Research in the Large: Challenges for Large-Scale Mobile Application research - A Case Study about NFC Adoption using Gamification via an App Store

Matthias Kranz, Lukas Murmann, Florian Michahelles

The adoption of NFC technology has taken longer than expected after its inception in 2004. Several projects on ticketing and payment are gaining momentum. However, the actual state of adoption of NFC is still unclear. As an alternative to consultants' prediction (which mostly prove wrong), this paper describes a gamification-based approach to motivate users themselves to report on NFC tags they spotted in their environment. As part of a trading card context users get rewarded with gadgets and points for documenting the existence of NFC technology in their environment. This paper describes the development of this game and the experience of two release cycles. The paper concludes with lessons learned and provides an outlook on next steps.

Research in the Large: Challenges for Large-Scale Mobile Application research - A Case Study about NFC Adoption using Gamification via an App Store

Matthias Kranz (matthias.kranz@ltu.se)[1], Lukas Murmann (lukas.murmann@tum.de)[2], Florian Michahelles (fmichahelles@ethz.ch)[3]

[1] Luleå University of Technology, Department of Computer Science, Electrical and Space Engineering, Luleå, Sweden
[2] Technische Universität München, Munich, Germany and University College London, United Kingdom
[3] ETH Zurich, Auto-ID Labs, Zurich, Switzerland

Pre-Print – to appear in: Intl. Journal of Mobile Human-Computer Interaction (IJMHCI)
Preliminary BibTex available at: https://vmi.lmt.ei.tum.de/publications/bibtex/114.bib

Table of Contents

Research in the Large: Challenges for Large-Scale Mobile Application research - A Case Study about NFC Adoption using Gamification via an App Store

Abstract:
The adoption of NFC technology has taken longer than expected after its inception in 2004. Several projects on ticketing and payment are gaining momentum. However, the actual state of adoption of NFC is still unclear. As an alternative to consultants' prediction (which mostly prove wrong), this paper describes a gamification-based approach to motivate users themselves to report on NFC tags they spotted in their environment. As part of a trading card context users get rewarded with gadgets and points for documenting the existence of NFC technology in their environment.
This paper describes the development of this game and the experience of two release cycles. The paper concludes with lessons learned and provides an outlook on next steps.

1. Introduction and Motivation

In 2004, Nokia, Philips and Sony jointly proposed a wireless communication standard, Near Field Communication (NFC), to establish communication among mobile device by touching or bringing them in close proximity of less than a few centimeters.

As this proposed NFC standard builds upon established standards of wireless smartcards and Radio Frequency Identification (RFID) technology, NFC enabled devices to act both as a smartcard or RFID tag as well as reading smartcards and RFID tags. Thus, NFC enabled mobile phones to connect to a completely new group of users to everyday items and, as such, to extend the role of mobile phones by new forms of communication between people and objects.

NFC technology operates at 13.56 MHz (HF Frequency) and comprises two standards. First, the NFCIP-1 standard (approved as ISO/IEC 18092) specifies the air interface and transmission protocol for NFC devices. This standard allows reading wireless smartcards (ISO/IEC 14443). Second, NFCIP-2 (approved as ISO/IEC 21481) enables access to read RFID tags (ISO/IEC 15693) and determines which communication standards are going to be used at the beginning of a communication session.

Since its inception contactless payment transactions, electronic ticketing, data exchange, and simplified setup of more complex communications such as Wi-Fi or Bluetooth have been proposed. Just recently Google Wallet has been launched in the US as a first NFC-based implementation allowing consumers to store credit card and store loyalty card information in a virtual wallet and then use an NFC-enabled device at terminals that also accept wireless smart card transactions. Several local transport companies have trialed NFC ticketing systems for public transport. However, the predicted break-through of NFC technology has not happened yet.

Despite the huge variety of applications based on wireless payment, couponing, ticketing, linking of information to places and things, handset manufacturers have been slow and reluctant to integrate NFC technology into their devices. From the first phone ever produced in serial production with NFC in 2004 (with the Nokia 6131) it took until 2010 when the first smartphone was released with NFC (e.g. the Samsung Nexus S).

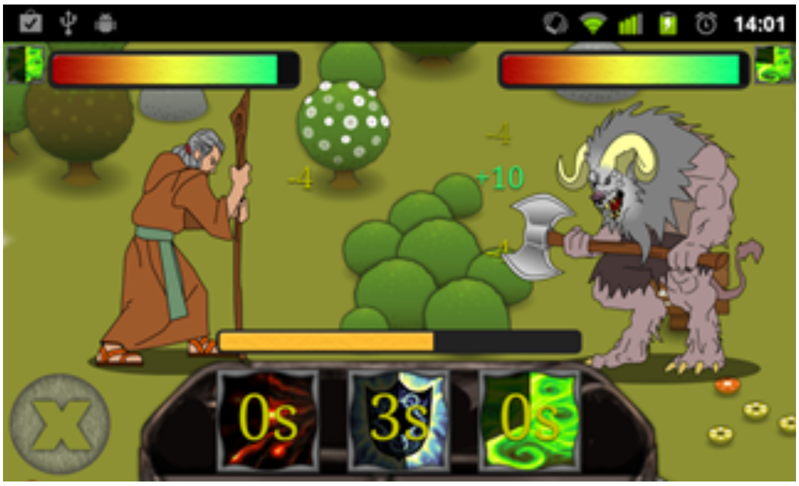
Finally, a larger number[1] of phones from various manufacturers have become available which could trigger the adoption of this technology. NFC is getting used in a wider range of applications, from gaming consoles like the upcoming *Wii U* (Engadet, 2012) to payment solutions. Recent versions of the Android SDK and Windows 8 provide libraries that allow easy access to the underlying NFC hardware.

Albeit several approaches from mobile phone and computing equipment manufacturers to push this technology into the market and numerous scientific publications on the technology and potentials for advanced human-computer interaction, the success so far has only been limited in the year 2012. Furthermore, there is still doubt on the potential of NFC to actually be widely applied – Apple therefore refrained from integrating NFC in their current iPhone 5 mobile phone released in September 2012.

With our research, we want to investigate further on the potentials of the NFC technology, and for those having an NFC-enabled mobile device, one question still remains: 'Where are the NFC tags and what can I do with those?' As there is no central registry or database of NFC applications or tags, the goal of this project is to capture the current state of deployment of NFC solutions using a crowd-sourced approach. We want to motivate users to collect locations, uses, and picture NFC tags deployed in the real world in order to capture the current stage of adoption and deployment. With this we want to provide an alternative to predictions, expert views and opinions, but rather base maturity of NFC technology on facts and users' experiences.

Figure 1 The player has selected three spells and fights a randomized opponent.

Filename: game.tif



To learn about the situation 'in the wild', we have released a research app to the public via the *Play Store* as a proof-of-concept for capturing the state of NFC deployment by a gamified approach. In the following we describe the rationale, design, and release of NFC Heroes'[2] that makes use of the platform's NFC capabilities and gives users in-game incentives to scan and upload information about deployed NFC tags.

[1] http://www.nfcworld.com/nfc-phones-list/

[2] https://play.google.com/store/apps/details?id=com.heroesgame (Möller, et al., 2012)

We present the process of publishing the game on Google's *Play Store* and how we integrated Facebook as an identity provider. Our goal is to bring a research application to a consumer platform. We share the lessons we learned during that process, both in terms of direct user feedback and number of users our game did attract. We also report on our (Cramer, Rost, Belloni, Bentley, & Chincholle, 2010) insights gained from maintaining and updating app deployed using app stores. We conclude by summarizing our findings and experiences.

2. Related Work

We discuss related work with a general focus on large-scale application deployments for user studies, with a focus on security, and with respect to the specific study scenario we have chosen (NFC-based gaming).

2.1. Research in the Large – Research in the Wild

Researchers have only recently begun to exploit the new opportunities for research in the large. Technological development, widespread adoption, and technology proliferation in the mass and consumer market and decreasing cost now allow employing commercial systems to conduct research beyond the limitation of the lab, out in the wild. Examples for these recent developments are the availability of embedded networked systems allowing to interact with the so-called 'Internet of Things' (Kranz, Holleis, & Schmidt, 2010), cloud computing – and app stores. Considering these as tools, such as Amazon's Elastic Compute Cloud (Amazon EC2), researchers are extending their ability to approach scientific problems in a fundamentally different way – removing some of the final limitations: scale.

The idea we follow here to use app stores and markets for UbiComp research has been discussed by Cramer et al. (Cramer, Rost, Belloni, Bentley, & Chincholle, 2010). With the advent of so-called 'app stores' for modern smartphones, it is now possible to conduct 'research in the large'. While tools like SurveyMonkey[3], MobileWorks[4], or Amazon's Mechanical Turk[5] allow asking many users (study participants), it is not possible to study actually the effects of the real system (nor in a real context). Therefore, extending, complementing or even substituting these questionnaire tools with users that experienced the app (without possibly being to much aware of the scientific nature or research questions) can provide valuable additional insights.

The difference to prior studies conducted 'in the wild' is significant: more users, more devices, more contexts, and more diversity! No longer are we constrained anymore by our owns lab setup, infrastructure, or biases and can pursue our goal to justify and proof our interaction design, application architecture, interaction metaphor or research methodology with a 'real' user study. We, by the utilization of app stores, no longer need to ask our colleagues or students, do not have to conduct the study in our own institutions' laboratory or potentially introduce biases by our behavior. As Henze (Henze, Hit it!: an apparatus for upscaling mobile HCI studies, 2012) describes the problem of lab-internal studies: 'Such common studies can have a high internal validity but often lack external validity. The findings cannot always be generalized to the

[3] http://www.surveymonkey.com
[4] https://www.mobileworks.com
[5] http://www.mturk.com

behavior of real users in real contexts. In contrast, researchers recently started to use apps as an apparatus for mobile HCI research.' It is also theme and motivation for new conferences and workshop series (e.g. the LARGE series[6]) to investigate these new potentials – and pitfalls. Leaving our safe ground and going to release our apps in the wild allows us to study app usage in context – with all benefits and problems associated to it. We report in this work on our experience and summarize them as lessons learned at the end of this article.

We today find a focus on mobile phones as – so far – the only true ubiquitous interaction devices seems, giving the sheer numbers of devices and users (and the fact, that many users already have several personal portable devices, including smartphones and tablets). Given the steadily increasing number of embedded sensors of these devices, more and more options arise. Lane et al. provide a comprising survey on the potentials of mobile phone sensing (Lane, Miluzzo, Hong, Peebles, Choudhury, & Campbell, 2010). Henze et al. (Henze, Pielot, Poppinga, Schinke, & Boll, 2011) report on several of their experiments involving the publishing of research-related apps for the Android platform and identify success factors and potential pitfalls. Their individual apps allowed them to identify specialties of app store-based deployments, such as short usage times, the need for collecting additional qualitative feedback e.g. through user comments and email and the lack of a representative sample of users – despite users from the whole world could participate by using the apps. Ferreira et al. (Ferreira, Kostakos, & Dey, 2012) complement these experiences by reporting on biases and side effects of the usage of app stores might introduce. They also discuss findings with respect to e.g. the recruitment and distribution.

2.2. Security-related Issues for Research in the Large

While inclusion in the Apple *App Store* requires a review process[7], Google Play is free of constraints for uploading apps. However, apps are scanned for viruses and malware (Lockheimer, 2012) and in case of malicious content deleted. This is, however, just a method to uncover software that obviously tries to do 'evil' things, but not to detect programming bugs or security holes.

Automatic analysis of security problems during the submission process to digital market places has been proposed using several approaches (Gilbert, Chun, Cox, & Jung, 2011) (Shabtai, Kanonov, Elovici, Glezer, & Weiss, 2012). Di Cerbo et al. (Di Cerbo, Giradello, Michahelles, & Voronkova, 2010) present a methodology for mobile forensics analysis to detect 'malicious' (or 'malware') applications. The methodology relies on the comparison of the Android security permission of each application with a set of reference models, for applications that manage sensitive data. Thus, this research is focusing more on protecting the user from malicious apps whereas our paper focuses on capturing the (non-) compliances of users to install fixes of a trusted developer.

It has also been found that Android apps often require permissions that are actually unneeded. Extensions to Android's permission model have consequently been proposed which focus particularly on improving the (initially quite coarse) granularity of permissions (Nauman, Khan, & Zhang, 2010) (Vidas, Christin, & Cranor, 2011) or

[6] Workshop on Research in the Large – App Stores, Wide Distribution, and Big Data in MobileHCI Research, http://large.mobilelifecentre.org

[7] https://developer.apple.com/appstore/guidelines.html

remove them in hindsight by inline reference monitoring, e.g. as done by AppGuard[8]. Fewer rights inherently also decrease the probability for security-relevant bugs.

Miluzzo et al. (Miluzzo, Lane, Lu, & Campell, 2010) looked at implications and challenges of large-scale distribution of research apps through the Apple *App Store*. They pointed out that insufficient software robustness and poor usability may lead to a loss of confidence on the part of the users, but did not quantitatively examine this phenomenon (such as the number of uninstalls due to dissatisfaction). AppTicker[9] is a project that allows monitoring mobile app usage, (un-) installation and more to gain information about usage patterns on smartphones. To our knowledge, the particular phenomenon of update behavior in app stores has not been examined yet. Despite the security approaches and measures we presented in this section, keeping the software up to date remains the central requirement for a stable and secure system.

2.3. Barcodes, Visual Codes and NFC/RFID-based Mobile Gaming

Gaming systems have been integrating physical or virtual tag readers since the early 90's. As cameras and tag readers are now ubiquitously available in smartphones, developers finally start implementing many of the concepts known from previously dedicated gaming consoles on mobile devices. At the same time, HCI researchers develop games to evaluate new interaction methods made possible by NFC sensors or other sensing technologies, such as accelerometers (Möller, et al., 2012) or capacitive sensors (Wimmer, Holleis, Kranz, & Schmidt, 2006), incorporated in pervasive mobile devices.

Visual markers have been employed in many mobile game researches. Markers, such as black and white 1D bar codes (e.g. known from product labels) or 2D markers (e.g. such as Quick Response (QR) codes), enable game designers to extend and connect the virtual game to the real world. Rohs (Rohs, 2007) extends the concept of hyperlinking physical and digital world by the inclusion of spatiality and exploiting the physical relation between mobile device and marker as additional input parameter. Lam et al. (Lam, Chow, Yau, & Lyu, 2006) use the game concept of a trading card game, providing a virtual table as context, utilizing physical cards and extending them with a virtual 3D environment.

In the early 90's, the *Barcode Battler*[10] handheld devices were released in Japan and later also in Europe and the US. Players could swipe special cards with barcodes to unlock items in the game. The first *Barcode Battler* was a stand-alone console, but the *Barcode Battler 2* could also be connected to the NES and SNES gaming consoles[11].

Nintendo pursued the idea of using real-world, physical cards to influence game events further. In 2001, they released the *e-Reader*[12], an accessory to the Gameboy Advance that could read proprietary visual codes.

[8] http://www.backes-srt.de/produkte/srt-appguard

[9] http://projects.hcilab.org/appticker/

[10] http://en.wikipedia.org/wiki/Barcode_Battler

[11] http://barcodebattler.co.uk

[12] http://www.nintendo.com/consumer/downloads/ereader_english.pdf

Recent games do not rely on dedicated hardware to read barcodes, but make use of the camera integrated into modern smartphones. In *Barcode Empire* (Budde & Michahelles, 2010; Böhmer, Hecht, Schöning, Krüger, & Bauer, 2011), players can collect real-world products in order to expand their 'Empire'; *Barcode Beasties*[13] is a fighting game that lets players improve their avatar (beast) by scanning barcodes before they battle against a randomized opponent.

The Mattel *Hyperscan*[14] released in 2006 was a gaming console featuring an NFC reader that could read game-specific NFC cards. The cards were sold in separate booster packs, very much like traditional trading cards.

Murmann et al. (Murmann, Michahelles, & Kranz, 2012) used a gamification approach to encourage users to discover NFC tags distributed in the real world. Their app was distributed via Google's *Play Store*. The goal was to learn about the numbers, distribution and locations of tags in the wild. They offered the incentive to obtain virtual game items after the scanning and discovery of novel tags.

Pellerin et al. (Pellerin, Yan, Cordry, & Gressier-Soudan, 2009) used NFC tags to provide adaptable and personal content in multiplayer games to combine physical and virtual game elements. Nokia (Nokia , 2012) also follows the approach of the introduction of tangible elements in mobile gaming. With their game 'Shakespeare Shuffle' users can read NFC tags by wiping their phone over the tags to listen 'magically' to quotes from Shakespeare. The task is to physically rearrange the tags (quotes) in the correct order so they form the full quote. Other comparable games in beta and experimental state (as of 2012) are 'World Flags' or 'Nursery Rhyme Shuffle'. Due to the innovative nature of mobile physical gaming, Nokia states: 'Since these games offer a radically new kind of mobile gaming experience, your feedback would be very helpful to us'. In contrast to the approach of gathering feedback from a wide audience by distributing the game via a comparable app store, Nokia requires a registration to access these games.

Broll et al. experimented with NFC-based games on public displays (Broll, Graebsch, Scherr, Boring, Holleis, & Wagner, 2011). Nokia Research launched a website dedicated to NFC-based games (Nokia Research Center Palo Alto, 2011). At the time of this writing, three games are featured. With the *Wii U*, Nintendo will allow mobile games to interface with real-world objects through an NFC reader in the console's controller (Engadet, 2012).

3. Study: A NFC App Game in the Wild

3.1. The App NFC Heroes: Concept and Core Game Design

NFC Heroes is a virtual trading card game for Android phones, slightly inspired by the *Magic: The Gathering*[15] trading card game. Users can scan NFC tags to unlock more powerful spells or heroes in the game. The spells can then be used to fight against monsters, collect coins, and compete against other players on a leaderboard. The integration with Facebook lets players share their victories and collected cards.

[13] http://barcodebeasties.com

[14] http://service.mattel.com/instruction_sheets/k4386-0920.pdf

[15] http://en.wikipedia.org/wiki/Magic_the_gathering

NFC Heroes is a fast-paced fighting game where a computer-controlled monster competes against a hero controlled by the player (see Figure 1). The player must choose a hero and can then set three spells from his card deck to be active in the game. There are a variety of different spell types available: Players can optimize their selection of shield, offensive, and healing spells and whenever they unlock a new spell, it might be necessary to adjust the set of active cards in order to make room for the new spell. This cycle of incremental improvements is intended to motivate the user and the tradeoffs between the different spells add tactical depth to the game.

3.2. Installation and First Start

To reach a large number of players for our initial studies, the game was made available on Google's *Play Store*. As most users are unaware of the game's purpose as a research project and expect the same level of visual quality than from any other free game offered in the smartphone's application store, particular attention was given to the design of promotion graphics and in-game screenshots.

When users first start the game, they are asked for a name or alias to appear on the game's leaderboard. They can now start playing with an account tied to their smartphone. Alternatively, they may choose to link their game progress to a Facebook account and will then be able to continue playing on other devices. The two authentication methods were chosen to pose the lowest possible barrier of entry. In neither of the methods are users required to enter account information or passwords. When they choose to start playing without Facebook, a unique ID is stored on the device and will subsequently be used for authentication. When they authenticate through Facebook, the authentication steps are delegated to the *Facebook for Android* [16]application. A local account can be upgraded to a linked account at any later point.

3.3. Using NFC to unlock new Spells and Heroes

After logging in, users can start fighting monsters, climb up the leaderboard, and share their progress on Facebook. Ultimately however, they will want to use their NFC-enabled phone and scan NFC tags, which will reward them with more powerful spells, and rarely an additional hero.

Once users touch an NFC tag with their smartphone, the tag's unique ID, manufacturer, and standard compliance is uploaded to the *NFC Heroes* server and added to the user's tag collection (see Figure 2 for an example). The ID is used as a seed for the random card generator algorithm. The algorithm tries to generate more powerful cards for rare NFC tags in order to incentivize users to look for tags even in unlikely locations.

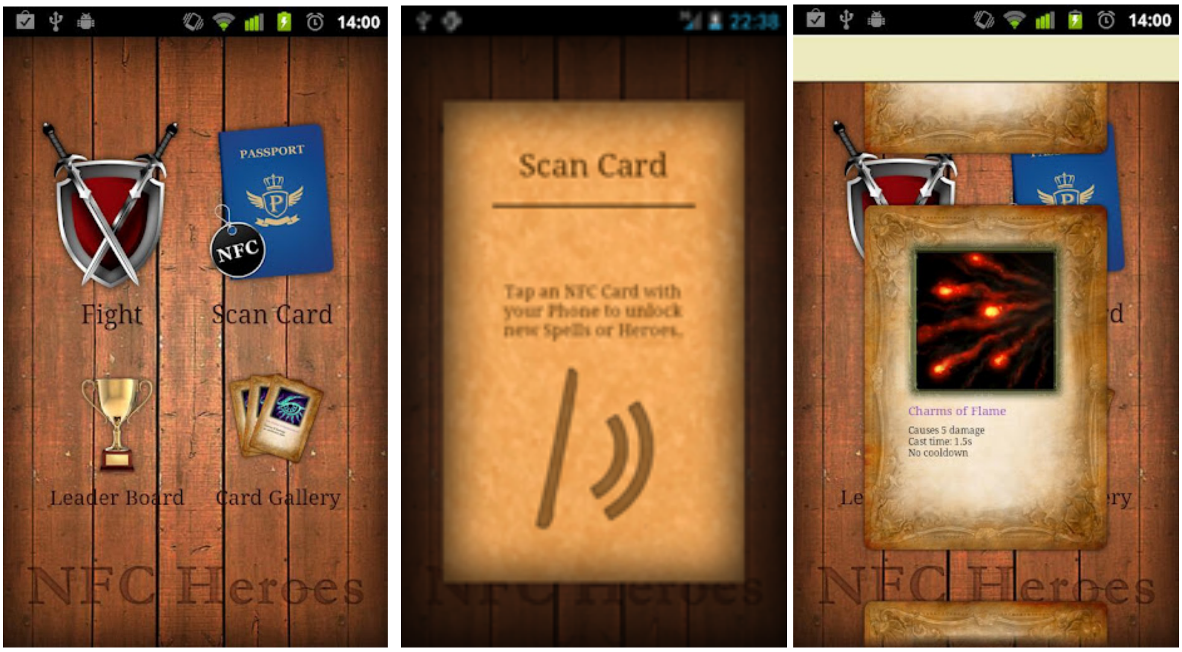
Back on the phone, the generated card is shown to the users who can now optionally upload a photo and description of the tag they just scanned.

Finally, they are offered to share the new addition to their card deck with friends on Facebook. In case they did not link their account with Facebook yet, they can choose to do so now.

[16] https://play.google.com/store/apps/details?id=com.facebook.katana

Figure 2 The user selected `scan card' from the main activity, scans the card, and received three spells and fights a randomized opponent.

Images from left to right: startscreen.tif, scancard.tif, scancard2.tif



Tap an NFC Card with your Phone to unlock new Spells or Heroes.

Charms of Flame

Causes 5 damage
Cast time: 1.5s
No cooldown

3.4. Progress and Leaderboard

Games that aim to provide long-term motivation to players must provide ways for players to progress in the game (Zichermann). *NFC Heroes* provides two ways how player can measure their progress: first, they can collect more powerful spells and heroes, similar to a role-playing game.

Second, we added a more immediate and visible progress indicator: For every defeated monster, a player will be awarded a number of coins proportional to the strength of the opponent. At the same time, the more coins a player collects, the harder the randomly generated opponents will become. A player's amount of earned coins can be shared with friends on Facebook and is shown on an in-game leaderboard.

3.5. Implementation and Technologies

As NFC is the focus of our research, Android was the only viable mobile platform for our game. For the implementation of the web server, we used a setup consisting of *Node.js*[17] for our application logic and MongoDB as a non-relational database. The game architecture is based on the client-server model, with the client app running on the personal portable device and the server running on an internet-connected backend system.

NFC Heroes supports Android Devices running on Android 2.3 or higher and thus more than 76.6% of all devices that were active in July 2012[18]. All Android phones with NFC support (Android version 2.3 or higher) and are thus supported by our game.

[17] http://nodejs.org

[18] http://developer.android.com/resources/dashboard/platform-versions.html

The Facebook SDK[19] was used to facilitate the integration of social features and the use of Facebook as an identity provider. We further used Google Analytics to gather information beyond our server logs and the data that is available from Google's *Play Store*.

The *NFC Heroes* server was written in JavaScript using the *Node.js* platform. All communication between client and server is secured by TLS encryption. Node.js is a rather young technology, but it is easy to learn and allowed us to develop the server component in very little time. Its event-based IO system is particularly suited for real-time application like games and allows developers to handle HTTP requests, as well as socket-based communication in the same process.

Data about scanned tags and user progress is stored in a *MongoDB* database. Just as *Node.js*, *MongoDB* was chosen because of its ease of use and short development cycles. There further exist good support libraries for using *MongoDB* from *Node.js* and an active developer community provides documentation and example code.

3.6. Study Wrap Up

As illustrated by Figure 5, the number of active *NFC Heroes* installations is growing at a constant rate. Still, the goal of the project, to create an engaging game with a significant number of users that will help create a database of NFC-enabled products, has not been achieved yet.

As of this writing, 180 NFC tags (including duplicate tags with IDs already known to the server) have been uploaded and for 40 tags an additional photo or description was provided. Users have fought a total number of 706 battles; the most active day was on May 28 2012 with a total of 54 fights on a single day.

We still find ourselves early in the life cycle of the game, and Android phones with NFC support are only starting to gain traction. Still, initial reactions on version 1.0 have shown good receptions among interested users who have spent a significant amount of time playing the game. We reflect on the implications and findings of this research and present our lessons learned at the end of this article.

4. Study: Updating Apps in the Large

4.1. Motivation

Platform-specific marketplaces, such as the Apple *App Store* or *Google Play* (formerly *Android Market*), are nowadays an important source for mobile app distribution (Research and Markets, 2011). In March 2012, Apple reached in total 25 billion iOS app downloads[20]. Until 2011, 10 billion Android apps have been downloaded in total over *Google Play*[21]. Smartphone users find their applications bundled at one place and are informed about available updates (via a badge symbol on the *App Store* icon on iOS, or a message in the notification bar on Android). However, neither on iOS or Android,

[19] https://github.com/facebook/facebook-android-sdk

[20] http://www.apple.com/pr/library/2012/03/05Apples-App-Store-Downloads-Top-25-Billion.html

[21] http://www.wired.com/gadgetlab/2011/12/10-billion-appsdetailed/

application updates are installed automatically. Android has a setting for installing updates without confirmation, but it is disabled by default.

This update mechanism implementation can be seen as a potential risk for security. Unfixed security holes increase the vulnerability of a device. As users need to take charge of keeping their system up to date themselves, important updates might not be installed timely or at all. Especially for research apps, e.g. (Möller, et al., 2012) (Möller, et al., 2011), or at the beginning of an app's market lifetime, regular installation of updates is important. Being in state of development, such apps often are less stable and require more frequent fixes. Until the end of 2011, more than 20,000 new apps per month were published in Google Play[22], so that potentially a large number of apps are affected by this phenomenon. Security flaws become even more severe for the novel and upcoming category of apps that integrate with the home or automobile (so-called in-car apps, see e.g. (Diewald, Möller, Roalter, & Kranz, 2011)), since in that case not only the app itself, but also the connected property becomes insecure. In a case study, we observed users' update behavior of an Android app we have placed in *Google Play*. We gained insights on the correlation between published updates and their actual installation (Möller, Michahelles, Diewald, Roalter, & Kranz, 2012) and discuss the consequences and recommended actions on the part of the developers.

4.2. The App VMI Mensa

For our case study, we are looking at *VMI Mensa*[23], an Android application developed by the research group of the authors of this paper for our university's students. *VMI Mensa* shows meals and prices of cafeterias and canteens of university campuses in our city. The application, targeted at students and university employees, has been available in *Google Play* since July 21, 2011 and meanwhile (as of July 2012) reached 2,294 downloads in total. It has received 123 ratings (averagely rated with 4.8 out of 5 stars) and 40 user comments. Since its launch, the app has continuously been extended in its functionality, e.g. by a location-aware canteen finder, details on ingredients, accessibility information (e.g. on elevators), and much more.

4.3. App Update Installations Analysis

Since *VMI Mensa* was first available in Google Play, we have shipped 21 updates. For our analysis, we used the built-in statistics tools of the Android Developer Console in *Google Play*. They allow keeping track of the number of installations over time, monitor installed app versions and a lot more. All data is anonymous and cannot be related with individual users. As stated before, updates may install automatically or manually by user confirmation. We cannot track whether automatic update installation was enabled on users' devices.

For our analysis, we looked at the latest five updates, published at December 22, 2011, January 17, January 26, February 24 and April 02 (all 2012). The average time between updates was 26 days, which we consider not as an unreasonable effort for users to regularly install them. All updates added new functionality to the app and/or fixed small problems, but none were critical for security. For each update, we observed how many users downloaded the update on the initial day of publishing and in the 6 consecutive

[22] h ttp://www.androlib.com/appstats.aspx

[23] https://play.google.com/store/apps/details?id=de.tum.ei.lmt.vmi.mensa

days. We calculated the update installation ratio by relating the download count to the total count of active device installations on the respective days.

In addition to the anonymous update installation statistics, we considered available user communication in form of feedback emails, comments and ratings in *Google Play* for our analysis. We will bring in these findings in the discussion section.

4.4. Results of the Analysis of the Update Behavior

In the following, we describe and visualize the quantitative results of our case study. Update Behavior Table 1 shows the installation percentages on the update publishing day (day 0) and the six consecutive days (day 1 to day 6), averaged over all five updates that were considered in this study. The exact ratios are very similar for all updates, which is implied by the low standard deviations (see last column of the table). In average, 17.0% installed the update on day 0. On the following days, the numbers continuously and exponentially decrease: 14.6% installed the update on day 1, only 7.8% on day 2, and 5.1% on day 3. On day 6, only another 2.3% downloaded the update.

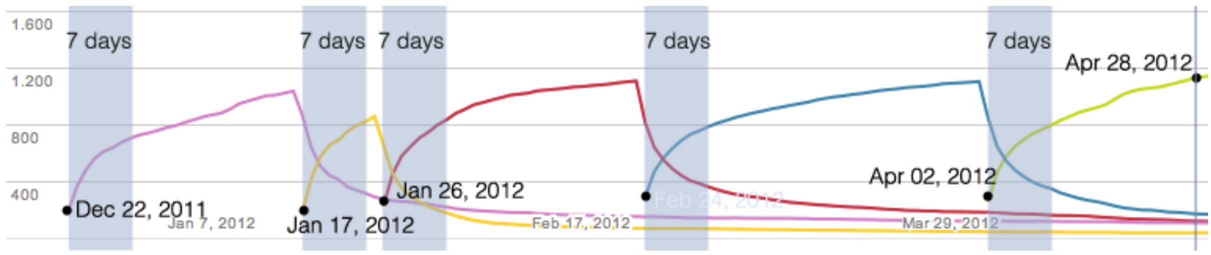
Day after Update	Update Installed	Standard Deviation
Publishing Day	17.0%	2.7%
Day 1	14.6%	2.0%
Day 2	7.8%	1.3%
Day 3	5.1%	0.9%
Day 4	3.5%	0.7%
Day 5	2.8%	0.5%
Day 6	2.3%	0.4%
Total in 7 days	53.2%	2.7%

Table 1: Percentage of all users who installed an update within 7 days after it was published. Only slightly more than half of all users installed a recent update within one week. Data was averaged based on five subsequent updates published within 102 days. Standard deviation is related to the five individual updates we observed in our use case.

This trend is visualized in Figure 4 and can be summarized as follows: Most of those users who actually do install updates install them quickly. We hypothesize that the relatively high ratios of the first two days might partly be due to the automatic update option. Users that did not install the update early are also not likely to do so in the subsequent days. In total, just 53.2%, slightly more than a half, had the most recent update installed one week after publication.

Figure 3 Visualization representing the number of installations by version (vertical axis); the colored lines indicate the five latest versions. The diagram reveals how long old versions are active on user's devices. The 7-day periods after an update has been published are highlighted. Modified diagram based on Android Developer Console statistics.

Image: installs_by_version.tif

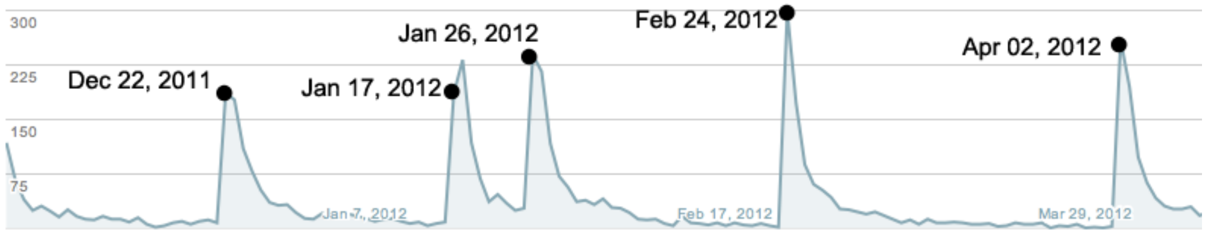


4.5. App Version Distribution

We also looked at the distribution of the latest five versions of the app on users' devices, illustrated by different colors in Figure 3. The seven-day periods after an update has been published are slightly shaded for illustration. The visualization shows the spread of new versions due to cumulative installs (visualized with a steep graph that flattens out more and more), and the decrease of older versions. It also becomes evident how long outdated versions (up to four versions older than the latest one) are still circulating. As an example, we look at April 28, 2012, which is two weeks after the latest update has been published: Only 56.4% of all users have installed the latest version (v.27) at this time. The previous four versions were still in use by 8.5% (v.26), 6.0% (v.25), 5.5% (v.24) and 2.1% (v.23). Most severely, 21.5% had even older versions installed on their devices at that time.

Figure 4 Visualization representing the number of five subsequent update downloads (vertical axis) over time. The graph shows maxima on the update publishing day (possibly also due to activated auto-updates) and exponentially decreases thereafter. Modified diagram based on Android Developer Console statistics.

Image:updates.tif



4.6. Discussion of the App Update Behavior

Results from our case study reveal a problematic update behavior: Even one week after their publication, updates were installed only by about 50% of users. The rest used different outdated versions; one fifth even did not install even one of the last five updates. This implies two potential groups of users: those who update in an exemplary manner, and those who barely update at all. Hence, developers must not make the mistake to rely on the belief that at least the penultimate version of their app would run on most devices.

If we project this result to general update behavior, our findings imply a critical security situation. The harmless feature updates in our case study could be important security-related fixes in another app. On average, almost half of all users would use a vulnerable app version even 7 days after the fix has been published. The time from detection of a security hole to the final update shipment is not even considered here. Further reasons indicate that the 'real' update situation could even be worse than in our exemplary case analysis. A high number of installed apps could further decrease the amount of up-to-date apps, since more time would be required for individual updates. Furthermore, the fact that users are presumably highly engaged with our examined canteen app could have an impact on update frequency as well. We see an even more critical situation with apps that are not regularly used, but for which security is crucial just then (e.g. for online banking apps). In-depth usage monitoring (Böhmer, Hecht, Schöning, Krüger, & Bauer, 2011) is required for better understanding the relation between usage frequency and update behavior.

We also looked at users' behavior in case of problems. Our app contained a 'Give feedback' item in the preferences menu that allowed sending an email to the developers. In the app description in *Google Play*, we asked users to give us feedback using this function. We also linked to a Q&A page from which users could contact the developers as well. Our experience revealed that few users actually used these opportunities.
They rather made use of the rating functionality in *Google Play*. For example, the download of the daily menu was not working for one day due to a server migration. Several users immediately left a bad rating in *Google Play*, complaining about the app not working any more. Apparently, they had not read the requests to provide feedback per mail or not found the feedback link in the app. A similar case illustrates as well that not all users read the description texts in *Google Play*: One user commented that it would be good to have an English translation. In fact, the app is fully localized to 6 languages (amongst them English), and localizations automatically adapt to the device's system language. Similarly, this user rated the app worse because of this complaint. For developers, our observations have three consequences.

First, they show how quick users are with bad ratings, which may be problematic especially for commercial apps – other work already stated that user reviews could be brutal (Miluzzo, Lane, Lu, & Campell, 2010). Hence, it is important to keep the application bug-free and provide timely updates in case of problems.

Second, developers cannot rely on users reading instructions and employing the built-in feedback functions. We gained the insight that ways to further improve such functions should be found, and we also learned that keeping track of ratings and comments in *Google Play* is important. Otherwise, in some cases, we would not have been aware of potential problems. In our case, they were related to usability and minor issues, but they could have been security bugs as well. This is especially important since security holes not necessarily go along with unresponsive or crashing apps and thus are not covered by the built-in error reporting function of Google Play.

Third, as a first step towards an improved security on mobile phone platforms and in light of sometimes difficult download mechanisms (Cramer, Rost, Belloni, Bentley, & Chincholle, 2010), we encourage developers to support users in updating, e.g. by built-in update checks within their application and/or forwarding users to the platform market place, as we use it in our research apps (Möller, et al., 2011).

4.7. Study Wrap Up

In this study, we have analyzed update behavior and security implications in application markets at the example of an Android application we developed and offer for download in *Google Play*. We found that, in average, half of all users did not install an update even seven days after it has been published and thus would use a potentially vulnerable application. Although generalizations of our initial findings must be carried out carefully and further studies will be necessary, we raised the awareness for potentially slow update propagation on Android (until Version 4, after that an option to 'auto-update apps' exists – but is disabled by default) and other mobile platforms.

Further automatic quality assessments for uploaded apps in digital market places and more automated update mechanisms could be ways to increase the level of security on mobile devices.

5. Lessons Learned

We take some key learning about the release of these research projects on Google's *Play Store* with us.

5.1. App Stores make short development cycles possible

We learned that Google's *Play Store* allows researchers to release applications in an early state and to get immediate feedback from actual users.

We split the development phase of 9 weeks into two iterations. A preview version of *NFC Heroes* was released after only five weeks. This allowed us to apply an iterative user-centered development process: we were able to take user feedback into account while we were still implementing the remaining features.

5.2. An early release can give guidance in the design process, but may cause mediocre first reviews

In our case, the preview version consisted of just the features identified by us as key features for playing the game, so that we could evaluate feedback relating to the core game mechanics. In the preview, the player started out with a fixed set of three spells and two predefined heroes. Neither Facebook integration, nor the leaderboard where players can compare their progress was implemented in this version. We were curious how many players would actually download what we announced as 'Gameplay Preview' and how the initial reviews on the store would be.

The preview version attracted a fair number of users with 80 users downloading it during the first week. Some of those were attracted by a post we did in a popular web forum on Android[24], some were users that stumbled upon the game while browsing the store, and a small number were hand-picked testers that we contacted via email.

However, the reactions on this preview were mediocre. Some users really liked the idea, giving it 5 out of 5 stars, another user liked the idea, but gave it only 3 stars because of the missing features, and yet others seemed almost offended by the early release, rating it with the minimum number of one star. Our takeaways here are that the store can be used to distribute preview versions of the application and store ratings will provide researchers with honest feedback. One has to be aware of the risk of bad reviews, but as the total number of reviews for such early releases is rather small, they will have only little impact once the app is completed and more and more reviews are added.

5.3. A visually appealing presentation will attract enough users for medium-scale observations

For both our preview version and the feature-complete release, we created promotional graphics and chose a neutral name for our game that did not disclose its nature as a research project, but did rather seem like a game of an independent development studio. Making an offer on the store appealing to users in this way has shown to be enough to make several hundred users download and try the app. We thus learned that the sole appearance in Google's *Play Store* provides an application with enough visibility to attract enough users for a medium-sized study.

5.4. Many downloads on non-NFC phones

[24] http://androidforums.com

Statistics from the app store indicate that many of the actual downloads and active installations of the app are originating from non-NFC phones. This is due to the fact that the many successful Android phones (e.g. HTC's Desire HD) do not come with an NFC reader (yet?).

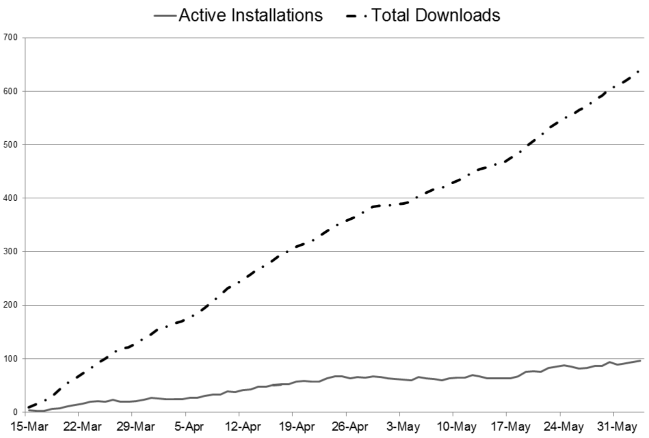
However, our data also shows that the two device models the app is installed on the most both support NFC: Samsung's Galaxy S2 and Google's Nexus S. We take away that support for non-NFC phones helps increase the number of downloads, but users of those phones are likely to uninstall the app soon. Uninstalls on NFC phones occurred, in our experiences, much less often. We from this conclude to make your app only visible to supported devices.

5.5. Scaling the game will require a fair amount of marketing and maintenance effort

At the time of this writing, the feature-complete version of *NFC Heroes* has been available for download for one month. The release of the gameplay preview has been slightly more than two months ago. The number of total downloads during this time was increasing at a constant rate per day. However, we at the same time faced uninstalls (also constant rate per day), so that the total number of active installations was still increasing rather slowly, totaling at 100 installs one month after the feature-complete version was released (see Figure 5).

Figure 5 Total number of downloads and active installations are growing at a constant rate.

File: installs.tif



We take away that in order to increase the growth of our game we have to fine-tune our game mechanics to reduce our relatively high bounce rate of up to 80%. Once more downloads turn into active installations, we will acquire users more actively and emphasize the game's viral aspects.

We acknowledge the fine-tuning of our game mechanics and the marketing efforts required to grow our total number of users will require roughly the same amount of resources as the initial development of the game. Researchers interested in performing large-scale studies with the help of app stores should carefully watch how many downloads actually turn into active installations and plan how they will scale their application once they are satisfied with those key metrics.

5.6. Diversification could give access to different user groups

The game presented in this paper is a trading card game. As we could show, the game was attractive to its users. However, as games dependent on personal preference for future evaluations of NFC adoptions we should consider to release various different games (e.g. arcade, role play) including rewards for scanning NFC tags in order to increase the user base by no longer being limited to trading card players only. Furthermore, we also consider to listen to NFC intends from other apps on the phone in order to capture NFC interaction outside the game context.

5.7. Studies in the large are constrained by users' contexts

As mentioned above the deployment of NFC technology and the awareness in the general public about it have not yet evolved as initially predicted. While wireless terminals for reading credit cards and transport tickets being deployed and starter kits distributed occasionally[25,26], the understanding of NFC has not yet arrived at most of the phone users. Thus, in contrast to a lab study where the subjects can be instructed to conduct a certain task, a study in the wild has to build upon self-selection, knowledge and motivation of the users choosing to download the app representing the study part. Accordingly, we plan to repeat our study in a few years from now once NFC might have been adopted in larger quantities. Then, we expect to collect much more data about deployed NFC tags, which could then be summarized in deployment maps and service directories of NFC.

6. References

Böhmer, M., Hecht, B., Schöning, J., Krüger, A., & Bauer, G. (2011). Falling asleep with Angry Birds, Facebook and Kindle: A large scale study on Mobile Application Usage. *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHIC 2011)* , pp. 47-56.

Broll, G., Graebsch, R., Scherr, M., Boring, S., Holleis, P., & Wagner, M. (2011, February). Touch to Play - Exploring touch-based Mobile Interaction with Public Displays. *Proceedings of the 3rd International Workshop on Near Field Communication (NFC 2011)* , pp. 15-20.

Budde, A., & Michahelles, F. (2010). Product Empire - Serious Play with Barcodes. (F. Michahelles, & J. Mitsugi, Eds.) *Proceedings of 2010 International Conference on Internet of Things: IoT for a green Planet (IOT 2010)* .

Cramer, H., Rost, M., Belloni, N., Bentley, F., & Chincholle, D. (2010). Research in the Large: Using App Stores, Markets, and other wide Distribution Channels in UbiComp Research. *Proceedings of the 12th ACM Internation Conference on Ubiquitous Computing - Adjunct (UbiComp '10 Adjunct)* , pp. 511-514.

Di Cerbo, F., Giradello, A., Michahelles, F., & Voronkova, S. (2010). Detection of Malicious Applications on Android OS. (H. Sako, Y. Franke, & S. Saitoh, Eds.) *Proceedings of the 4th International Conference on Computational Forensics (IWCF 2010)* , pp. 138-149.

Diewald, S., Möller, A., Roalter, L., & Kranz, M. (2011, December). Mobile Device Integration and Interaction in the Automotive Domain. *AutoNUI: Automotive Natural User Interfaces Workshop at the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2011)* .

[25] www.samsung.com/us/tectile/

[26] http://www.cde.at/en/references/mobilkom-austria-en/

Engadet. (2012). *Leaked Raymen Legends for Wii U trailer showcases NFC feature*. Retrieved September 20, 2012, from http://www.engadet.com

Ferreira, D., Kostakos, V., & Dey, A. (2012). Lessons Learned from Large-Scale User Studies: Using Android Market as a Source of Data. *International Journal of Mobile Human Computer Interaction , 4* (3), 16.

Gilbert, P., Chun, B.-G., Cox, L., & Jung, J. (2011). Vision: Automated Security Validation of Mobile Apps at App Markets. *Proceedings of the 2nd International Workshop on Mobile Cloud Computing and Services (MCS 2011)* , pp. 21-26.

Henze, N. (2012). Hit it!: an apparatus for upscaling mobile HCI studies. (J. Konstan, E. Chi, & K. Höök, Eds.) *CHI Extended Abstracts* , pp. 1333-1338.

Henze, N., Pielot, M., Poppinga, B., Schinke, T., & Boll, S. (2011). My App is My Experiment: Experience Studies in Mobile App Stores. *International Journal of Mobile Human Computer Interaction , 3* (4), 21.

Kranz, M., Holleis, P., & Schmidt, A. (2010). Embedded Interaction - Interacting with the Internet of Things. *Internet Computing , 14* (2), 46-53.

Lam, A., Chow, K., Yau, E., & Lyu, M. (2006). ART: Augmented Reality Table for Interactive Trading Card Game. *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications (VRCIA 2006)* , 357-360.

Lane, N., Miluzzo, E., Hong, L., Peebles, D., Choudhury, T., & Campbell, A. (2010). A Survey of Mobile Phone Sensing. *Communications Magazine , 48* (9), 140-150.

Lockheimer, H. (2012, February). *Google Mobile Blog: Android and Security*. Retrieved September 20, 2012, from http://googlemobile.blogspot.de/2012/ 02/android-and-security.html,

Möller, A., Michahelles, F., Diewald, S., Roalter, L., & Kranz, M. (2012, September). Update Behavior in App Markets and Security Implications: A Case Study in Google Play. *3rd Workshop on Research in the Large – App Stores, Wide Distribution, and Big Data in MobileHCI Research* .

Möller, A., Roalter, L., Diewald, S., Scherr, J., Kranz, M., Hammerla, N., et al. (2012, March). GymSkill: A Personal Trainer for Physical Exercises. *IEEE International Conference on Pervasive Computing and Communications (PerCom 2012)* , pp. 213-220.

Möller, A., Thielsch, A., Dallmeier, B., Roalter, L., Diewald, S., Hendrich, A., et al. (2011, June). MobiDics - Improving University Education with a mobile Didactics Toolbox. *9th International Conference on Pervasive Computing - Video Proceedings (Pervasive 2011)* .

Miluzzo, E., Lane, N. D., Lu, H., & Campell, A. T. (2010, September 26). Research in the App Store Era: Experiences from the CenceMe App Deployment on the iPhone. *1st International Workshop Research in the Large: Using App Stores, Markets, and other wide distribution channels in UbiComp research (LARGE 2010)* .

Murmann, L., Michahelles, F., & Kranz, M. (2012, September). NFC Heroes - Observing NFC Adoption through a Mobile Trading Card Game. *3rd Workshop on Research in the Large – App Stores, Wide Distribution, and Big Data in MobileHCI Research* .

Nauman, M., Khan, S., & Zhang, X. (2010). APEX: Extending Android's Permission Model and Enforcement with User-Defined Runtime Constraints. *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2010)* , pp. 328-332.

Nokia . (2012). *Nokia Beta Labs*. Retrieved September 20, 2012, from Nokia Shakespeare Shuffle: http://betalabs.nokia.com/trials/nfcgames/

Nokia Research Center Palo Alto. (2011). *Nokia NFC Games*. Retrieved September 20, 2012, from http://betalabs.nokia.com/apps/nfcgames/

Pellerin, R., Yan, C., Cordry, J., & Gressier-Soudan, E. (2009, January). Player profile management on NFC smart card for multiplayer ubiquitous games. *International Journal of Computer Games Technology - Special issue on cyber games and interactive entertainment* (7).

Research and Markets. (2011). *Application Distribution Channels 2011.* Evans Data Corp.

Rohs, M. (2007). Marker-Based Embodied Interaction for Handheld Augmented Reality Games. *Joural of Virtual Reality and Broadcasting , 4* (5).

Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., & Weiss, Y. (2012). "Andromaly": A behavioral Malware Detection Framework for Android Devices. *Intelligent Information Systems , 38* (1), 161-190.

Vidas, T., Christin, N., & Cranor, L. (2011). Curbing Android Permission Creep. *Proceedings of the Web , 2*.

Wimmer, R., Holleis, P., Kranz, M., & Schmidt, A. (2006). Thracker - Using Capacitive Sensing for Gesture Recognition. *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCS Workshops)* , pp. 64-68.

Zichermann, G. *Gamification by Design.* 2011: O'Reilly Media.