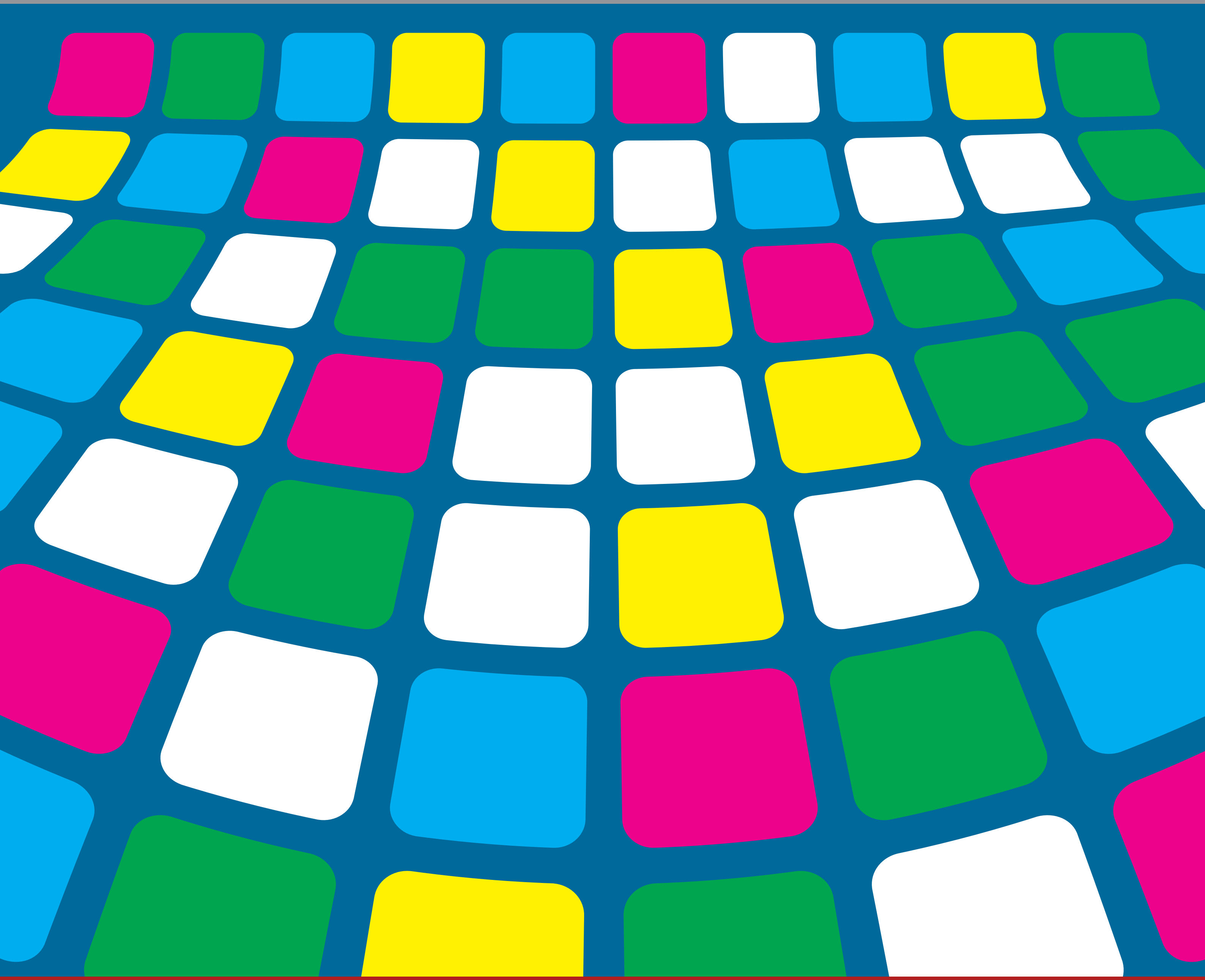
research in the LARGE



Proceedings of the 3rd International Workshop. Held in Conjunction with Mobile HCI.

Benjamin Poppinga, Henriette Cramer, Matthias Böhmer, Alistair Morrison,
Frank Bentley, Niels Henze, Mattias Rost, and Florian Michahelles

September 21, 2012.

table of contents

On the trail of ecological validity: Using a mobile campus application for analyzing user behavior in the field

Tilo Westermann
Quality and Usability Lab
Telekom Innovation Laboratories
TU Berlin, Germany
tilo.westermann@tu-berlin.de

ABSTRACT

In this position paper, we present *MoCCha*, a mobile campus application used not only as a subject of research, but as a research platform for a number of scientific disciplines. Following the approach of using an app that is available from a mobile application store, we want to study user behavior in the field with the aim for ecological validity that human-subject studies in lab environments are potentially missing.

Author Keywords
field study; app store; user behavior, mobile applications; ecological validity

ACM Classification Keywords
H.5.2 User Interfaces: Evaluation/Methodology

General Terms
Design; Experimentation; Human Factors

INTRODUCTION

Multimodal interactive services are used in situations where - in addition to the specific design of the application - contextual factors play an important role regarding the use and the user experience. While mobility and multimodality offer greater flexibility in principle, this is not always for the user's benefit: she or he is often rather overwhelmed than supported by modalities or the application's design.

Human-subject studies in Human-Computer Interaction (HCI) research are often conducted in a lab environment where all contextual factors may be controlled. While these studies lead to results with high internal validity, they might lack of *ecological validity* supposed that the user behavior differs in a realistic context (i.e. not in a lab environment). There is a trend towards utilizing mobile applications under real-world conditions as a research tool for studying HCI [3, 1, 2]. While many human-subject studies suffer from low numbers of participants or low diversity among the participants, mobile application stores such as Google Play or Apple's App Store enable researchers to reach a large number of users.

LARGE 3.0, September 21, 2012, San Francisco, CA, USA.

MOCCHA: MOBILE CAMPUS CHARLOTTENBURG

MoCCha is a mobile campus application that serves as a research platform. Using this app, we want to analyze user behavior in the field. At the time of writing, *MoCCha* is available from the Apple App Store[1] and first scientific studies are in the preparation phase. In this section, we will briefly present the concept of this app, outline data we are collecting, and discuss opportunities (i.e. research questions that may be answered) and challenges that are emerging using this approach.

App-in-App Concept

In order to motivate a regular use of the app and to get valuable data, we tried to create an offer that is both attractive and useful for the target audience. For our implementation we focused on the iOS platform. *MoCCha* is based on an App-in-App concept, mimicking the iOS home screen: various buttons are arranged on the screen, each of which leads to a specific sub-app. Currently, it includes a canteens app, a course catalog, an event calendar, Twitter, a contacts app (listing friends who also use *MoCCha*) and the program of the Berlin State Opera. The choice of sub-apps will be continuously expanded and tailored to meet the requirements to address the research questions listed below. With the exception of the Twitter sub-app, which is read-only up to now, all apps provide additional functionality in addition to pure information retrieval. For instance the user can export events to the device's calendar app or arrange to meet with friends who also use *MoCCha*.

The concept of nesting contents-wise differing offers into a single app makes *MoCCha* a research platform for a number of scientific disciplines: computer science, communication studies, engineering, and psychology. This characteristic distinguishes *MoCCha* from other apps that were previously used for research purposes.

Data Acquisition

Data is collected without presence of the experimenter which likely decreases effects of users feeling observed and could lead to more reliable data. Due to the diversity of sub-apps,

[1] `http://itunes.apple.com/en/app/moccha/id527216534?mt=8`

we are able to collect different kinds of data. Since we are using the iOS platform that is highly restricted in terms of interaction/logging outside of the sandboxed application, only data produced within *MoCCha* will be collected. Timestamped events are stored locally on the user's device and are uploaded to a central server when leaving the app. The collected data may be categorized into the following groups: time of interaction, location information, user preferences, social interaction and communication. In addition, *MoCCha* is able to record data from in-app questionnaires.

Challenges

A major issue when using an app as a research tool that is available from a mobile application store is the lack of knowledge regarding the user. At the beginning of a typical human-subject study information such as age, gender, or prior experience with technical devices are collected through questionnaires and the experimenter has the opportunity to ask in case of doubt. Using an app to collect demographic data requires more effort, as described in [3].

In the literature we often found the use of games as a research tool which is reasonable, as mobile gaming becomes increasingly popular. We took a slightly different approach in targeting students, employees and visitors of an university areal encompassing two universities. While this approach is limited with respect to the sample size of about 40.000 possible users (around 34.000 students and 9.000 employees) and educational standard (mostly high school graduates), this group represents the typical smartphone user aged between 18-34 years [4]. The well known issue of achieving equal sample sizes with respect to gender in human-subject studies seems to be negligible, as smartphone users are equally distributed among gender [4].

Using an app allows for the collection of large data sets. While this is valuable for analyzing user behavior, the researcher takes over great responsibility in dealing with this data, especially with regard to information about the user himself. In addition to offering a privacy policy, we will still recruit test participants for studies that require more knowledge about the user.

Opportunities

Using *MoCCha* as a research platform, we will analyze user behavior in the field in order to answer the question *where*, *when* and *how* users from diverse user groups use mobile multimodal services, and which contextual factors need to be considered for designing such services. In an ongoing research project we will analyze and model how different sensory perceptions integrate into an overall picture, the influence on perceived quality, and which conclusions can be derived for the design of such services. Our research questions for this project can be assigned to the following groups:

Fusion and fission of modalities

- Which models can be derived from field studies regarding fusion and fission of modalities?
- Under which condition does adaptivity for fusion and fission of modalities make sense?
- What impact on acceptance and use of speech do context, task and user group have?

Changes from lab environment to the field

- What effect on use in the field do contextual factors have?
- How to replicate contextual factor in the lab?
- How to study concurrent interaction in the lab?

Adaptivity and personalization

- How to measure the quality of adaptive systems in laboratory environments and in the field?
- What is the optimal compromise between adaptivity, adaptability, and learning?
- How do the use and the experience vary over time?

Attention and perception of multimedia

- What clues do models for attention and distraction provide for the design of applications for navigation, and how do they need to be modified?
- What is the impact of context on the distribution of attention during pedestrian navigation?
- How to measure multimedia perception in an ecologically valid manner?
- What is the relationship between multimedia perception, quality and use of media?

DISCUSSION

In this position paper, we discussed challenges and opportunities that go along with using a mobile campus application for research purposes. As previous research has shown, the approach of releasing an app as a research tool on a mobile application store enables findings from a realistic context that may differ from studies conducted in a lab environment. We believe that this approach complements current research practices, and is a promising step towards ecological validity of human-subject studies in HCI research.

REFERENCES

1. Böhmer, M., Hecht, B., Schöning, J., Krüger, A., and Bauer, G. Falling asleep with Angry Birds, Facebook and Kindle: a large scale study on mobile application usage. In *Proc. MobileHCI '11*, ACM (New York, NY, USA, 2011), 47–56.

2. Henze, N., Rukzio, E., and Boll, S. 100,000,000 taps: analysis and improvement of touch performance in the large. In *Proc. MobileHCI '11*, ACM (New York, NY, USA, 2011), 133–142.

3. McMillan, D., Morrison, A., Brown, O., Hall, M., and Chalmers, M. Further into the Wild: Running Worldwide Trials of Mobile Systems. In *Proc. Pervasive'10*, Springer-Verlag (Berlin, Heidelberg, 2010), 210–227.

4. The Nielsen Company. State of the Media: U.S. Digital Consumer Report, Q3-Q4 2011. `http://goo.gl/qlSq2`, 2012.

Update Behavior in App Markets and Security Implications: A Case Study in Google Play

Andreas Möller,
Stefan Diewald, Luis Roalter
Technische Universität München
Munich, Germany
andreas.moeller@tum.de,
stefan.diewald@tum.de,
roalter@tum.de

Florian Michahelles
ETH Zurich
Auto-ID Labs
Zurich, Switzerland
fmichahelles@ehtz.ch

Matthias Kranz
Luleå University of Technology
Department of Computer Science,
Electrical and Space Engineering
Luleå, Sweden
matthias.kranz@ltu.se

ABSTRACT

Digital market places (e.g. Apple App Store, Google Play) have become the dominant platforms for the distribution of software for mobile phones. Thereby, developers can reach millions of users. However, neither of these market places today has mechanisms in place to enforce security critical updates of distributed apps. This paper investigates this problem by gaining insights on the correlation between published updates and actual installations of those. Our findings show that almost half of all users would use a vulnerable app version even 7 days after the fix has been published. We discuss our results and give initial recommendations to app developers.

Author Keywords
Mobile applications; digital market places; update behavior; security

ACM Classification Keywords
D.4.6. Operating Systems: Security and Protection

INTRODUCTION AND 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 [13]. In March 2012, Apple reached in total 25 billion iOS app downloads[1]. Until 2011, 10 billion Android apps have been downloaded in total over Google Play[2]. 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, 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. [11, 10]) 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[3], so that potentially a large number of apps is 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. [5]), 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 and discuss the consequences and recommended actions on the part of the developers.

RELATED WORK

While inclusion in the Apple App Store requires a review process [1], Google Play is free of constraints for uploading apps. However, apps are scanned for viruses and malware [8] 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 [6, 14]. Di Cerbo et al. [4] 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)

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

[2] http://www.wired.com/gadgetlab/2011/12/10-billion-apps-detailed/

[3] http://www.androlib.com/appstats.aspx

LARGE 3.0, September 21, 2012, San Francisco, CA, USA.

granularity of permissions [12, 15] or remove them in hindsight by inline reference monitoring[4]. Fewer rights inherently also decrease the probability for security-relevant bugs.

Miluzzo et al. [9] 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 [7] 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.

CASE STUDY

For our case study, we are looking at *VMI Mensa*[5], an Android application developed by the research group of the authors of this paper. *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. 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.

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

User Communication Analysis

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.

Results

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 Fig. 1 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.

Version Distribution

We also looked at the distribution of the latest five versions of the app on users' devices, illustrated by different colors in Fig. 2. 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%

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

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

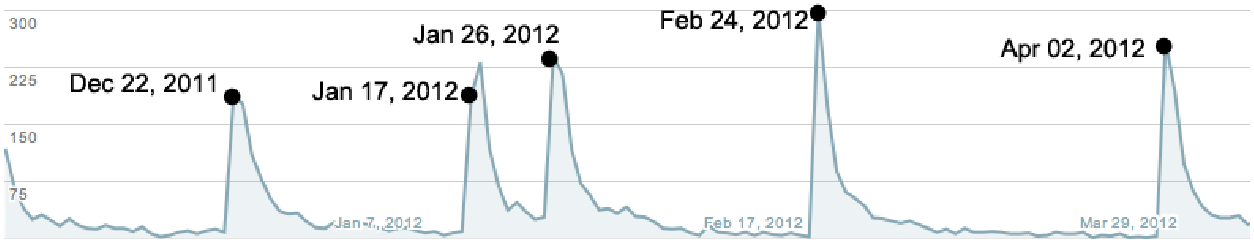


Figure 1. Visualization representing the number of five subsequent update downloads (vertical axis) over time. The graph shows maxima on the update publishing day (possibly also due to activated auto-updates) and exponentially decreases thereafter. Modified diagram based on Android Developer Console statistics.

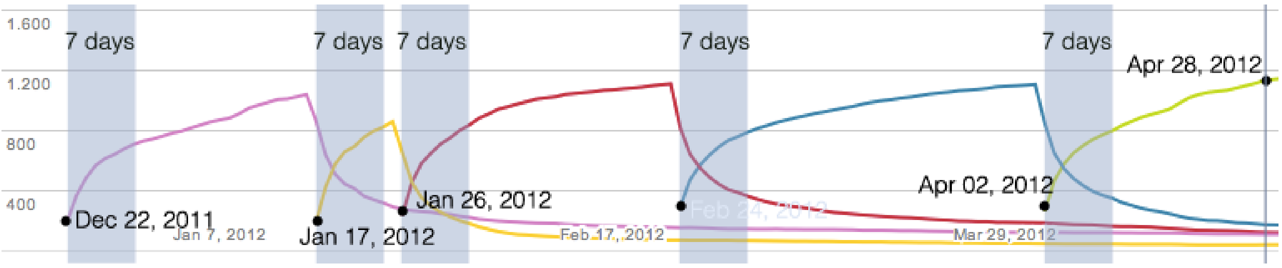


Figure 2. Visualization representing the number of installations by version (vertical axis); the colored lines indicate the five latest versions. The diagram reveals how long old versions are active on user's devices. The 7-day periods after an update has been published are highlighted. Modified diagram based on Android Developer Console statistics.

(v.24) and 2.1% (v.23). Most severely, 21.5% had even older versions installed on their devices at that time.

DISCUSSION

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 [2] 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 can be brutal [9]. 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 [3], 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 [11].

CONCLUSION

In this paper, 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 a potential slow update propagation on Android 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.

Acknowledgments

We thank all of our students involved in the development of *VMI Mensa* and our other research apps.

REFERENCES

1. Apple Inc. App store review guidelines. `https://developer.apple.com/appstore/guidelines.html`, 2012.
2. Böhmer, M., Hecht, B., Schöning, J., Krüger, A., and Bauer, G. Falling asleep with angry birds, facebook and kindle: a large scale study on mobile application usage. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, ACM (2011), 47–56.
3. Cramer, H., Rost, M., Belloni, N., Bentley, F., and Chincholle, D. Research in the large. using app stores, markets, and other wide distribution channels in ubicomp research. In *Proceedings of the 12th ACM international conference adjunct papers on Ubiquitous computing - Adjunct*, Ubicomp '10 Adjunct, ACM (New York, NY, USA, 2010), 511–514.
4. Di Cerbo, F., Girardello, A., Michahelles, F., and Voronkova, S. Detection of malicious applications on android OS. In *Proceedings of the 4th international conference on Computational forensics (IWCF'10)*, H. Sako, K. Y. Franke, and S. Saitoh, Eds., Springer (Berlin, Heidelberg, 2010), 138–149.
5. Diewald, S., Möller, A., Roalter, L., and Kranz, M. Mobile Device Integration and Interaction in the Automotive Domain. In *AutoNUI: Automotive Natural User Interfaces Workshop at the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2011)* (Nov.–Dec. 2011).
6. Gilbert, P., Chun, B.-G., Cox, L. P., and Jung, J. Vision: automated security validation of mobile apps at app markets. In *Proceedings of the second international workshop on Mobile cloud computing and services*, MCS '11, ACM (New York, NY, USA, 2011), 21–26.
7. Henze, N., and Sahami, A. Appticker. `http://projects.hcilab.org/appticker/`, 2012.
8. Lockheimer, H. Google Mobile Blog. Android and Security. `http://googlemobile.blogspot.de/2012/02/android-and-security.html`, February 2012.
9. Miluzzo, E., Lane, N., Lu, H., and Campbell, A. Research in the app store era: Experiences from the CenceMe app deployment on the iPhone. In *Proc. Ubicomp* (2010).
10. Möller, A., Roalter, L., Diewald, S., Scherr, J., Kranz, M., Hammerla, N., Olivier, P., and Plötz, T. Gymskill: A personal trainer for physical exercises. In *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on* (march 2012), 213 –220.
11. Möller, A., Thielsch, A., Dallmeier, B., Roalter, L., Diewald, S., Hendrich, A., Meyer, B. E., and Kranz, M. Mobidics – improving university education with a mobile didactics toolbox. In *Ninth International Conference on Pervasive Computing (Pervasive 2011), Video Proceedings* (San Francisco, CA, USA, June 2011).
12. Nauman, M., Khan, S., and Zhang, X. Apex: Extending android permission model and enforcement with user-defined runtime constraints. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ACM (2010), 328–332.
13. Research, and Markets. *Application Distribution Channels 2011*. Evans Data Corp., Sep. 2011.
14. Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., and Weiss, Y. "Andromaly": a behavioral malware detection framework for android devices. *Journal of Intelligent Information Systems 38* (2012), 161–190. 10.1007/s10844-010-0148-x.
15. Vidas, T., Christin, N., and Cranor, L. Curbing android permission creep. In *Proceedings of the Web*, vol. 2 (2011).

The data chicken and egg problem

Håkan Jonsson
Sony Mobile Communications & Lund University
Nya Vattentornet, SE-22188 Lund
hakan1.jonsson@sonymobile.com

ABSTRACT

Creating context aware consumer applications or other user data-driven applications require user data in order to model the user and her context. For users to give up personal data they want value from the data collecting application. So how can we provide the value to get the data when we need the data to provide the value, without intruding on the user's privacy or violating her trust in the application? In this paper we name the problem, formulate a data usage design principle and discuss some potential solutions to the problem.

Author Keywords
User model, context awareness, privacy, design

ACM Classification Keywords
E.m. Data: Miscellaneous.

General Terms
Design

INTRODUCTION

"You only had 50.000 users in your study?" is a question Andrew T. Campbell, Dartmouth College, claimed will soon be asked by scientific paper reviewers, at the Mobiquitous 2011 conference. Perhaps a bit exaggerated, but he has a point. The possibility to recruit study subjects for scientific studies through the mobile web in general and app stores specifically, is radically increasing the number of subjects we can recruit and the amount of data we can collect. However, it also has implications, for example on marketing, quality and cost [1].

One of the challenges is visibility and discovery: How can we get users to find and install our research application in competition with hundreds of thousands of commercial quality applications? Spending money on marketing can help, but research project have limited budgets. Another solution is to spread the application virally, which require it to provide a genuine end user value in order for users to recommend it to friends. To create a great application that provide end user value is however not trivial. A special problem is that many research applications need the user data in the first place to provide valuable functionality. Another problem is that forcing users to give up data before getting value from application functionality is questionable from a privacy point of view, as we will see below.

LARGE 3.0, September 21, 2012, San Francisco, CA, USA.

PRIVACY: MANAGING USER EXPECTATIONS

The vast majority of Internet users do not read terms of service or privacy policies [2] (or data usage policies as Facebook now calls them). They just click trough them when signing up for a new service. If the user discovers that information he considers private or sensitive has been disclosed to parties he did not expect, he is surprised and annoyed. Knowing this we cannot consider users who have agreed to terms of service or a privacy policy by clicking an OK button to have given informed consent to data collection. Annoyed users can be dangerous to business in more ways than through legal actions. Detractor influencers [3] can cause widespread defection from a service and generate negative publicity. Thus, the terms of service only protect the service provider from a legal perspective, and not from a business or user perspective. Several social network providers have learned this the hard way [4]. The main lesson to learn from these experiences is: *Don't surprise the user*.

Managing privacy is about managing expectations of the user, and not about making sure the terms of service is correct. Neither does it help to provide fine-grained controls of privacy control settings if the user is not aware of them, application developers ignore them or if default privacy control settings do not meet user expectations. Application permission let's the user determine what data is being accessed, if used correctly, but not why. Major application vendors have been known to abuse the permissions frameworks of major application stores, or failed to give acceptable motivations for some permission [9].

Phone OEM's have an additional problem: They can in general not use application permissions to declare permission at install time, since their applications are pre-installed. This means that they can only rely on what they know about user expectations on privacy, or provide additional information or permission prompts in runtime. OEM's have a lot to loose in terms of brand value and must in general be a lot more careful about breaking user expectations on privacy than application developers for an academic research project.

In addition to user expectations, there are regulations that need to be considered when using app stores for research. The EU regulations [5] and directives in general dictate that data collection and processing must be fair, specific and explicit to the user: The data collected must be adequate, relevant and not excessive in relation to the purposes for which they are processed. The purposes must be explicit and legitimate and must be determined at the time of collection of the data.

The Obvious Data Usage Principle

We propose the Obvious Data Usage Principle (ODUP), a design principle which purpose is to fulfill the requirements above when applied to design of applications that collect data:

Any data collected from the user should be reflected in the functionalities of the application collecting the data, in such a way that it is obvious to the user what data is being collected and how it is being used.

ODUP can be seen as a being based on the economic theory of privacy signals in a lemons market [8]. By applying this principle, the user is less likely to ever be surprised that some data is being collected from her. This applies even if the user does not read EULAs or privacy policies before he starts using the application.

Compliance to ODUP does not solve all privacy issues and it is not always straightforward how to apply it. For example, some functionality may need data to be collected for some time before it can be used. For this functionality it will be hard to make it obvious to the user why the data is being collected. Letting the user know that the data is being collected, but not allowing the user to use the functionality does not make for a very good user experience. In these cases it may be better to delay showing the functionality until it becomes available to the user, with the risk of surprising the user, or indicating that the functionality is not yet available. In the recommenders systems community this is known as the cold start problem [10].

Another issue is of course that what is obvious and not differs between users. Thus, ODUP should be seen as a guiding principle that needs to be applied in each case rather than a requirement to be interpreted formally.

THE DATA CHICKEN AND EGG PROBLEM

ODUP require us to not collect any data that we can't reflect in application functionality. However, many of the data driven applications that are the end goal of much research, especially in mobile sensing, require user data or user-generated data to develop the functionality that the user wants in exchange for data. For example, to develop an application that depends on named entity recognition in SMS text messages in multiple languages and regions, huge amounts of SMS data needs to be collected before the functionality can be offered since SMS datasets are not generally available. Another example is an application that learns the user's movement behavior to adapt functionality to the users home and workplace locations. The home and workplace locations cannot be inferred until a certain amount of data has been collected.

So how do we solve this chicken and egg problem?

Different feature, same data

One solution is to provide a different feature or application than the one you are aiming for, that you can bootstrap using other data. For example: If you need Bluetooth proximity data to be able to do social group context detection, provide an application that needs proximity Bluetooth data to detect single individuals in proximity, e.g. to trigger reminders. This allows you to collect the data required to do the analysis to provide the group context detection application. However, there are a legal and privacy issues associated with this approach since it can break ODUP and EU Data Protection Act, since you collect the data for a different purpose than the one the user sees in the application. From a legal point of view, you can only collect data for the purpose you declare to the user. This means you can provide a terms of service that allows you to collect the data for the group context detection purpose, and then not make the usage obvious in you reminder application. This breaks ODUP but not the EU Data Protection Act and risk surprising the user.

Bootstrapping

Another solution is to bootstrap using data from existing available data sets of same type. For example, if you need to collect location data and have a location history in order create end user value, bootstrap using Google Latitude Location History from the user if available.

Depending on the problem, it may also be possible to bootstrap using data from existing available data sets of a similar type, e.g. bootstraping an application that needs SMS data with twitter data. We used this approach successfully [6] to develop an application that extracted trending topics from SMS messages. However, this approach did not work well for named entity recognition in SMS messages. While trending topics is only about counting of word frequencies, named entity recognition needs to capture language features, which turned out to be too different for SMS and tweets for the two languages we investigated: Swedish and English [7].

Stereotyping

Instead of creating a user model based on actual data, use an existing stereotype model. Use some information you have or can collect when the user first starts using the application, to determine which stereotype the user belongs to. Use this stereotype as the initial model, while collecting enough data to create an individual user model. A trivial example of this is asking the user for zip code to select a stereotype from an existing demographical model.

Common sense rules or heuristics can also be used to create simple stereotypes when data is lacking. For example, if you want to determine a users home and work place from location data, but don't have the location data, an example stereotypical common sense rule model could be one that assumes users are at work 9-17 and at home 22-6 on weekdays. Based on this, you can provide an application that will initially work quite well for a lot of people, and you can start collecting actual data to create a better model based on real data.

CASE STUDIES

The ODUP principle is an attempt to condense the experience drawn from three case studies into a single guideline. In these cases we studied data collection and processing of sensitive personal information. In all of them it was quite obvious to the user what data was being collected and how it was used, since we applied ODUP, but it also prevented us from collecting some desired data, and rightly so.

SMSTrends

In the first we tried to collect SMS data from users in order to develop named entity recognition for SMS. Raw SMS data is not generally available as research datasets due to its sensitive nature. Some datasets are available that has been anonymized, but none of them would work for named entity recognition research. We first asked Sony employees to simply give us the data, and developed an app to make it easy to upload from the user's phone. We got no contributions, which was expected. Next we developed an application, SMSTrends (Figure 1), that provided a little user value, showing locally trending words extracted from the SMS, and tried to find users willing to try it out. A small group was, but none were willing to upload data until privacy controls were put in place, allowing the users to mark certain messages as secret. None of the users ever used this possibility. In this application it was obvious to all users that both location and SMS data was required to provide the service, and they were aware of the data collection for research purposes and had given their consent. For this small group that was well informed about the purpose of the data collection we felt safe about using the collected data for named entity recognition research. Still, we were unsure this approach would be feasible for a larger scale since the named entity recognition had no connection to the user experience or functionality of the application, and we would thus break the expectations on usage of the data collected, for users who would not read the terms of service.

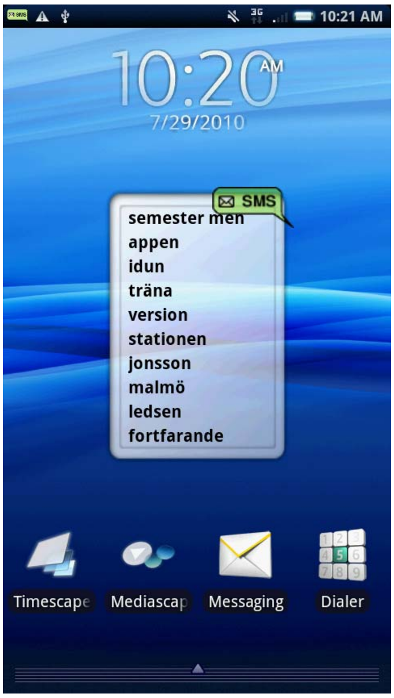


Figure 1 SMSTrends widget

Contacts widget

We developed a widget application (Figure 2) that showed the most frequently used contacts, giving the user quick access to communication with them. It also included a feature that forwarded missed call notifications and unread SMS messages to the user's email. This was launched on Android Market and has had about 300.000 downloads. In this case, even though we had the users permission to access the body text of the SMSes and that it was required to deliver the functionality, we thought it would abuse the user expectations of privacy if we would use that data for named entity recognition research, and decided to refrain from using it.

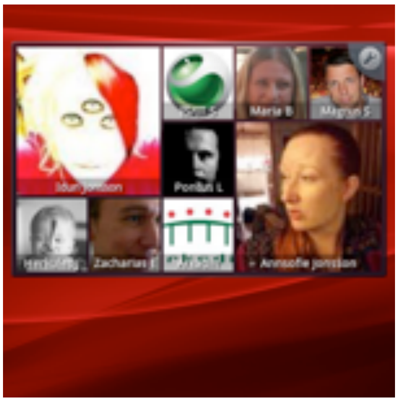


Figure 2 Contacts widget

Reminders

In a third and ongoing study we collect movement and proximity data in a small user group to allow the user to set location based reminders. For this project our approach is to introduce features or improvement in features in an incremental way such that each small improvement is acceptable to the user and gives the users additional value. If the step is too big we will put that functionality in a separate application and allow the user to bootstrap the new application using data from the first, allowing us to agree on a new terms of service, permissions and purpose of data collection. In this way we hope to approach the goal of studying proximity dynamics on a large scale without violating user privacy or having to create a new Facebook.

CONCLUSION

Developing valuable data-driven applications for research purposes that will market themselves though viral spread on app stores is not trivial, since you often need the data you want to collect to develop the application in the first place. Privacy regulations and user expectations on privacy complicates the picture even more. In this paper we have suggested the ODUP design principle to meet these requirements and listed some general methods of bootstrapping data for application development. ODUP is currently a loosely formulated design guideline. To give it more substance in terms of theoretical and empirical grounding, the next step should be to estimate the cost and benefits of the privacy signals for different applications.

REFERENCES

1. Miluzzo, E., Lane, N. D., Lu, H., and Campbell, A. T. (2010) Research in the App Store Era : Experiences from the CenceMe App Deployment on the iPhone. *Proceedings of The First International Workshop Research in the Large Using App Stores Markets and other wide distribution channels in UbiComp research.*
2. Böhme, R., and Köpsell, S. (2010) Trained to accept? *Proceedings of the 28th international conference on Human factors in computing systems CHI 10.*
3. Reichheld, F. (2006) "The Ultimate Question: Driving Good Profits and True Growth", *Harvard Business School Press*.
4. B. Slattery, (2010) "Google Hits Restart on Buzz Privacy Settings", *PC World*, April 6th.
5. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data, (1995) *Official Journal L 281*, 31-50.
6. Jonsson, H., Nugues, P., Bach, C., and Gunnarsson, J. (2010) Text mining of personal communication. *14th International Conference on Intelligence in Next Generation Networks*.
7. Ek, T., Kirkegaard, C., Jonsson, H., and Nugues, P. (2011) Named Entity Recognition for Short Text Messages. *Procedia - Social and Behavioral Sciences*, 27, 178-187.
8. Vila, T., Greenstadt, R., and Molnar, D. (2003) Why we can't be bothered to read privacy policies: Models of privacy economics as a lemons market. *Proceedings of the 5th international conference on Electronic commerce ICEC 03*, 403-407.
9. Ngak, Ch. (2012) Facebook denies allegations of reading users' text messages. http://www.cbsnews.com/8301-501465_162-57385763-501465/facebook-denies-allegations-of-reading-users-text-messages/, CBS News.
10. Schein, A., Popescul, A., Ungar, L. and Pennock, D. (2002) Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '02)*.

Long Tail or Superstar?
An analysis of app adoption on the Android Market

Nan Zhong
Computer Science, ETH Zurich
Switzerland
zhongn@ethz.ch

Florian Michahelles
Information Management, ETH Zurich
Switzerland
fmichahelles@ethz.ch

ABSTRACT
Many online markets are found with a long tail in sales distribution. With the analysis of a large data set of transactions in Android Market, this work first brings the examination of long tail to the mobile application market. The results suggest that, rather than being a "Long Tail" market where unpopular niche products aggregately contribute to substantial portion of sales, the Android Market is more a "Superstar" market strongly dominated by popular hit products. Hit apps are also found to have higher user consumption and satisfaction rate. Besides, we investigate the impact of price and finds that some expensive apps constitute unproportional large sales. Our findings reveal possible different market structure of mobile app market and point out challenges to app developers.

Author Keywords
Mobile application market; long tail; sales distribution.

ACM Classification Keywords
H.4.m. Information Systems Applications: Miscellaneous

General Terms
Economics; Measurement.

INTRODUCTION
A number of digital markets are found to be "Long Tail" markets where the aggregated sales of the huge amount of niches contribute a sizable fraction of the total revenue [2, 5, 4]. Nevertheless, some other markets are found to be "Superstar" markets where the blockbusters strongly dominate the revenue [7, 8].

Mobile app markets can be seen a long-tailed sales distribution. Among the tens of thousands of apps listed in Android Market, blockbusters such as Angry Bird have been downloaded millions of times, while numerous niches have only been downloaded dozens of times. Therefore, the examination of the long tail in mobile market could provide insights to developers in understanding the market structure and evaluating profitability of the long tail.

Despite research based on limited data [11], the general lack of data in sufficient size hinders research in mobile app market. Thus, this research, to our knowledge, is among the first to examine sales distribution of a mobile app market. In particular, we analyze a large data set of transactions in Android Market and examine the long tail in detail. We find evidence indicating that the Android Market whose downloads and sales are largely dominated by hit apps, is more a Superstar market than a Long Tail market. We also show that though most downloads of paid apps are from cheap apps, some expensive apps accounts for unproportional large revenue.

In the next section, we review related work, describe the dataset and methodology of research. Then results are presented and analyzed. Finally, we conclude our findings and summarize implications to business strategies.

RELATED WORK
The term "Long Tail " was coined by Chris Anderson to describe how aggregated sales of niches products of online retailers can contribute to large portions of sales [2]. For example, 30% of Amazon's sales of books and 20% of Netflix revenue of movies come from titles unavailable in largest offline stores [2].

However, the value of long tail is in dispute in academia. There is evidence from video [7, 6, 12] and music markets [8] that online market sales concentrate further on hit products, therefore retailers should continue emphasizing the hit products.

Regarding the cause to the long tail, researchers have pointed out that 1 Low stocking and distribution costs that enable abundant supply; 2 Easy searching tools and smart recommender systems that allow users to access otherwise unnoticed niche products, are key factors [2, 5, 4, 3]. The mobile app market possess these factors and our work firstly examine the long tail of it.

DATA
The data of this work has been provided by *42matters AG*[1] which captures installations, updates and removals of apps in real time and shares this information among its users [9]. Its central database receives records of transactions from Appaware clients running in users' Android phones, which is authorized on the terms of use when users install Appaware. A record contains user id, time, type of transaction (install, removal, and update), app name, app price, app rating, and etc. The dataset is part of those records and Table 1 shows some statistics of it. In general, this dataset consists of 208 thousand anonymous users' 84.1 million transactions from

[1] www.appaware.com

LARGE 3.0, September 21, 2012, San Francisco, CA, USA.

Users	208,187
Paid Apps	16,214
Free Apps	175,087
Transactions	17,609,041
Paid App Sales	1,887,175$
Paid App Downloads	530,168
Free App Downloads	6,079,398

Table 1. Statistics of dataset.

	Paid Apps	Free Apps
C	71.5%	76.0%
U	15.9%	14.3%
W	12.6%	9.7%

Table 2. Evaluation of dataset representativeness.

March 2011 to November 2011. This dataset is one of the few sources that are statistically large enough for studies in sales distortion and user consumption patterns in mobile app markets.

To show the representativeness of this dataset, we conduct an evaluation by comparing orderings of downloads in our data with those in Android Market. Intuitively, if an app x has a higher ranking of downloads than y in Android Market, then x should also have more downloads recorded than y in our data. With comparison of all possible combinations of two-app pairs, we could examine how much the dataset accords with Android Market available data.

In detail, for a given app, although the ground truth (precise number of downloads) is inaccessible, the range that how many downloads it has is listed in Android Market. These ranges are given by an ascending sequence of predefined consecutive intervals: $[1, 5]$, $[5, 10]$, $[10, 50]$, $\cdots [10{,}000, 50{,}000] \cdots$. Every app x fits in a range $r(x)$ and all the apps share the same sequence of ranges. We define $r(x) \succ r(y)$ if left bound of $r(x)$ is greater than or equal to right bound of $r(y)$. Let A be the set of all apps in the dataset, and $d(x)$ number of downloads of an app $x \in A$. We calculate:

$$C = |\{(x, y) \mid x, y \in A, r(x) \succ r(y), d(x) \geq d(y)\}|/N$$
$$U = |\{(x, y) \mid x, y \in A, r(x) = r(y)\}| /2N$$
$$W = 1 - U - C$$

where (x, y) is an ordered pair of apps. $N = |A|(|A| - 1)/2$ is the total number of possible pairs. C is the percentage of correct pairs, U unclear pairs, i.e. the ones are in the same range, and W wrong pairs. From Table 2 we could see that more than 70% of pairs have correct orderings in both paid and free apps. In short, the dataset preserves the ordering between apps in the Android Market fairly well.

METHODOLOGY

We first partition the apps into two categories: paid and free, taking account of intrinsic difference between paid apps and free apps. Table 1 already indicates the difference: there are a lot more free apps than paid apps, and free apps also have much larger total number of downloads than paid ones. In addition to that, top downloaded free apps has around 10 times more downloads than top paid apps recored. Therefore it is reasonable to distinguish these two kinds of apps. Regarding the definition of free apps, an app is defined as free if it never charges users for downloading. This is because app prices could change over time, sometimes a paid app is free for promotion for a short period of time.

For paid apps, we unify local payment currency used in different countries by Android Market by converting all payments to US dollar using conversion rate given by Google Currency on May 1, 2012. We believe this has minor impact on the calculation of total sales.

Afterwards, we calculate downloads that contribute to sales of paid apps. Android Market has a return time for paid apps, within which a payment could be refunded if the purchased app is deleted. Since late December 2010, this return time has been set to 15 minutes. Spurious downloads are neglected accordingly.

Finishing these preparations, following [6, 5, 10], we conduct the analysis of sales distribution of paid apps and downloads distribution of free apps.

RESULTS

User Consumption

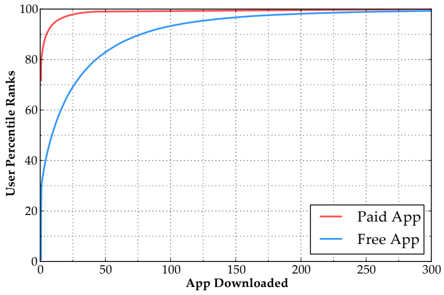


Figure 1. User percentile of downloads.

To begin with, Figure 1 depicts user consumption of apps. We find user consumption of paid apps is rather limited. For a given number of downloads x in the horizontal axis, the corresponding y value, i.e. percentile, is the percentage of users downloading less than or equal to x apps. For example, 72% of users have not downloaded any paid apps and only 2% of users have not downloaded any free apps.[2] Most users (90th percentile) download less than 3 paid apps and 75 free apps. It may be caused by the fact that most apps in Android Market are free and as some business observers speculate, users in Android market are less willing to pay than in other mobile markets [1]. This strong distinction between paid and free apps supports our previous partition.

Long Tail vs. Superstar

Then we examine the sales distortion. In Figure 2 we use the Lorenz Curve and Gini Coefficient to study the concentration of consumption. Apps are ranked according to its popularity ascendingly. For paid apps, popularity is defined as value of sales, and free apps number of downloads. The Lorenz Curve

[2] Due to the design of Appaware, users not downloading any apps cannot be recorded.

depicts the cumulative percentage of popularity of the bottom x percent most popular apps. The Gini Coefficient represents the deviation of the Lorenz Curve to the Line of Equality. A big Gini Coefficient indicates a Superstar market dominated by the hits, and a small Gini Coefficient shows a Long Tail market characterized by the long tail.

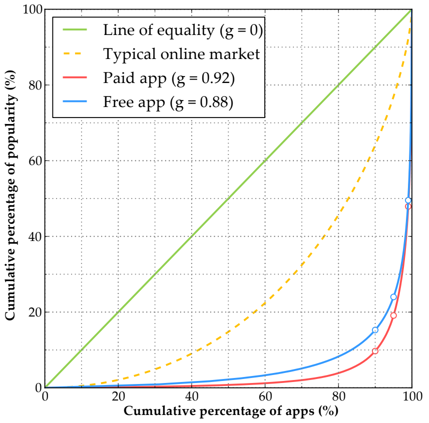
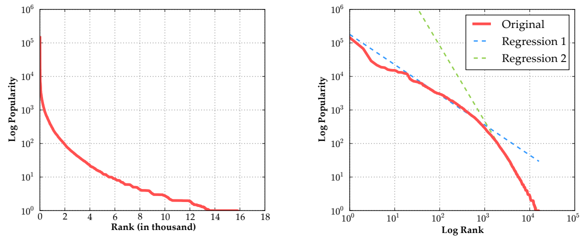


Figure 2. Lorenz Curve and Gini Coefficient.

We can see that the hits are dominating. For both sales of paid apps and downloads of free apps, top 1%, 5% and 10% most popular apps make up approximately 50%, 80% and 90% percent cumulative popularity. This dominance of hit products is even stronger than the well known Pareto Principle which claims that 20% most popular products possess 80% of popularity. These curves are also far different from Lorenz Curve of typical online market [5].



(a) Tall head and flat tail.

(b) Piecewise power law.

Figure 3. The long tail of sales and downloads in absolute terms.

These patterns are depicted in absolute terms in Figure 3, where apps are ranked by popularity descendingly in x-axis, and its popularity value is in y-axis. Figure 3(a) takes logarithmic scale in y-axis, and shows that the popularity decreases sharply as the rank increases. Instead of having a long tail, the Android Market has a tall head and a flat tail. When logarithmic scale in both axises in Figure 3(b), curve does not posses a global power law which was found or assumed in similar studies in other online markets [5, 4]. Nevertheless, a piecewise power law is observed using two linear regressions segmented at $x = 10^3$, which divides the apps into two groups: the top 10% hits and the bottom 90% niches. The coefficient of determination (R^2) of Regression 1 and Regression 2 are 0.9997 and 0.9973 respectively implying the good fitting of the regression. The slope of the regression 2 is much deeper than that of regression 1. Namely, when compared to hits, popularity of niches drops much quicker. This suggests that for the niche apps, discovery is still an intractable task, especially in a market where most users download no more than 3 paid apps.

Natural Monopoly and Double Jeopardy

The dominance of hit apps is further shown in the two phenomena of sales distribution: *natural monopoly* and *double jeopardy* [6, 10].

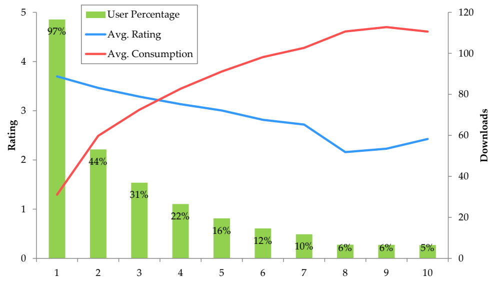


Figure 4. Natural Monopoly and Double Jeopardy

Natural monopoly claims that not only does popular products attract disproportionate share of customers, but also these customers purchase more popular products than unpopular ones. We find evidence supporting this theory. In Figure 4 and Figure 5, apps are sectioned into ten deciles where the most popular 10% apps are at left most and least popular 10% right most. The green bars in Figure 4 represent the percentage of users downloading at least one app in this decile[3]. Almost every user download the most popular apps while very few users download the least popular ones. Additionally, the red line shows the average number of apps downloaded by users downloading at least one app of a decile. It tells that, consumers of niche apps download more than those of hit apps. When we drill down these downloads in Figure 5, in which the top 10% apps are tittled as Head and bottom 90% as Tail. Light users in 1st decile , i.e. those who download most popular apps, have larger portion of apps downloaded from most popular apps.

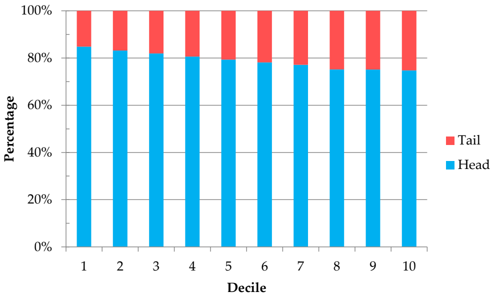


Figure 5. Distribution of downloads in tail and head.

Double jeopardy describes that the unpopular products have both less consumers and lower satisfaction rate, therefore in a "double jeopardy". This is shown in Figure 4 by the descending bar chart and blue line, which represent number of consumers and their average rating of apps.

[3] Users not downloading any paid apps are not shown in this chart.

To sum up, the majority of users download hit apps and the few minority users download niche apps; all users consume much more hit apps than niche apps; and hit apps have higher user ratings than niche apps. This accords with the natural monopoly and double jeopardy observations, which clearly demonstrate the superiority of hit apps.

Price Distribution

At last, we analyze the distribution of sales and downloads of paid apps versus prices. In Figure 6, the height of a bar is the percentage of total apps in a section of prices, and corresponding percentages of total sales/downloads of all apps in this section are represented by the red and blue lines. Most apps are rather cheap, actually the average price of all paid apps is 2.6$. Interestingly, among cheap apps which are below 3$, the usual 1$ apps have less aggregated downloads and sales than apps whos prices are ranging from 1$ to 3$. However, counter intuitively, a few expensive apps acquire unproportional large revenue, whose price are dozens of times higher than cheap apps, thus a few downloads results in big revenue. These apps are usually professional apps, such as navigation, which may have different market position than games and daily apps.

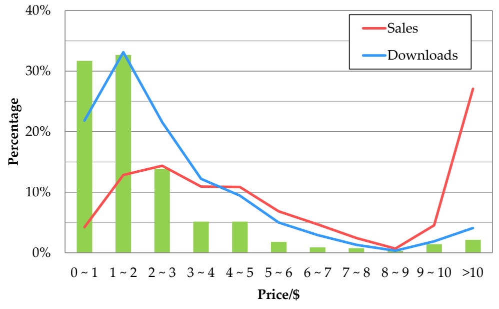


Figure 6. Distribution of sales and downloads of paid apps.

DISCUSSIONS

We found that the Android Market is a Superstar market largely dominated by hit apps. Among the limited number of apps downloaded or purchased by most users, hit apps make up the vast majority and achieve better user rating.

Thus, developers should focus on hit apps to achieve a spot in the relatively small screen of smart phones which physically constraint user choices. Our results also suggest developers to employ more flexible pricing policy. Also, we do not find any pattern of affection of discount promotion in the data.

Our findings suggest that, mobile app market may follow a different market structure than other online markets. First of all, in a highly connected world full of social networks and social apps, mobile market could be influenced by the tyranny of network effect which let users tend to choose the same app. Studies investigating the impact of social features on mobile app market would be beneficial. A second consideration is the diversity of users' tastes. Do users really have diverse needs in choosing most apps? Unlike books or music whose perception is highly subjective, a user's need for an app, e.g. a navigation app, tends to be more objective. However, for different categories of apps, e.g. games, the perception may be subjective as well. Diversity of consumer needs of different categories of apps is another point of research. All these open research problems could help researchers and developers in understanding the underlaying mechanism of mobile app market.

Developers or market operators may have the chance to change the market structure by providing a smart recommender system which better help consumers reach the niches. This has been proven to be beneficial in other online markets [8]. Currently, the recommendation is more based on current popularity of apps which contributes to the dominance of hit apps. How could recommender systems better enable users to explore the growing long tail where thousands of new apps are added to everyday? Is this able to change the market structure?

Finally, we want to mention two limitations of our work. Firstly, our data-set is limited to AppAware users who would yet have to be proven to be representative for the total Android user population. Secondly, we had to neglect the impact of in-app purchase and revenue of add's, which has been important sources of revenue to developers, too, besides the price of the app.

ACKNOWLEDGMENT

We thank 42matters AG for providing data to foster research.

REFERENCES

1. Iphone owners more willing to pay for apps than android phone owners, 5 2012. `http://www.cellular-news.com/story/39327.php`.
2. Anderson, C. *The long tail: Why the future of business is selling less of more*. Hyperion, 2006.
3. Brynjolfsson, E., Hu, Y. J., and Simester, D. Goodbye Pareto Principle, Hello Long Tail: The Effect of Search Costs on the Concentration of Product Sales. *Management Science 57*, 8 (June 2011), 1373–1386.
4. Brynjolfsson, E., Hu, Y. J., and Smith, M. D. From Niches to Riches : The Anatomy of the Long Tail. *MIT Sloan Management Review 47*, 4 (2006), 67–71.
5. Brynjolfsson, E., Smith, M. D., and Hu, Y. J. Consumer surplus in the digital economy: Estimating the value of increased product variety at online booksellers. *Management Science 49*, 11 (2003), 1580–1596.
6. Elberse, A. Should you invest in the long tail? *Harvard Business Review 86*, 07/08 (July 2008), 88–96.
7. Elberse, A., and Oberholzer-Gee, F. Superstars and underdogs: An examination of the long tail phenomenon in video sales. *Marketing Science Institute 4* (2007), 49–72.
8. Fleder, D. M., and Hosanagar, K. Blockbuster Culture' s Next Rise or Fall : The Impact of Recommender Systems on Sales Diversity. *Management Science 55*, 5 (2009), 697–712.
9. Girardello, A., and Michahelles, F. AppAware: Which mobile Applications Are Hot? In *Mobile HCI* (2010), 431–434.
10. Goel, S., Broder, A., Gabrilovich, E., and Pang, B. Anatomy of the Long Tail : Ordinary People with Extraordinary Tastes. In *WSDM* (2010), 201–210.
11. Lee, G., and Raghu, T. S. Product Portfolio and Mobile Apps Success: Evidence from App Store Market. In *AMCIS* (2011), 444–454.
12. Tan, T. F., and Netessine, S. Is Tom Cruise Threatened? An Empirical Study of the Impact of Product Variety on Demand concentration. In *ICIS* (2011), 1–18.

Ethnography for Large Scale User Trials

Donald McMillan
University of Glasgow, U.K.
Donny.McMillan@Glasgow.ac.uk

Matthew Chalmers
University of Glasgow, U.K.
Matthew.Chalmers@Glasgow.ac.uk

ABSTRACT

In this paper the spread of ethnographic techniques from anthropology to HCI and their applicability to large scale mobile trials is discussed. Each of the three main ethnographic tools, observation, analysis of recorded data and interviews, are described and the challenges such trials present for each of them explored.

Author Keywords
Mass Participation, Ethnography, User Trials

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

General Terms
Human Factors; Theory.

INTRODUCTION

The growth in the popularity of smart mobile devices has been rising as swiftly with their power and usability. In 2006 the number of mobile phones shipped worldwide topped 1 billion [1] with 64 million of them smartphones [2]. In 2011 the number of smartphones shipped was over 488 million, including tablets this number rises to 551 million mobile networked devices – 200 million more units than the combined global shipments of desktop, laptop and netbook PCs [4].

Smartphone ownership had risen to 42% of mobile subscribers in the USA and 44% in Western Europe by the end of 2011 with the UK and Spain above 50%, and is set to continue increasing [23].

While still not the predominant type of handset globally, with the number of smartphones shipped forecast to reach 1 billion units annually by 2016 [3] it can be seen that smartphones have been adopted into mainstream use and are no longer the solely for the highly technically competent or the early adopters.

Understanding the social practices of people engaging with mobile applications is of central importance not only to being able to design better systems but also in finding research methodologies that better deal with the challenges of mass participation trials, in much the same way that workplace studies have informed the design of CSCW systems

However with the movement to remote and mobile participants the practicalities of studying situations using traditional ethnographic immersion become incredibly difficult, if not impossible. In this paper the history and use of ethnography is explored, along with how its methods may possibly be extended to meet the challenges faced in understanding use in the wild.

ETHNOGRAPHY

Ethnography has its roots in anthropology and sociology and has, in some forms, become a poplar model for research in the HCI community [9]. Ethnographic research methods are being applied to a variety of projects, including the evaluation of mind map software [11], a proximity based mobile game [15], the working practice of nightclub DJs [5], the results of severing IT workers from their email [17] and the analysis of a mobile, interactive performance [32] at this year's ACM CHI alone.

Unlike many other scientific research strategies, the ethnographer as researcher is not typically a detached or uninvolved observer. The ethnographer collects data and gains insight through firsthand involvement with research subjects or informants.

From the standpoint of ethnography, the only plausible way to study social and cultural phenomena is to study them in action [20]. The formalised ethnography in anthropology is generally seen to have grown from the foundations of the then mainstream practice of ethnology, comparative analysis of different cultures using observational data, in the late 1910's [27] along with the rise in modern fieldwork.

Where previous ethnographic researchers relied on pidgin or interpreters and augmented their data sets with third party accounts from sailors and travellers the new wave, like Malinowski and Radcliffe-Brown, guided by Boas lived among their subjects for extended periods of time. They learned the local language, recorded local myths, customs and ceremonies is much greater detail than had been done before [27].

Ethnography developed as the study of cultures. Originally, the idea of a culture was tied to the notion of ethnicity and geographic location, but the areas in which ethnography has been applied has broadened this definition to include virtually any group or organisation. Using ethnographic methods researchers are able to study the 'culture' of a business, a sports club or the users of a particular system.

Ethnography in HCI

The emergence of ethnographic enquiry as a method of choice within HCI can be attributed to the fields of Computer Supported Cooperative Work (CSCW) and Participatory Design (PD), which imported ethnographic methods from anthropology and sociology to study the use of technology in

LARGE 3.0, September 21, 2012, San Francisco, CA, USA.

situ. The use of ethnographic enquiry within HCI has been argued to be a method of rich requirements capture [9]. Apart from the insight gained into social practice, its merits with respect to the design process include providing "a useful contrast to traditional methods of requirements capture" [29] and the engagement of users in the design process [30].

The difficulty ethnography faced when attempting to influence design has been highlighted [9], and the suitability of ethnography for the task of generating implications for design has been questioned [30]. It is not suggested that this in unobtainable, but simply that implications for design don't necessarily follow from ethnographic findings [30], or that ethnographic findings are more suited to identifying how people cope with existing technologies rather than inventing new ones [10]. The most influential ethnographic studies in CSCW did not provide design recommendations "but instead tried to uncover, in minute detail, the ways in which social order is produced in cooperative work settings." [28]

More recently the notion of evaluating the merit of ethnographic work carried out within HCI by the presence or absence of implications for design has been challenged [9, 10]. Dourish [9] suggests that HCI needs to distinguish between ethnography to inform system design and ethnography to study human computer interaction.

While ethnography purists argue that employing methods such as quasi-experimental statistical tests or dependent and independent variables results in 'dead knowledge' and that 'it is much better to be deeply interesting than accurately boring' [24] when human action and interaction are the subject of the research these tools are often employed in HCI research to support the claims of ethnographic enquiry and add confidence to claims of generalisability.

Ensuring that in reporting the results of ethnographic research the observations are given in context, following the ethnographic principle that people will behave differently under different circumstances, allows the ethnographic part stand on its own, without relying on other tools yet using them to report findings to the research community and leave the design to the designers. An ethnographic description may contain a large amount of information with direct value to design and evaluation but it is still a largely unconstrained and arbitrary narrative account. This raises problems of abstraction, generalisation and comparison and leads to a lack of cumulative research results [21].

THE TOOLS OF ETHNOGRAPHY

The formalised multi-method form of ethnographical research used today reduces the risks stemming from reliance on a single kind of data and makes triangulation possible, allowing the researcher to compare data collected by different methods to aid understanding [31].

This type of research is concerned with the interaction of events, actors and system — the study of any one of these hold very little meaning without the others, and the research itself is embedded in the social, interconnected world under investigation. Researchers practicing Ethnography therefore recognise that they are part of the world they are studying, and that they will have an effect on the subjects under investigation.

Two key issues in any ethnographic study are those of access and of field relations [6]. In trials conducted in the large, the depth and breadth of access, and therefore the researchers' relations with their participants, is inherently different than a traditional ethnographic enquiry. Where, in other types of ethnographical observation, the researcher will negotiate access to the setting and begin the relationships with the actors then, defining what is and what is not to be part of the research, here the researcher intervenes in the user's the setting by introducing, and potentially insisting upon the use, of an application — the application within and through which the actions and actors will be observed.

Blomberg et al. [7] characterise ethnography with four principles and three main techniques: it takes place in natural settings; it is based on the principle of holism, that is, particular behaviours must be understood in their respective contexts; it develops descriptive understanding; and it is grounded in a member's point of view. The main techniques they use are observation, video analysis and interviews. The use of each these techniques and their applicability to remote, large scale trials is discussed separately.

Observation

Observation is the primary means by which a researcher can examine the actions of a participant and the broader context in which the actions take place. In purely observational studies the actions can be open to possible misinterpretation by the researcher, a risk reduced in ethnographic studies by the researcher's long-term immersion in the environment.

There are two modes of observation, direct and indirect. In direct observation, the researcher is present in the subject's environment and watches the subject go about their everyday routine or perform a particular task. With no mode of recording the events that are being observed, one limitation is that events of interest may be missed by the researcher [30] and that there is no way to revisit the data [22].

The most commonly stated limitation of observational studies is the Hawthorne effect.

Proponents of the Hawthorne effect say that people who are singled out for a study of any kind may improve their performance or behaviour not because of any specific condition being tested, but simply because of all the attention they receive [26].

Such a view seems to indicate that the degree of attention paid to those participating in a study is positively correlated with any subsequent Hawthorne effect; a commonly held assumption being that no human-centred study is completely free from the Hawthorne effect [16]. However, the generalisability of the Hawthorne effect has recently been called into question [16]. Macefield [16] presents a full discussion on the limitations of such a generalisation with respect to usability evaluations. Similarly, Crabtree and Rodden propose that the Hawthorne effect is often overestimated when considering ethnographic studies in the workplace and home, simply because when in these environments people "have better things to do than impress or worry about the ethnographer" [8].

The cost of conducting direct observational studies as a primary means of enquiry in a large scale trial are, because of the global spread of participant, far beyond the perceived benefit. Indeed the use of observational studies of the use of a system, rather than the interaction with it, even in small scale mobile deployments are generally reserved for very short term studies of use in a particular location. Even if the large scale deployment resulted in enough users within a reasonable distance from the researcher to practically allow observations to take place, by no means a certainty in any App Store style deployment, turning users into participants willing to be observed in person can only be expected to be even more difficult than we found turning users into participants willing to be interviewed over the telephone was in the trial of Hungry Yoshi [18]. However, it would be possible to use these techniques with the local participants of a hybrid trial, such as those we recruited in [19], and use the data from a large body of remote participants to back up the claims of generalisiability should the type of system lend itself to observational studies.

Video and other Recorded Data

There are a number of advantages to video recording in ethnographic research. One advantage is the density of data that a visual recording provides [13]. In an ethnographic approach to research, the goal is to study real people in real situations, doing real activities. Video can provide more contextual data than audio data alone [12, 14], and the addition of further sources of data as are available on modern smartphones even more [25], such as the usage logs, audio recordings and mini-surveys. Indirect observation also helps alleviate the problem of the Hawthorne Effect.

This can give a more complete sense of the participants, the setting in which they function and the types of activities they engage in.

The greatest advantage of recording is permanence [13]. This allows an event to be experienced repeatedly, and with each repeated viewing, the observer can change focus to things not noticed at the time of recording or on previous viewings. Replaying the event also allows more time to contemplate before drawing conclusions, and hence serves to ward off premature interpretation of the data. Even a rare event, when captured, can be replayed repeatedly for a thorough analysis and intensive study.

A recording contains very little information on how typical an event is. Whether the event is frequent, unusual or unique must be supplemented by the ethnographer, by drawing on the time spent in the field as a participant-observer, or triangulating with other methods of data collection such as usage logs and survey responses.

The unspoken thoughts and feelings of a participant cannot be probed while watching a recorded event. Tacit knowledge and influential experience cannot be accounted for when relying only on observation. However it can be played back to the participants [14] in order to attempt to get them to recall and describe their thoughts, feelings and reactions at different points in time during a given event, thus giving information about the unobservable. A version of this technique was performed with the data, specifically location of use, recorded from participants in [19] presented back to them to explore the nature of their understanding of the trial process.

No recording can show every observable thing that happened, but only that which was occurring within the range of the camera lens, or the equivalent limit on the accelerometer, GPS or magnetometer sensors being recorded. The camera can no more provide accurate observations in the dark than a GPS can provide accurate locations inside a building. More of the context can be understood by recording a larger area and for a longer time than the specific event under investigation strictly requires, however a balance must be sought between the participants needs and expectations of privacy and the researchers desire for greater fidelity of data.

The ever reducing cost and increasing fidelity of recording instruments combined with the increasing complexity of tools available to synchronise and examine their output means that the quality of analysis that can be drawn from such data is getting better. However, as mentioned above, any ethnographic research is inherently contextual. The understanding of context that can be inferred from recording device's display, accelerometer, GPS, microphone and even front and rear facing cameras could only possibly be as good as its similarity to the understanding of the closest context which the researcher has previously experienced. The costs of such data, in terms of data transfer fees and the invasion of privacy of the user, seems to preclude using such a technique in the majority of cases.

Interviews

There are many limitations to interviewing as an investigative technique, the most obvious being the widely acknowledged discrepancy between what people do and what people say they do. Interviewees may also tailor their answers to suit what they think the interviewer wants to hear [22] or to maintain their presentation of self to the interviewer.

A less obvious limitation is that interviewing relies on a degree of reflective expertise on the part of the subject, and the ability to articulate their thoughts, feelings, and experiences even though one purpose of interviews is to gain insight into the thoughts, feelings, and experiences of subjects that may not otherwise be easily observed [22].

Additionally, the questions that are asked are limited by the assumptions of the researcher. While this may be useful in situations where the research question has a narrow focus, in more exploratory studies this may delimit the subsequent scope for potential and valuable findings. The implication of incorrect assumptions is most damaging in structured interviews, in which the researcher follows a script predetermined questions with no opportunity for deviation, clarification or explanation. Semi-structured interviews offer some purchase on this problem in that the researcher enters the interview with a loosely defined schedule and willingness to let the course of the interview be guided by issues that are raised as relevant by the subject.

While we found gaining access to remote participants for such interviews to be difficult [18] the fidelity of the data collected, and the ability to enter into a conversation with a participant to explore and verify the researcher's understanding of the context in which the activity under investigation occurred was invaluable. Indeed, the majority of the benefits of the hybrid trial conducted in [19] can be directly attributed to easy access to a local group of participants for repeated and in depth interviews. Building closer relationships with a subset of users of an application by offering influence in the development or early access to new features could provide a similar cohort of users researchers are able to call upon in the same way.

CONCLUSION

In conclusion, while the mainstay of traditional ethnography — direct observation — is incredibly difficult to apply to large scale mobile trials the methods developed for ethnographic analysis of recorded data and, to some extent, interviews can be expanded into the field of mass participation user trials. There remains a question of to what extent the understanding of context that is lost without direct observation can be gained through sensor data and remote interviews.

An interesting potential avenue for further work would be to run a hybrid mass participation trial, as conducted in [19], with two independent investigators. One researcher would conduct an ethnographic study on the local participants, the other would have access to the remote participants and conduct the trial without performing direct observations. The extent to which the lack of direct observation can be overcome by analysis of large volumes of high fidelity log data combined with remote interviews could then be determined by a comparison of the results.

REFERENCES

1. Global Mobile Phone Shipments Top 1 Billion Units in 2006. Strategy Analytics.
2. 64 million smart phones shipped worldwide in 2006 . Canalys, Feb. 2007.
3. 1 Billion Smartphones To Be Sold Each Year, July 2011.
4. Smart phones overtake client PCs in 2011. Canalys, Feb. 2012.
5. Ahmed, A., Benford, S., and Crabtree, A. Digging in the crates: an ethnographic study of DJS' work. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, ACM Request Permissions (May 2012).
6. Banister, P. *Qualitative methods in psychology: a research guide*. Open University Press, 1994.
7. Bloomberg, J., Giacomi, J., Mosher, A., and Swenton-Wall, P. Ethnographic field methods and their relation to design. Participatory design: Principles and practices, 1993.
8. Crabtree, A., and Rodden, T. Domestic routines and design for the home. *Computer Supported Cooperative Work 13*, 2 (2004), 191–220.
9. Dourish, P. Implications for Design. In *Proceedings of the 24th international conference on Human factors in computing systems*, ACM Press (2006), 541–550.
10. Dourish, P. Responsibilities and implications: further thoughts on ethnography and design. In *Proceedings of the 2007 conference on Designing for User eXperiences*, ACM (Nov. 2007).
11. Faste, H., and Lin, H. The untapped promise of digital mind maps. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, ACM Request Permissions (May 2012), 1017–1026.
12. Gass, S. M., and Houck, N. *Interlanguage Refusals: A Cross-cultural Study of Japanese-English* . Walter de Gruyter, 1999.
13. Grimshaw, A. D. Sound-Image Data Records for Research on Social Interaction. *Sociological Methods & Research 11*, 2 (1982), 121–144.
14. Iino, M. Issues of Video Recording in Ethnographic Studies. In *Annual Meeting of the American Association for Applied Linguistics* (1998).
15. Licoppe, C., and Inada, Y. 'Timid encounters': a case study in the use of proximity-based mobile technologies. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, ACM Request Permissions (May 2012).
16. Macefield, R. Usability studies and the Hawthorne Effect. *Journal of Usability Studies 2*, 3 (2007), 145–154.
17. Mark, G., and Voida, S. "A Pace Not Dictated by Electrons": An Empirical Study of Work Without Email. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems* (2012), 555–564.
18. McMillan, D., Morrison, A., Brown, O., and Hall, M. Further into the wild: Running worldwide trials of mobile systems. In *The 8th International Conference on Pervasive Computing* (2010), 210–227.
19. Morrison, A., McMillan, D., Reeves, S., Sherwood, S., and Chalmers, M. A Hybrid Mass Participation Approach to Mobile Software Trials. *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems* (Jan. 2012), 1311–1320.
20. Murchison, J. *Ethnography Essentials*. Designing, Conducting, and Presenting Your Research. Jossey-Bass, Jan. 2010.
21. Nardi, B. A. Activity Theory and Human-Computer Interaction. In *Context and Consciousness: Activity Theory and Human-Computer Interaction*, B. A. Nardi, Ed. The MIT Press, 1996, 7–16.
22. Preece, J., Carey, T., Rogers, Y., Holland, S., Sharp, H., and Benyon, D. *Human-Computer Interaction*. Addison Wesley Publishing Company, 1994.
23. Radwanick, S., and Aquino, C. Key Insights from 2011 and What They Mean for the Coming Year. Tech. rep., Mar. 2012.
24. Reason, P., and Rowan, J. *Human inquiry: a sourcebook of new paradigm research*. J. Wiley, 1981.
25. Reeves, S. Studying social organisation with video. In *Research Squared Conference* (July 2011), 1–10.
26. Rice, B. The Hawthorne effect: Persistence of a flawed theory. *Psychology Today 16*, 2 (1982), 70–74.
27. Risjord, M. The politics of explanation and the origins of ethnography. *Perspectives on Science 8*, 1 (2000), 29–52.
28. Schmidt, K. The critical role of workplace studies in CSCW. In *Workplace Studies: Recovering Work Practice and Informing Design,*, P. Luff, J. Hindmarsh, and C. Heath, Eds. Cambridge University Press, 2000.
29. Shapiro, D. The Limits of Ethnography: Combining Social Sciences for CSCW . In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, ACM Press (1994), 417–428.
30. Sharrock, W., and Hughes, J. A. Ethnography in the workplace: Remarks on its theoretical bases. *Team Ethno Online 1*, 1 (2001), 1–21.
31. Suchman, L. A. *Plans and situated actions: the problem of human-machine communication*. Cambridge University Press, 1987.
32. Tolmie, P., Benford, S., and Flintham, M. Act natural: instructions, compliance and accountability in ambulatory experiences. In *Proceedings of the 2012 ACM annual conference extended abstracts on Human Factors in Computing Systems* (2012), 1519–1528.

NFC Heroes - Observing NFC Adoption through a Mobile Trading Card Game

Lukas Murmann
Technische Universität München
Arcisstraße 21, 80333
München, Germany
lukas.murmann@tum.de

Florian Michahelles
Auto-ID Labs,
ETH Zurich
Switzerland
fmichahelles@ehtz.ch

Matthias Kranz
Luleå University of Technology
Department of Computer Science,
Electrical and Space Engineering
Luleå, Sweden
matthias.kranz@ltu.se

ABSTRACT

Near-field Communication (NFC) technology finally starts to proliferate on modern smartphones, enabling researchers to conduct researchin the real world. The research question for this work is to learn about the distribution of NFC tags in the wild. As there is, for good and for bad, no central registry or database of NFC tags, we propose a game-based approach to capture the adoption of NFC solutions and technologies.

We first report on the development process an NFC-based game. We then present the game logic and implementation, share our experiences from two release cycles on Google's *Play Store* and finally report on initial results and lessons learned during the whole process.

Author Keywords

mobile games; apps; research in the large; barcode; NFC

INTRODUCTION AND MOTIVATION

The number of products using NFC [6, 8] technology is increasing rapidly. As there is no central registry or database of NFC applications or tags, the goal of our project is to capture the current state of deployment of NFC solutions.

NFC is used in a wide range of applications, from gaming consoles like the upcoming *Wii U* [4] to payment solutions like Google's Wallet[1]. There is an increasing number of (as of mid of 2012 only) Android smartphones that incorporate NFC readers[2]and recent versions of the Android SDK provide libraries that provide easy access to the underlying hardware. To learn about the situation 'in the wild', we release our research app to the public via the *Play Store*, disguised as free game, called *NFC Heroes*[3].

We describe how we designed an approachable game for Android that makes use of the platform's NFC capabilities and gives users in-game incentives to scan and upload NFC tags.

We will present the process of publishing the game on Google's *Play Store* and how we integrated Facebook as an identity provider. Our goal was to bring a research application to a consumer platform to conduct actual studies on human-computer interaction (HCI). We will share the lessons we learned during that process, both in terms of direct user feedback and number of users our game did attract.

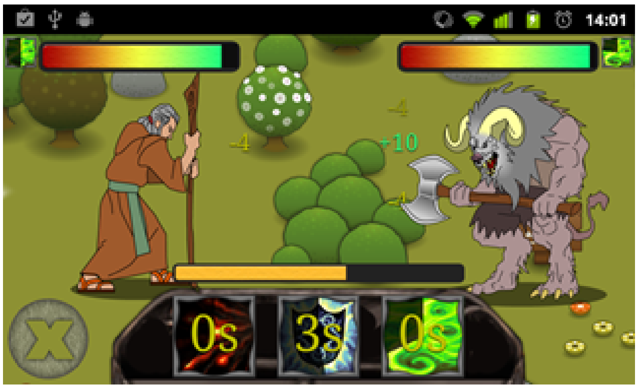


Figure 1. The player has selected three spells and fights a randomized opponent.

RELATED WORK

The idea we follow here to use app stores and markets for UbiComp research has been discussed by Cramer et al. [3].

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 [5] or capacitive sensors [9], incorporated in pervasive mobile devices.

Barcodes and Visual Codes

In the early 90's, the *Barcode Battler*[4] 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,

LARGE 3.0, September 21, 2012, San Francisco, CA, USA.

[1] `http://www.google.com/wallet`
[2] `http://www.nfcworld.com/nfc-phones-list/`
[3] `https://play.google.com/store/apps/details?id=com.heroesgame`
[4] `http://en.wikipedia.org/wiki/Barcode_Battler`

but the *Barcode Battler 2* could also be connected to the NES and SNES gaming consoles[5].

Nintendo pursued the idea of using real-world, physical cards to influence game events further. In 2001, they released the e-Reader[6], an accessory to the Gameboy Advance that could read proprietary visual codes.

Recent games do not rely on dedicated hardware to read barcodes, but make use of the camera integrated into modern smartphones. In *Barcode Empire* [2], players can collect real-world product in order to expand their 'Empire'; *Barcode Beasties*[7] is a fighting game that lets players improve their avatar (beast) by scanning barcodes before they battle against a randomized opponent.

NFC

The Mattel *Hyperscan*[8] 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.

Broll et al. experimented with NFC-based games on public displays [1]. Nokia Research launched a website dedicated to NFC-based games [7]. 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 [4].

CONCEPT

NFC Heroes is a virtual trading card game for Android phones, slightly inspired by the *Magic: The Gathering*[9] 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.

Core Game Design

NFC Heroes is a fast-paced fighting game where a computer-controlled monster competes against a hero controlled by the player (see Fig. 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.

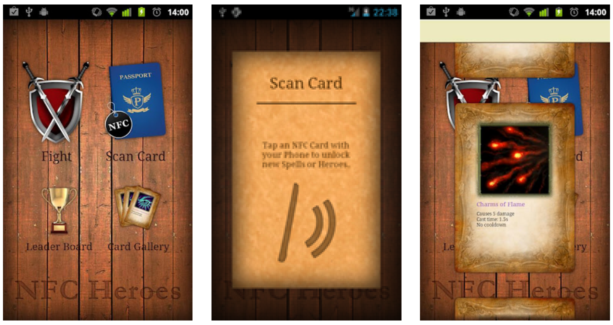
Installation and First Start

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* application. A local account can be upgraded to a linked account at any later point.

Using NFC to unlock new Spells and Heroes

After logging in, the 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.



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

Causes 5 damage
Cast time: 1.5s
No cooldown

(a) Main activity (b) Scanning the card (c) Receving a bonus

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

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 Fig. 2(a) 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.

[5] See `barcodebattler.co.uk` for more information

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

[7] `http://barcodebeasties.com`

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

[9] `http://en.wikipedia.org/wiki/Magic_the_gathering`

Progress and Leaderboard

Games that aim to provide long-term motivation to players must provide ways for player to progress in the game [10]. *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.

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[10] for our application logic and MongoDB as a non-relational database.

Android Client

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[11]. All Android phones with NFC support (Android version 2.3 or higher) and are thus supported by our game.

The Facebook SDK[12] 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*.

Server

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.

LESSONS LEARNED

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

App Stores make short development cycles possible

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

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

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[13], 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.

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.

Many downloads on non-NFC phones

[10] `http://nodejs.org`

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

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

[13] `http://androidforums.com`

Statistics from the app store indicate that many downloads and active installations are on 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.

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 much less often. We from this conclude to make your app only visible to supported devices.

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 is increasing at a rate at 10 downloads per day. However, we at the same time face uninstalls (constant rate of 8 uninstalls per day), so that the total number of active installations is increasing rather slowly, totaling at 100 installs one month after the feature-complete version was released. 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.

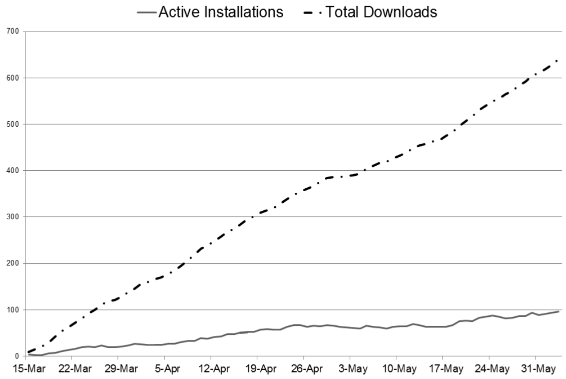


Figure 3. Total number of downloads and active installations are growing at a constant rate.

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.

CONCLUSION

As illustrated by Fig. 3, 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.

So far, 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.

REFERENCES

1. Broll, G., Graebsch, R., Scherr, M., Boring, S., Holleis, P., and Wagner, M. Touch to play – exploring touch-based mobile interaction with public displays. In *Proceedings of the 2011 Third International Workshop on Near Field Communication*, NFC '11, IEEE Computer Society (Washington, DC, USA, 2011), 15–20.

2. Budde, A., and Michahelles, F. Product empire - serious play with barcodes. In *IOT*, F. Michahelles and J. Mitsugi, Eds., IEEE (2010).

3. Cramer, H., Rost, M., Belloni, N., Bentley, F., and Chincholle, D. Research in the large. using app stores, markets, and other wide distribution channels in ubicomp research. In *Proceedings of the 12th ACM international conference adjunct papers on Ubiquitous computing - Adjunct*, Ubicomp '10 Adjunct, ACM (New York, NY, USA, 2010), 511–514.

4. Engadet. Leaked Rayman Legends for Wii U trailer showcases NFC feature. http://www.engadget.com, 2012. 27 April 2012.

5. Möller, A., Roalter, L., Diewald, S., Scherr, J., Kranz, M., Hammerla, N., Olivier, P., and Plötz, T. Gymskill: A personal trainer for physical exercises. In *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on* (march 2012), 213 –220.

6. NFC Forum. About NFC. `http://www.nfc-forum.org/aboutnfc`, 2012.

7. Nokia Research Center Palo Alto. Nokia NFC Games. `betalabs.nokia.com/apps/nfcgames`, 2011. 15 October 2011.

8. Want, R. An introduction to rfid technology. *IEEE Pervasive Computing 5* (2006), 25–33.

9. Wimmer, R., Holleis, P., Kranz, M., and Schmidt, A. Thracker – using capacitive sensing for gesture recognition. In *Proc. 26th IEEE International Conference on Distributed Computing Systems Workshops ICDCS Workshops 2006* (2006), 64.

10. Zicherman, G. *Gamification by Design*. O'Reilly Media, 2011.

Measuring Latency of Touch and Tactile Feedback in Touchscreen Interaction Using a Mobile Game

Niels Henze
Institute for Visualization and Interactive Systems
University of Stuttgart, Germany
niels.henze@vis.uni-stuttgart.de

Benjamin Poppinga
OFFIS – Institute for Information Technology
Oldenburg, Germany
poppinga@offis.de

ABSTRACT

Touchscreens currently become the dominant means of interaction with mobile phones. As with all interaction technologies, users face latency when using touchscreens. After touching the screen with the finger there is a delay until the touch is recognized. If the phone provides tactile feedback through a vibration motor there is a further delay until the motor starts to move. Latency can limit the user experience and even the users' performance. In this paper we analyse the timing of touch events and activating the vibration motor. We distributed a game to players around the globe and measure the timing using the phones' accelerometer. Based on a small dataset we analyse latency for a single device type. Results suggest that the time between the finger hits the screen and the event is delivered to applications is about 70ms. We further find that the vibration motor starts to fully hit the device around 90ms after activation.

Author Keywords

touchscreen, latency, tactile feeback, virtual keyboard, mobile phone, public study.

ACM Classification Keywords

H.5.2 User Interfaces: Haptic I/O; H.5.2 User Interfaces: Input devices and strategies; H.5.2 User Interfaces: Interaction styles

General Terms

Design, Human Factors, Experimentation.

INTRODUCTION AND BACKGROUND

At least since the introduction of the iPhone, mobile phones with touch screens began to dominate the smartphone market. Today, all major phone makers have touchscreen devices in their portfolio. In contrast to earlier devices, today's smartphones are operated by directly touching the screen with the fingers and only very few devices have a physical keyboard. Therefore, most users rely on virtual keyboards and other widgets that are operated by touching virtual buttons shown on the screen. While touch screens are well suited for direct manipulation, they suffer from the lack of feedback they provide. In contrast to physical buttons, users cannot feel the position of virtual keys and do not get tactile feedback if a button is hit. A common approach to compensate this lack of tactile feedback is imitating natural feedback using vibration. The standard Android 2.1 keyboard, for example, activates the phones' vibration motor every time the user hits a key.

While it is intended that touchscreens provide direct feedback, sensing and processing needs time. After a finger touches the screen it is sensed by the touch screen. Afterwards, the generated touch event is processed by the phone operating system and UI framework. Finally, application or build-in UI widget has to react to the touch event. All these steps require a certain amount of time and this latency affects the user's experience [2] and can even reduce the user's performance [4]. It is clear that the latency introduced at the various steps should be as short as possible to increase users' satisfaction.

Kaaresoja and Brewster tried to analyse [3] latency of mobile phones. They build a custom multimodal latency measurement tool for touchscreen interaction. The tool is used to measure latency of four mobile phones. The tool is used to measure latency for the respective phones' native dialer application and a text editor. As we do not know about the processing inside these application and the applications differ between the phones we learn nothing about general capabilities of the phones itself. A practical limitation of a custom apparatus to measure a phone's capabilities is that the investigator has to have access to the respective phone. Considering the large amount of different smartphones available today, it seems impossible to address even a small fraction of them. Furthermore, not all individual devices of a certain type might be equal. Some devices will have malfunctions and there might be even large variations between different devices of the same model. Measuring each device by hand does not scale very well.

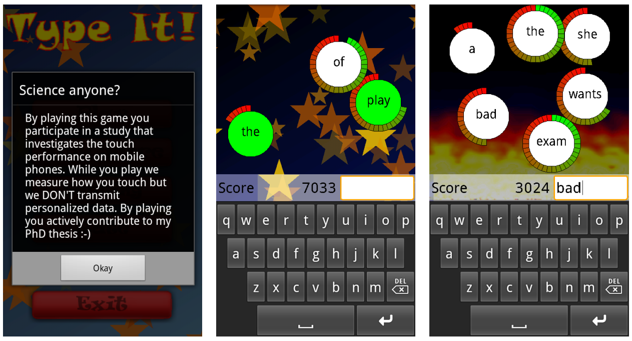
In this paper we describe an approach to measure the latency of recognizing touch events and providing tactile feedback that naturally scales to a large number of devices. We use a mobile game to collect touch events. While players touch the screen and eventually the vibration motor is activated we record the phone's movement using the build-in accelerometer. The apparatus is made publicly available to record data from all kinds of devices. We provide a preliminary analysis of the latency by exemplarily analysing the data received for Samsung Galaxy Y devices.

APPARATUS FOR FOR COLLECTING TOUCH EVENTS

We used the game *Type It!*[1] [1] to collect keystrokes on a virtual keyboard from a number of different devices and diverse participants to investigate the latency regarding touch events

[1] Type It! in the Android Market: `http://bit.ly/Type_It`

LARGE 3.0, September 21, 2012, San Francisco, CA, USA.



Science anyone?

By playing this game you participate in a study that investigates the touch performance on mobile phones. While you play we measure how you touch but we DON'T transmit personalized data. By playing you actively contribute to my PhD thesis :-)

Okay

Figure 1. Screenshots of the apparatus for analysing text entry showing microlevels with different numbers of words.

and tactile feedback. Words are presented to the player and the task is to type these words using a virtual keyboard.

Game play

The game play focuses on collecting basic keystrokes that form independent words. Words are presented to the player and the task is to type these words. The game is structured in three stages called stars, water, and fire. Each stage contains four levels and each level consists of multiple words that must be typed. As shown in Figure 1 the keyboard is displayed in the lower half of the screen and the words are shown in the upper part of the screen. While playing, words are presented in white circles with a fixed size. A circular progress bar around the circles shows the remaining time until the word must have been typed. The bar is coloured from red to green to also highlight the remaining time. While the time to type a word expires, the progress bar gets shorter. The available time to type a word depends on the level and the number of characters. Depending on the level, multiple words are presented simultaneously and can be typed in any order.

A word's characters must be typed to complete it. While typing, the characters appear in a textbox just above the keyboard. The player must confirm the words by either tapping the space bar or the enter key. If a word has been typed correctly the word's background becomes green, the progress bar accelerates, and a rattle sound is played. If the progress bar gets empty the word disappears. To make a game out of the basic task the player must complete a word in a certain timeframe. The timeframe is reduced from word to word while the player proceeds through a level and also depends on the word's number of characters. Players receive a penalty point if a word has not been completed in the given timeframe. The game is lost when the player collected three penalty points in one level. Players receive scores when they complete a word. The faster a word is typed the higher the score.

To increase the study's internal validity, the same keyboard is used for all devices. We used the source code of the standard Android 2.2 ('Froyo') keyboard as basis. The Android keyboard is designed to scale across different devices, screen sizes, and resolutions. We adapted the keyboard by removing keys that are not required to play the game and added code to measure the players typing behaviour. We made the game visually appealing to motivate intensive usage. Each stage has a different animated background shown in Figure 1. The total score is shown above the keyboard next to the text box. Furthermore a player receives "badges" when successfully completing a level or achieving other goals. To increase the long term motivation we implemented a global and a local high score lists. Players can share their score via twitter if they achieve a high score.

Measures and consent

We collected various data about the characteristics of the used devices and the performance of the players. An unique identifier for each installation is derived from a device's "Android ID" using a hash function. Furthermore, we collect the user's locale (e.g. "en_GB" or "es_ES"), the device's type (e.g. "GT-S5360" for the Samsung Galaxy Y), the time zone, and the resolution. We log the words presented to the player and all keystrokes both with a timestamp.

The properties of the used device are transmitted to our server when the game is started. The data collected while playing is transmitted after a level is completed no matter if it was successfully completed or not. The data is stored internally on the phone and retransmitted after the next level is completed if the transmission fails.

We do not collect any data that allows identifying individual players or installations. We decided to clearly inform players about the fact that data is collected in order to act ethically and to conform to corresponding legislation in many countries. The modal dialog shown in Figure 1 (left) tells players that they are about to participate in a study when the game is started for the first time.

ANALYSIS OF LATENCY

We intended to measure two aspects of latency. First, we are interested in measuring the time it takes between a user touching the screen and an according touch event is delivered to the application level. Second, we wanted to learn about the time it takes until output of the vibration motor kicks in and tactile feedback is provided to the user.

Design

To measure the timing of touch events we just intended to observe the devices' acceleration before and after the event is delivered to the application level. To investigate the effect of the vibration motor we designed an independent measures experiment. In the control condition players do not receive tactile feedback. In the experimental condition we activate the vibration motor right after it is recognized that the finger lifts of from the keyboard again, i.e. a touch up event is detected.

Apparatus

We used the game TypeIt! described above to collect touch events. In the experimental condition the vibration motor is activated for 100ms right after the touch up event is delivered to the application layer.

To measure how the phone is accelerated we injected code that continuously records the data delivered from the build-in accelerometers. The frequency of the measurement depends on the respective phone model. For typical devices (e.g. the Google Nexus S) the frequency is 50Hz. That means that one sample is recorded every 20ms. The Android sensor framework delivers the acceleration for the three axes as 32bit floating point values and an additional timestamp for each sample is also required. Thus, measuring one second of interaction results in 800 bytes of data, 10 minutes result in 480 kilobytes, and 10 minutes by 10,000 players result in 4.8 gigabytes. To reduce this amount of data we convert the acceleration values to 16 bit fixed point values. In addition, the recorded data is compressed using GZIP. Thereby, we can reduce the amount of data that needs to be send to our server for 10 minutes of play to about 210 kilobytes. The data is submitted to our server via HTTP to ensure that requests are not blocked by mobile network providers. We use multipart HTTP post request to minimize the overhead induced by the protocol. If the recorded data cannot be transmitted to our server due to a lack of network connectivity the data is simply discarded.

Participants

After integrating code to measure the devices acceleration we published an update of TypeIt! in Google Play (Google's application store for Android devices formally known as the Android Market) on the 5th of June 2012. Over three days we collected data from 937 installations. In total users played 6,077 levels and produced 248,354 touch events. We received data from 484 devices that played 3,518 levels and produced 145,283 touch events for the control condition. For the experimental condition, we received data from 453 devices that played 2,559 levels and produced 103,071 touch events.

Preparation

Since we collected the data only for three days and the individual device types have to be treated individually we can only report preliminary results for one selected device. The most common device in the dataset is the Samsung Galaxy Y. The Galaxy Y is typical entry level Android device. Its typical operating system is Android 2.3.5 Gingerbread and it has a 3.0 inch capacitive touchscreen with a resolution of 240x320 pixels. For this particular device we recorded 13,556 touch events from 67 devices for the control condition and 18,128 touch events from 68 devices have been recorded for the experimental condition.

To analyse the timing of touches we select a window of ± one second around each touch event. For this window we compute the magnitude of the acceleration for each sample. We interpolate the sample to get granularity of one kHz. The magnitudes in the window are normalized to reduce the effect of outliers. Almost the same process is used to determine the effect of the vibration. Instead of using the pure acceleration we derive the jerk from the acceleration. Jerk is the first derivation of acceleration just like acceleration is the first derivation of speed.

Results

To derive the average time between a touch down and the according touch up event we removed outliers by removing data more than three standard deviations from the average. The average time, for our game, between a touch down and the according touch up event is 87ms (SD=33ms). Figure 2 show the acceleration for the control condition of the Galaxy Y centred around the time the touch down event is delivered to the application layer (i.e., at 0ms). We can distinguish between 5 events during a touch. The first event starts around -70ms when the device starts to accelerate. At -10ms the maximal force is applied to the device and it starts to swing back again. 10ms later the touch event is delivered to the application layer. The device finishes swinging back at 20ms and the touch is completed at 90ms. The whole process of a single touch takes around 160ms.

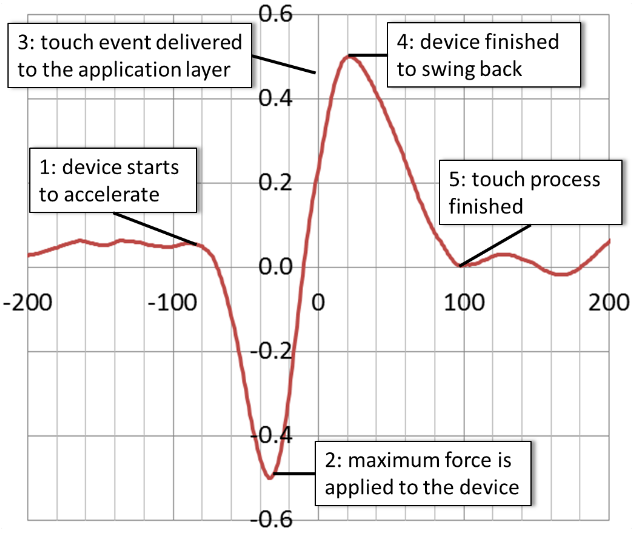


Figure 2. Average acceleration of Samsung Galaxy Y devices 200ms before and after a touch down event has been handed to the application layer.

Figure 3 compares the jerk for the control condition and the experimental condition. The figure is centred on the time the touch down event is delivered to the application layer (i.e., 0ms). Around 10ms after the touch up event was delivered to the application layer the device perceives the maximum jerk from the finger that lifts of the screen. We assume that at around 30ms the vibration motor slowly starts to kick in. It seems to take until 90ms for the vibration motor to apply the full force. At 270ms the both conditions reunite and the devices in both conditions go back to normal.

Discussion and Limitations

The whole process of a touch without tactile feedback by a vibration motor takes around 160ms. This includes the time between the touch down and touch up events are received by the application layer. For the used game that requires to type on a virtual keyboard the average time between up and down event is 87ms. The latency for touch down events is 70ms. If the vibration motor is activated after the touch up event is

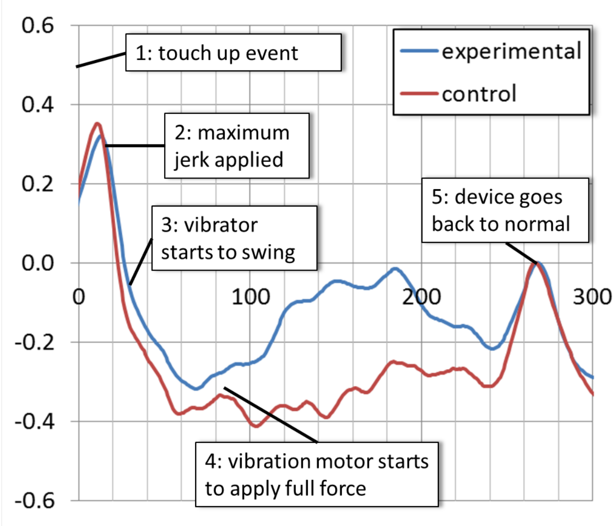


Figure 3. Average jerk of Samsung Galaxy Y devices after a touch up event has been handed to the application layer. The vibration motor has been activated right after the touch event in the experimental condition only.

received it takes more than 90ms until the vibration motor is fully activated and the vibration impacts the device up to 270ms. Our results suggest that there is quite a bit of latency at least for the Samsung Galaxy Y. Providing tactile feedback using the build-in vibration motor massively extends the duration of a touch procedure. We can only guess but it might be assumed that the tactile feedback might therefore not always be beneficial for the user.

The conducted study is seriously limited in a number of ways. First of all, our analysis is only based on a small dataset that only allows a brief analysis of a single device type. Furthermore, we analyse devices using the devices' own sensors. In the unlikely case[2] that the accelerometer has a relevant latency this would certainly limit our conclusions. We only record when the touch events reach the application layer. The UI framework receives the events earlier and tactile feedback produced by the UI framework might has a lower latency. However, at least for the standard keyboard, this is not the case. We cannot calibrate the used sensors but it would be required to, at least, compare our measurements with professional external measuring tools for selected devices.

CONCLUSION

In this paper we analysed the latency around touch events. Using a game published in Google Play we collected data from 937 installations. While players touch the screen we measure the acceleration of the phone using its build-in sensors. To investigate the effect of the vibration motor we conduct an experiment that compares touch events with and without providing tactile feedback. We report preliminary results from an analysis of the data received from Samsung Galaxy Y devices. Our data suggest that touch events are recognized by the phone approximately 70ms after the user initiated the touch. The whole process of one touch seems to take 160ms. Providing tactile feedback can take 270ms and it can take up to 90ms until the vibration motors applies its full force.

The latency of the vibration motor is not only relevant for interaction with the touchscreen. Researchers started to use the vibration motor also for other purposes. Pielot et al. [5] as well as Rümelin et al. [6], for example, used tactile feedback produced using the phones' vibration motor to provide navigation instruction. Knowing about the characteristics of the feedback might be helpful for designing such tactile patterns and potentially even to adapt the feedback to the respective device.

In our future work we intend to extend the analysis to further devices. This would enable to compare different devices and different form factors. Furthermore, it is required to compare our measures to calibrated measurement tools. Another interesting direction is to look at current smartphones' other sensors. In particular, for common Android devices, the gyroscope provides a much higher sampling rate than the accelerometer. The Google Nexus S delivers gyroscope data at about 1,000Hz while accelerometer data is typically only delivered at 50Hz. However, we need to find efficient ways to transmit, store, and process the huge amount of this data first.

REFERENCES

1. Henze, N., Rukzio, E., and Boll, S. Observational and experimental investigation of typing behaviour using virtual keyboards on mobile devices. In *Proc. CHI* (2012).

2. Kaaresoja, T., Anttila, E., and Hoggan, E. The effect of tactile feedback latency in touchscreen interaction. In *Proc. WHC* (2011).

3. Kaaresoja, T., and Brewster, S. Feedback is late: Measuring multimodal delays in mobile device touchscreen interaction. In *Proc. ICMI* (2010).

4. Kaaresoja, T., Hoggan, E., and Anttila, E. Playing with tactile feedback latency in touchscreen interaction: two approaches. In *Proc. INTERACT* (2011).

5. Pielot, M., Poppinga, B., Heuten, W., and Boll, S. Pocketnavigator: Studying tactile navigation systems in-situ. In *Proc. CHI* (2012).

6. Rümelin, S., Rukzio, E., and Hardy, R. Naviradar: a novel tactile information display for pedestrian navigation. In *Proc. CHI* (2011).

[2] Seriously, that is really unlikely!

imprint

Research in the Large, LARGE 3.0
Volume 3 - September 21st, 2012

Editor
Benjamin Poppinga
OFFIS - Institute for Information Technology
Escherweg 2, 26121 Oldenburg, Germany
Phone: +49 441 9722 - 138
E-Mail: poppinga@offis.de

organisers

Benjamin Poppinga
OFFIS - Institute for Information Technology
Henriette Cramer
SICS @ Mobile Life
Matthias Böhmer
DFKI
Alistair Morrison
University of Glasgow
Frank Bentley
Motorola Mobility
Niels Henze
University of Stuttgart
Mattias Rost
SICS @ Mobile Life
Florian Michahelles
ETH Zürich

acknowledgements

Thanks to the programme committee for providing excellent reviews: Antti Oulasvirta, Janko Timmermann, Karen Church, Kasper L. Jensen, Martin Pielot, and Matthew Chalmers.

copyright