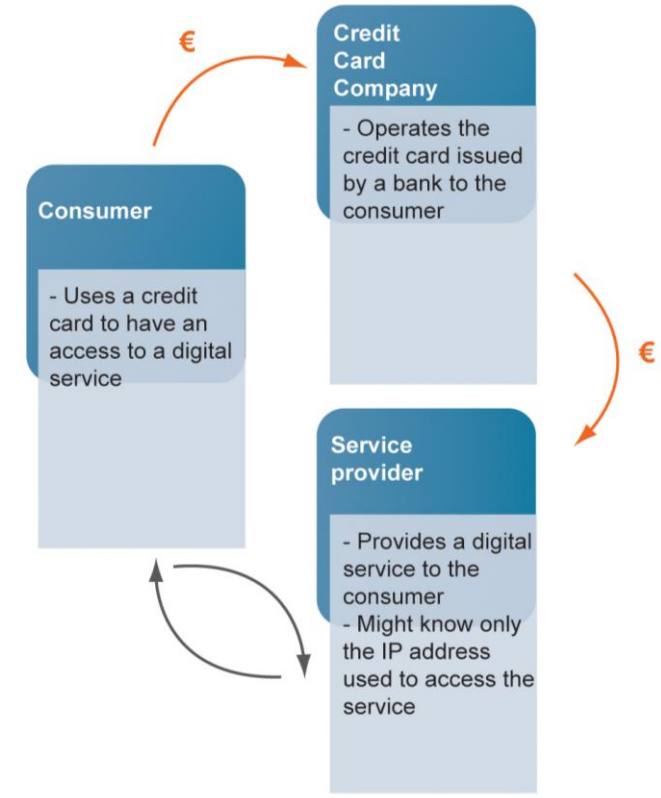
3. Payment by phone
Used especially in Europe



4. Payment by ISP billing
Used especially in Europe



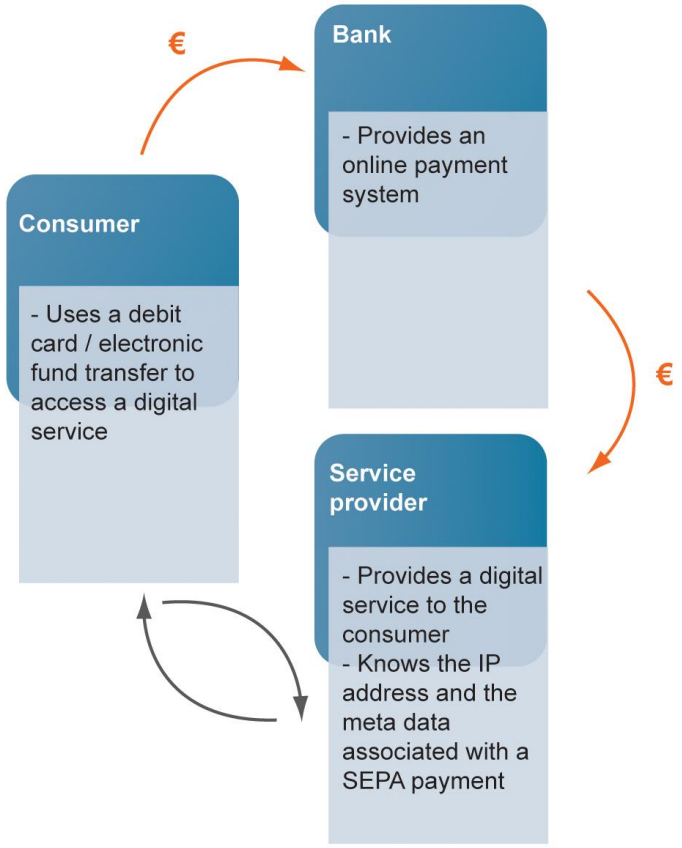
5. Payment by cash invoices
Used especially in Latin America, but also in Portugal, examples include Dineromail and Western Union Quick Pay



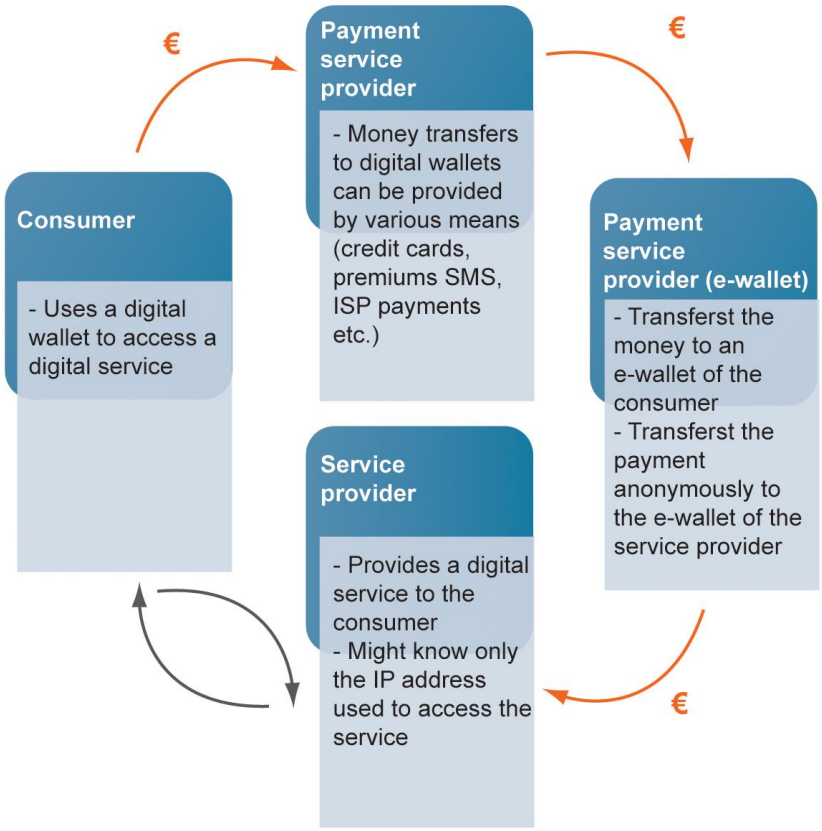
6. Payment by debit/credit card
Used all around the world, especially in the USA and Europe, examples include VISA, Mastercard, Visa electron, Maestro. It is worth noting that services like MB NET by Multibanco[2] or Paypal have been introduced to secure that no information associated with the consumer's credit card is transferred to a service provider.

[2] http://www.multibanco.pt/pt/

7. Payment by direct debit / electronic fund transfer
Used in Europe, examples include SEPA payments



8. Payment by a digital wallet
Used especially in Europe and Latin America, examples include Skrill, Hipay and Dineromail



13. ANNEX 6: HTML5 Report

1. Summary

This paper seeks to clarify problems and opportunities with the HTML5 platform, primarily as seen from the perspective of European mobile games developers.

Taking the commercial potential inherent in a global games platform already to some extent, through PC browser mainly, reaching literally billions of users, we chart some areas of particular interest, that emerge from our conversation with the game developer community, such as Augmented Reality, Geo-Localisation, Context Awareness, User Privacy, and Data Mining. We give consideration to issues regarding monetization, payment systems but also basic issues such as addressable markets, competition and barriers to entry to desirable markets, platforms and channels.

With some of these considerations in mind we move on to the actual game development and the technological challenges that will be encountered en route to market. We find that the performance often is highly unsatisfactory, as games often make the highest demands on systems and standards implementations. Differences in hardware, OS and browser versions, and variances in HTML5 and JavaScript implementation, along with varying support for core supporting technologies, lead to a high degree of fragmentation, and the resulting uncertainty about the actual function and user experience of the finished product is a very big problem for small SMEs doing games development. Some of these worries can to some extent be off-loaded onto providers of middleware and other resources, but this still demands a lot of resources devoted to testing and evaluating experiences gathered from other developers.

At that point we try to describe how far we have come in fulfilling the promise of a standard global games platform reaching literally hundreds and hundreds of millions of users that have adequate hardware and software in place. We go on to briefly discuss what remains, how game developers best put their limited efforts to use in influencing the nature and features of this future mobile games marketplace.

We also reason about specifically European policy implications drawn from the direct barriers to market for European game developers, but also considering important side issues that also have bearing on neighbouring fields, such as other games, other applications for mobile platforms, etcetera.

Finally we arrive at our main conclusions, where at the... ***[summary text ends]***

2. Introduction

a. PURPOSE OF THE PAPER

The fragmentation of platforms is an industry issue since the very beginning of digital gaming. The commercial control over the platform (the console for example) allowed business models based on a closed vertical value chain. Similar problems exist even with PC's. Games are dependent on the platforms and when these platforms are not available the digital content dies, an important subject in the context of our cultural digital heritage (see the KEEP project).

In the mobile gaming space, the fragmentation of handsets has always been one of the main sources of hindering growth. In many cases, the porting and testing on different handsets was more expensive than the development of the game. Today many mobile game developers prefer iPhone solutions as the system works on one single SDK family and Apple takes care of the whole billing and revenue distribution procedure in a closed eco-system.

Many game developers hope that HTML5 will have an important impact on the standardization of handsets and underlying technologies; many believe that this can even lead to a merger with the online world. The following paper tries to validate these opinions in practice. What are the problems that developers face? Do they usually encounter fairly trivial, if blocking, problems that can be expected to be solved at a predictable point in the near future, or are the problems more of a structural nature, and very hard to overcome, or is it even uncertain whether they will ever be solved? This paper tries to understand these problems from the technical, but also from the business angle, and to answer these questions in a strategic, but still practically applicable, manner.

It is worth special mention that particularly in SMEs, technological and commercial challenges go hand-in-hand, and are parallel concerns for key development staff. The mobile games field is evolving rapidly, both from market and technological standpoints, while almost without exception, European games developers must be regarded not only as SMEs, but among the lowest tier of SMEs in terms of staff, and often in terms of revenue and resources. At the same time they are active in a highly dynamic environment and striving to maximize the user experience building on the creative use of technology. Thus it is of the outmost importance to treat these challenges equally and holistically.

b. WHY LISTEN TO GAME DEVELOPERS?

Game development is often called the "hardest" software discipline. To an uninformed reader, this may seem puzzling, as it is hard to make a distinction between a high-end business system (like a banking platform) and a massively multiplayer game. There is a quite simple explanation as to why this is so. Game development is hard and strict on demands, because it involves every development discipline within the software sphere. Everything from tight optimization, parallelization, database handling and network coding, to graphical programming, sound design, and video. All of these widely varying concepts are combined into a single product where integration must be flawless and performance immaculate. In the

optimal case, of course. There are of course quality variations in banking systems just as in video games.

In general, this means that even a fairly simple game involves most of the techniques involved in software development, and that it suffers if any one of those parts has been poorly executed or integrated. In wide-platform technologies, such as HTML5, implementation effort seems distributed randomly between the different feature sets. Some feature may wait for years before even the simplest game can use it, while others are implemented at once.

Allowing game developers to be involved in the demand-setting system might appear intimidating, as the answer might be anticipated as; "We need everything to be perfect". This needs not be the case. Instead by sequentially focusing on the correct sub-areas of technologies, the standard evolves at a much better pace for everyone. "We need a little of everything, but done well!"

As game development already involves all development disciplines of the software spectra, all other non-game development projects should benefit equally.

3. Commercial challenges for HTML5 mobile games development

a. THE COMMERCIAL POTENTIAL OF INNOVATION

In this section we will try to give a broad idea on how we can evaluate the commercial potential of specific innovations in mobile games. We have, based on a number of interviews with actors through the value chain of mobile games, and based on the practical knowledge gained in practice by the consortium members, identified the following elements, relevant for the commercial potential of mobile games:

- **The news value of the technology applied**

If the technology is very new, the chances that media will cover it are larger than when the technology is already around for several years. The use of new technology for the first time also constitutes a chance to obtain "first mover advantage" and to build a dominant position in a specific genre of mobile games or applications.

- **The monetization potential of a new technology**

Certain innovations offer unique monetization opportunities. A good example can be found with GPS games and applications that can generate shop traffic and with that traffic enter the market of immaterial coupons and loyalty cards and be paid by these shops or retail chains.

- **Premium pricing**

Certain technologies are perceived as valuable by consumers, such as 3D. They can therefore be priced higher than average, non-innovative games. The same, of course, applies to established, well-known brands, from trusted, resourceful sources, just like in any industry, especially those dealing directly with consumers, or end-users.

- **Cross-platform potential**

Cloud gaming and cloud standards offer the possibility to publish games on a multitude of platforms, including mobile devices, tablets, Smart TV, PC and consoles and maximize revenues.

- **Costs**

The costs of innovation can plummet the commercial potential of innovative mobile games. These costs can be related to existing patents, or development costs.

- **Availability/fragmentation/market reach/addressability**

Is the technology or the innovation available on all smartphones or just a limited number of devices? Is it easy to deploy on multiple platforms ?

- **Other commercial aspects**

Is it possible to launch this innovation worldwide, or can it only be launched locally? Is there currently a lot of competition in this type of innovative game? Are there any specific revenue opportunities and/or spin-offs related to this technology such as Advergames, serious games, transmedia potential, etc?

From this analysis a handful of categories emerged, that seem to be central to the innovation at the time of writing, and these are: Augmented Reality, Geo-Localisation, Context Awareness, User Privacy, and Data Mining, here in the order of importance they emerged. These will thus be covered with particular emphasis in this paper.

b. CONTEXT AWARENESS

The commercial potential for context awareness in gaming is identical to the situation for context awareness in e-commerce and advertising. The more a publisher knows about players, the better they can serve them.

Analytical tools give us information on the mobile games markets, the app stores. In-app analytics provide information on how players use our games, for instance how much time they spent to move from level one to level two and when they decide to buy virtual goods.

Context awareness is the next step in consumer analytics, which enables publishers to know where their players are and what they are doing at a specific moment of the day. They can use this information to make commercial suggestions or to promote a a game that seems appropriate, given the current context.

Somebody who finds himself in a shopping area on a Saturday might be more interested in playing a game than in when he is in an office on Tuesday at 10 AM, especially if the publishers knows he is never playing a game in the AM during weekdays.

Successful use of context awareness in marketing reduces the risk of being perceived as a spammer and contributes in an important way to the success of promotional campaigns, marketing and even sales.

c. AUGMENTED REALITY AND LOCATION-BASED GAMES

Introduction

In this section we cover Augmented Reality and Location-based Games together, because of the many relations and overlaps of these categories, which are types of games in constant development, thus to some extent avoiding definition. However, we will still try to identify them as clearly as possible below.

Definition of AR technology used in mobile games
In this context Augmented Reality (AR) is a live, direct or indirect, video camera view of a physical, real-world environment whose elements are augmented by computer-generated sensory input such as sound, video, graphics or GPS data.

Geographic location-based AR uses GPS, compass and other sensors in a user's mobile phone to provide a "heads-up" display of various geo-located points-of-interest. Vision-based AR uses many of these same sensors to virtually display digital content in context with real-world objects – like magazines, postcards or product packaging – by tracking the visual features of these objects.

Definition of location-based game
A location-based game (or location-enabled game) is a videogame in which the game play somehow evolves and progresses via a player's location. Thus, location-based games almost always support some kind of localization technology, for example by using satellite positioning like GPS. "Urban gaming" or "Street Games" are typically multi-player location-based games played out on city streets and built up urban environments.

Current research and development trends are looking to other embedded mobile technologies such as Near Field Communication, Bluetooth, and UWB.

A few examples, AR:
In the following we present a selection of well-known or otherwise noteworthy examples of augmented reality games and applications.

Layar (NL)
Amsterdam-based Layar specializes in mobile augmented reality – the most popular medium through which the average person interacts with AR content. The mobile AR world consists largely of two different types of experiences: geolocation- and vision-based augmented reality.

Int13 (FR)
Paris-based Int13 specializes in vision-based AR. Their games combine real-life toys, such as cars with AR animations, generated in the mobile phone, such as AR Defenders and the Kweekies.

Ogmento (US)
Los Angeles and New York based Ogmento specializes in location-based augmented reality games. Their games include Paranormal Activity, an AR game showing paranormal AR images on top of the real world.

The most elaborate Location-Based game (LBG) is Shadow Cities made by Finnish studio Grey Area. Other specialized studios with different games are, Mekensleep (France), C4Mprod (France), Mosca (Belgium), Gbanga (Switzerland), 7Scenes (the Netherlands), Ogmento (USA), Giant Mechanic (USA).

A few examples, LBG:
It can be claimed that there are also identifiable sub-sets of the Location-Based Games, such as "hard-core", "mid-core" and "casual LBG" as well as semi-LBG.

Hard core LBG
These games require an active involvement of the player, usually outdoors and in many cases the players need to run. Examples are Meatspace Invasion and Code Runner.

Mid Core LBG
These types of games can be played anywhere anytime and do not require the gamer to walk or run, but the location of the player and the other actual players remains an important and determining element in the gameplay. Shadow Cities is an example of a mid-core game.

Casual LBG
The Japanese game Castle Maker is an example of a casual LBG. The player can pick up items on the road and trade these items with other players. The more you travel the more items you can gather.

Semi-Location-Based Game.
A game which is using location, but where location is not essential for the game. Examples are MyTown (Booyah) and Parallel Kingdom.

How AR is monetized
AR is still considered as distinctive feature in games, although it is already applied in games since 2005 with AR Tennis, created by Hitlab and the Norrköping Institute of technology.

The use of AR in connection with brands has proven to be successful at several occasion, such as a game based on the French brand Lustucru. An application created by French studio C4Mprod allowed consumers to play an AR game when they were pointing their smartphone camera on the Lustucru logo. Similar applications can be found in the previously mentioned Layar app.

It seems that the combination of location based AR and advertising or other forms of engagement of consumers with brands are an area we see explored in many countries.

The following monetization models are used in AR games:

[1] License fees
Many games with AR features are premium games in the EUR 0.79 – 1.79 ranges. The premium games are generally games without any advertising or in-app purchase.

[2] In-app purchase of virtual goods
A growing number of AR games allow users to buy virtual goods, power ups or other additional features in a game. Usually a player needs to buy virtual currency with real money and this virtual currency allows them to buy these virtual goods. Companies such as TapJoy provide ways to players to get virtual currency as a reward for looking at advertising or downloading games.

[3] Advertising
Companies such as Google, inMobi and others are providing in-game advertising to game publishers

[4] Merchandising (toys)

[5] **Mobile advertising**[3] AR redefines the way people interact with products. Companies such as GoldRush experiment with AR by bringing retail experiences to the home; you can virtually try your new jeans, shoes or jewelry at home using their platform.

These technologies offer new opportunities for serious game studios who create games for products and brands.

Location based mobile advertising is used in successful games such as My Town (USA) and offers incentives to players to pick up virtual coupons or to take advantage of special product promotions in stores around them. Location-based AR advertising campaigns allow users to see advertisement through the camera of their phone in certain shops. This can drive traffic to stores, points. This is done by the car industry, the fashion industry and the food industry. Virtual coupons, contests and other incentives get consumers to engage in sales promotions.

How LBG are monetized

Location Based Games are considered as distinctive in mobile games, although it is already applied in games since 2001with Botfighters, created by It's Alive from Sweden.

The gameplay of LBG is often, but not always, connected to one of the following areas of activity:

1. Jogging, sports. (Fast Foot Challenge, Code Runner, Meatspace Invasion)
2. Discovery, tourism (Castle Maker, Aroune Me)
3. Teambuilding, (La Mosca, Triangler)
4. Find deals nearby (Castle Maker, SCVNGR)

Games in the first category are behavior with license fees and the exploitation of Virtual Goods/ in-app purchase. Games in the second category are monetized with advertising and specifically location-based advertising. The third category is characterized by a monetization of a game related service. Games are instantiated by a game leader who accompanies the players and who is most of the times paid to do so. Finally, the last category is financed by deals with advertisers and virtual goods. The Japanese game Castle Maker is making several millions a month selling advertising and virtual goods.

d. USER PRIVACY

Information about players is extremely valuable (see also Context Awareness) in all phases of the lifecycle of a game: development, production and publishing. It helps developers and publishers to understand players better.

Of course there are many threats for consumers if their personal information is used for the wrong purposes. In the games industry Habba Hotel, a social platform for teenagers, was accused of connecting paedophiles with young girls.

Other examples include Sony and Blackberry, whose databases with credit card numbers was hacked, which created a huge threat for consumers worldwide.

[3] http://mashable.com/2012/06/20/augmented-reality-retail/

The main challenge for the games industry is to provide adequate guarantees to players that their data are safe with them. The problem here is that they cannot give a 100% guarantee. Nobody can.

e. DATA-MINING

Data mining is essential to the commercial potential of games whereby the retrieval of user data is central to the business model and consumer offering.

Retrieval of user data
Today, in Q2 2012, a very large segment of all mobile games build for smartphones, are equipped with in-game analytics, tracing in detail how gamers interact with the game.

Studios and publishers are using tools from companies such as Flurry. These tools are provided by Flurry in exchange for a payment and provide the game publisher with detailed analytics of the usage of the game. These analytics are used to improve the game in terms of gameplay and in terms of profitability. Analytics allow studios to optimize the use of virtual items and/or virtual currency.

Retrieval of gamers' ID
The success of Japanese companies GREE and DeNA is based on their ability to aggregate very large numbers of players around a social networking platform offering a large quantity of mobile games and apps.

Their impressive success in Japan, where consumers sometimes spend more than 60 Euro on virtual items per month, has allowed these companies to internationalize their services by acquiring other networks such as ngmoco and Open Feint.

The European network Scoreloop offering a similar service, has been acquired by Canadian Research in Motion (RIM/BlackBerry) in 2011.

Monetization of user data
User data are very valuable and this value is monetized in various ways.

One is aggregation. DeNA and GREE can be seen as TV channels with a large quantity of regular viewers. Their viewership allows them to sell advertising, games and virtual currency across games and apps. Bringing users to games is the most valuable proposition at this moment in time.

Companies such as Foursquare and Google also sell user behaviour data to a wide range of companies, wishing to understand consumer traffic and behaviour. This data generates millions of Euros in revenue for the sellers.

f. TRANS-MEDIA, CROSS-MEDIA AND RELATED FIELDS

Mobile games which are part of a more complex intellectual property, such as a movie, a TV series, a book or a massive multi player on line game are often referred to as transmedia games.

The player, already familiar with the property or the story is tempted to buy the game, expecting to find the same thrill as he or she found in the movie, the book or the other 'original property'. Unfortunately, most of the time the player is disappointed and finds a game of poor quality, with a gameplay that has no relation to the original. In mobile games, this is a reality with very few exceptions in mobile games.

The reasons for these recurring phenomena from the side of the games studio or publisher can be:

- The price for the license is high, or the price for the minimum guarantee is too high, which means there are not enough resources available for the production of the game.
- The license period is short (1 to 2 years) and therefore there is not enough time to produce a great game.
- The prominence of the property is considered to be high and the developer or publisher thinks that a success is guaranteed.
- The developer does not have access to any art, stories or other assets of the IP, which makes it difficult to produce a great game.

The reason for these recurring phenomena from the side of the owner can be:

- The studio or owner may have the rights to license the right to produce a game based on the title, but has no access to actors, art, stories, marketing tie-ins, etc,
- The studio or owner may have no understanding at all of the games industry and is only interested in the money.

However, the commercial challenges for transmedia properties are important:

- A mobile game can allow a parallel gameplay and therefore enhance the experience of a movie, a TV series or an online game. Players would be willing to pay for this enhanced experience.
- A mobile transmedia game can stretch the life cycle of a property, by offering an interaction with the IP beyond the launch of the movie and the ending of the TV series.
- Mobile transmedia games are strong marketing tools, at launch, during distribution or broadcast.
- TV and Cinema are strong aggregators for mobile games and can bring many players (and payers) to mobile.

g. OS FRAGMENTATON

In the current state of affairs, Apple's App store offers by far the most chances for publishers and developers to make money. This is largely due to the fact that their model offers the best protection against piracy, the least handset fragmentation, and finally the most favourable revenue split in combination with a widely established and trusted marketplace.

All other OS are less interesting commercially due to a limited installed base (RIM OS, Windows, Bada) or piracy and in-OS fragmentation (Android).

The biggest challenge today is how to avoid a situation of a single OS market, which would give too much power to Apple.

h. PAYMENT SYSTEMS

There is a wide diversity of payment systems in use all over the world, and many of them are also in use in many or all European countries. However the picture is not homogenous, and particularly so in Europe.

Please refer to Annex I for a comprehensive overview of mobile payment systems.

i. DEPLOYMENT

A number of choices need to be made very early in the game development process, involving an iterative and far from trouble-free decision-making process. The level of uncertainty facing the typically tiny SME is almost overwhelming, making either for very tough and resource-drawing decisions, or operating more on intuitive feeling or just plain taking chances, chances where the probabilities often are unknown.

Size of addressable market
For the game developer the potential number of end-user customers that can be reached at an acceptable cost is a function of the choices made that minimizes the fragmentation, or that maximizes the adherence, but not necessarily to standards, but the dominating interpretations and implementations of standards.

Barriers to entry
Reaching markets that experience less competition may increase revenues for the developer, but often involves thresholds in porting or even rewriting substantial parts of code and re-doing a lot of graphics. So, the marginal revenue after new-market investment is of course very important.

Competition
The size of the market share for a certain platform must be weighed against the numbers of existing and coming games for the same platform. It is a matter of expected revenue from a certain effort, a regular cost-benefit analysis is needed. However, much of the necessary data is hard to find for or even deliberately hidden from the developers. Increasing competition, more on offer for a certain platform in relation to the buys of the audience, of course results in diminishing returns for the same effort.

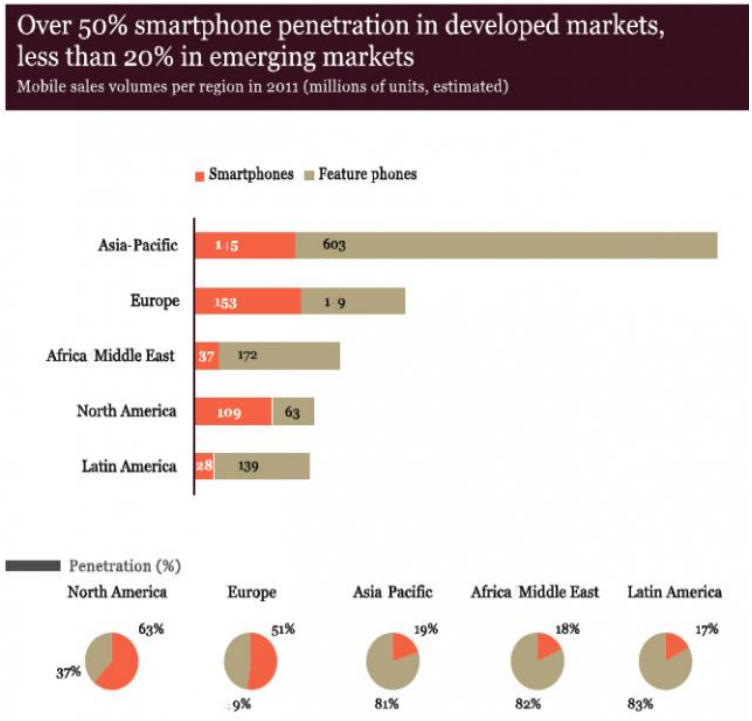
Examples of consideration
As we are not aiming to make any recommendations here, we limit ourselves to a small handful of illustrations of the complex data to be thoroughly analysed when planning to make a mobile game.

Figure 1. Market shares for different operating systems. (Source: "3 out of every 4 smartphones now shipping runs Android" at http://www.pocketgamer.biz/r/PG.Biz/IDC+news/feature.asp?c=46309 as accessed on 2012-11-21.)

Operating System	3Q12 Shipment Volumes	3Q12 Market Share	3Q11 Shipment Volumes	3Q11 Market Share	Year-Over-Year Change
Android	136.0	75.0%	71.0	57.5%	91.5%
iOS	26.9	14.9%	17.1	13.8%	57.3%
BlackBerry	7.7	4.3%	11.8	9.5%	-34.7%
Symbian	4.1	2.3%	18.1	14.6%	-77.3%
Windows Phone 7/ Windows Mobile	3.6	2.0%	1.5	1.2%	140.0%
Linux	2.8	1.5%	4.1	3.3%	-31.7%
Others	0.0	0.0%	0.1	0.1%	-100.0%
Totals	181.1	100.0%	123.7	100.0%	46.4%

Thus it may seem that Android is a good choice. However, this summer it was claimed by Apple that More than 80% of Apple customers were running the latest version of iOS while about 7% of Android customers were running the latest version. (Source: "Apple Rubs Android's Fragmentation Problem In Google's Face" at http://www.businessinsider.com/chart-of-the-day-ios-vs-android-fragmentation-2012-6#ixzz2CtP2zrBN as accessed on 2012-11-21.) And just the operating system does not necessarily tell us very much about the actual capabilities of the handsets actually in use.

Figure 2. Geographical variances in smartphone penetration. (Source: "Global Smartphone Adoption Approaches 30 Percent" http://www.wired.com/gadgetlab/2011/11/smartphones-feature-phones/ as accessed on 2012-11-21.)



4. Technical challenges for HTML5 mobile games development

a. INTRODUCTION

In this section we revisit the areas of particular commercial interest that have been covered before, but pay special attention also to the technical challenges, in the form of handset, OS, browser-implementation of standards, and other issues that are inherently the core of the development process and the decision-making that precedes it.

b. CONTEXT AWARENESS

The specific application should be aware of the current usage situation. This applies both to the content as processed and presented by the application, and to the current connection with network, peers and servers.

A very basic instance is where HTML content is presented differently when sizes and aspect ratios of web browser windows differ. A technical challenge more specific for HTML5, as applied to games development, related to context awareness is the matter of online and offline content. In HTML5 applications can be offline and the choice of going online may not always be as straight forward. Does the application choose to go in online mode even though the connection quality is poor? Perhaps the user experience will be worsened going into offline mode at that point. A similar scenario exists in reverse, as the connection quality drops, the application may choose to go offline for the user's own benefit.

Imagine a music streaming service that supplies streamed music to the user and allows the user to keep a local library of media. When quality of service goes down, perhaps because the user entered a shopping mall with poor reception, the application can choose to increase the user experience by going offline and allowing the user to only play his local media. Stuttering and buffering issues are avoided due to the low connection and the user retains his high quality experience.

Another HTML5-specific context-awareness issue is of course cross-platform support. Not all of the new features are supported by the myriad of devices now available to consumers. There are hundreds of different mobile handsets that all support HTML5 in their own way. A very simple – but to the point – example would be the way events are handled in javascript. A PC application might listen to mouse events, while a mobile application must listen to touch events. Displaying the application correctly on a big number of handsets is an old problem with no easy solution. In the worst case, to ensure a high-quality experience on most devices a big testing effort must be undertaken.

5. PRIVACY

The managed local, off-line, storage that HTML5 introduces has many technical and user-experience advantages. From a privacy perspective, it may be less than advantageous, though. A shared computer using local storage is in principle not private, but shared. It can be argued

that this is not necessarily a HTML5-specific technical challenge, as local caching of on-line content may pose more or less equivalent security and privacy breach risks.

However, in HTML5 the built-in support for offline storage does increase this risk in three ways. First: user protection. You must protect your users' privacy when storing sensitive data in the local database. How easy is the local database breached on the huge number of platforms available? Are there critical security flaws on that one platform you didn't test that much on?

The second issue is a more abstract one, but no less important. With HTML5 and the evolution of web content, we are creating more and more advanced applications. Some even with hardware-support (E.g. WebGL). This leads to bigger chances of the software harboring compromising flaws. Especially in a field that is very susceptible to security breaches.

The third issue relates to the first and is connected to the privacy laws issued by countries world-wide. How can you guarantee that you are storing your content in a secure-enough manner to facilitate a global launch? If your data is stored on the cloud, can you be sure you comply with all different national quality demands? Especially the question of liability is an interesting one. If rules are not followed to an adequate extent, who is legally liable? In most cases it might be the poor, clueless, developer. Without a global agenda on similarizing or standardizing such national privacy demands, the job of the developer becomes much tougher. Especially considering that most applications today store private data either online or offline – many of those through cloud storage.

a. DATA-MINING

Ensuring the privacy of your users connects this point to the aforementioned privacy issues, although it does have a unique twist. All usage information you may store about your users, to streamline their experience in your system, can be compromised. Before, this was done with cookies and was very easy to compromise.

With HTML5 the security is increased (on most platforms) by using the local storage database, included with the standard. Instead of storing information in cookies, developers must switch to using the more secure (albeit not perfect) local storage. This may not always be possible, since there is a difference in how cookies and the local storage work. This forces developers to re-think their development strategy in order to just keep their users' information safe. Having said this, it is still worth to again mention the security problem here. Still on some systems, the local storage is implemented in the same manner as cookies, making it extremely exploitable. This brings us to the next point.

Encryption is a challenge now most developers face if they wish to guarantee the safety of their users' data. This is not a unique challenge to be faced in HTML5, but it is much more important to be overcome in an age where all applications can freely store and install data on the user's device – that may or may not be secure. By encrypting data, it becomes more secure, of course, but it also introduces a latency that may not be acceptable.

Finally, if developers wish to data mine in HTML5, they face segregation at the technology level. With many different places for users to place their data through a myriad of applications they are interfacing with, data mining is considerably more difficult. This is of course

considered a good thing from the perspective of a user. However, providing services that may actually benefit from tracking the usage and habits of a user – with the user's full consent, of course – are harder to make today.

b. OS FRAGMENTATION

User experience is absolutely critical for games and thus also for the technical aspects of HTML5 games development. A consistent experience is not only desirable, it might be absolutely vital in a games environment. A simple inconsistency emanating from slightly different viewport sizes may not only break the challenge and game-play experience in regard to what the designer intended. It may also introduce "fairness" and "cheating" phenomena. A player may have a distinct advantage or disadvantage by using a specific browser on a specific operating system, leading to a situation where player skill may become a secondary issue in a contest-like usage context.

To specify further, the new graphical options available in HTML5 – either through the Canvas tag, or webGL – differ wildly between operating systems. The support might exist on a wide range but the performance is in most cases far from equal. Especially if we involve mobile handsets. The loose requirements on implementers of operating systems regarding these technologies leave developers in the dark, forcing them to develop less-appealing applications for a mass market, or more-appealing applications for a narrow segment. Neither of these options is particularly attractive. A stricter adherence to standards on the requirements of quality would be preferred. WebGL support on a system that offers only a fraction of the frame rate another, hardware-equal system, but with a different operating system does offer, is not a bonus. In fact, developers are more troubled by the fact that their application works very poorly on certain systems than by it not working at all. In the end, the developer might be forced into exclusion strategies, where systems that are marketed with support for certain features (whether it is Canvas or WebGL) are not supported by a developer's application in order to keep the quality of user-experience consistent.

We have elsewhere touched upon the security-related issues that may arise from operating system fragmentation. However, it is worth noting yet again, that without a standardization of certain core features, the developers face a tough time dealing with security aspects throughout all operating systems.

c. IMPLEMENTATION DIFFERENCES

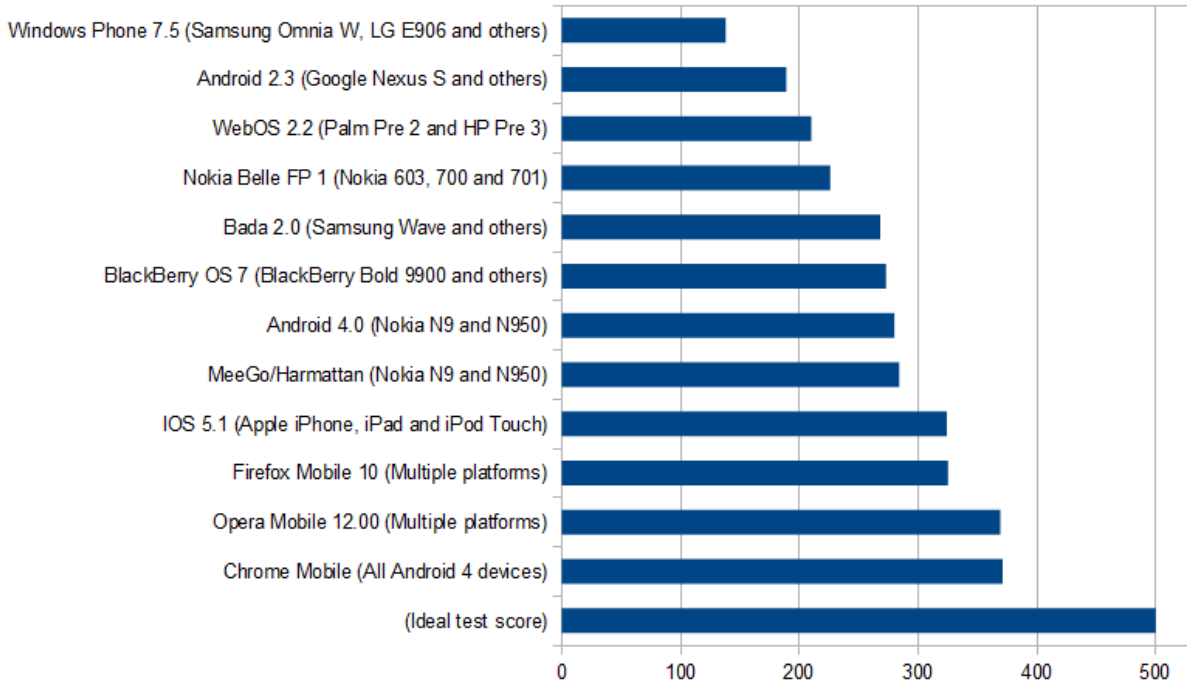
Furthermore, which is again not a game-specific issue, the cross-platform viability of HTML5 is actually in question already on fairly obvious, expected different implementations. As a very basic example (that has been previously touched upon), mouse-related events do not exist on tablet computers, which use touch events.

Finally, segregation in relation to performance comes through differing implementations of the JavaScript execution engines. This segregation is mainly seen in the choice of browser. However, the implementation of a certain browser regarding its JavaScript engine might differ completely between different operating system versions, thus creating a large rift in performance. This is not only a worry browser developers must cope with, but also application providers. If a browser of choice performs less than ideally on a different platform performance-wise there is very little the developer by themselves can do to alleviate the issues

(here we of course assume that all necessary optimization has been done). To summarize; in order for HTML5 to be a valid platform for developers to develop media-rich applications for, the consistency across different operating system implementations must increase. Both support-wise, but also (and some might say more importantly) quality-wise. Otherwise it is up to developers to secure comparable quality through dodgy means that seldom lead to a well-defined product. And a browser upgrade may break all such well-meaning tricks retroactively.

Below is an overview of the major browsers and their current implementation on different operating system and hardware platforms. They have been tested and scored on a number of features. Please note that the scores do not aggregate. This means that out of an ideal test score of 500, two browsers' implementations could at a theoretical extreme both score 250 and still have no support at all for the same features.

Figure 3. Variations in HTML5 browser support as tested on a range of handsets. (Data source: http://html5test.com/results/mobile.html as accessed on 2012-08-24. Selection and graphic presentation source: Own.)



It is not only the mobile field that suffers from lacking adherence to standards, and bowsers not providing the support for content that they claim to. As can be indicated above, it is far from satisfactory in the mobile field. However, on more established browser platforms, such as PCs, game developers can still encounter insurmountable obstacles, as is demonstrated in Annex II, where a fairly straight-forward WebGL (browser-based 3D) deployment failed in a highly disappointing way, seemingly due to shortcomings in standards implementation.

d. DE-FRAGMENTATION INITIATIVES

The Firefox Marketplace, as an example, and in extension; the Firefox OS, are two initiatives taken by the Mozilla group to lessen segregation for end users and developers. The technology means to supply a common platform for developers to create with - HTML5 - while users access their applications in the same way whether it's on their desktop computer, their tablet or their phone. The Marketplace provides a common ground for app distribution, similar to Google Play or the AppStore, the difference being it's not locked to a specific OS. Instead, both Android, RIM, Windows and iOS users can download and use the same applications that (in theory) behave identically.

Unfortunately, though the solution sounds good in theory it's practical implementation, at the time of writing, is far from flawless. It should be noted that the technology is still in its pre-beta-stage. The solution to the segregation/browser problem is (intuitively for the Mozilla corporation) to only use Firefox on the platforms you wish to access your apps. The thesis is that the execution and feature-implementation control is set by a single browser. This does do away with browser segregation, but does not solve many of the other issues that riddle HTML5 applications; performance and hardware segregation. The performance factor carries much weight, as developers cannot expect their end-users to settle for less, performance-wise, than the competing platform.

A direct, exploratory test of this technology, carried out by consortium members at the time of writing, yielded poor results across three to us readily available Android-based mobile devices of fairly recent manufacture (Samsung S2, Samsung Nexus S, and Sony Ericsson Xperia). Even though the HTML5 technology is (relatively) widely spread across the Android platform, most (if not all) applications did not work. Testing other, advanced HTML5 games and applications on stationary, PC platforms, directly from the web-based source would yield better results, so the problem seems not to lie in the HTML5-capabilities of the devices but rather in the Mozilla implementation of their new platform; Aurora.

A developer's intuition may very well say that attempting to force users to choose a single outlet of technology (in this case, a browser) is not the correct solution to this problem. Instead, we should seek strong standardization that not only focuses on the HTML5-standard as a whole, but also places demands on its implementation. The ideal case being that game developers would be able to place some of these demands that are specific to games themselves. Demands such as performance, network communication, etc.

e. LOW-LEVEL API'S

From the perspective of the HTML5 browser layer, WebGL is an example of a basic foundation building block, providing 3D graphics capabilities. Related to the previous issue of OS and browser fragmentation, WebGL is not today implemented in Microsoft Explorer.

Furthermore, when dealing with low-level APIs performance becomes much more critical. A normal HTML5 application worries less about performance than an embedded product on a constrained platform would. When working with these low-level APIs the developers themselves must re-think development strategies and approaches. It is both a habit and competence issue. For companies that normally focus on pure web development the transition to a high-performance-environment might be like entering alien territory. The experience

might be poor or lacking completely, and surmounting this may require very unfamiliar thinking models. This will lead to poor results in prototypes and ultimately may cause companies to abandon this new and exciting technology.

To continue on the performance theme, we must also discuss how applications are implemented today in HTML5. Real-time logic that interfaces with these low-level APIs must today be written in JavaScript. As mentioned in the previous section, regarding operating system segregation, code performance can vary largely. This means that a very basic challenge of the developer lies in the smart use of low-level APIs. Reliance on JavaScript logic should be kept to a minimum – but this is far from possible even in the most ideal of cases. Developers then face the choice between empowering their application through low-level APIs but suffer consequences of poor performance – at no fault of the developer – or ignore using advanced features altogether. Unfortunately this choice dominates much of next generation web development and is only extra highlighted in this case, when developing games.

Quality-wise we developers must consider the underlying hardware. A low-level API promises only a link to some low-level system; graphics, sound, network etc. However, we already know that hardware can differ wildly and thus to ensure the quality of a certain low-level API implementation the developers face two large challenges. One of testing and one of simulation.

First of all, testing is even more paramount when dealing with low-level interfacing code. If quality is to be preserved things may have to be done differently to work consistently across a large range of hardware. Even though this is normally a segregation issue, it is more prominent in the field of low-level programming than anywhere else.

The second challenge lies in simulation. When hardware differs and the quality of the achieved results also differs developers may have to simulate certain lacking features in order to achieve the highest apparent consistency. This simulation may not always be straight-forward nor possible. As a simple example, imagine a low-level sound processing API that allows application developers to de-code heavily compressed audio on the fly, for playback. Now imagine two different sets of hardware, that both decode the audio correctly, but with different volumes. The developer must then cover for this hardware-based flaw by creating a volume normalizer.

f. DEVELOPMENT TOOLS AND PLATFORMS

There is a lot of middleware to choose from. A complete set can be obtained, for free or for a price. However the overwhelming multitude of different solutions can be confusing and requires research on the developers' part.

A lot of time is needed to find and evaluate enough products to make educated picks. Evaluation needs to be done concerning overall quality and suitability in relation to price. Different middleware with the same intended task perform to different levels of quality. Some middleware are distinctly specialized for specific challenges of game development while others are closer to complete libraries. A middleware may be well suited for a specific game project while the price may not, or vice versa.

The reality of many game developers is most likely researching a handful of candidates through web forums and articles and then picking one and running with it. The community of game developers would be well served by increased interaction with peers on what works and what does not.

One of the leading multi-platform development systems is Unity, allowing developers to "code once" and "publish many", on several different hardware platforms. Unity will eventually support HTML5 applications but are waiting for "the moment HTML5 is right for games" (Unity CEO David Helgason, as quoted on 2011-11-30 in "Unity: We'll support HTML5" http://www.develop-online.net/news/39242/Unity-Well-support-HTML5, as accessed on 2012-11-21.). At the time of writing Unity does not support HTML5.

Please refer to Annex III for a brief overview of examples of resources for mobile game developers working with HTML5, ranging from single component libraries, to whole suites of middleware and deployment platforms.

g. PERFORMANCE CONSIDERATIONS

HTML5 is an amalgam of two major systems, HTML and JavaScript. In a normal scenario, a piece of software is executed as machine code instructions, directly interpreted by the hardware CPU on a given computer, within the confines of an operating system. A program created with HTML5 technologies has a different ladder of execution. First and foremost, the HTML5 program is sandboxed in a virtual environment for the end-user's safety. Then depending on browser, the HTML code is parsed and converted to a Document Object Model. The JavaScript itself is either interpreted, or compiled (JIT) and executed in this sandbox that is in the domain of the browser itself. The browser then behaves like the classical program, running on the CPU.

As the astute reader may infer, there is a big difference and a huge by-design-introduced latency in execution. For games, performance is of vital importance. This is true regardless if the game is a simple 2D title, or an advanced 3D production. No delay in updates and immediate response on player interaction are necessary for a satisfying gameplay experience.

We must ask ourselves, are we pushing this platform in the right direction? Are we emphasizing the right areas? If we get better database support, better streaming video standards and better audio playback, are we still not trapped by the actual blueprints of the underlying system?

Should we instead push for consolidation and standardization of the actual platform, regarding its execution environment and sandboxing? Of course, segregation is a huge issue here, as all browsers use some unique flavor of JavaScript engine. A comparable platform is Java, that uses a single virtual machine sandbox that compiles Java byte code into native code when needed. A Java developer knows that his code will be compiled to machine code and execute at maximum performance on any system - allowing the hardware to be the only limitation.

As HTML5 developer faces the same limitations in the end, in the hardware. However, we introduce a whole intricate web of other limitations, too. Things such as browser, sandbox and JavaScript compiler/interpreter, just to name a few. Perhaps before going further into a discussion of the details of the platform itself, and the implementation results for individual features across browsers, maybe we should consider focusing on the standardization of the platform's execution environment, to guarantee as similar results as possible, across platforms, on the lowest level possible.

6. HTML5 for Mobiles, State-of-the-Art and roadmap

a. State-of-the-Art, or Where Are WE?

As can be seen or inferred from the preceding sections, HTML5 is in principle a very exciting platform for game development, also on mobile handsets. Full, true implementations on handsets would really enable cross-platform games development, linking online and offline play and content.

It has also been demonstrated that we might be at least "half-way there", even if core user experience expectations on audio and 3D graphics are far from in place. The question is more "will we ever get there?", as the work on the HTML5 standard seemingly might split. The two forces to be counted, the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG) seem to be falling out of step. The W3C is an international community where member organizations in academia, public service and industry, a full-time staff, and the public work together to develop Web standards. WHATWG mmembership is by invitation only, and consists of a number of representatives from various browser manufacturers. ("[whatwg] Administrivia: Update on the relationship between the WHATWG HTML living standard and the W3C HTML5 specification" by Ian Hickson, 2012-07-19, at http://lists.w3.org/Archives/Public/public-whatwg-archive/2012Jul/0119.html as accessed on 2012-11-20.)

We already know from own experience and that of other developers that the HTML5 standard and specifications are far from implemented. Much of this could be argued that it is the worry of middleware providers. Game developers need not check every singe thing they do on resources like "Can I use..." (See http://caniuse.com), because middleware and tools providers should already have checked that. Still, many of the tools are unproven, and it is a huge undertaking or a small games developer just to evaluate what is available. Middleware thus does not solve the implementation issues, more than to some degree, and that degree that is not transparent to the developer.

b. Roadmap, or How and When Do We Get There?

Joining standards bodies
W3C - HTML
WHATWG – HTML implementation - not open
Khronos Group – WebGL and more
ECMA International – JavaScript

Identify most important hurdles
Mobilizing gamer community, finding browser manufacturer pressure points
Alliance(-s) with individual browser manufacturers

Financing activities
EGDF Mobile SIG supporting memberships
Awards
Certifications

7. European Policy Implications

a. Introduction

A few fields are worth particular attention from the European perspective, and could thus possibly be considered for focused efforts in European policy-making. These fields are briefly revisited below.

b. Augmented Reality and Location-Based Games

Location-based game development has from the beginning been a European issue. The connection between the geolocation through mobile phones and gaming has been the subject of research on European and national level. Projects accessing public space using mobile phones (such as Blinkenlights) have been developed in Europe first. However in Europe location based games have not reached yet commercial validation. Commercial projects failed (Orbster sponsored by Bigpoint). It seems as if geolocation based projects (Gbanga from Switzerland) come into existence particularly in Europe. Even though the market is not mature enough yet it could well be that Europe is the place where location based gaming takes off first. In this field there is therefore a clear European dimension of interest.

c. Data-mining, privacy and related issues

Data mining and data accumulation encounter regulatory user specific resistance. Currently the EC is discussing further regulation (cookie law). This development is seen highly critical by self-publishing game developers. It must be seen, that they compete in a virtual and global market place against companies, who are much less restricted.

d. Transmedia, Cross-media, and Linking online and offline games

One of the largest expectations towards the new development is the trans platform development of games between mobile and online. Developers hope, that the separation between those two worlds will blurr and that gaming in mobile and online at the same time will be possible. Some commercial projects do have an integrated approach. From a European point of view this is specifically interesting as it enlarges the content scope.

e. OS fragmentation

In the time, where European mobile games still were dominant world wide the OS fragmentation of operating systems was the largest obstacle. Whether HTML5 as a free of charge standard can however serve as a solution remains to be seen.

f. Low-level API's

The development of low level API's has been a European strategy ever since platform fragmentation exits in Europe. The European industry never mastered the hardware and therefore needed to develop strategies from a start to overcome fragmentation of platforms.

One of the solutions was middleware, which resulted in a number of commercial products from Europe with a word wide impact (e.g. Shark3D, Havoc/Trinigy, CRY engine, Unity etc.). Similar developments we find in the mobile space (e.g. Marmalade). Other developers have developed multi-platform mobile pipelines, which makes them quite agnostic to low level API's.

g. PAYMENT SYSTEMS

See ANNEX I or n overview of the plethora of payment systems in existence in the mobile field, which offer a global addressable market, of course, but requiring commercial agreements and technology implementations for each and every game of European origin, a great undertaking for SMEs and maybe uniquely so for small European SMEs in games development.

8. Conclusions

(This section is in editing – input most welcome.)

Answers needed to the introduction: *(Many developers hope that HTML5 can have a serious impact on the standardization of handsets and underlying technologies; many believe that this can lead even to a merger with the online world. The following paper tries to validate this opinion in practice. Which problems do developers face? Do they usually encounter fairly trivial, but blocking, problems that can be expected to be solved at a predictable point in the near future, or are the problems more of a structural nature, and very hard to overcome, or is it even uncertain whether they will ever be solved? This paper tries to understand these problems from the technical, but also from the business angle, and to answer these questions in a strategic, but still practically applicable, manner.)*

Just some tentative answers:
(Instead, we should seek strong standardization that not only focuses on the HTML5-standard as a whole, but also places demands on its implementation. The ideal case being that game developers would be able to place some of these demands, that are specific to games themselves.)

> *(The reality of many game developers is most likely researching a handful of candidates through web forums and articles and then picking one and running with it. The community of game developers would be well served by increased interaction with peers on what works and what does not.)*

ANNEX VI-I: Major payment models used in games industry

The following payment methods are just simplifications summarizing the most important actors in the value chain. Quite often it is possible that the value chains are even longer than demonstrated elsewhere.

1. Prepaid cards
Used especially in Europe and Africa (e.g. Neosurf and Paysafe) and North-America (e.g. Zeevex, Ultimate Game CArd and Rixty). Payment service providers include PlaySpan, for example.

2. Premium SMS
Used especially in Europe, Asia, Latin America and Africa). Examples of aggregators include Allopass, Playspan, Boku.

3. Payment by phone
Used especially in Europe, this involves calling a premium-toll number for a one-time or per-minute charge.

4. Payment by ISP billing
Used especially in Europe, this adds the cost to your monthly bill from the operator.

5. Payment by cash invoices
Used especially in Latin America, but also in Portugal, examples include Dineromail and Western Union Quick Pay

6. Payment by debit/credit card
Used all around the world, especially in the USA and Europe, examples include VISA, Mastercard, Visa electron, Maestro. It is worth noting that services like MB NET by Multibanco[4] or Paypal have been introduced to secure that no information associated with the consumer's credit card is transferred to a service provider.

7. Payment by direct debit / electronic fund transfer
Used in Europe, examples include SEPA payments

8. Payment by a digital wallet
Used especially in Europe and Latin America, examples include Skrill, Hipay and Dineromail

[4] http://www.multibanco.pt/pt/

ANNEX VI-II: Sample case: HTML5 3D Evaluation by Redikod AB

"At Redikod we had a project that was going to utilize **high-quality 3D rendering in a web product**. In the initial product design phase, we chose to create a prototype of the technology using **HTML5 and WebGL**.

As the programming environment, we chose Google's GWT 2.4 (currently stable version) and GwtGL 0.9.1 as the WebGL binding. Prototyping an application that renders 3D content was fairly simple. The development environment was quite good and facilitated easy testing. Everything was fine until we started cross-browser testing of our technology. Immediately we encountered compatibility issues and bugs that not only spanned browsers, but also computers. On some computers Firefox wouldn't fire up a WebGL canvas to render in, while on others it would. The same could be said for almost all browsers. To add to our frustration, many of the issues we encountered while prototyping were **known bugs in bug-tracking tools of the various browsers that hadn't been resolved for months and even years**. No priority was given to these issues.

Being PC, mobile, games console and web developers, we're quite used to compatibility issues and segregation. However, in this particular field two of the problems forced us to seek solutions elsewhere. First and foremost, the issues that many browsers had were well-known and old, no priority was given to them and we did not want to put our fate in the hands of someone else's' priority sheet. Who knows when someone decides that it's finally time to fix WebGL issues. Secondly, the amount of bugs in the web browser itself added another layer to an already very layered architecture. Instead of working with code that ran on a computer that interfaced with hardware we ran code that ran in a sandbox, in a virtual machine, in a web browser that runs on a computer that interfaced with hardware.

After a period of trying to fix the myriad of issues that arose, we chose to transfer our technology to a different web-based platform where none of the issues were encountered. For this particular purpose **we chose Java Applet technology, with native OpenGL bindings to render hardware-accelerated 3D graphics directly in the browser**. The drawback is of course the need to download a small JRE package from Oracle (which many already have installed) while the benefit is a minimum-fuss very powerful solution that fits both small prototypes and large enterprise-scale projects."

Mikael Baros
Chief Technology Officer, Redikod AB, Sweden
October 2012

ANNEX VI-III:
Examples of middleware and component libraries for making games with HTML5

	Description	*Notes*
3D Graphics Engines		
Three.js	Well established, high performing 3D engine	
DivSugar	CSS-based 3D Graphics Engine Scene, Animations, 3D-Geometry classes, input handling	
J3D	Javascript / WebGL Engine	Provides tool pathway from Unity 3D.
PhiloGl	Thin-ish layer on top of WebGL	
PlayCanvas	Cloud-based 3D HTML5 engine and toolset. Full JS engine, real-time collaborative designer tools. Support for 3D art packages.	
2D Graphics Engines		
IvanK Lib	Renders 2D graphics with GPU accelerated WebGL	Doesn't work in iOS.
2D Game Engines		
GameMaker HTML5	Commercial 2D game engine	
Crafty.js	2D HTML5 Game Engine, complete and flexible engine (collision detection, renders with canvas or DOM, sprite map support, event system, input etc...).	Active forum, monthly updates.
Construct 2	Drag and drop solution. rendering and physics. complete with tools.	Frequent updates.
Impact (with ejecta)	2D JavaScript Game Engine, Flexible Level Editor, Canvas & Audio Implementation for iOS	
Irenic	General purpose Javascript engine both for the server side (Node.js) and the client side (HTML5).	
Cocos2D	Cocos2d-javascript is a 2D game/graphics engine based on cocos2d-iphone but designed to run in the web browser. It uses the latest features available in HTML 5 allowing real-time rendering of 2D graphics without the need for plug-ins such as Adobe Flash.	
GammaJS	Gamma is a new Javascript library which can be used to create 2.5D	

	platform games for a web browser using the power of HTML, JavaScript, CSS and WebGL.	
Gamvas Web	Javascript / HTML5 Game Framework with Box2D physics integration.	
Jaws	Jaws is a HTML5 2D game development library written in javascript. Sprite, viewport, gameLoop, input handling etc. The core to create a 2D game. Comes with examples.	
Jest	Jest provides a way to rapidly create JavaScript (HTML5) games using the canvas element. Includes a few samples currently with more on the way.	
Pulse	Focuses on high performance 2D games using HTML5 canvas.	
Traffic Cone	Traffic Cone is a 2D and isometric tile based game engine written for html5. It makes complex animations of sprites and tiles based worlds fairly simple.	
Geom	Javascript / HTML5 Game and Content Engine with Pro Editor.	
Impact	Javascript / HTML5 Game Engine with world editor.	
Isogenic Engine	HTML5 Isometric & 2D Game Engine. Emphasis on real-time persistent worlds, MMO games & plugin architecture, micro-transactions, cloud-based hosting. Realtime networking. Support for canvas and HTML-based output, Node.js + MongoDB. Physics support using Box2D.	
Playcraft Engine	Large-scale, professional HTML5 game engine featuring: multiplayer, tiling, physics, sprites/animation, sound, input handling, entity systems, native acceleration wrapper (allows HTML5 games to be run as native apps with performance acceleration), scenes/layers and AI.	
LimeJS	Special attention to touchscreen devices. HTML5 game framework for fast, native-experience games for all modern touchscreens and desktop browsers.	
2D/3D Game Engines		
Turbulenz	Turbulenz offers the ability to build, publish and monetise high-quality games that react like no others, with immersive 2D and 3D effects and real-time physics that open up a whole new world of unprecedented and extraordinary web content.	

lycheeJS	JavaScript, HTML5 and native OpenGL (DOM, canvas, WebGL and native OpenGLES with GLU and GLUT). Independent of the environment, runs in V8GL, NodeJS and even in Spidermonkey (setTimeout/setInterval required, nothing more for server-side).	
2D Physics Engine		
Box2D html5	Html5 implementation of well documented, well used physics engine.	
Other		
clay.io	Service for user account and social features (achievements, leaderboards, In-game purchases, multiplayer rooms,... etc)	Commercial. Users can log in using google, Twitter, or facebook account.
Complete solutions		
LibGDX	Android/HTML5/desktop game development framework, 2D,3D,input,audio,physics(box2D),tools(particle editor, texture packer, ...)	Supports Windows, Linux, Mac OS X, Android and HTML5.
Marmalade		As of 2012, the multi-platform SDK Marmalade has expanded its support to HTML5.
API's		
Geolocation(GPS)	Wireless Geolocation(html5 geolocation api) vs. IP Geolocation. Used for locating devices almost anywhere in the world.	The wireless Geolocation with html5 can be more accurate then IP Geolocation but may not always be able to detect all locations, At this time not all browsers support it either.
Orientation API(accelerometer)	Used to determine how the device is rotated in the world.	
WebGL(GPU)	Interface to make 3D in browsers.	
Web Audio API(audio hardware)	Recording and playing sounds.	
HTML5 Camera API	For taking pictures, filming.	
Sources: github.com, gamemiddleware.org, and others, as accessed on 2012-11-19 and 20.		

-----END OF DOCUMENT-----