Karlsruhe Institute of Technology

Studies and design of user interaction for an acoustical object recognition program via smartphone

Bachelor's Thesis
from

Marcel Danz

Chair of Pervasive Computing Systems/TECO
Institute of Telematics
Department of Informatics

Reviewer #1:	Prof. Dr. Michael Beigl
Reviewer #2:	Prof. Dr. H. Hartenstein
Supervisor:	PD Dr. Andrea Schankin

Editing Time: 05/04/2015 – 10/02/2015

Abstract

Participatory sensing is made possible in today's life through the ubiquity of smartphones, and can thereby be used to increase public health care, social and political involvement of normal citizens, or help managing resources on a large scale. But where other sensing methods do not need any, or only tangible user involvement in the measuring process, acoustic object recognition needs expert users to generate viable audio samples. One example for acoustic object recognition is the task to find out what is contained in a Kinder surprise egg with the help of the sound the egg makes, when it is shaken. In order to study the user interaction needed to bring users to record viable audio samples via smartphone, a formative process is conducted. During this process a low-fidelity prototype is implemented which is evaluated in a formative evaluation. The findings of this evaluation are then used to implement a high-fidelity Android™ prototype which is evaluated in a final summative evaluation. As first variable to evaluate the prototype, the user experience is measured. The results show that the prototype is intuitive to use and instructs the user well, which results in a good user experience. As second variable, the recorded audio samples are analyzed. The results show a difference between laboratory and field tests. However, this effect appears due to the lack of a good audio preprocessing and the use of a poor classifier (31.13% correct classification rate). This suggests that the used classifier needs to be noise resistant to perform well in all environments. Between expert and amateur users no significant differences could be found. Concluding it can be said that amateur users can perform similar to expert users in recording audio with a smartphone, when given the right instructions. Hence participatory sensing yields the potential to measure data using amateur users, which otherwise would require expert users to make the measurement.

Kurzfassung

Participatory Sensing ist heutzutage möglich, da Smartphones allgegenwärtig im täglichen Leben zu finden sind. Dadurch können sie dazu eingesetzt werden um die öffentliche Gesundheitsvorsorge zu verbessern, um einfache Bürger politisch und sozial besser zu integrieren, oder um dabei zu helfen Ressourcen in großem Rahmen zu verwalten. Wo jedoch einige Messverfahren keine, oder nur wenig Nutzer Partizipation für den Messprozess benötigen, benötigt akustische Objekterkennung erfahrene Nutzer um nutzbare Tonaufnahmen zu generieren. Ein Beispiel für akustische Objekterkennung ist die Aufgabe, am Schüttelgeräusch eines Kinder Überraschungseis herauszufinden was es enthält. Um herauszufinden wie die Nutzerinteraktion mittels Smartphone gestaltet sein sollte, um Nutzer dazu zu zu bringen verwendbare Tonaufnahmen zu erzeugen wurde ein formativer Prozess durchgeführt. Während dieses Prozesses wurde ein Papierprototyp implementiert, der dann in einer formativen Studie evaluiert wurde. Die Ergebnisse dieser Studie wurden dazu verwendet einen High-Fidelity Android™ Prototyp zu implementieren, der wiederum in einer abschließenden summativen Studie evaluiert wurde. Die erste Variable zur Evaluation dieses Prototyps war das Nutzererlebnis welches zeigte, dass der Prototyp intuitiv zu benutzen ist und den Nutzer gut instruiert. Als zweite Variable wurde die Qualität der Tonaufnahmen evaluiert. Die Ergebnisse zeigen einen Unterschied zwischen den Labor- und Feldtests. Dieser Effekt ist jedoch dadurch zu erklären, dass keine gute Tonvorverarbeitung und nur ein mäßig funktionierender Klassifikator (korrekte Klassifikationsrate von 31.13%) zur Verfügung standen. Das lässt annehmen, dass ein Klassifikator für diesen Anwendungsfall eine gute Rauschresistenz benötigt. Zwischen Experten und Amateuren konnte hingegen kein signifikanter Unterschied festgestellt werden. Deswegen kann abschließend festgehalten werden, dass Amateure in der Lage sind ähnlich gute Tonaufnahmen aufzunehmen wie Experten, wenn ihnen die die richtigen Instruktionen gegeben werden. Dadurch hat Participatory Sensing durch Smartphones auch das Potential Amateure zum messen von Daten zu verwenden, welche ansonsten für den Messvorgang Experten benötigen würden.

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet habe.

Karlsruhe, den 1. Oktober 2015

M. Dahl

Contents

List of Figures

1. Introduction

Today's smartphones are omnipresent and thereby provide a vast amount of mobile sensors. These sensors can be used for uncountable applications, from activity tracking as used in [Inc.15a, Inc.15b, Inc.15d], up to great scale sensor networks as in [Foth08, LEMM+08, BEHP+06, MIT15]. But acquisition of audio samples for acoustical object recognition is a widely unexplored field, even if microphones had been the first sensors established in mobile devices.

In this thesis, studies are conducted and designs are explored to find promising approaches for teaching amateur users to record viable audio samples for acoustical object recognition via smartphones. This applies especially to acoustic object recognition applications where the users have to generate the acoustic signals themselves. These applications often need some kind of expert knowledge, about how to generate the desired acoustic signal, and how to record it properly to ensure that it is clean enough to be usable by the background classifier. Examples for such applications are diagnosis of car engines, finding a ripe melon among other non-ripe melons, or even find out what is contained in Kinder surprise eggs without having to open them.

Here the last example will be explored in greater detail. The focus of the present thesis is on the design of the user interface (UI) used to perform the analysis and to collect the recorded samples. Approaches on the background classifier will be discussed in a the collaborative master thesis by Christian Voigt and therefore are not described here. The result of these two collaborative works is a mobile application named "SkillStore – Surprise Edition" for the Android™ operating system (OS).

To approach the matter of this thesis, a formative process was conducted. This process consists of a conceptual analysis of the new system (section 3.1), which resulted in a low-fidelity paper prototype (section 3.2). The low-fidelity prototype was then evaluated in a formative study (section 3.3). The findings of the formative evaluation were then used to implement a high-fidelity Android prototype (chapter 4), which was evaluated in a final summative study (chapter 5). At the end of this thesis conclusions are drawn from this formative process, and some further ideas on the matter are given (section 6).

The summative study approaches the following questions:

1. Does the prototype instruct the users well enough to perform the task successfully?
2. Is the task designed intuitively for the users?
3. Does the prototype perform similarly in the laboratory and in the field (supermarket)?
4. Does the prototype instruct amateur users (in the field of audio recording) well enough, that they perform as good as experts would perform?

Before answers to these questions will be given, the next chapter will present some related work on this topic.

2. Related Work

Participatory sensing is involved in a wide range of research areas. Burke *et al.* imagined applications in the following areas: urban planning, public health, cultural identity and creative expression, and natural resource management [BEHP+06]. *"Participatory sensing will task deployed mobile devices to form interactive, participatory sensor networks that enable public and professional users to gather, analyze and share local knowledge"* [BEHP+06]. That is how Burke *et al.* envisaged the future of sensor networks and shaped the term participatory sensing in 2006, when the technology was on the verge to the release of the first smartphones. Nowadays smartphones are ubiquitous in out society, with 1,244.89 million units sold in 2014 worldwide [Stat15], see A.1 for more details. Additionally small, wearable gadgets have been developed, which provide extra functionality (e.g. sensors) and can be accessed wirelessly via smartphones. Thus all tools required for participatory sensing are present in everyday life.

Today several research groups are engaged in the matter of participatory sensing. These research groups concentrate their efforts on applying the technology in the urban environment, thereby the terms urban sensing [LEMM+08] and citizen science [Foth08] were formed. Urban sensing is concerned with using the ever growing mass of mobile sensors for big data projects like Google© Street View [Inc.15c], or MIT SENSEable city lab [MIT15]. Citizen science on the other hand researches the issue of how the citizens can be empowered in such a way that they can change their environment, providing them with the tools to create and participate in movements that aim to aid scientific inquiries, environmental health policy, and communities' decision-making [Foth08, p. 418]. Paulos *et al.* for example provided taxi drivers and students with GPS loggers and carbon monoxide sensors. These devices had to be carried around Accra, in Ghana, and dropped off at the lab every evening. In order to improve this technique Paulos *et al.* also developed a carbon monoxide sensor, which can be integrated in mobile phones, and sends its measurements every time the user makes a phone call or writes a text message [Foth08, p. 429]. In another study, Dutta *et al.* developed a system, which uses a handheld air pollution sensor in combination with a mobile phone, to measure the air pollution values around the user and send them to a web portal for processing [DAKM+09].

Both of these introduced systems do not require the user's active participation in the measuring process. Especially in the approach introduced by Paulos *et al.* [Foth08, p. 429] data is transmitted without asking the user for permission. Similarly, the user is not asked to make measurements by himself, instead the device is the operator. The approach of Paulos *et al.* is thus more opportunistic than participatory.

Campbell *et al.* [LEMM+08] compared opportunistic and participatory sensing. They described participatory sensing as placing demands on the mobile device's user, and thereby restricting the user pool willing to put efforts in participating in the sensing system. Opportunistic sensing on the other hand does not require the user to be aware of the system's participation in the sensing system, and thereby is only restricted by the device's ability to detect the right context to measure and send data. Campbell *et al.* also argue that participatory sensing systems require a lot more appeal and must be engaging for the user in order to generate a critical mass of participation (and sensor data). Therefore they suggest to use opportunistic sensing over participatory sensing [LEMM+08].

A lot of mobile applications follow this memento today, especially in the area of health care and fitness applications. For example Google© Fit [Inc.15b] automatically tracks the activity of the user, and is able to distinguish between walking, running, and cycling activity. As a result the application then presents the burned calories on that day. Another example is Map My Run, by MapMyFitness Inc. [Inc.15d], which tracks the user's location using GPS when he is going for a run, and presents the average speed, the distance covered and the burned calories.

None of the beforehand presented examples requires the users to measure the needed characteristics themselves, and none of the measured characteristics needs an expert in the specific field to generate usable data. But how should one treat characteristics which are dependent on the user's participation?

An example for the need of the user's participation is heart rate measurement without a wearable pulse sensor. An application solving this problem is Instant Heart Rate, by Azumio Inc. [Inc.15a], which measures the user's heart rate by instructing him to lay a finger on the smartphone's camera and then measuring the skin tone changes. Everyone with fingers can execute this task, so it is to be put into the same category as having to carry an air pollution sensor around.

A much more difficult-to-measure characteristic is sound. It has a broader array of features that can be used for different purposes, as for example air pollution values. This thesis approaches the issue of acoustical object recognition. An application which uses acoustical object recognition is Shazam, by Shazam Entertainment Limited [Sha15]. It is able to classify songs played in the area or room the user is located in. In Shazam the user has to click one button and after a few seconds the title of the played song is presented to the user. Due to the strong noise resistance of the underlying algorithm [Wang+03] the only thing the user has to pay attention to is not to cover the device's microphone. The algorithm is so accurate, that it is able to distinguish between live and studio recordings of the same song [Wang+03].

It must be pointed out that some factors are also dependent on the user's performance, meaning the user heavily influences the acoustical characteristics. An example for this kind of object recognition is the task to find a ripe melon, among other, non-ripe melons. Another example is the recognition of the object inside a Kinder surprise egg without opening it. Moreover, the task of detecting whether a car's engine is running correctly falls into this field of research. All these tasks can be solved by using the sound the different objects are generating (melon when knocked on, egg when shaken, and engine when running).

Two points are essential for the successful conduction of these acoustical object recognition tasks: First, for each of the tasks the user must be instructed how to generate the sound sample, and second, the user must be instructed how to record a viable audio sample. That is because not all users are experts in the respective fields, and especially in the field of audio recording. In this thesis studies were undertaken to find solutions for these problems, concentrating on the Kinder surprise egg example.

3. Design

This chapter covers the processes and attempts undertaken to design the beforehand described system. First the conceptual process will be depicted, which helped to find this system's limitations, and also assisted in understanding the range and coherences of it. Afterwards, using the insights gathered in the conceptual process, a low-fidelity prototype was designed. For evaluation of various designs, three slightly different versions of the prototype were developed. Finally, a formative study was performed, which was used both to evaluate the overall design of the low-fidelity prototype, and to determine the most promising variant of the three different designs.

3.1 Conceptual Design

The conceptual design aims to line out the characteristics of the intended mobile application and looking at it from many different perspectives, in order to find a promising solution for the issues at hand. Initially a PACT analysis [Beny10, p. 27-45] is performed to find limiting factors and to understand the system's environment. From this analysis arise several persona [Beny10, p. 56f.] which are used afterwards in user stories. These user stories are generated to describe the main tasks performed with this system. The conceptual design process also includes an ethnographic study for a better comprehension of the user. After this part the system is understood and well-known. It is then ready to be modeled in form of a low-fidelity prototype.

3.1.1 PACT Analysis

A PACT (People, Activities, Contexts, and Technologies) Analysis is an elemental tool in interaction design. The analysis is used for understanding the new product and the people who will use it. Benyon described the idea behind this framework as follows: *"People use technologies to undertake activities in contexts"* [Beny10, p. 27]. He also provides an informative schema (Figure 3.1) showing the relations between the parts of the PACT model. The framework is anthropocentric and thus – by putting people first – it helps to center the design process right from the beginning on the people instead of the technology.

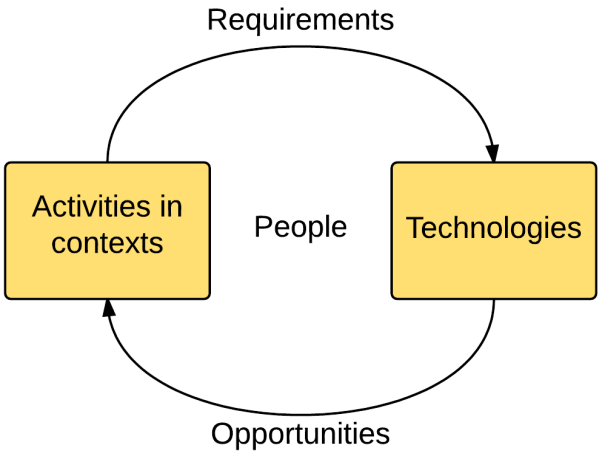


Figure 3.1: Activities and technologies [Beny10, p. 27]

People. The people most likely to use the new system are individuals from all ages and technological backgrounds trying to find "miniatures" in the surprise eggs. Furthermore people with color blindness have to be taken into account, as general rule for interactive systems design. Speaking of general design rules, multiple languages should be supported, because the final system will be available in several countries through the Google© Play Store [Pla15]. The system targets all people who do not possess the expert skill to "hear" by shaking a surprise egg what it contains. The application can help people to save money while exercising their hobby with more success, or provide an additional fun factor for buying surprise eggs. But also experts with the skill to "hear" the egg's content might use this application, even if it is just to see if their intuition is right, to help training the classifier, or even just for fun. It seems to be natural that another user group might be children trying to find the "coolest" content, as surprise eggs are targeting this consumer group. Furthermore, parents could also use it in order to find something suitable for their children.

Activities. People will perform two main tasks with the system. In the first main step, they will record the sound of surprise eggs and give these samples to a classifier for results about the egg's content. In a second optional step, they will be able to give feedback about whether the classifier's results were correct, and about what actually was contained. Hereby people could give detailed information about the content, such as which miniature from a certain series was contained, possibly proving this with a photo for the database or their own collection. Also thinkable are community features like sharing results and personal collections with friends, or showcases for the own collection.

Contexts. The contexts in which the system will be used are the same as for any other mobile application. For example, it might be used indoors directly in the supermarket where it is loud, or it also might be used at home where a quiet environment is given. To follow up on the store scenario, if the individual is in a public environment such as a supermarket, using the application could possibly discomfit

the user, letting him feel awkward or even give him the feeling of doing something forbidden. On the other hand analyzing surprise eggs in the store could be a group action for fun. Finally the system should work in any environment and social situation. In order to perform the required analysis task people will be required to use both hands, one for shaking the egg and one for holding their smartphone, since there might be no table available to lay the smartphone on.

Technologies. Technologies used for this application are all already included in every smartphone available today. The system can use the smartphone's touch screen for input, and the smartphone's microphone to record the audio samples. Moreover it will use the smartphone's data storage to store the audio samples and analysis results. Additionally a connection to a web server will be needed for training the classifier, and sending additional information. The data to be sent over the Internet is small, only a few bytes for the information and several hundred kilo byte for the audio samples. On the receiving side the data returned to the system includes just a few bytes for the results. Sending pictures as a proof takes several megabytes, so a "Send-only-when-in-Wifi"-Feature could be convenient. For usability reasons and to ensure better audio recordings, it should be obvious to people when and how to interact. The system should also display information about what it is doing in the background.

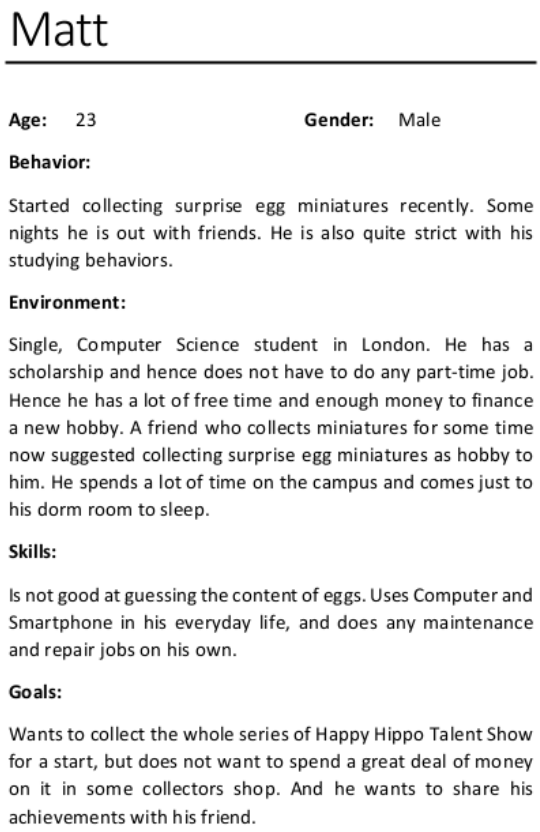
3.1.2 Personas

In the course of understanding the people which will use the introduced application, six persona where developed. Persona represent the different types of people a product is targeted at. For better usage in the following design process they were given a name. In order to better understand the group of people represented by a persona, each persona has a background. They have skills and an environment they live in. Persona should also project ambitions and aims of the people. All these parts help to form their character. Recently this last aspect became more prominent, as persona should try to execute specific things and tasks with the to-be-designed system [Beny10, p. 56f.]. The persona for the "novice collector" people group (Figure 3.2) can be seen, below. This persona was used throughout the whole design process from understanding the proposed system, up to the implementation of the SkillStore application prototype. All persona developed for this thesis can be seen in A.2.

3.1.3 User Stories

With the general conditions for the system being clear its main tasks can be drafted. For this purpose user stories are employed, wrapping the task in a short story or experience from a persona's perspective. The story is abstract on the system's design, and shows the general idea to keep the design itself at this early point of the design process as unlimited as possible. These stories will be refined later to use cases, which will screen an abstract design solution for one or more stories [Beny10, p. 65].

Analyze egg. As mentioned before, the main aim of this software, is to transfer expert skills to amateurs. This expert skill is recognizing objects, or characteristics of objects by use of their acoustical properties. In the scenario treated in this thesis,



Matt

Age: 23 **Gender:** Male

Behavior:

Started collecting surprise egg miniatures recently. Some nights he is out with friends. He is also quite strict with his studying behaviors.

Environment:

Single, Computer Science student in London. He has a scholarship and hence does not have to do any part-time job. Hence he has a lot of free time and enough money to finance a new hobby. A friend who collects miniatures for some time now suggested collecting surprise egg miniatures as hobby to him. He spends a lot of time on the campus and comes just to his dorm room to sleep.

Skills:

Is not good at guessing the content of eggs. Uses Computer and Smartphone in his everyday life, and does any maintenance and repair jobs on his own.

Goals:

Wants to collect the whole series of Happy Hippo Talent Show for a start, but does not want to spend a great deal of money on it in some collectors shop. And he wants to share his achievements with his friend.

Figure 3.2: Persona Matt

the system should transfer the skill to "hear" what is contained in a surprise egg by shaking it. The user story containing this task is presented in section A.3.1. It can be seen how the persona Matt's goal is achieved using an abstract system description, but the major ideas for the system's usability and user experience become transparent for the reader.

Give feedback. The second major task of this system is to provide people with a simple mechanic for correction of wrong analysis results. Thereby the classifier is supplied with new samples to train on in a participatory fashion. This operation is depicted in section A.3.2. It can be seen, that Matt is able to correct wrong results. Thereby he gives the developers expert insight on the results, which they in return can use to improve the classifier. There are two main scenarios of giving feedback: Directly when a classification result is received shown in section A.3.2, or at a later date after several other classifications, demonstrated in section A.3.3. Apparently the second scenario claims some mechanism to log past classifications. A solution for this issue will be introduced later in section 3.2.3.

3.1.4 Ethnographic Study

An important part in understanding this new system is to comprehend the active participation of a person during the analysis of objects. For the acoustic object recognition, people have to interact with the object to be recognized, so that it emits acoustical signals. Additionally the acoustical signal has to be recorded with a smartphone, which involves the necessity of people knowing how to record a viable

audio sample with this device. In the case of surprise eggs, people have to shake the egg to generate the needed sound. But how do people shake eggs, and do they know how to make a viable audio recording?

In order to answer these questions an ethnographic study was executed. Ethnographic studies where originally performed by anthropologists to understand foreign cultures and life styles. They observed activities, language, traditions and other cultural aspects. For this purpose, in the early twentieth century anthropologists spent long time spans living in foreign communities observing them. Today ethnographic studies are widespread and also used in interactive system design [Beny10, p. 167].

Test setup. For this study 31 students between the ages of 19 to 32 where asked to record with their smartphones the sound that a surprise egg makes when it is shaken. The audio sample should have an approximate length of five seconds, and should be as free from noise as possible according to the participants' understanding. For later interpretation, all participants agreed to being filmed while executing the task.

Shaking method. The first observation was the way participants held the egg. Of all participants 14 held the egg on the long sides, but shook it in direction of its peaks (Figure: 3.3a). Five participants changed their grip on the egg while recording, presumably to generate a louder sound with the egg. The other participants

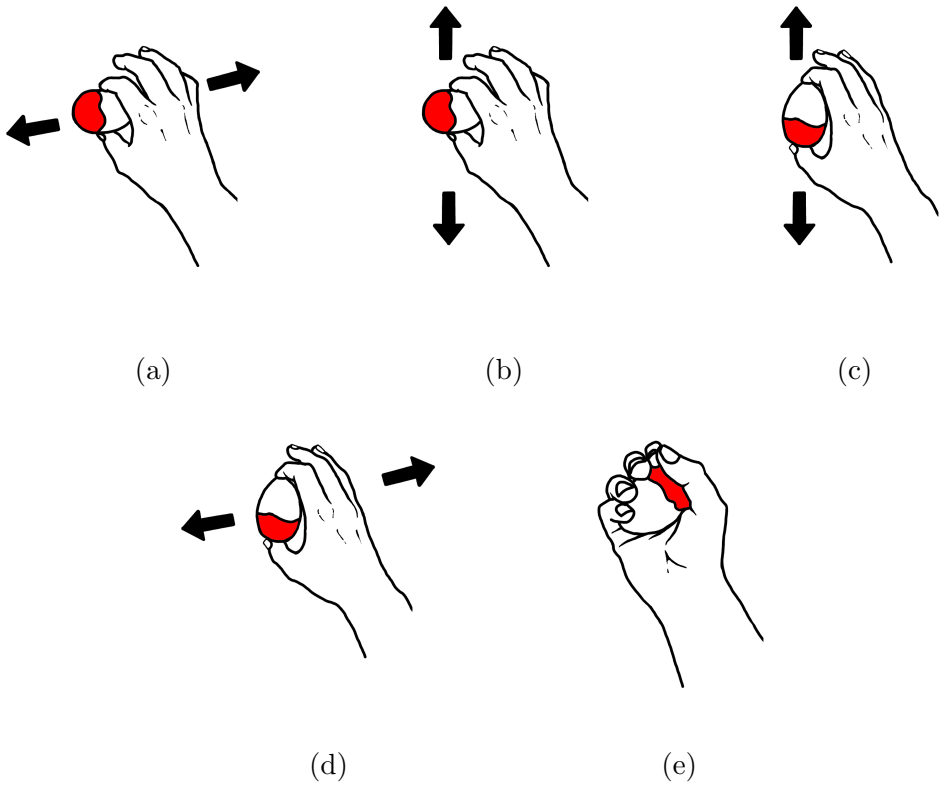


Figure 3.3: Different shake variations found during ethnographic study: (a) Holding egg at the long side, shaking in direction of peaks, (b) holding at long side shaking perpendicular to peaks, (c) holding at peaks of the egg, shaking perpendicular to peaks, (d) holding at peaks, shaking in direction of peaks, (e) holding the egg with full hand.

showed four other grip variants. The first of these variations was to hold the egg on the long side and shake it perpendicular to the peak axis (Figure 3.3b). The second variation was to hold the egg at its peaks and again shake it in direction of the peak axis (Figure 3.3c). Each of these variations was performed intuitively by six participants. Grabbing the egg with the whole hand was the third variation (Figure 3.3e), and was performed by four participants. The last two participants were shaking perpendicular to their pinch grip at the egg's peaks (Figure 3.3d). These results are summarized in Figure 3.4. One question arising from this observation is: Does the generated sound differ in a way that it is no longer recognizable by the classifier, meaning is a dedicated grip technique needed and has to be taught to people for good analysis results? Concerning this, the author found that a pinch on the long sides of the egg and shaking perpendicular to the peaks axis seems to produce a louder sound. Furthermore, the inner egg seemed to get stuck more often when shaking the egg in direction of the peaks' axis.

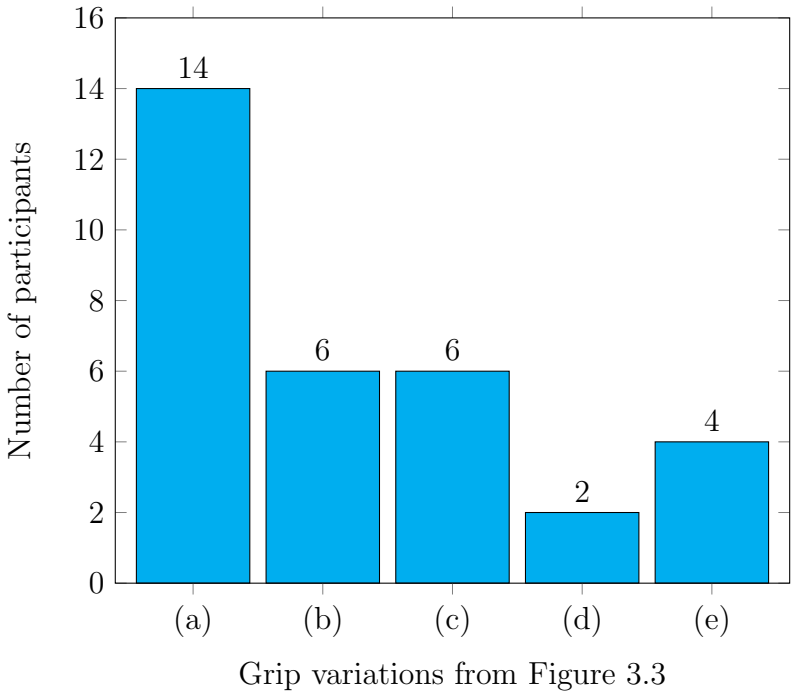


Figure 3.4: Distribution of participants over grip variations.

Hand usage. A second feature observed was with which hand participants held the egg, and with which the smartphone. Here no significant difference could be found. 16 participants used their strong hand to shake the egg and held the phone in their weaker hand, and 15 held it the other way around. So this feature seems to be minor for the shaking process, from the perspective of interaction design, except for the fundamental guideline to design for right and for left handedness.

Participant negligence. After that the effect of the instruction to record a viable audio sample, was investigated. From this perspective the participants made two major negligence. The first – from lesser lower importance – was not to take off their watches and bracelets from their shaker-arm, and thereby made it impossible to make a truly viable recording. The main problem was the location where the participants shook the egg in relationship to the smartphone. Generally the smart-

phones microphone is located at the bottom side, so that the voice is recorded best while making a call. However only two of the participants shook anywhere near the microphone. The rest shook somewhere behind or in front of the phone, or at one of the sides (left or right) but not even close to the microphone.

Possible reasons for these behaviors could be misconceptions – either shaking near the screen, because that is where normally all smartphone interaction takes place, or shaking behind the phone, because the camera is located there, which is especially relevant in the case of participants recording the sound with their video recorder application. In case of shaking at one of the sides a possible explanation could be that the participant thought it would be irrelevant where to shake as long as it was anywhere around the phone. This behavior shows that the mental model of where the microphone is located in their smartphone and how to record viable samples is not clear at all for the participants. It can't be assumed that the participants know about the details of the technical realization of an audio record. For the system's design it has to be stated that clear instructions have to be given, about where the microphone is located in a smartphone and how to record a viable audio sample.

Viable audio recording. In order to record a viable audio sample in general, three things are essential: First, the location of the microphone needs to be known. Otherwise the sound could be generated too far away from the microphone and hence more environmental noise than actual sound is recorded. Second, a environment with little noise needs to be chosen, and everything that could generate noise must be avoided or turned off, in order to reduce possible noise that covers the actual sound signal. Third, it must be known how to generate the needed sound signal, else the classifiers using the sound samples or trying to analyze the samples is not able to perform its task properly.

Synopsis. In summary, recording viable audio samples with a smartphone is not an elementary task to perform with a smartphone among students nowadays. Therefore the application has to teach people how to perform this task, and which aspects to pay more attention to. Despite that, it must be mentioned that this task was embarrassing for some participants, so it could be that the final system might not prove popular.

3.2 Low-Fidelity Prototype Design

Now that the intended application is understood and its main functions are lined out, a first prototype can be built. This prototype is a low-fidelity or paper prototype as described by Benyon [Beny10, p. 187]. Three slightly different prototype versions were developed addressing a user experience problem in the Analyzer part of the application, see section 3.2.2 for details. These prototype versions will be evaluated in the later described formative study, and the most promising design version will be found.

For this low-fidelity prototype the user stories were refined and merged to use cases. Benyon introduces use cases as descriptions of interactions between people and devices. Use cases are more specific than user stories which only contain an abstract view on the system to design. The use cases on the other hand cover a broad band

from specific pseudo code, to abstract descriptions of what the system does and how people interact with it [Beny10, p. 67f.]. The cases used in this thesis are more specific than abstract, but do not contain any pseudo code. They are concentrated on the interaction design part rather than on the implementation part.

3.2.1 Navigation

A great part of interaction design is focused on navigation and on the question how to guide people through the different system functions. In order to address this issue three use cases where developed, depicting three different navigation styles. The specific use cases are attached in A.4. The SkillStore application is implemented for the Android OS, hence the Android design guidelines were factored in for the design process. More information on the decision for Android will be provided in section 4.1.4.

Static menu page. The first of the three navigation methods is a static menu page, see Figure 3.5a. This is a dedicated screen to which the user will return every time he wants to navigate to some other part of the system. This behavior can be observed in section A.4.1. But here also the first great drawback of this method can be seen, as the navigation menu is a bottleneck in terms of interaction speed and fluent usage of the prototype's functions. Due to the fact that the average time spent in a mobile application, during one session, is only 72 seconds [BHSK+11], this navigation style might impair user experience. The second drawback of this method is that it alone does not show the current state. This invisible status problem is solved when using the static menu in combination with the `ActionBar` as recommended in the Android design guidelines [Act15a]. On the other hand an advantage of the static menu is its low implementation costs. In summary it can be said, that a static menu poses a so called "quick and dirty"-solution for the navigation problem.

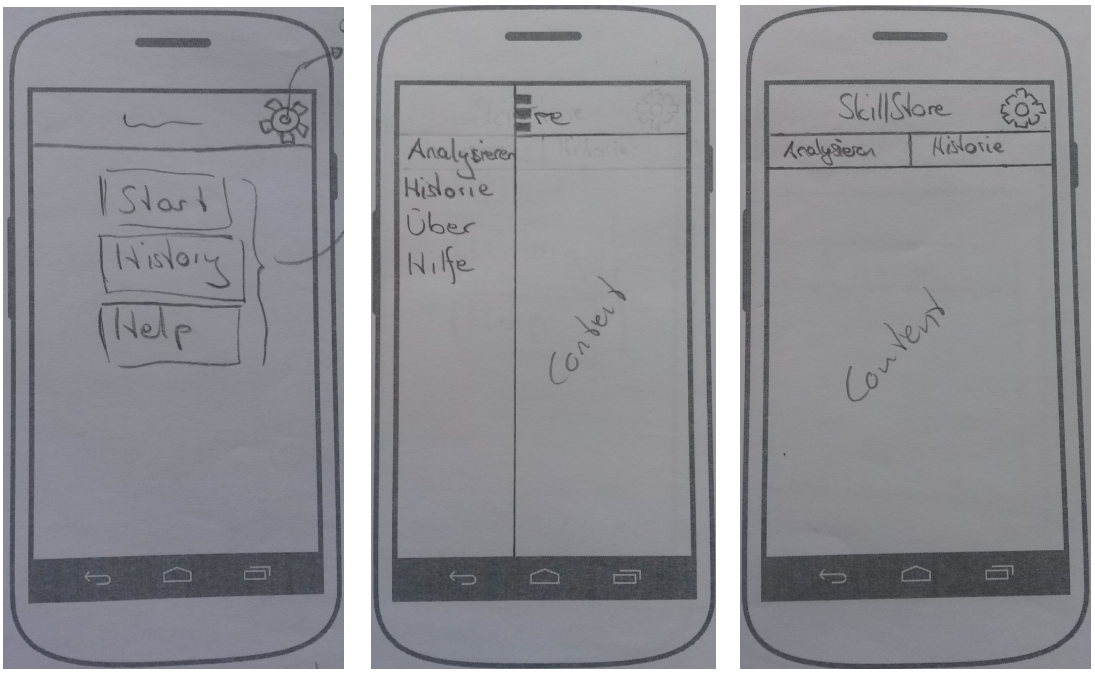
Swipe menu. In order to improve the navigation's usability, it can be implemented as a swipe menu. It is accessed by swiping with the finger from a screen boarder, see Figure 3.5b. Using this method the bottleneck of the first navigation style is eradicated, so that the work-flow is much more fluid (see section A.4.2). But if no permanent visible access button is implemented, gesture control is needed and thus this menu style has to be counted suitable only for experts, because gesture control is an invisible shortcut. New users will not find the menu or struggle to remember how to access it. Besides it still suffers from the invisible status problem, but as explained earlier in Android this issue can be solved easily. In terms of implementation cost this navigation style is much more expensive than the static menu, which could be worth the efforts due to the probably better user experience. This method is used by popular application like Maps from Google© Inc. [Map15], or Chefkoch from pixelhouse GmbH [Che15]. It is also used by Facebook® [Fac15] in their mobile application, for hiding the list of all friends which are currently online.

Tab navigation. Since the prototype has so little functionality, the swipe menu will be mostly empty, and thus such a swipe menu would take up an unnecessary big amount of space. Android gives a good alternative to the swipe menu to solve the problem: Tabs in combination with an `ActionBar` menu, see Figure 3.5c. Tabs are space saving and always visible, which solves the invisible status problem. Additional as shown in section A.4.3, it does not interfere with the work flow of the

main functionality of the application. The user is able to switch between the two major functionalities with just one click or one swipe, whereas he needed at least two actions to switch functionality using one of the other two navigation styles. This navigation style splits the navigation in two parts: main functionality and less needed functions. The main functionality is located in tabs, always visible and accessible for the user. With the intention to save space and keep the particular tabs easy to find, this method is only suitable for a very small number of entries (2-4). These entries can be accessed by clicking on a tab. In addition, swipe gestures can be introduced as an expert shortcut to switch between tabs. When switching tabs, only the displayed fragment will be changed, keeping the rest of the screen still. This supports the user in finding orientation in the application. For access of less frequently used functionality, an additional drop-down menu is introduced, which originates from the right corner of the `ActionBar`. When accessing one of the less frequent used functions, the screen will switch and no longer display the tabs. But it will keep the `ActionBar`, updating its visible status and now providing a back button, so that the user may return to the main screen [Nav15]. Thus the user always knows the current position in the application and how to go back at a glance. This combination enables the user to access the important functionality really quickly, and still gives him access to the less used functions. It is also used in many popular applications, like WhatsApp [Wha15], and Facebook® [Fac15]. Thus tabs seem to be the best navigation method for this prototype due to the small functionality.

3.2.2 Analyzer

Since the easy access to the analyzer – as the main feature of the prototype – got ensured now, the design for the implementation can follow. Here major questions are, how the user should shake the surprise egg, and what is needed by the classifier



(a) Static menu (b) Swipe menu (c) Tabbed menu

Figure 3.5: The three different menu variations.

working in the background.

At first the classifier constraints have to be clear. As mentioned in chapter 1 the present thesis originated in cooperation with a master thesis concerning the classifier and the server backend used for the SkillStore application. The input required by the classifier are viable audio samples. From an interaction design perspective this leaves free reign to the design process.

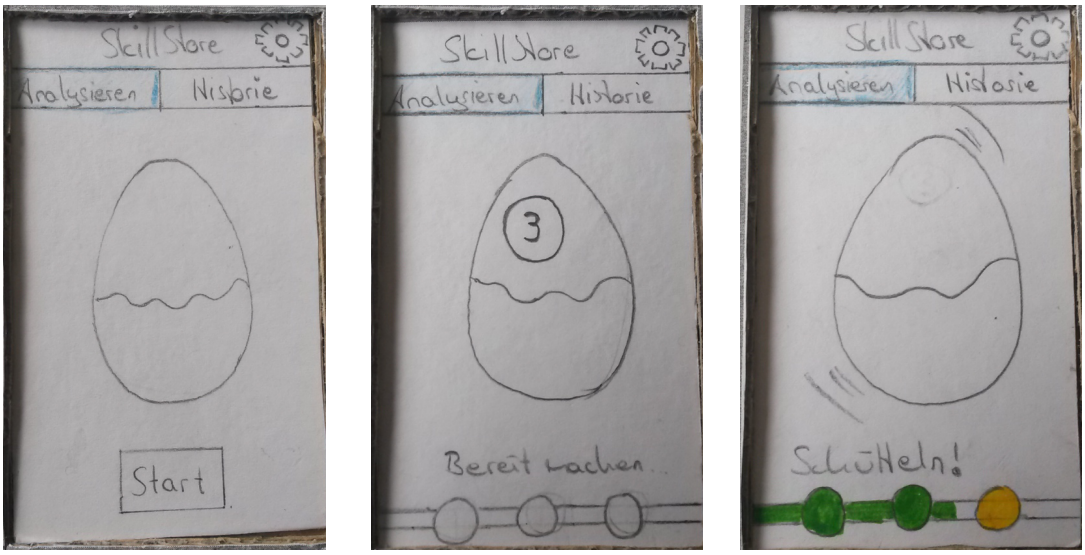
Realizing the results of the ethnographic study, and with respect to the classifier's requirements, it was decided to let the user shake the surprise egg vertically pinching it at its long sides, as depicted in Figure 3.3b. The choice fell on this, because it seems to be the most common intuitive grip variation among people, and it was found that the inner egg gets more often stuck when shaking it in direction of its peaks.

Shaking methods. Three different shaking methods were analyzed. For every analysis session each of the methods was repeated three times. The idea behind the three dedicated samples was to make preprocessing easier. Moreover it mediates structure to the user, which makes him feel better than a chaotic procedure. The first variation is to shake just one time up and down, so that in sum three shaking peaks are generated in one analysis session. After preprocessing the sample and cutting it down to the shaking peaks, this method will be the cheapest in terms of Internet usage. It also gives the user the impression that the single peaks are important, and therefore might help to form a better mental model of the analyzer. The second shaking method instructs the user to shake three times in one sample. Thus, nine shaking peaks are generated in sum. Here the benefits of structure are still preserved, and in addition it could be more satisfying for the user. This seems natural because he gets to shake more, which is based on the intuitive reasoning: "More is better". For the last variation the constraint on the number of shaking peaks is suspended, in order to amplify the effect of the "more is better" reasoning. These methods are interesting for learning when people feel they are recording more viable audio files, either when shaking without structure, or when they have to keep structure in the shaking procedure.

Analysis process. The next task to be executed is the design of the user interface. The main requirement here is that the analyzer needs to be intuitive for people to use. It must be mostly automatic so the user can concentrate on shaking the egg. Additionally the interface needs to be suitable for single hand usage, because the user holds the egg in one hand and the smartphone in the other.

These constraints lead to the implementation of a "Start"-button (Figure 3.6a), which will start the analyzing process when clicked on. Then the application runs automatically until the results of the analysis are presented. After having clicked the "Start"-button, there needs to be a short period in which the user can prepare for shaking. Without this period the user feels hounded and overwhelmed with the task at hand, which in return lowers the quality of the audio sample and reduces the user experience. This applies especially to the first session in which the system is used by a new user. The first impression is very important for a good user experience. Therefore a short countdown of three seconds is installed, see Figure 3.6b. This is

not too long for an expert user to get bored, or getting hindered in their work-flow, but enough to prepare.



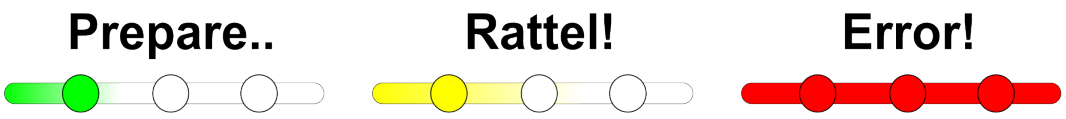
(a) Start screen (b) Prepare screen (c) Recording screen

Figure 3.6: Three screens of the analyzer tab.

After this countdown there must be some signal for the user to start shaking. For this a picture of a shaking egg is introduced, see Figure 3.6c. This metaphor is easily understandable by all of the used persona, in order not to surprise the user when suddenly a wild egg appears on the smartphone screen, this egg is already displayed in the preparation period in a still pose. The egg will stop after a short shaking period, and the countdown will begin again. This will be repeated until the three samples are recorded and it can be classified.

While the sample is analyzed for a short while there is nothing to do for the user. To discourage the user from interrupting the background work of the application, the application's status should be displayed. Therefore a status text is displayed below the egg picture, see Figure 3.7. By this, the user gets feedback about the active state of the application. In addition the opportunity for instructions and hints to the user is provided. This can be helpful in the preparation period, and helps in understanding of the meaning of the displayed shaking egg.

At last it seems to be a good idea to give feedback to the user about the duration of the automated action of the application. For this purpose a progress bar was added to the bottom of the screen, see Figure 3.7. With an additional color code on this progress bar also information about status and success of the application can get

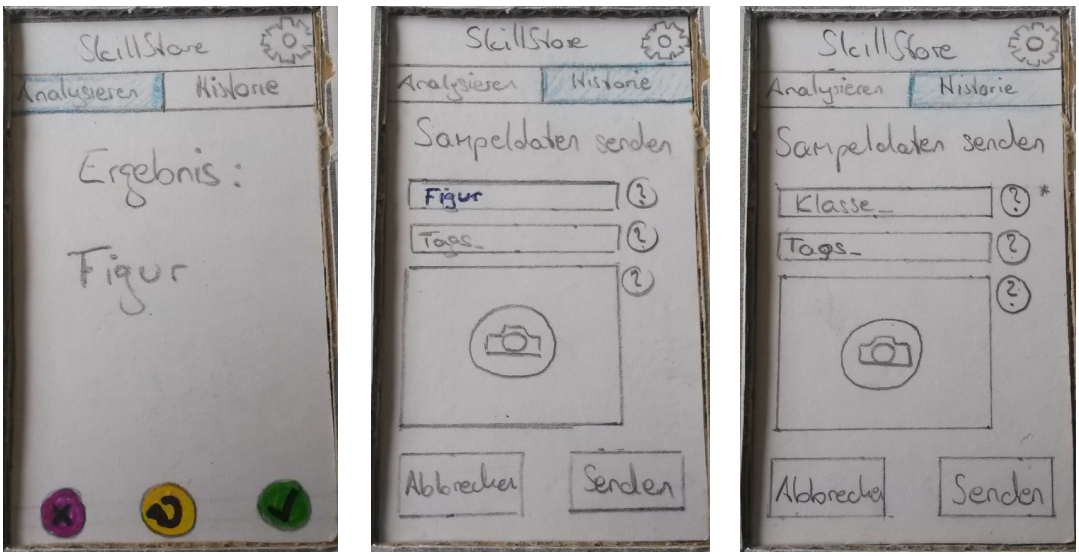


(a) While preparing (b) While recording (c) When error occurred

Figure 3.7: Progress bar and status text in different states.

transported. Green color means that everything is all right (Figure 3.7a). This is displayed in each preparation period, and at the end when the results are presented and no error occurred. A red color means that an error occurred (Figure 3.7c). And when the application is doing work in the background like recording audio, or interacting with the Internet or classifier, the bar will turn to yellow (Figure 3.7b). This color pattern was chosen due to its usage for traffic lights all over the Western Hemisphere, and therefore is learned and understood by everyone from a young age on. Sometimes blue is used instead of green, but these colors make a similar calm impression on the observer. At this point people with color blindness have to be kept in mind. For a later version of this application prototype it can be considered how to make this color pattern also understandable for color blind people, but for this first prototype this matter will be postponed.

Giving feedback. When everything was successful a result screen will be displayed (Figure 3.8a), showing the result of the analysis and three buttons: A yellow button with a rotating arrow on it, a “true”-button in green with a tick symbol, and a red “false”-button with a “X” symbol. The yellow button lets the user repeat the analysis. Via the other two buttons the user can give feedback. If the result was right he clicks the green, ”true”-button, in which case he is redirected to a feedback form filled with the results, see Figure 3.8b. There he can provide additional information like tags and a picture, and send all of this to a server. If he clicks the red, “false”-button, he is redirected to a blank feedback form (Figure 3.8b), where he shall fill in the correct result, and details. This design decision gives the user the feeling to have more control over the feedback process, and thus he might feel more in charge and his contribution to the analyzer being more valued by the developers. The feedback form is located in the history tab introduced in the next section.



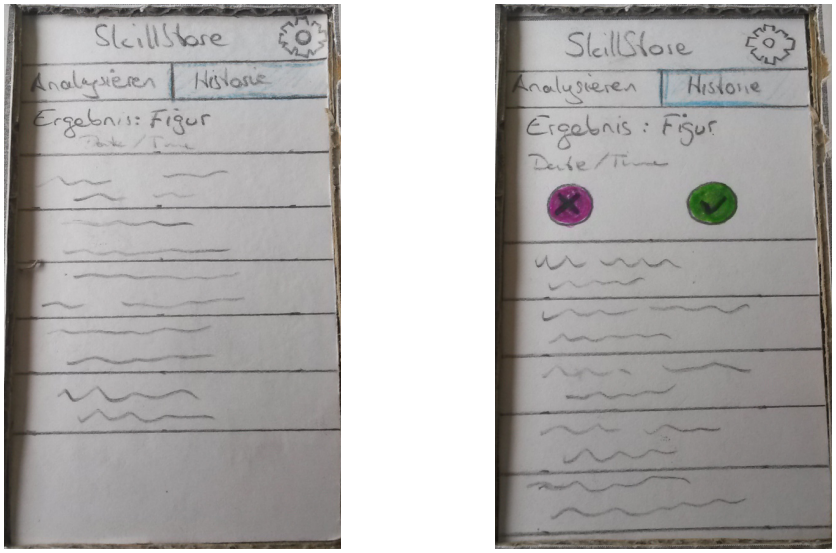
(a) Result screen (b) Feedback form filled (c) Feedback form empty

Figure 3.8: Result screen, and feedback form in different states.

3.2.3 History

The History tab was introduced to approach the log problem mentioned in section 3.1.3. Here the user can access all past analyses he made. They are displayed

in list form, sorted by date hence the user can quickly find the right egg. Each entry displays the result and date of the classification, see Figure 3.9a. Additionally – if available (e.g. provided by the user) – a picture of the eggs content will be displayed. When the user clicks on an item, it will unfold into a detail view (Figure 3.9b), displaying additional tags for this item, and two buttons. These buttons are the green, "true" and red, "false"-buttons from the result screen, which show the same behavior as for the result screen.



(a) (b)

Figure 3.9: (a) History screen, and (b) History screen with unfolded entry.

3.2.4 Manual and Help

It is good practice to provide help to the user and give instructions about how to use the application. This is to reduce the efforts a user needs to muster to remember how to execute tasks with the system, and to help him recover from errors.

In order to support the user in learning how to use the system, a manual will be shown directly when the prototype is started for the first time. For the prototype a simple screen with text on it is chosen, see Figure 3.11a. The manual text itself is subdivided into three steps, see Figure 3.10. Step one tells the user how to hold the egg. Step two is slightly different for each of the three shaking designs, but in general it gives instructions on how to shake the egg. And the third step informs the user that this shaking process is repeated three times before the results will be presented. All three manuals for the paper prototype can be seen in A.5. This short manual page should give users all information they need for performing their first analysis. A "don't show again"-checkbox is provided at the bottom of the manual page, to give the user the control to skip the manual next time, see Figure 3.11a. This preference can be reset in a settings menu located in the `ActionBar`'s drop-down menu. The manual can also be accessed there at any time, via the last entry in the menu "Help" in the way it is recommended in the Android design guidelines [Hel15].

In order to help the users recovering from an error which occurred while analyzing or in general using the application, an error screen was implemented, see Figure 3.11b. Depending on the error that occurred, error messages and recovery hints are displayed. Some examples of errors are network connection errors, asking the user to make sure that the connection is stable, and audio recording errors, giving the user instructions how to make viable audio recordings.

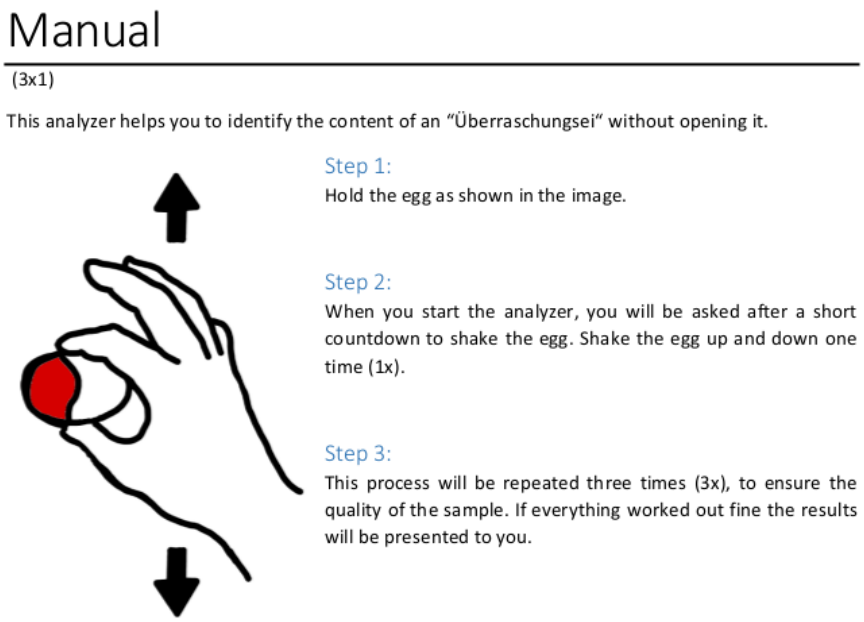
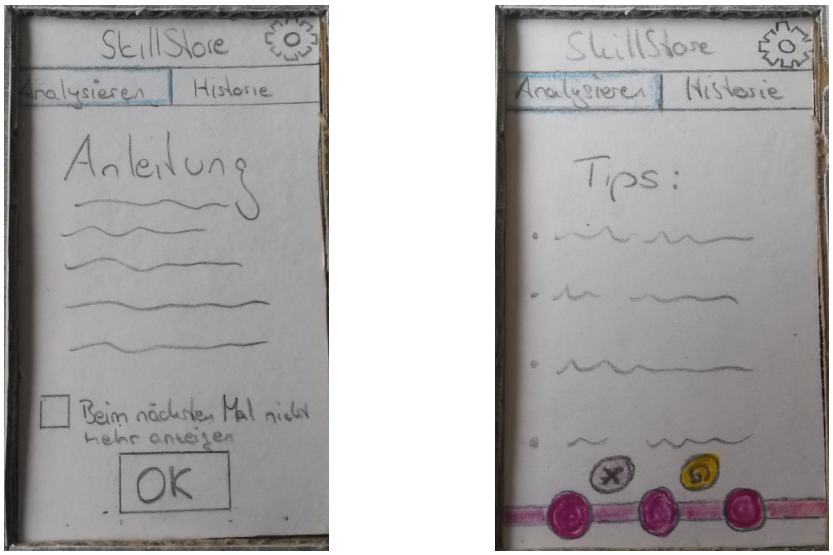


Figure 3.10: One of the manual texts of the paper prototype.



(a) Manual screen (b) Error screen

Figure 3.11: Manual and error screens.

3.3 Formative Evaluation

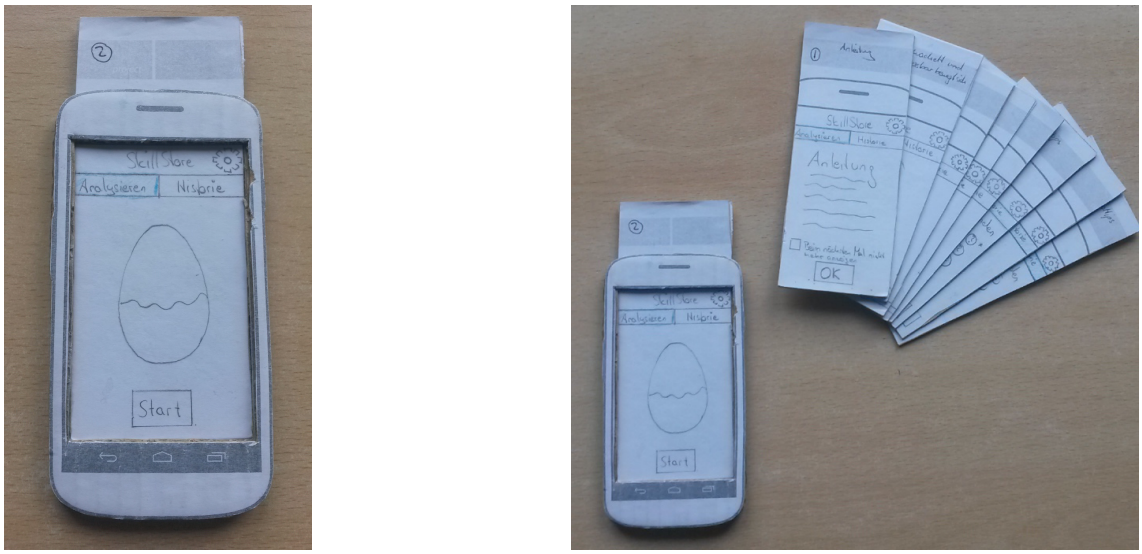
For evaluation of the new designs described above a user study was performed. This study was designed according to the descriptions given by [Beny10, p. 234f] for

controlled experiments. In addition an expert study was performed to get a second view on the designs. Here the test set-up was a mix of the formative study and the expert study given by Benyon [Beny10, p. 228].

3.3.1 Study Design

The controlled study focused on three variables: The egg shaking method, the manual, and the general user experience. The first independent variable was to evaluate which of the three ways to shake the egg would be the most promising. The dependent variables used here were the System Usability Scale (SUS) [Saur11], and a question for the users opinion which design was the best in their eyes. Another independent variable was to evaluate the manual given to the users. For its evaluation some questions and the observations made during the test were used. The third focus was on the general user experience, evaluated again through the SUS.

Participants. Six participants were acquired for this study, all in the age range of 20 to 25. They were balanced regarding their gender, three female and three male subjects. Even though the applications prototype was designed for Android users, one of the participants was an Apple® user and another one used a Blackberry® smartphone in her everyday life. Nevertheless it was no drawback to have platform-alien participants, instead it rather helped to control for consistency[1], affordance[2] and familiarity[3], since thereby it could be evaluated how intuitive this prototype was to use.



(a) (b)

Figure 3.12: (a) Paper prototype, and (b) paper prototype with screen cards.

Test setup. Used in this study was the paper prototype described in detail in section 3.2 see Figure 3.12. It was operated by one of the testers, and the participants

[1]Being consistent when using design features, mechanics, or concepts through out a system, or throughout similar systems [Beny10, p. 90].

[2]Designing features, navigation, controls etc. in a way that it is clear to the user how to interact with them, how they behave, and what they achieve [Beny10, p. 90].

[3]Using language and symbols fit for the intended audience of users. For example transferring metaphors from domains the user is familiar with to this new domain [Beny10, p. 90].

interacted with the system. The whole test setup is depicted in Figure 3.13. Participant and prototype operator were seated at a table, which was recorded from above by a camera, for later analysis of the interaction with the system. The role of the operator was to simulate the actions of the prototype and present assets like manual pages, hints, or error messages at the right moments to the user. The second tester sat a bit apart from the other two persons, his role was to observe the tests and take notes on behavior and observed difficulties the participants might have while using the prototype.

Figure 3.13: Experiment set-up for the formative study. Prototype operator and participant seated at a table. The participant is using the prototype. A second tester is seated apart from the other two, observing the trials. The table surface is recorded on video and saved on a computer.

Task. Each participant had to perform the same task, namely performing an analysis of a provided surprise egg using the paper prototype. This task had to be executed three times, so that each participant tested all three different shaking designs. The sequence in which the different designs were presented to the participant permuted, in order to control for learning effects. Also the outcome of the analysis – its success or failure – was controlled to let the participant test the error recovery mechanisms implemented in this prototype. Each test session was performed as "think aloud session", in which the participant has to describe what he was doing, or thinking right at the moment. This method gives the testers a better insight in the mental process of participants and their problem-solving methods [VSBSo94].

Test session. The typical test session started with a short briefing in which the whole test setup was explained to the participant, beginning with the role allocation of the testers, and the participants' task. Afterwards the paper prototype and the think aloud session were explained. After this briefing the three different trials with the different designs were conducted. After each trial the participant was asked to fill in a short questionnaire, consisting of a SUS and a few questions to the overall user experience. The questionnaire is attached in A.6. When the trials were finished

participants were confronted with a few open questions on their overall experience with the design in the debriefing. In the course of this they were asked to rank the three designs, as they thought the best audio samples would be generated. The session was closed by answering all the participants' questions.

3.3.2 Results

Statistic results. The results of the formative evaluation were crucial to the further development of the system's user experience. Furthermore it was highly informative for the testers. The main question in this study came to the anticipated result that the design variation – which abandons shaking structure for better user experience – was perceived as the one generating the most viable audio samples and resulting in the best analysis results. This was shown in the open questions asked during the debriefing. In the question where the participants had to rank the three designs, this design obtained an average score of 1.5 on a scale from 1 to 3, where 1 is the best score. Whereas the variations with three shakes and one shake reached average results of only 2 and 2.3. For more clarity this relationship is depicted in Figure 3.14. The SUS evaluation showed the same tendencies as the design ranking before. The unstructured variant scored on average for 91.5 out of 100 possible points, closely followed by the one shake variant, with a score of 91, and last came the three shakes approach scoring for 88. Overall the prototypes design scored for 89.7 in average over all SUS surveys filled in during the study, which is a satisfying result. These results are depicted in Figure 3.15. Participants explained their decision for the unstructured design with it generating both the loudest clack sounds, and also the most clack sound in the given time. Thereby they thought the analyzer would had more data to work with. They also mentioned that it felt most satisfying to them, as predicted while designing this method. Another reply was that it was the easiest to use: participants said they had to think least using this

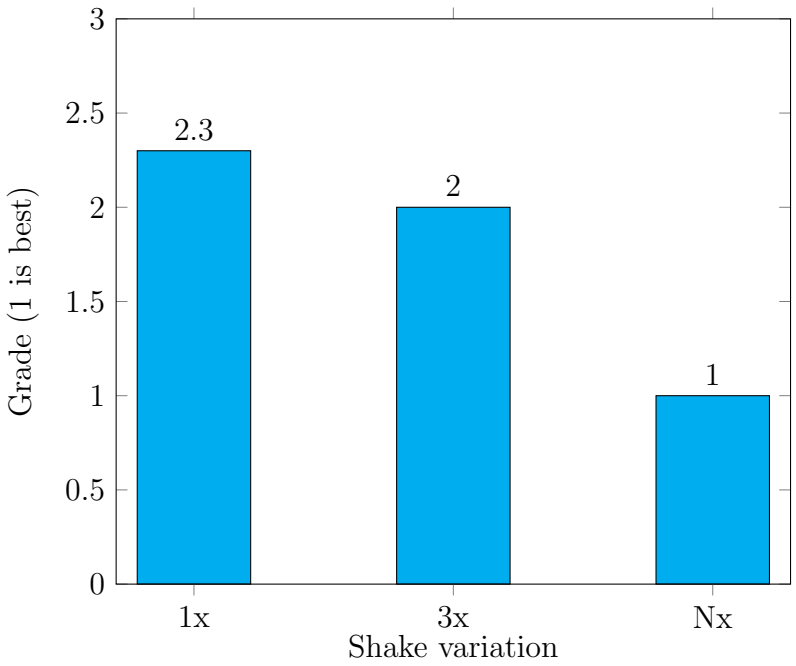


Figure 3.14: Results for the task to order the three designs by pleasure to use, a lower score means that the design was more pleasing.

design, whereas they had to count shakes in the other two variants. Thus even the act of shaking the surprise egg a fixed number of times – either once or three times – seemed to pose a difficult task. Probably this could be due to the fact that the participants were so concentrated on how and when to shake that no capacity was left to count correctly.

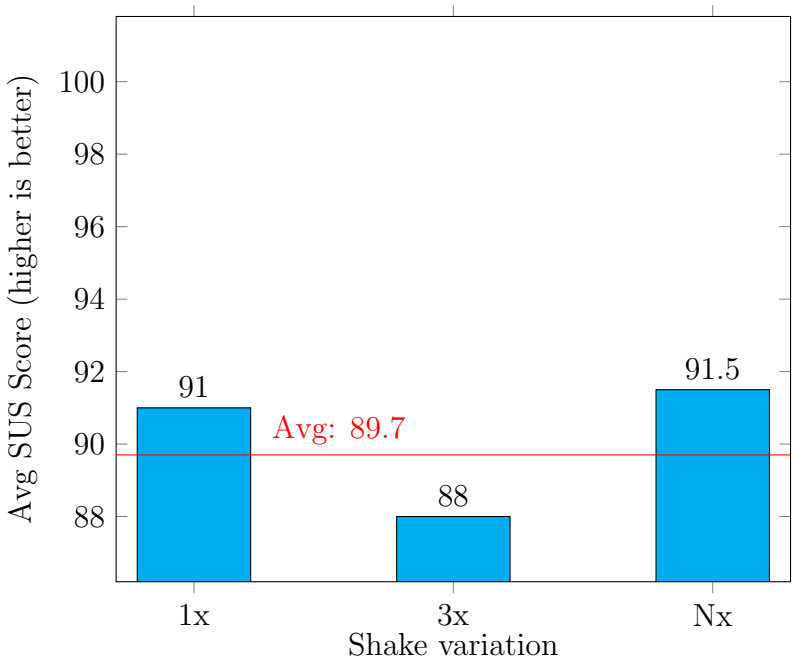


Figure 3.15: SUS score for each design, on a scale from 0 to 100 where values above 80.3 are graded as A, and everything below 51 is graded as F [Saur11].

Manual evaluation. The next interesting topic to investigate is the user manual and other guiding measures introduced for supporting the amateur in recording of viable audio samples. The results of the study show that the manual had several flaws, which impaired the participants' mental concept of how to use the prototype and how to perform the given task. One great flaw was the text length given on the manual page. There was too much text on one page. That participants did either not read it to the end, or just skimmed it. Particularly this occurred in the second and third trials, where just small passages changed in the manual, what was not recognized by the participants. Therefore participants shook the surprise eggs not for the correct number of times, or they shook them while the countdown was running and stopped shaking when the prototype simulated to start recording. Whit the intention to improve the manual and reduce the confusion among users, the manual in the next prototype should be split into a slide show, containing one slide for each step. This approach should improve the transparency and should help to focus on the single steps. Consequently the user's efforts will be reduced because he can concentrate on one step at a time. The slide show will also prevent the user from not reading single steps, except he is just going through the slides as fast as possible without paying any attention. Also all of the slide show steps need to be revised for the next prototype version.

In the first step of the manual a hand holding an egg is shown, see Figure 3.16. The egg should be colored to improve clarity, and thereby help the user understand

the image quicker. Steps two and three could be combined on one page if their text would be reduced. It also could be beneficial if in these steps an animation of the countdown and the shaking egg was displayed. People understand strong symbolic metaphors much faster and more intuitive than long texts [Beny10, p. 325]. Besides, many participants complained about the inconsistency between the picture of a hand holding a surprise egg, and the egg shaking during the analysis. First of all they were confused about the orientation of the egg in both pictures. The egg in the manual was held horizontally, but the one in the analyzer was displayed vertically, see Figures 3.11a, and 3.6c. This confused the participants greatly and the most frequent reaction to it was to turn the egg to fit the vertically displayed egg – which was not the way to hold the egg as envisaged by the designers. So in the next version this inconsistency should be removed to help users recognize what to do rather than to put thought in it themselves. Here the suggesting solution is to use the hand holding the egg. This metaphor is a more direct mapping of the underlying metaphor than the single egg. When speaking of the shaking metaphor, it seemed to the testers, that the participants did not really understand the displayed shaking egg as a metaphor for their own task to start shaking the egg, at least not in the first trial. This might be rather due to the way it was represented during the study, than due to a badly chosen mapping. It seemed to the testers that participants could not conclude on the underlying metaphor when given the static picture of the paper prototype combined with the oral information that the displayed egg was shaking. These assumptions where confirmed during debriefing, where participants which had understood the metaphor mentioned that it would have been more clear to them if the picture had been animated.

Figure 3.16: Hand holding egg, picture from the first manual prototype.

Concerning the manual, the study also had shown that important information was missing in the manual. Hence it seems that two more steps are needed which will be added to the next prototype version. The new third step should explain the result screen, otherwise users would not know what happens after they shook the egg. The fourth step should explain the need to give feedback to help improving the application in the future, and how to access the feedback functionality. In particular this step should introduce and explain the history tab to the user, which was not mentioned in the paper prototype.

Other design flaws. Overall the prototypes design scored for 89.7 in average over all SUS surveys filled in during the study, which is a satisfying result, for more

information see Figure 3.15. The major flaw in the general design were the “true/-false” buttons appearing on the result screen and in the History list when an entry is viewed in detail view. They were misunderstood as “Accept” and “Cancel” buttons, and thereby transported a wrong clue for the mental map of the prototype. For more clarity of the navigational model underlying the prototype it seems to be better to the designers to omit the “true/false” buttons and thus automatize their functions. In the result screen this will result in the two buttons “Feedback”, and “Later”. The “Feedback” button combines the “true/false” functionality, and the “Later” button replaces the repeat button, for reasons of consistency - either all buttons located in one bar are symbolic, or none is. For the menu entry detail view (Figure 3.9b) it seems cleaner to remove it entirely, and guide the user with one click to the feedback form, which has already been filled with the existing data. The detailed information is not lost, because it seems to fit neatly in one menu entry, which makes the detail view redundant anyway. A minor flaw confusing some participants was the lack of information about what happened with the data when they gave feedback and were then returned automatically to the start screen. In the next version of the application this could be solved easily by using the status text more or some pop-up message informing the user that his data was saved. In general it should become more transparent for the user when a secure state is reached during a session, where he can close the application without the risk of data losses, thereby exit points are created [Ext15].

3.3.3 Expert Evaluation

As a second evaluation for the paper prototype, an expert evaluation was conducted. Here the same task as beforehand in the formative study was performed, this time with one interaction design expert. The expert had to note for every flaw she found in the prototypes design. She also had to find the heuristic [Beny10, [p. 89f.] connected with this flaw, and give an idea for improvement. This procedure is described by Benyon [Beny10, p. 228f.].

Surprisingly this single expert found the same design flaws as the amateur participants in the formative study, and even more. She additionally noted that the three buttons provided, that were provided when a recording error occurred, could be confusing for the users. Besides she also misunderstood the “true/false” buttons, and she noted that having to navigate manually to the feedback menu when a “silent egg” – an egg which gives no sound when it’s shaken – occurred might demand too much effort from the user. In order to improve this flaw she gave the advice to reduce the buttons to a simple “Ok”-button which would improve clarity, and would make the navigation clearer. Concerning the “silent egg” problem, she mentioned the idea of providing a link for sending direct feedback about silent eggs, which would reduce the efforts for this task immensely. Another issue confusing for her was that the active tab changed from “Analyzer” to “History” when entering the feedback screen. This seems not to fit with the general mental map of the application. It would be better to keep the tab unchanged or even provide a whole new page when entering the feedback form. Concerning the tabs, their naming was also confusing, due to the fact that the analyzer tab was called “Analyzer”, but in the application the tab was named “Analyzing” which is an inconsistency and would be better changed to “Analyzer” in both cases. A small note at the end was to exchange the text field for

the class in the feedback form, with a drop-down menu of the different classes available. This would reduce efforts needed by the user and likewise the error-proneness of the training data.

3.3.4 Summary

In summary this low-fidelity prototype yields high potential for a good user experience. From the three proposed shaking methods using the method without structure (Nx) promises the best user behavior. For further improvement the instructions given in the prototype must be shortened and simplified. These two studies helped a lot to improve the next prototype, which was to be implemented in Android. The findings from this first, simple prototype could already be taken into account for the design of the Android implementation. The specific implementation details and design improvements will be shown in the next chapter of this thesis.

4. Implementation

This chapter describes the aforementioned implementation of the high-fidelity prototype which had been already mentioned in the introduction. High-fidelity, or hi-fi prototypes are similar in their appearance and behavior to the final system which will be released to customers. The development of such prototypes costs a lot more time than developing a paper prototype as shown in section 3.2, and it provides a more detailed view on the features and design elements anticipated for the final product. High-fidelity prototypes are also double-edged swords, because people believe they are real, due to their high immersiveness. So every small flaw in the implementation which impairs its authenticity can confuse the study participants, and thereby reduce the perceived user experience and validity of the experiment [Beny10, p. 185f.].

Firstly the frameworks used to implement this prototype are introduced, and secondly the design of the mobile application software with its three-layered architecture will be explained. During the implementation special care was taken to avoid the flaws found in section 3.3. Finally the high-fidelity prototype was evaluated in a summative evaluation, for details see chapter 5.

4.1 Frameworks

Here the frameworks and development environment used to implement the high-fidelity prototype are introduced and explained. Frameworks in computer science are software packages which provide reusable and customizable functionalities [Wik15a].

4.1.1 Xamarin

Xamarin is a cross platform framework written in C#. It targets the platforms iOS®, Android, Windows®, and Mac®. Xamarin allows to share code over all platforms, and write native applications at the same time. Hence it offers the possibility to port the application quickly to other platforms without sacrificing the native nature of the application. Because of using C# Xamarin brings full Microsoft Visual Studio® integration which makes development and management of bigger software projects like mobile applications easier [Xam15].

4.1.2 Microsoft Visual Studio®

Microsoft Visual Studio® was chosen as IDE (integrated development environment) due to its broad spectrum of well-developed features, and its high usability standards [Vis15]. Compared to Oracle Eclipse® it has a much higher quality user interface, and Xamarin is fully integrated in Visual Studio®, to name just two of the benefits.

4.1.3 C#

C# is a type-safe, object-oriented, C-style programming language which runs on the .NET Framework, and is developed and supported by Microsoft® [CSh15]. C# was chosen over the more popular Java due to its strong integration in Microsoft Visual Studio, its support by Xamarin, and the competent support Microsoft® provides for the programming language. C# and Java just differ in minor details, see [Wik15b, Rade03].

4.1.4 Android

Currently Android is one of the most wide-spread operating systems worldwide, and it is developed under open source licenses by Open Handset Alliance, based on the Linux Kernel [RLMM09]. Figure 4.1 visualizes the number of smartphone users in millions in September 2013 [Stat13]. It can be seen that Android is market-leader with 70.9% of the users, iOS® follows next with 21.5%, and at the lower end Windows® Phone and BlackBerry® are following with 6%, and 1.5% of the users. In June 2015, 75.1% of the German smartphone market was served by Android, followed again by iOS® with 13.2%. Windows® has gained 4.5% compared to 2013 and now holds 10.5% of the market. BlackBerry® is still at the lower end with 0.7%, see Figure 4.2. This shows that Android has the broadest user base in Germany, but also worldwide the situation is quite similar as shown at [Worl15]. Additionally Android provides easier access to this broad user base by charging lower fees for ap-

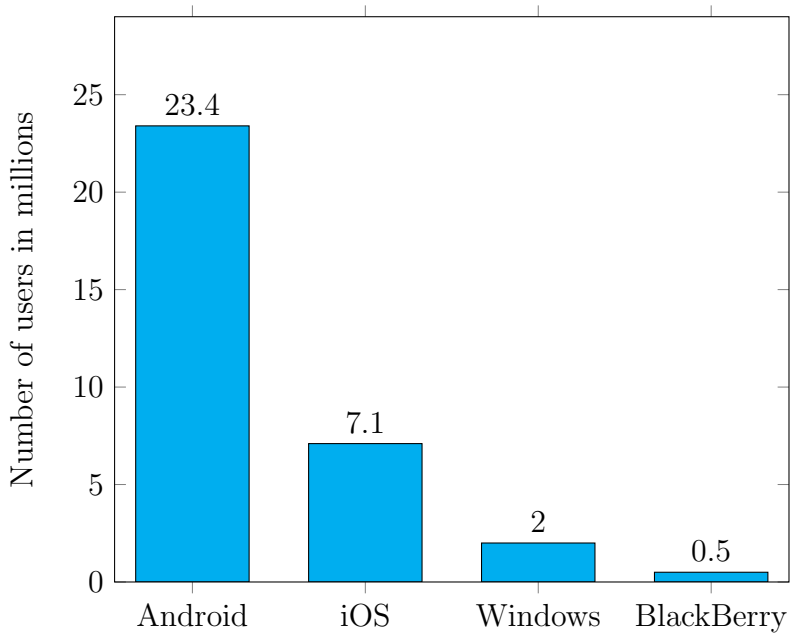


Figure 4.1: Number of smartphone users in Germany in millions, in 2013 [Stat13].

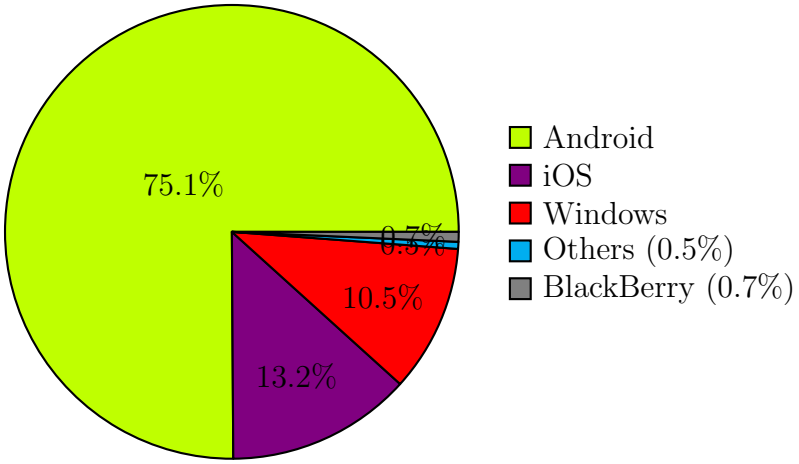


Figure 4.2: Percent of smartphone users in Germany, in 2015 [Worl15].

plication developers. For example Google© Play, as the largest Android distribution platform [Gigg14, Wik15c], charges a one-time registration fee of 25$ [Goo15]. In comparison to an Apple® App Store℠ membership, that costs 99$ per year [App15]. These facts finally led to the decision to choose Android as implementation framework for the high-fidelity prototype in this thesis.

Below the elemental components of an Android application are explained. They were then used for the Android prototype in section 4.2.

4.1.4.1 Activity

`Activities` are the most fundamental component of an Android application, they could be compared with a window on desktop computers. Nearly all `Activities` interact with the user, and have the purpose to display UI elements, or to process user input. They also can be used for control of program flow of the application. Especially small applications can be run in a single `Activity`.

Every Android application follows the *Activity Lifecycle* [Act15b]. As depicted in Figure 4.3 it is the central control center of the application. The `Activity` can be in one of three states: *Running* (visible), *paused* (partially visible), or *stopped* (hidden). The transition between two states subsequently calls the callback method annotated near the transition arrows in Figure 4.3. The `Activity`'s behavior can be declared by overwriting these callback methods. The *running*-state is reached via two sub states: *Created*, and *started*. These two states are used to initialize and connect the UI, and creating objects needed when the application is in *running*-state. In the *running*-state the `Activity`'s content is visible on the device's screen, and the user can interact with them. When the `Activity` is obscured partly – e.g. by a popup window – the `Activity` goes into *paused*-state. While in the *paused*-state the user cannot interact with the `Activity`'s content, and the `Activity` cannot execute any code, but it is still partially displayed. When the `Activity` is backgrounded – e.g. when the user pressed the "Home", or the "Return"-button – it is being stopped. In this state the `Activity` cannot execute any code and is no longer visible to the user, but it maintains its internal attribute state. When in *stopped*-state the `Activity` can be destroyed by Android when there is no memory left for running `Activities`, or manually by the user through the task-manger.

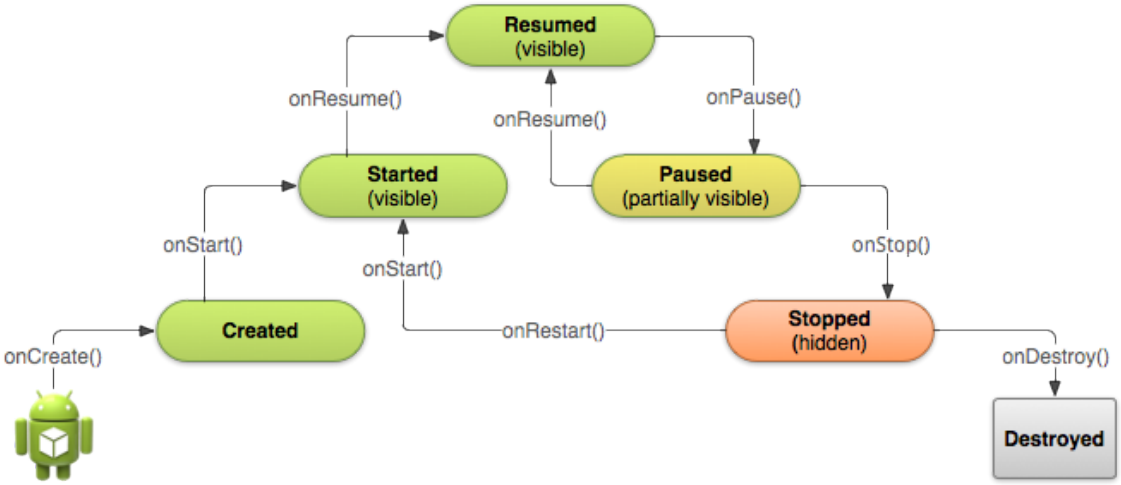


Figure 4.3: Android Activity Lifecycle [Act15b]

4.1.4.2 Fragment

Android applications are using `Fragments` in order to simplify changes in the behavior of the user interface. Its basic tasks are displaying UI elements processing of user input and manipulating UI elements within the `Fragment`. It cannot be used alone, it needs to be attached to an `Activity` through a `FragmentManager`. The `Fragment` has its own life cycle which is based on the `Activity`'s life cycle, and is subordinated to it – e.g. when the `Activity` is stopped, all attached life cycles will go into their "stopped"-status too [Fra15].

UI elements and their layout are defined through XML-files, this can also be done in the `Activity`. The advantage in using `Fragments` for UI behavior over `Activities` is, that the UI can be exchanged easier, and it can be reused to fit into different device formats [Fra15].

4.1.4.3 Service

`Services` are used in Android whenever it is necessary to perform long-term tasks in the background of any application. They are not meant to manage any user interaction, but to process data. There are two types of `Services`: Started, and Bound. The first of them is used for tasks which should keep running in the background until they finish their execution even if the starting component is already destroyed – e.g. downloading a file. The latter `Service` type is bound to an application component, and hence is bound to this components life cycle. This kind of `Service` offers interfaces to interact with other components, like sending requests, or getting results. Similar to `Activitys` and `Fragments`, every `Service` has a life cycle, which can be seen in Figure 4.4. On the left the life cycle of a "started"-`Service` can be seen, and on the right is the life cycle of a "bound"-`Service` depicted [Ser15]. For the high-fidelity prototype just "bound"-Services were used, these `Services` are explained in detail in section 4.2.2.3.

4.1.4.4 Intent

The last component listed here is an `Intent`. They are used for messaging purposes, requesting actions from other components. There are three major use cases for `Intents`: Starting `Activities`, starting `Services`, or broadcasting. For the transition

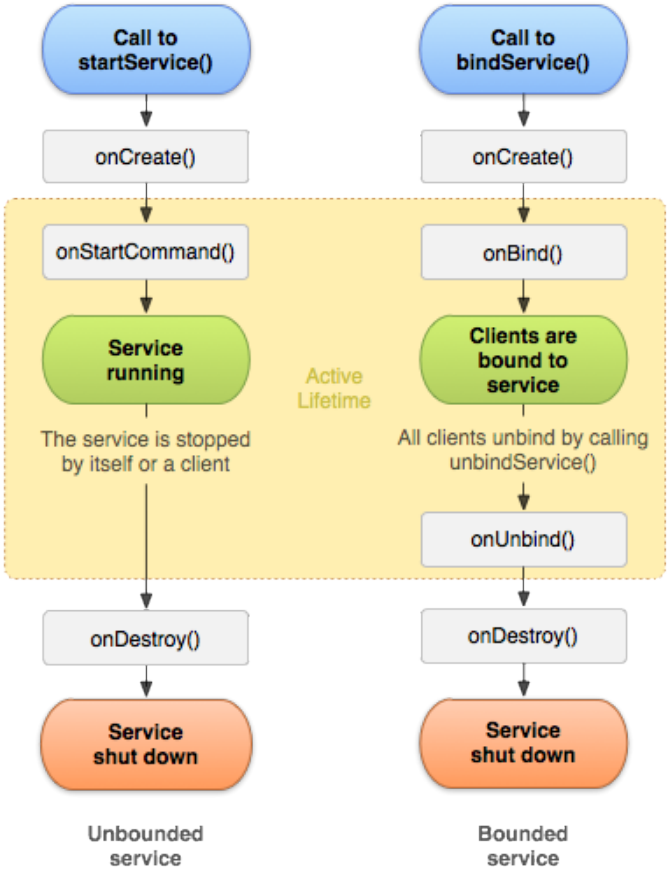


Figure 4.4: Android Service Lifecycle [Ser15]

between `Activities` the first case is used. The second case is used to start `Services` during execution of an application. The last case is used when a system wide broadcast is needed, this broadcast can be caught by any application on the system [Int15]. For the high-fidelity prototype just the first two cases were used.

4.2 High-fidelity Prototype

The implementation of the final, high-fidelity prototype for this thesis, consists of an Android application and a Java server. The prototype is conceptualized according to the agreement with the master thesis of Christian Voigt, as Client-Server architecture [Oluw], communicating through Representational State Transfer (REST) behavior [RiRu08]. The Client-Server architecture is depicted in Figure 4.5. The Android prototype itself is developed as standard three layer architecture [Mic15]. The Java server is not part of this thesis, hence it will only be outlined abstractly in this thesis.

4.2.1 Server

The server was implemented by Christian Voigt as part of his master thesis. It consists of a SQL database, a file system and the classifier. The SQL database is used to store all additional information sent with an audio sample, and connects the database entries with the associated audio sample, which is stored in the file system. The classifier is trained with samples stored in the servers file system, and the trained

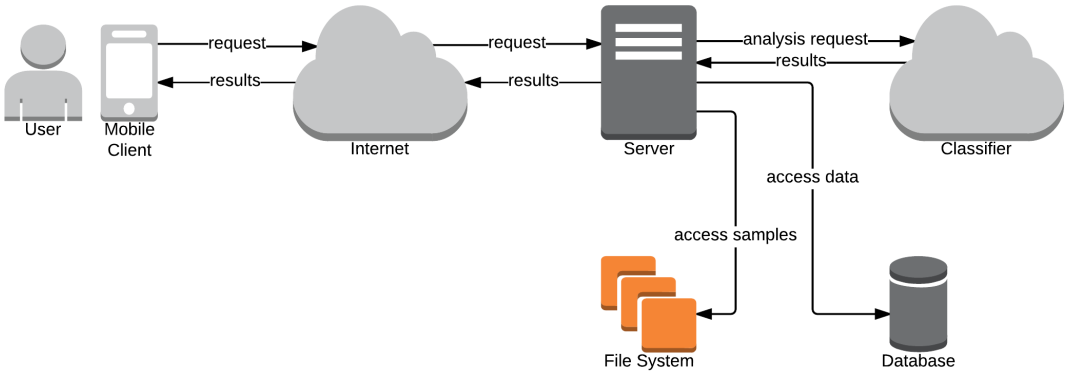


Figure 4.5: **Server-Client-Model** The mobile client can send analysis requests with an audio sample over the Internet to the server, which analyzes the received sample with a classifier, then saves the sample on its file system, and creates a database entry connecting the sample with the classification result.

classifier takes the audio samples sent by analyze requests as input. As a result the classifier returns the series of the surprise egg's content, more details on this decision can be seen in Chrisitain Voigt's work, which will be released shortly after this thesis. The server communicates over the Internet with the Android prototype, using RESTful Client-Server-Model. In order to communicate answers to the requesting Android client, a JSON object is used. This `Response` JSON object contains an identification string provided in the request from the client, a classification result string, and optional additional information in form of two string variables ("tags", "confidence"), see Figure 4.6a.

```
1 "response": {
2 "id":         "",
3 "class":      "",
4 "tags":       "",
5 "confidence": ""
6 }
```

(a) `Response` used for communication with the server.

```
1  {
2  "audio_path":   "",
3  "class":        "",
4  "confidence":   "",
5  "device":       "",
6  "id":           "",
7  "notes":        "",
8  "picture_path": "",
9  "tags":         ""
10 }
```

(b) `DataObject` used to store information on the mobile device.

Figure 4.6: JSON objects used in the high-fidelity prototype.

4.2.2 Android Application

Here the three layered approach of the application will be explained in detail. The application is composed of three logical layers with different tasks: Presentation layer, business layer, and data layer. In this thesis the same definitions and declarations are used to explain the architecture as given in Figure [Mic15]. The simplified

overview over the three layered model used for this prototype can be seen in 4.7, showing the three layers – presentation, business, and data – plus their components. The more detailed diagram is given in A.7.

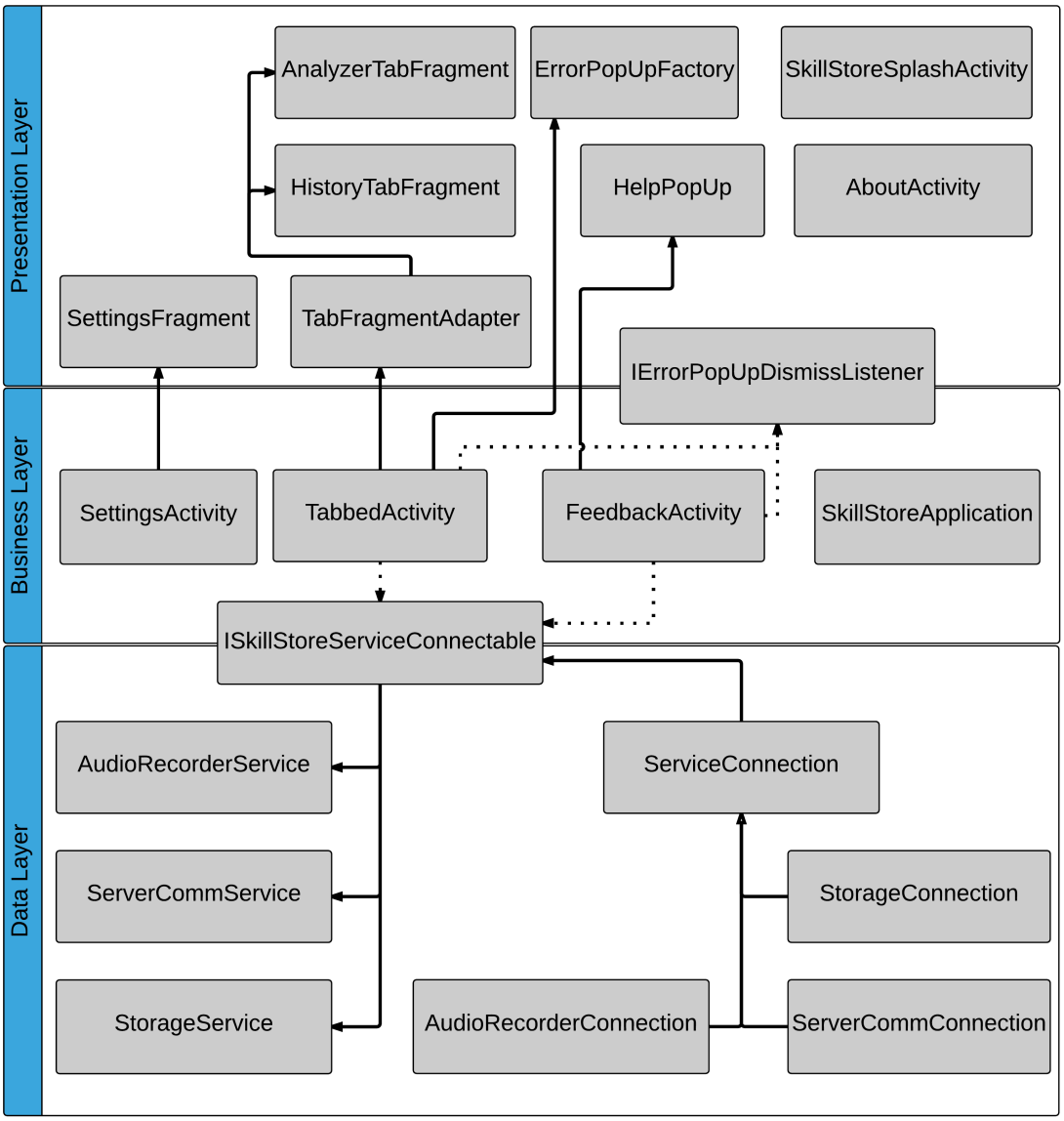


Figure 4.7: Simplified three layered architecture of the Android prototype.

4.2.2.1 Presentation Layer

The top layer is the presentation layer, which is user oriented and contains all functionality concerned with user interaction. It is responsible for displaying content and forward user input to the underlying business layer [Mic15]. This design approach gets along well with the Android design principles when `Fragments` are used [Fra15]. As explained in section 4.1.4.2, `Fragments` fill the role of the presenter in Android, sometimes also by `Activities`. In case of the high-fidelity prototype the `Activity` itself is to be located in the business layer, and is responsible for supplying the `Fragment` with data to display and managing which `Fragment` is to be shown.

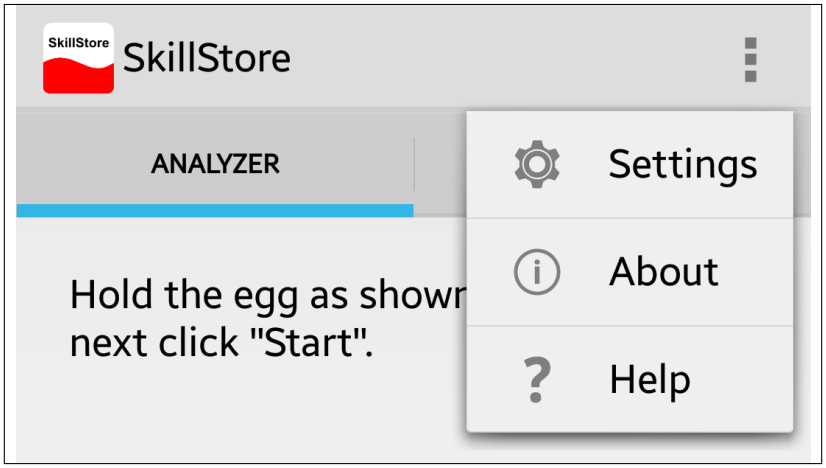


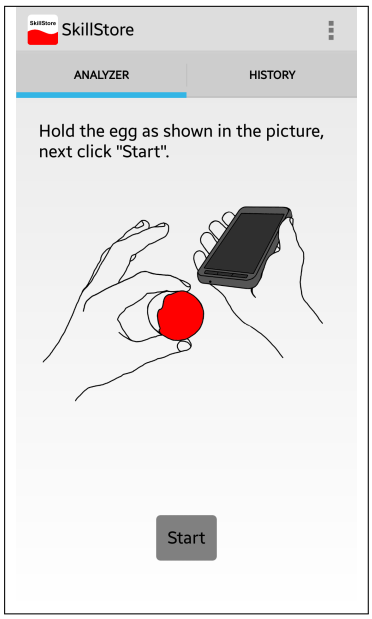
Figure 4.8: `ActionBar` with tabs and opened drop-down menu.

The two fundamental `Fragments` used in this prototype are: The `AnalyzerTabFragment`, and the `HistoryTabFragment`. They are both housed in, and controlled by a `TabbedActivity`, which violates the layered model, as its `ActionBar`'s tabs are used as main navigation mechanic. The `ActionBar` was configured according to the approach chosen for the paper prototype. It has two tabs: "Analyzer", and "History", which when selected displays the `AnalyzerTabFragment`, or the `HistoryTabFragment`. Furthermore, for the less used functionality the `ActionBar` provides a drop-down menu with the three entries: "Settings", "About", and "Help". The "Help" entry is located at the bottom of the menu to fit the Android design guidelines [Hel15], see Figure 4.8. For better usability for veteran users, swipe gestures as expert shortcut were implemented additionally to the tab navigation.

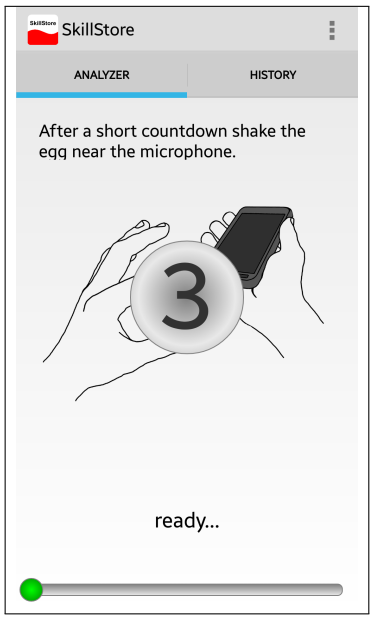
AnalyzerTabFragment. The first `Fragment` users will be confronted with – in the high-fidelity prototype – is the `AnalyzerTabFragment`. Users will spend most of their time in the application, interacting with this `Fragment`. It is responsible for showing the start screen (Figure 4.9a), the result screen (Figure 4.9d), and the animated screens for preparation and recording in between (Figures 4.9b, 4.9c). The main changes in comparison with the low-fidelity prototype are: The integration of the manual into the Analyzer, and the reduction and simplification of the shaking task.

Manual and analysis. In the process of implementing the manual a usability test with two users was conducted to test the slide show style manual. It was found that the users became confused when the application started with the now animated manual slide show. Users thought the analysis process already started, hence they shook the surprise egg during the animation and got confused that it did not stop, because the animation was looped. This happened even when the visible page was titled "Manual". Therefore the manual was integrated into the Analyzer itself, and each instruction is given at the time when it is needed (Figure 4.9). The text was reduced too, which was criticized during the formative evaluation (section 3.3). On the start screen (Figure 4.9a) the user is instructed how to hold the egg as shown in the picture and to press the "Start"-button when ready. Then the user has three seconds for preparation (Figure 4.9b), and he has time to read the instruction: To shake the egg near the smartphone's microphone when the countdown ends. By

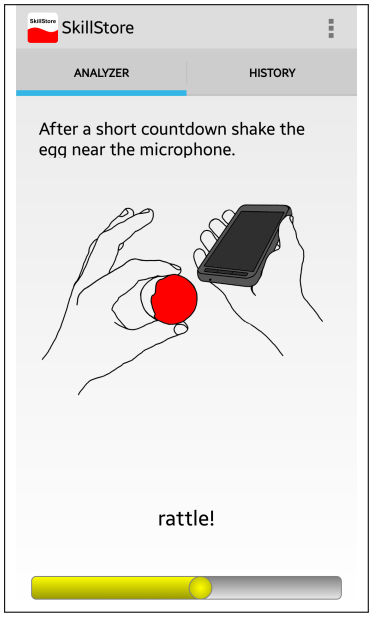
integrating the manual into the Analyzer the inconsistency problem illustrated in section 3.3.2 was solved too. This is because only one picture has to be displayed in this prototype. In Figure 4.9a can be seen that the picture with the hand holding a surprise egg was kept. The hand holding the egg was kept, because it shows the user exactly how to hold the egg and how to shake it, in comparison to just displaying a shaking surprise egg. In order to evaluate the draft of the integrated manual a usability test with three participants was conducted. The study showed that users got confused about where the smartphone's microphone is to be found. Hence the second hand holding a smartphone in relation to the shaking egg was added to the analyzer as depicted in Figure 4.9a. When the recording succeeded, the user comes to the result screen (Figure 4.9d), where he gets information about the result of the



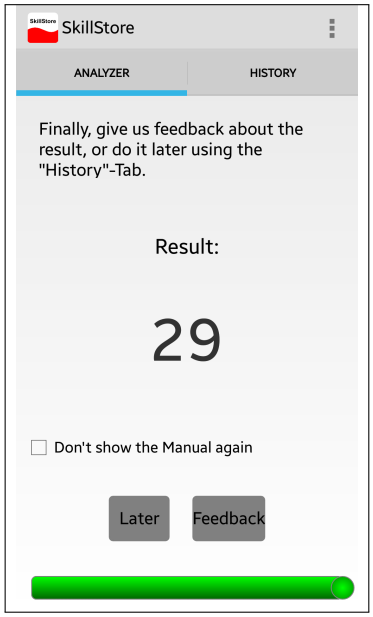
(a) Start screen



(b) Prepare screen



(c) Recording screen



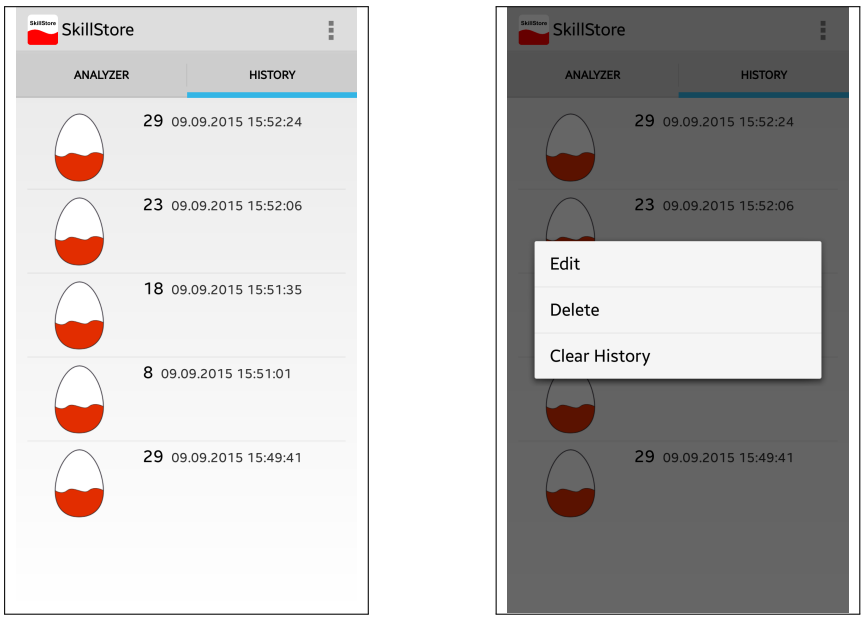
(d) Result screen

Figure 4.9: The different states of the "Analyzer"-tab.

analysis, and where he finds instructions how to give feedback. Either to give immediately feedback, or later using the "History"-tab. The formerly colored buttons were removed and simplified to a "Later", and a "Feedback"-button, which makes the navigation model more clear. Additionally the wording used on this screen is the same as in the instruction ("Feedback", "Later", and the "History"-tab is also visible), which supports the recognition rather than having the user recall the manual when he already dismissed it. On the result screen the user can – as before in the slide show manual – decide if the manual should be shown again during the next analysis. This decision can be reset at any time using the "Settings", or "Help" entries in the `ActionBar`'s drop-down menu.

Due to the fact that the audio records made during analysis are not preprocessed in any kind before sending them to the server, the three separate audio records do not serve any purpose anymore. Therefore the three records were reduced to one record, which is as long as two of the former records. The records are not preprocessed in order not to erase any information which could be used to analyze the sample.

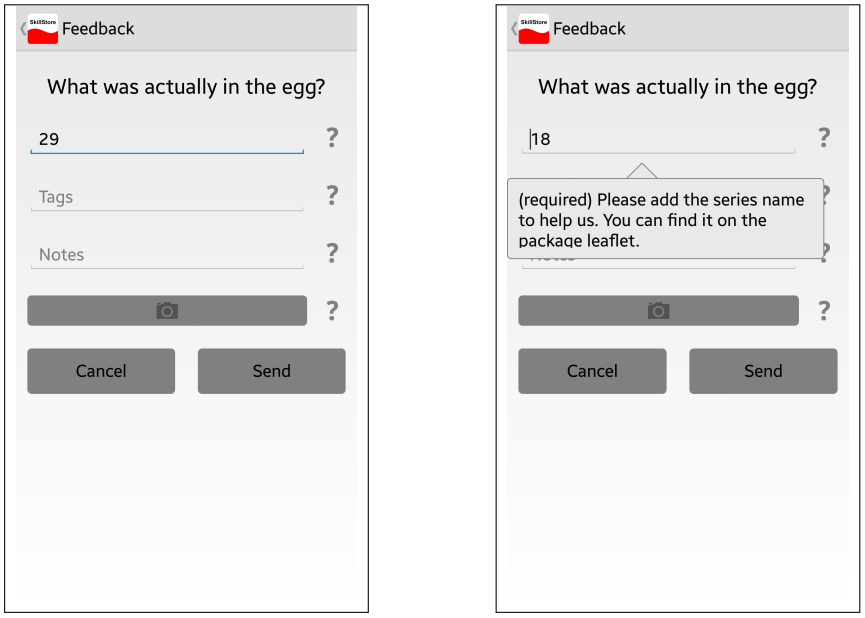
HistoryTabFragment. The `HistoryTabFragment` is the second `Fragment` housed in the `TabbedActivity`. As shown in Figure 4.10a, the listing style was adopted from the paper prototype, but the detail view was removed, because all information to be displayed fits in one list entry. Thereby the feedback function is reached with only one click, which reduces effort required by the user. Similar to the result screen (Figure 4.9d) mentioned earlier, the ambiguous colored buttons were removed. Thus, the navigation is designed more clear and easier. Additionally to the single short click, a long-click menu was added to the History's entries. This long-click menu provides the entries: "Edit", "Delete", and "Clear History" (Figure 4.10b). The first entry has the same functionality as a normal click on the entry. The second entry deletes the selected History entry, and the third entry deletes all data stored in the History.



(a) (b)

Figure 4.10: "History"-tab: (a) initial view, (b) with long-click menu.

FeedbackActivity. Concerning the feedback screen, which was before located under the "History"-tab it is now a standalone `Activity`, see Figure 4.11. Here again the layered model is broken. The `FeedbackActivity` is on one hand displaying a form sheet to enter the corrected analysis results, and on the other hand it communicates with the `Services` of the data layer, to send and save the feedback data. In later versions this `Activity` could be split up into a `Fragment` and an `Activity` to ensure the layered approach. It was decided to transfer the feedback functionality to its own `Activity` to make the navigational model more clear which was criticized during the formative evaluation (section 3.3)). Additionally with the `FeedbackActivity`'s `ActionBar` backward navigation to the formerly used `Activity` is possible [Nav15]. The tool notifications introduced in the paper prototype are inherited in this implementation. Here they are implemented as hovering popups beneath the related text field as can be seen in 4.11b. When entering the feedback form it is prefilled with all information already available to the system, and can be edited by the user here.

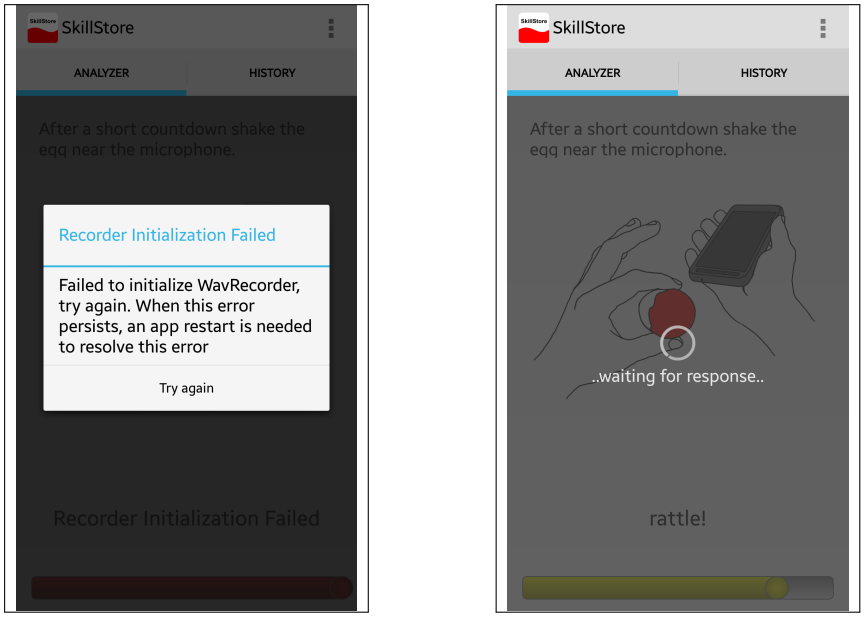


(a) (b)

Figure 4.11: Feedback form: (a) initial view, (b) with tool tip pop up.

Error and busy states. During implementation it was noticed that many more error cases occur than initially assumed in chapter 3. For that reason, the static error screen was replaced by popup messages which state error notifications and possible solutions, see Figure 4.12a. These popup messages are fit for reuse, a characteristic the static error screen could not account for. For better status visibility a "busy"-overlay (Figure 4.12b) was added to `TabbedActivity` and `FeedbackActivity`, which shows a rotating loading icon and a status feedback about what the application is currently doing in the background. These changes improve error recovery and make the internal status of the application visible to the user. The "busy"-overlay also helps to prevent the user, from making disturbing interactions, while the application is processing data, or is communicating with the server. However, without stripping him from control to cancel the running action. For canceling the running action users

simply have to navigate to another part of the application using the `ActionBar`, or by simply swiping while in `TabbedActivity`, which are both excluded from the overlay's disabling functionality.



(a) (b)

Figure 4.12: The overlays used: (a) error message overlay, and (b) overlay shown when the application is working in the background.

4.2.2.2 Business Layer

TabbedActivity. The presenter layer is governed by the underlying logic, or business layer. It encloses the system's main functionality, it is responsible to allocate work to the services in the data layer beneath, and it informs the presenter layer about the state to be displayed on the screen [Mic15]. The core of this layer consists of a `TabbedActivity`, which yields the application's main functionality. As mentioned earlier the business layer holds and controls the `AnalyzerTabFragment`, and `HistoryTabFragment` and provides the tabbed `ActionBar` for navigation. But it also starts services from the data layer, and processes their responses. The `TabbedActivity` implements the `ISkillStoreServiceConnectable` interface (Figure 4.7) and thereby binds the three data services `AudioRecorderService`, `ServerCommService`, and `StorageService` needed for all different interactions with data occurring in Analyzer, or History - more on the `Services` in section 4.2.2.3. In order to manage the interplay of the three services and the UI, `TabbedActivity` subscribes to three event handlers, one for each `Service`. In these three event handlers the main logic is implemented.

Layer communitcation. Figure 4.13 shows a top down view on a part of the program flow of an analysis process to be exemplary for the applications program flow and the interaction between the three layers. The analysis process is started when the "Start"-button in the `AnalyzerTabFragment` is pressed. This calls the `StartRecorder()` method of the `AudioRecorderService`, which then first stops all running worker threads contained in it with `StopRecorder()`, and then creates

and starts a new worker thread. While the worker thread is doing its work, a status object contained in `AudioRecorderService` is updated every time the worker reaches a milestone in its work process. For example the periodic update of the prepare countdown which is running on the worker. Every status update triggers a notification which is received by the `TabbedActivity`, which processes the notification and forwards the interpretation to the `AnalyzerTabFragment` to display the new status. When the `AudioRecorderService` finishes its work, the `TabbedActivity` calls the `ServerCommService`'s `PostAnalysis()` method to send the recorded data to the server. This suffices as example how the layers interact and how the `TabbedActivity` governs the program flow.

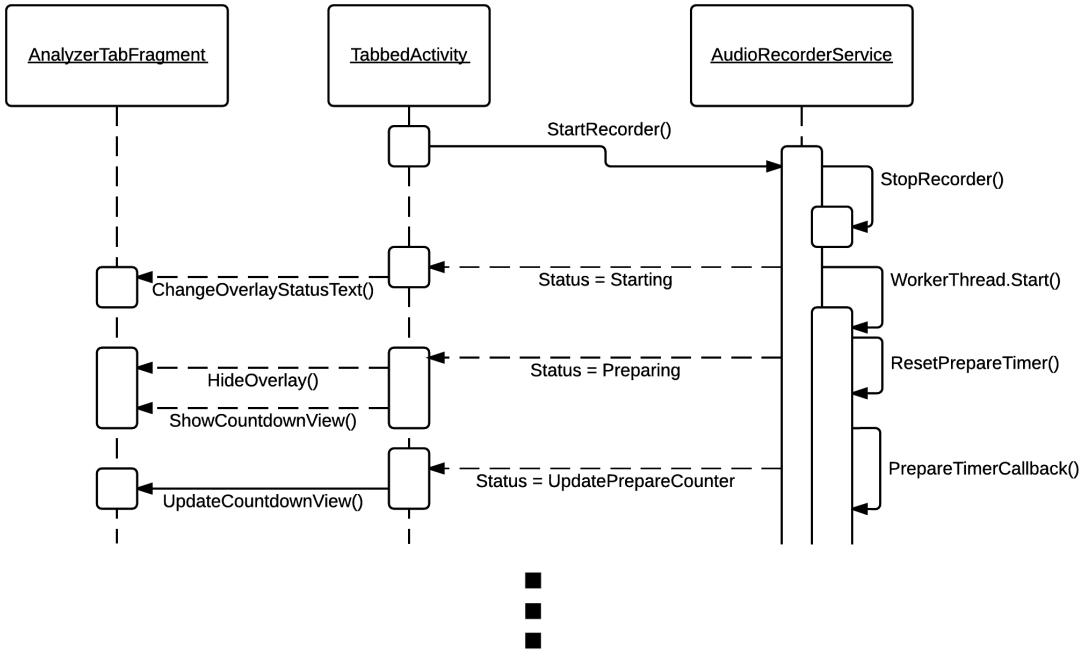


Figure 4.13: Part of the sequence diagram for an analysis.

FeedbackActivity. The `FeedbackActivity` reaches into the business layer too, and implements the `ISkillStoreServiceConnectable` interface. But it only binds the `ServerCommService` and the `StorageService` because the `AudioRecorderService` is not needed while giving feedback. When the "Send"-button in the `FeedbackActivity` is pressed the `ServerCommService`'s `PostAdditionalInfo()` method is called, which will send the given information to the server. When the `ServerCommService` finishes, the data returned by the server are saved with the `StorageService`'s `StoreData()` method. When saving succeeded, the `FeedbackActivity` ends its activation and gives the focus back to the `TabbedActivity`, from which it was called.

SkillStoreApplication. At some points in the system global states are needed. For example to get the `Application`'s context to access preferences, or store a `DataObject` while switching the `Activity`. For this task Android provides an `Application` class, which is here derived as `SkillStoreApplication`. Another task of the `SkillStoreApplication` is to start and stop the Android `Services`, which were used as base classes for the data services. By starting the `Services` in the

`OnCreate()` method of `SkillStoreApplication` some time consuming work can be done before the `TabbedActivity`'s `OnCreate()` method is called, where it will try to bind the `Services`.

4.2.2.3 Data Layer

The lowest layer is the service, or data layer. It provides interfaces for the overlying business layer to grant access to data within the system, and data exposed by other systems, e.g. over network [Mic15]. For this prototype the data layer consists of three services, which are all derived from Android `Service`. In order not to impair UI performance each of the three services derives a `Java.Lang.Thread` as worker, which processes the long tasks. Thereby the UI thread and the background services can run asynchronously to secure best responsiveness of the UI. In the following, each one of the three `Services` are described in more detail.

4.2.2.3.1 AudioRecorderService

The core `Service` is the `AudioRecorderService`, its task is to record the audio sample files needed by the analyzer. The audio samples are transmitted in raw

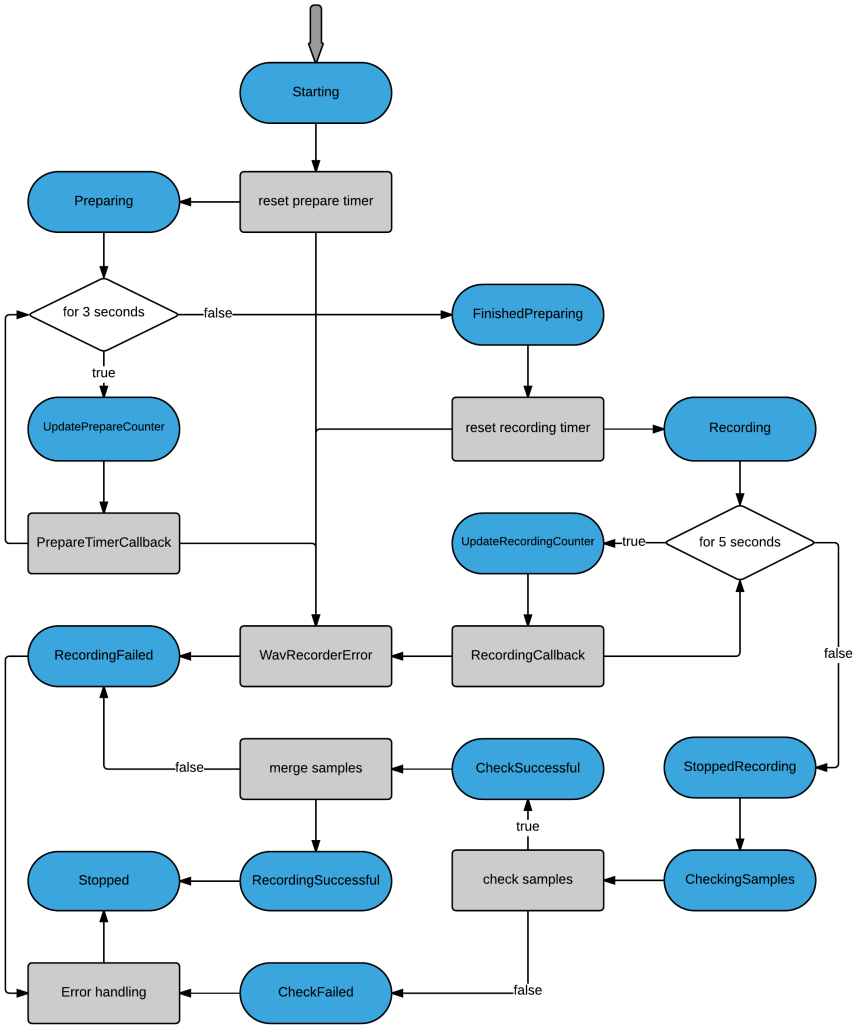


Figure 4.14: Flow chart of the `AudioRecorderService`.

WAVE file format [Rif15] to the server, so that no features which could be used, or are needed by the classifier are stripped from it. Preprocessing of the samples in this version of the prototype is reduced to checking the file length of the sample in order to ensure that the `WavRecoder` did record successfully. In addition it would be thinkable to check also the content of the sample; – for example to detect whether the user actually had shaken the egg but this would exceed the scope of this thesis.

Figure 4.14 shows the flow chart for one activation of the `Service`, from being started by the `TabbedActivity`, until its completion when it returns the path of the new sample, or ends execution because of an error. The flow is marked by two timer loops, one for the `_prepareTimer`, and one for the `_recordingTimer`, which schedule the preparation period and the length of the recording. During recording and preparation several errors could occur. These errors are mostly generated by the `WavRecoder` class, which wraps Android's `AudioRecord` class for PCM recording. Recorded files are saved to a temporary location in the smartphone's file system. This is done by the `AudioRecorderService` in this prototype, for later versions of this application, this task could be assigned to the `StorageService`.

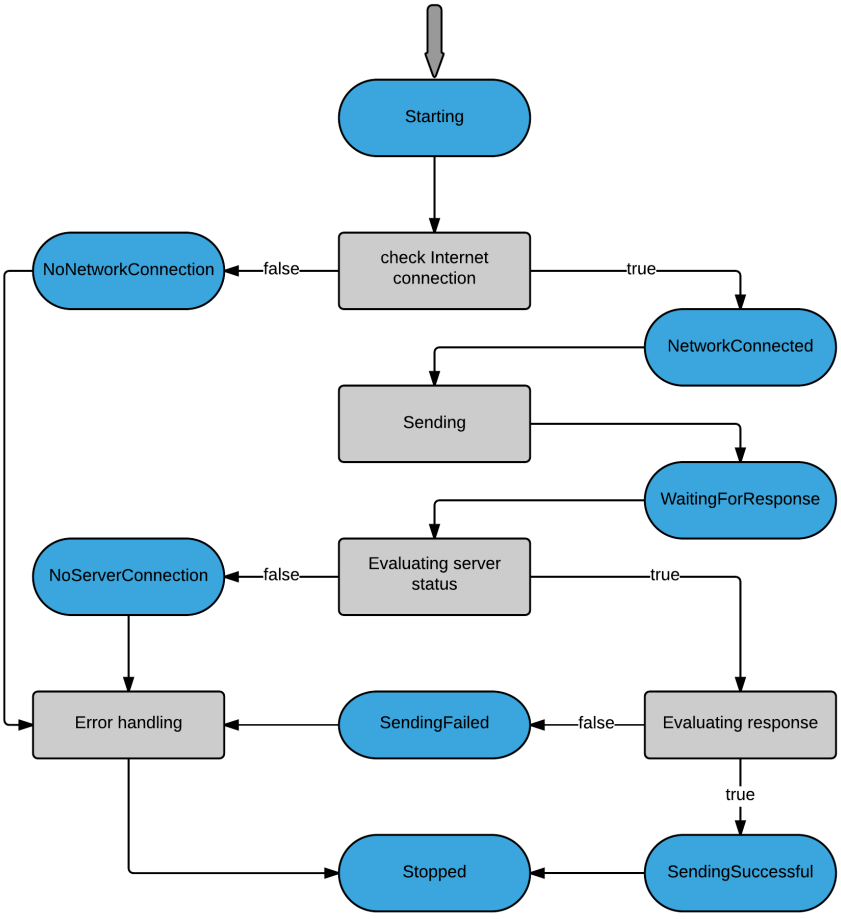


Figure 4.15: Flow chart of the `ServerCommService`.

4.2.2.3.2 ServerCommService

As the name suggests the `ServerCommService` is responsible for the communication between server and client. For that purpose there are two interfaces exposed to the business layer: `PostAnalysis()` and `PostAdditionalInfo()`. The flow chart depicting the general process of the `ServerCommService` is given in Figure 4.15. The two interfaces are running through this chart when executed, thereby they are sharing the same error, and intermediate states, and both of them will return to the "Stopped" status at the end of their execution.

The `PostAnalysis()` method asynchronously posts the current analysis' identification, device characteristic, and the recorded WAVE file, to the server. In return the Service awaits a JSON object as an answer. This `Response` JSON object has been depicted in Figure 4.6a, and was explained in detail in section 4.2.1. When receiving a response the `ServerCommService` inspects the received data: If the received identification matches the sent identification, and if the classification was successful. For verification of a successful run, a "FAILED" class was introduced to the classifier, which unites cases like noisy environment, or samples containing no shaking sounds. The results are then passed to the business layer, for further processing. The asynchronous post call sends once again the identification of the preceded analysis, the classification result string (edited by the user), and optional string variables ("tags", "notes").

The second interface `PostAdditionalInfo()` is used for posting the user's feedback, concerning the classification results. For verification of a successful post the `Service` receives the same `Response` JSON object as before during `PostAnalysis()`, see Figure 4.6a.

4.2.2.3.3 StorageService

The `StorageService`, provides several interfaces for data access and management. These interfaces are applied for example after successful posts of the `ServerCommService`, or when a reload of displayed data is needed. There are five interfaces implemented: `StoreData()`, `LoadData()`, `LoadAllData()`, `DeleteData()`, and `DeleteAllData()`. They will be explained in detail below.

The first of the interfaces is the `StoreData()` method, which saves given data in form of JSON objects, WAVE files, and image files on the file system. It is used every time when data have changed, e.g. after a successful analysis or when a feedback was sent. The flow of this program part is depicted in Figure 4.16. Data is either saved to the internal device memory, or if available to a Secure Digital (SD) card. Data is stored in form of serializable `DataObject`s (Figure 4.6b). `DataObject`s combine the different information parts mentioned in section 4.2.2.3.2 into one container. They can be serialized to JSON objects to ensure easy saving and loading from the file system. These `DataObject`s are also used to pass data between the modules of the application. The `DataObject` class provides merge interfaces, in case new data is given to an already existing identifier. These interfaces fill new data into the existing `DataObject`, or overwrite old values with newer one.

Stored data can be accessed through `LoadData()` or `LoadAllData()`. The former is

used to load a specific set of data, e.g. when the `FeedbackActivity` is started to edit an existing data set. However, when reloading the History's entries, the `LoadAllData()` method is used, which will load all stored data sets. When comparing the flow charts of `LoadData()` shown in Figure 4.17, and the one of `StoreData()` (Figure 4.16) it can be noticed that `LoadData()` is constructed inversely to `StoreData()`. Instead of serializing data, `LoadData()` deserializes stored JSON objects back into `DataObjects`. `LoadAllData()` works similarly to `LoadData()` with the difference that it loops over all stored data sets and loads them along the way.

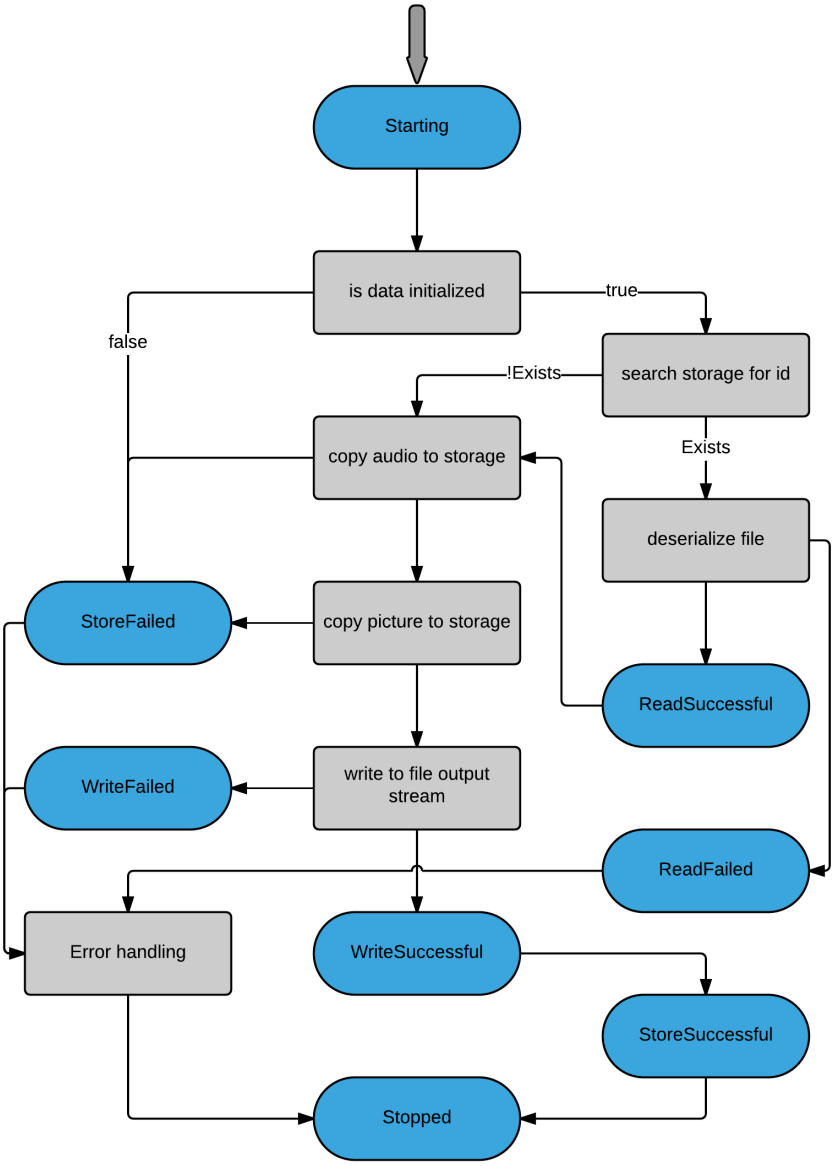


Figure 4.16: Flow chart of the `StorageService.StoreData()` interface.

In order to give the user more control over the data, and his device's memory, additional delete interfaces were introduced. First the `DeleteData()` method, which deletes a single data set from the file system, and second the `DeleteAllData()` method, which deletes all data sets stored by the application. These interfaces share one flow chart shown in Figure 4.18.

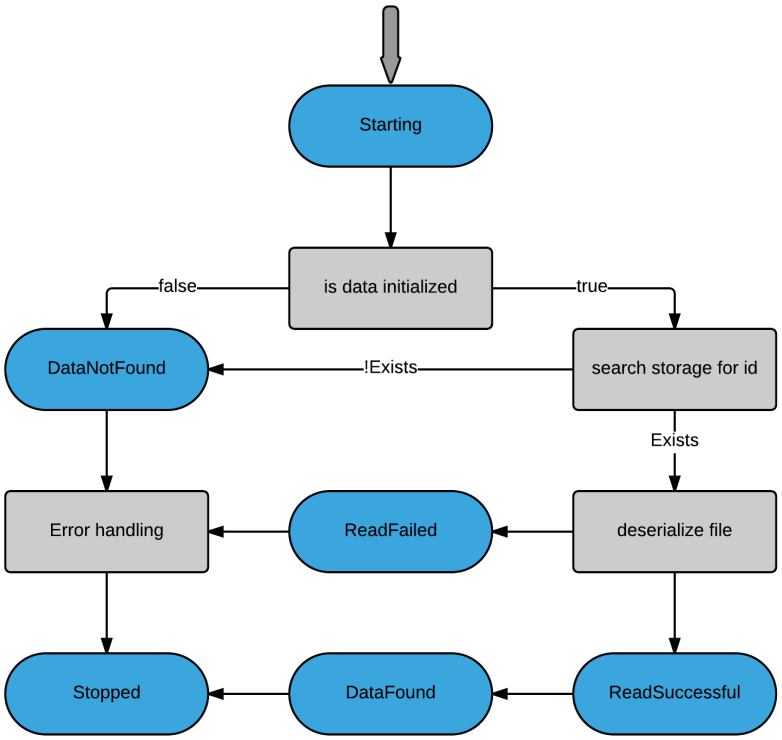


Figure 4.17: Flow chart of the `StorageService.LoadData()` and `StorageService.LoadAllData()` interfaces.

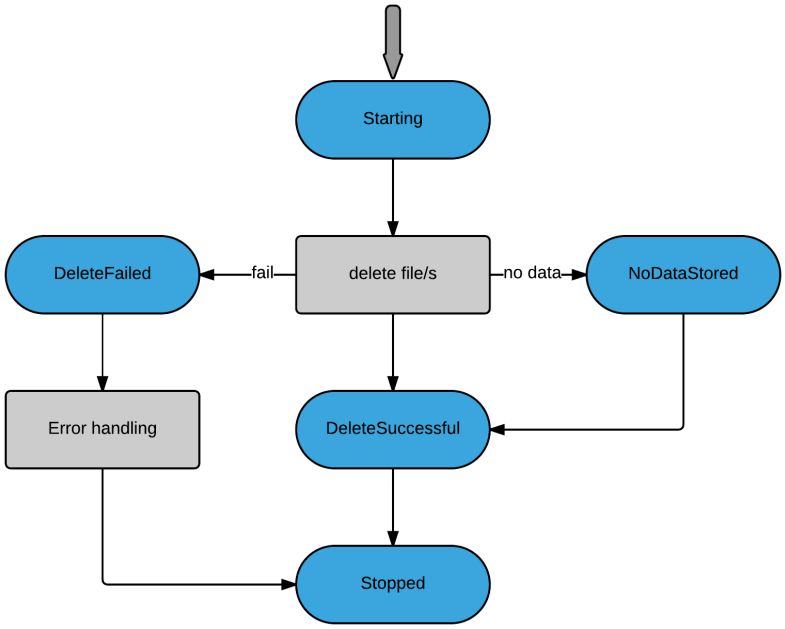


Figure 4.18: Flow chart of the `StorageService.DeleteData()` and `StorageService.DeleteAllData()` interfaces.

5. Summative Evaluation

In this chapter the final evaluation of the high-fidelity Android prototype is explained. The significant questions traced in this evaluation are: First, how is the user experience perceived by users? Second, does the prototype's design enable amateurs to perform as well as experts? And third, how does the users' performance differ between laboratory and field?

5.1 Study Design

Test setup. The study for this evaluation was designed as a Wizard of Oz (WoZ) experiment, in which the functionality of the prototype is partly, or completely simulated and does not work outside of the experiment [Kell84]. This was due to the fact that no working classifier for the surprise egg problem was available at this time.

In total 38 participants were tested between the ages of 14 to 53, twelve of the participants were female and the other 26 were male. The participants were divided into two groups: Amateurs, and experts, each with 19 participants. Each participant was tested in two different environments: Laboratory, and field. One half of the participants started with the laboratory test and the other half started with the field test, to control for learning effects. All participants had to perform the same task in both environments. The laboratory environment is characterized to have no or only little background noise. The field test on the other hand was performed in the entrance area of a supermarket, to simulate a realistic use case, and was characterized by loud background noise. One session with a participant, in both environments took approximately 30 minutes.

Task. In each environment participants had to test the high-fidelity prototype, which was installed on a the developer's smartphone with one test surprise egg, and eight other enumerated surprise eggs. The eggs had to be shaken in the same sequence each time to ensure that the wizard, which was faking the results worked properly, and thereby appeared trustful to the participants. After analyzing an egg the participants had to correct, or rather confirm the result to the server. This had

to be done, to label the recorded samples for later static analysis. When a participant had shaken all eggs he had to fill in a SUS [Saur11] about the prototype. At the end when a participant had executed both environments he was asked about his opinion on the prototype, and was enlightened about the wizard.

Amateurs and Experts. Participants were divided into amateur and expert groups according to their answers in a preceding questionnaire. This questionnaire and its results can be seen in A.8. Important characteristics for the expert group were: First, that the participant knew where the microphone was located at a smartphone (at the bottom edge), and that the microphone was used to record audio (Q7 in A.8). Second it is important to consider the environmental influence for higher quality audio samples, so to be classified as expert participants had to know that an environment with less noise was crucial for good audio quality (Q8 in A.8). The third question (Q9 in A.8) used for the group allocation, had the intention to see if the participants noticed that they should perform an audio recording and thereby understood what actually was asked of them.

Sample analysis. The static analysis of the recorded samples was performed afterwards to ensure that the poor performance of the classifier did not impair the perceived user experience. For the static analysis the Weka framework's J48 implementation of the C4.5 machine learning algorithm was used [Fran]. The participants' classification rate were evaluated against a baseline of 31.13% correct classifications. This baseline was generated through 407-fold cross-validation of the 408 sample trainings set. Additionally a second analysis was undertaken, taking the surprise eggs' content into account. This reduced the groups to classify to four, and thereby increased the baseline to 49.51%. The results of both analyses can be seen in section 5.2.2.

5.2 Results

The results to this study are divided into two parts: First, the user experience, and the design changes which can be derived from this evaluation. Second, the static analysis of the recorded samples, to evaluate the audio quality.

5.2.1 User experience

SUS score. The first metric to evaluate the user experience is again a SUS score. As mentioned earlier each participant had to fill in two SUS surveys, one after each environment test. The average results can be seen in Figure 5.1. As can be seen, there is no significant difference in perceived user experience, between both experts (89.25) and amateurs (88.1), and between laboratory (88.8) and field (88.55) test. This results in an average score overall filled SUS of 88.65, which can be graded as A (on a scale from A to F).

Observations. The second metric to evaluate the user experience are the observations made during the tests. In general participants behaved similar in both environments, with a few exceptions where participants felt awkward or slightly embarrassed to shake surprise eggs in the supermarket.

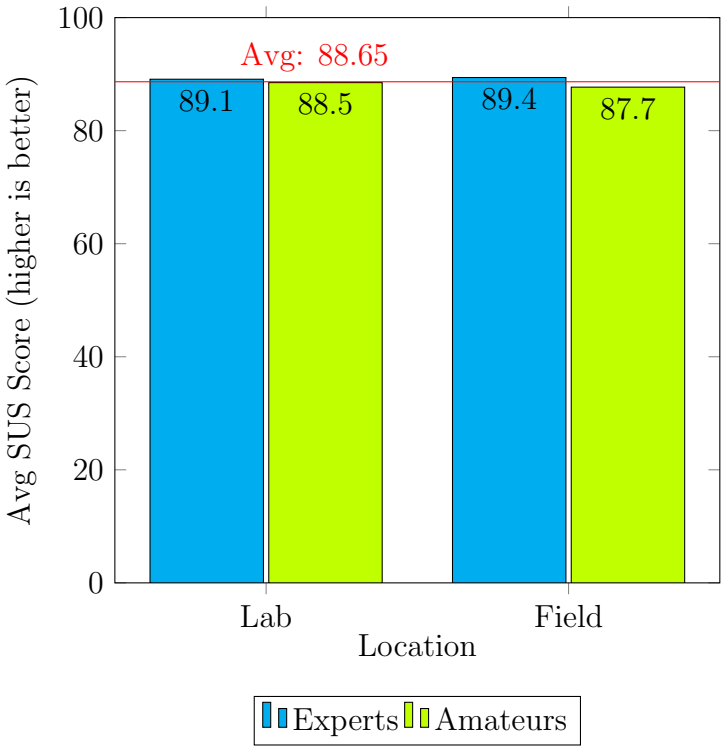


Figure 5.1: SUS score for each user group in each location, on a scale from 0 to 100 where values above 80.3 are graded as A, and everything below 51 is graded as F [Saur11].

Shaking method. A first observation made, was again the shaking behavior. 24 of the participants shook the eggs holding it at the flat sides and shaking it perpendicular to the peaks' axis (Figure 3.3b), as intended by the developers and presented via animation inside the prototype. Eight participants ignored the shaking direction of the animation and shook the egg in direction to its peaks, but still holding it on the flat sides, see Figure 3.3a. The other six participants ignored the holding instruction and the direction of the animation completely. These participants held the eggs at their peaks, and each three of them shook either in direction of the eggs' peaks (Figure 3.3c) or perpendicular to the peaks axis (Figure 3.3d).

Shaking position. A second observation concerning the prototype's main picture was that all participants, except for two, shook the egg at the right position at the bottom of the phone, where the microphone is usually located. These two exceptions had both a wrong mental model of a smartphone. One thought the microphone is located at the top of the smartphone where the front speaker for phone calls is located, and was confused himself about his action when he was asked about it afterwards. The other participant thought that the microphone is located at the back of the smartphone where the camera is located.

It could be assumed that the new picture with two hands, in the prototype has an effect on the participants, in such a way that it manipulates in which hand they hold the egg, and in which one the smartphone. Of the whole group of participants 30 had been right-handed, seven left-handed and one had no stronger hand (was ambidextrous). The smartphone was held by 21 participants in their left hand, and

by 17 in their right hand, correspondingly the egg was held in the respectively other hand. Among all participants 24 used their strong hand to shake the egg, from which were 19 right-handed and five left-handed. The other 11 right-handed participants held the egg with their weak hand, from which five explicitly changed their hand layout according to the picture in the prototype. So it can be inferred from this observation about the handedness, that the layout of the hands on the picture has a small effect on which hand the users will hold the egg, or respectively the phone. In order to reduce this effect further the picture could be mirrored, so that the right hand is holding the egg and the left the smartphone. This is because much more users are right-handed than left-handed, and from the data gathered here it seems more intuitive to hold the egg in the strong hand.

These findings are contrary to the findings of the ethnographic study (section 3.1.4) conducted earlier during the design process of the low-fidelity prototype. In the ethnographic study no significant difference between strong and weak hand usage for shaking was found. So the difference in this summative study could be explained in the way that the participants were more clear about what their actual task is. In contrast to the ethnographic study where the participants were taken from the street and had no time to think about the task they had been asked for.

In the summative study it was again confirmed that users forget to take off watches and bracelets which could make noise during the egg shaking. Eight participants kept jewelry on the shaking arm, and five on the phone arm. This negligence was already found earlier during the ethnographic study in section 3.1.4.

Design flaws. During the user experience evaluation several minor design flaws were found. The most important flaw concerned the manual. In the high-fidelity prototype it is integrated into the analysis process, and thereby instructions are shown when the countdown for preparation ticks down, see Figure 4.9b. This prototype behavior lead to two different participant behaviors: Ignoring the text instructions, or feeling hounded. The first behavior was the more frequent. Participants ignored written text in many cases completely and relied fully on the picture and its animation. On the other hand the second behavior could be perceived when participants tried to read the instructions while the countdown ticked down, which was not enough time. These two behaviors lead to two different design approaches: Either the instruction text is moved the start screen, in order to reduce stress. Or the text is removed completely since the animated picture is much stronger in regards to information transfer.

The slow shaking frequency of the animation confused some participants. They did not know if they had to shake strong so that a loud sound is generated, as the instruction “rattle!” indicates, or if they should shake slow as the animation suggests. In order to make it clearer to the user to really shake the egg this animation must be accelerated in the next version.

The error messages which occurred sporadicly while using the prototype in general had been to long. The participants did not read them, they just asked if they should click the “Try again”-button at the end of the message. As answer why they did not read the message, or did not know how to proceed the participants answered

that the terms used were too technical and the text too long. In order to improve these error messages a more simple language must be used and the text must be reduced significantly. This seems to be logical hence this application is also meant for children, and thereby should support their language instead of technical support requirements.

Another flaw is, that the progress bar is not filled completely when the busy overlay is shown. This lead to confusion among participants which took the progress bar as indicator for the timespan in which they should shake the egg. As a design change the progress bar should be filled when the recording time is over, to make the internal status of the application easier understandable for the user to the user.

Hindering for the work-flow of testing several surprise eggs in a row was the "data saved"-`Toast`[1] appearing over the "Start"-button on the start screen after sending a feedback to the server. Participants had to wait for several seconds, for the Toast to disappear and to continue with their task. Only the participants which developed Android applications themselves knew that `Toast`s could be removed by clicking on them. This overlap problem can be solved by increasing the size of the "Start"-button, which would also improve its visibility.

Giving feedback was also no trivial task, due to the fact that participants did not know what was meant with "Feedback", this word is chosen incorrectly. There are a few different approaches how to improve this design. One is to reintroduce the "true/false"-buttons used in the low-fidelity prototype (see section 3.2.2) with clearer labels. Another approach could be to combine the result screen and the feedback functionality so that no `Activity` switch is necessary.

Another poor text choice was made for the "don't show again"-check-box of the manual in the German version of the high-fidelity prototype which was used during the summative study. Here the check-box text said, translated to English: "Don't show this next time." Instead of: "Don't show the manual next time." This was misinterpreted by the participants and must be changed in the next version.

A general request by several participants was to make the prototype work without constant Internet connection, because a good Internet connection is not always guaranteed, especially in big supermarkets.

Synopsis. In summary, the high-fidelity prototype was perceived positively and the design of the main functionality was intuitive to use. Some minor flaws were found in the user interface which can be adapted in the next version of the application. By the means of the user experience evaluation no conclusions on the difference between laboratory and field can be made. But with regards to the amateur and expert groups it can be stated, that amateurs instructed by the prototype knew where to shake, and thereby where the microphone must be located, for presumably better audio recordings. Also all participants knew that they must not speak while recording audio.

[1] A little pop-up message in Android, which pops from the bottom edge of the screen like a toast.

5.2.2 Sample analysis

The subsequent analysis of the recorded audio samples is conducted to evaluate the quality of the recordings. As described earlier in section 5.1, analyses were executed: One with no information about the content of the eight eggs, and one with this additional information.

Analysis without content information. The first analysis, used a classifier trained to distinguish the eight eggs. For this purpose the eggs were numbered and each number represents one class. This classifier got a baseline correct classification rate of 31.13%, which is low but still above the 12.5% correct classification rate of guessing the class. The results of the analysis of all samples recorded by the participants can be seen in Figure 5.2. As anticipated the classifier performed better in the laboratory than in the field, with average classification rates of 12.75% in the laboratory and 12.7% in the field. But the difference between the both locations is not significant ($F(1,36)=0$, $p=1$). When comparing the results between experts and amateurs a bigger difference can be seen. Experts had an average classification rate of 14.8% in the laboratory, and a rate of 14% in the field. On the other hand the amateurs had an average classification rate of 10.7% in the laboratory, and 11.4% in the Field. But this approximately 4% difference in classification rate is also not significant ($F(1,36)=3.436$, $p=.072$), and the between-subjects analysis showed the same tendency ($F(1,36)=.17$, $p=.683$).

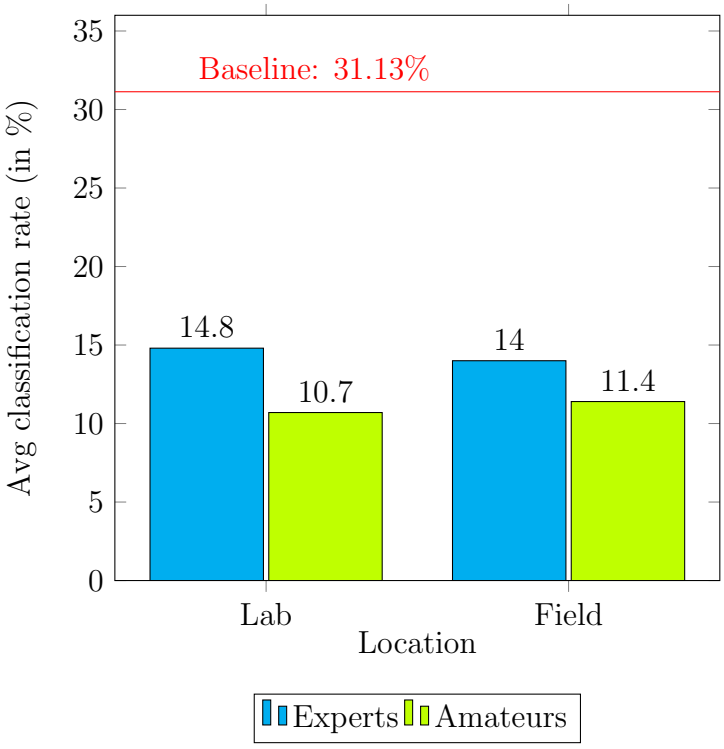


Figure 5.2: Average classification rate for each user group in each location against a base line of 31.13%.

Analysis using content information. The second analysis, used the information about the eggs' contents to increase the baseline, and participants' performance above guessing-level. The enhanced classifier only had to distinguish four classes: One class of three eggs (group 1), two classes of two eggs (groups 2 and 3), and one

class containing the remaining egg (group 4). The results can be seen in Figure 5.3. The baseline of the enhanced classifier increased by 18.38%, to 49.51%. By distinguishing only four classes the rate for guessing group 1 increased to 37.5%, guessing the groups 2 and 3 increased to 25% and the rate for guessing group 4 is 12.5%. Again the group evaluation between experts and amateurs had no significant results. Experts had average classification rates of 25.5% in the laboratory, and 34.9% in field test. Whereas the amateurs had classification rates of 22.5% in the laboratory and 35.2% in the field. The difference between groups is not significant ($F(1,36)=.178$, $p=.676$). Striking is the fact, that participants performed much better in the field compared to the laboratory which is counter-intuitively. In the field an average classification rate of 35.05% was reached, whereas in the laboratory only an average classification rate of 24% was reached, this difference is significant ($F(1,36)=17.366$, $p=0$). But this effect can be explained when looking at the detailed classification results of the field test. Here can be seen that the classifier only answered with group 1, which is the group with the most eggs. Hence an average classification rate of 35.05% is not surprising, it even is below expectations (37.5%). So in summary, the significant difference between laboratory and field is caused by the poor resistance of the classifier to noise. The between-subject analysis supports this observation showing that the difference is not significant ($F(1,36)=.591$, $p=.447$).

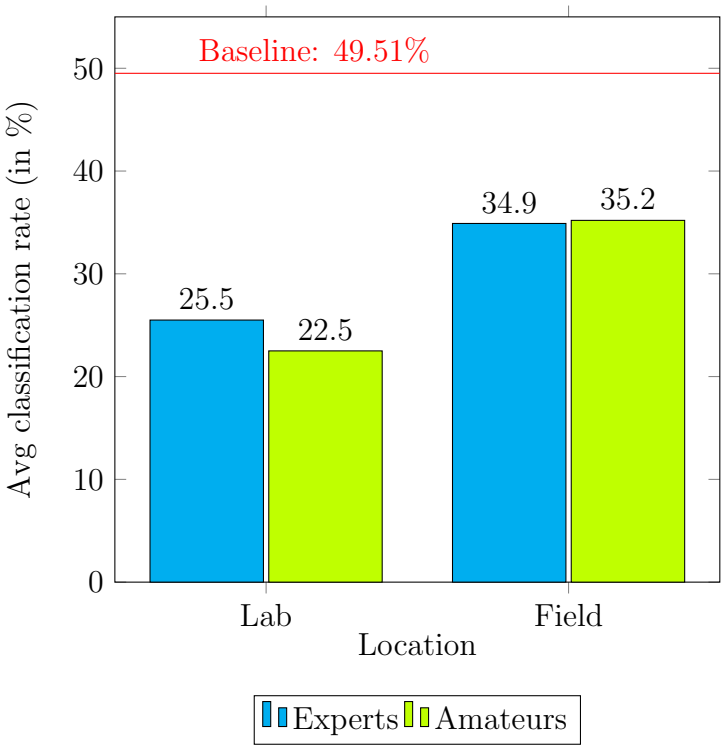


Figure 5.3: Average classification rate for each user group in each location. The egg content is used as additional information. Against a base line of 49.51%.

Conclusion. The conclusion to be drawn from the analysis of the samples, with the Weka J48 classifier is that experts and amateurs seem to perform similar when recording audio with the instructions given by the high-fidelity prototype. Furthermore, the location has no influence on the performance difference of experts and amateurs. For more detailed analysis of this observation a much better classifier

and probably preprocessing of the recorded samples is needed, but these actions exceed the limits of this thesis.

5.2.3 Summary

In summary the high-fidelity prototype was well received by the participants and led to the intended behavior. Analyzing the effects between different groups (experts, amateurs) no significant differences could be found neither by using the information about what was contained inside the surprise eggs, nor by ignoring the content information. The same results hold for the effects between locations when ignoring the content information. When using the content information a significant difference between locations can be seen. This effect can be explained by the poor noise resistance of the prototype version of the classifier and even if not directly contribution to the scope of the present thesis, at least the need of noise reduction means had been confirmed for the classifier.

6. Conclusion and Prospect

The objective of this thesis, as stated in chapter 1, is to study and design a system for acoustical object recognition on the basis of Kinder surprise eggs. The precise questions are:

1. Does the prototype instruct the users well enough to perform the task successfully?
2. Is the task designed intuitively for the users?
3. Does the prototype perform similarly in the laboratory and in the field (supermarket)?
4. Does the prototype instruct amateur users (in the field of audio recording) well enough, that they perform as good as experts would perform?

For answering these questions a formative process was undertaken.

First the system was conceptually designed, to understand its requirements and limitations. This included an ethnographic study, the development of a low-fidelity prototype, and a formative evaluation of the low-fidelity prototype. The ethnographic study – described in detail in section 3.1.4 – observed the behavior of people when they were asked to record the sound of a shaken surprise egg, and it was found that recording this sound is no trivial task. The low-fidelity prototype was implemented as a paper prototype and resulted in three different designs of instructions to the user, see section 3.2 for details. These three designs were evaluated in a formative evaluation as described in section 3.3. It was found that the design which instructed the user to shake for a given time instead of shaking in a given structure was the most promising. This is because this design burdened the participants with the least cognitive stress and resulted thereby in the best perceived user experience. In the formative evaluation it was also found, that the instructions given were not sufficient and had to be integrated better into the analysis task.

After the conceptual design process a high-fidelity Android prototype was developed

using the findings from the formative evaluation, see chapter 4. The high-fidelity prototype was evaluated in a summative evaluation, see chapter 5. The summative evaluation found that (Question 2) the system was easy to understand and intuitively to use. Therefore, the perceived user experience is high. The second important finding here was, (Question 4) that amateurs in audio recording could perform similar to experts in the field, when given the right instructions and hints. It was also found, that (Question 3) the performance of both groups was prone to noise in the environment. Taking the findings above into account, (Question 1) it can be argued that the users were instructed well enough to perform the given task. Though for a successful completion of the task a good performing classifier with audio preprocessing is needed additionally and cannot be replaced by expertise. While evaluating the instructions and hints given to the user, it was found that the instructions can be reduced even more, and a lot of information is communicated best with strong pictures and animations, immediately before and during the analysis.

A general improvement for the high-fidelity prototype would be if the classifier could be downloaded onto the smartphone. Because Internet connection is not guaranteed in every situation and location.

This high-fidelity prototype is highly specialized for the task to analyze Kinder surprise eggs, when it is thought about other applications for this prototype (e.g. car engine diagnosis, or finding ripe melons) it has to be modified for each use case, because of their strong differences.

For achieving similar user experience results in future interaction design projects in this field the following suggestions should be kept in mind. First, the interaction should be designed with the most unexperienced users in mind and it should be avoided to design for any kind of expert group. Second, the instructions given to the user should be held as simple and short as possible. Third, strong images and animations should be used instead of text, wherever possible.

A. Appendix

A.1 Smartphones sold worldwide 2007 to 2014

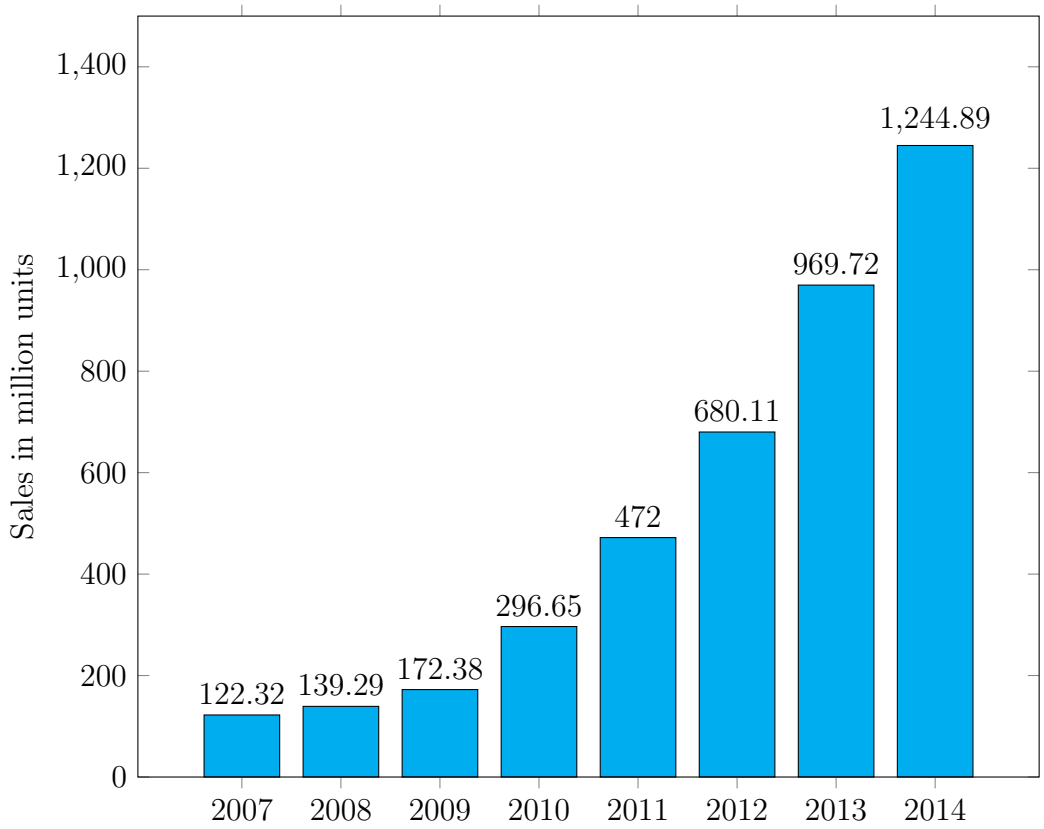


Figure A.1: Number of smartphones sold to end users worldwide from 2007 to 2014 (in million units) [Stat15].

A.2 Personas

Matt

Age: 23 **Gender:** Male

Behavior:

Started collecting surprise egg miniatures recently. Some nights he is out with friends. He is also quite strict with his studying behaviors.

Environment:

Single, Computer Science student in London. He has a scholarship and hence does not have to do any part-time job. Hence he has a lot of free time and enough money to finance a new hobby. A friend who collects miniatures for some time now suggested collecting surprise egg miniatures as hobby to him. He spends a lot of time on the campus and comes just to his dorm room to sleep.

Skills:

Is not good at guessing the content of eggs. Uses Computer and Smartphone in his everyday life, and does any maintenance and repair jobs on his own.

Goals:

Wants to collect the whole series of Happy Hippo Talent Show for a start, but does not want to spend a great deal of money on it in some collectors shop. And he wants to share his achievements with his friend.

(a) Persona Matt

Bernadette

Age: 72 **Gender:** Female

Behavior:

Continues collecting surprise egg miniatures in remembrance to her late husband, but she has trouble extending the collection with her skills. She meets a few friends sometimes for tea or while grocery shopping, and her children are visiting from time to time.

Environment:

Her husband died several years ago. Her children are grown up and she lives alone in her small house in Karlsruhe. Her husband collected surprise egg miniatures and had quite a collection. Her children also bought her a Smartphone to stay in contact with them and for any emergencies.

Skills:

Is able to call her friends and relatives saved in the phone and has found a preinstalled game on it which she is playing from time to time, if nothing else is up. She also has an old TV which she uses every evening. For anything further she calls her children for help. She has not the skills of her husband in collecting miniatures.

Goals:

Needs help with extending the collection.

(b) Persona Bernadette

Karl

Age: 37 **Gender:** Male

Behavior:

He is collecting surprise egg miniatures since he is a little boy. Uses his Computer and Smartphone every day. Is interested in using the newest technology in his everyday life. Is looking forward to teaching his newborn child all he knows.

Environment:

Married, since four years to his wife Beate, first child on the way. Living in Frankfurt, Germany. Close contact to his and his wife's family.

Skills:

Is able to use and configure his Computer and Smartphone alone, but prefers to see a specialist for any greater technical issue. Is able to dedicate what the content of a surprise egg is, only by hearing it shaking.

Goals:

Wants to give his knowledge to the community and wants a bit reassurance that his guesses are right when he is buying eggs. It would also be nice to have an easy way so store and see his figure collection, and statistic values like how many eggs have I opened by now, and how many where preferred eggs.

(c) Persona Karl

Leon

Age: 9 **Gender:** Male

Behavior:

Attends school and meets his friends after homework at the playground. Where they play video games on their smartphones. When playing video games gets boring they visit the nearest grocery store to spend their pocket money on candy.

Environment:

Lives with his parents in Karlsruhe and is in the 4th grade. He has a lot of friends in his class, and an elder Sister who is 16 and in the 11th grade at a "Gymnasium". For his birthday he got his first smartphone.

Skills:

His grades at school are "OK". And he is using the family computer at home to chat with his friends and his smartphone when is not allowed to use the computer. He knows how to install and use numberless apps via "trial and error".

Goals:

Likes surprise egg miniatures, but just the "cool one". And wants the best content from each egg he buys from his pocket money.

(d) Persona Leon

Figure A.2: The persona used during this thesis.

A.3 User Stories

A.3.1 US010/Analyze egg

Matt is at a supermarket in London, doing his weekly grocery shopping. Besides grocery he plans on buying a few surprise eggs to start his collection. He recently downloaded the new SkillStore application, and would like to test its performance today. When he is at the rack where the surprise eggs are stored he pulls out his phone and starts the SkillStore application. The application gives him instructions what to do next. When he is asked to shake the egg near the microphone Matt shakes the egg near the phone. As result the application presents "Minions" to him. Matt is not interested in this series, so he tries several other eggs until he found two which were titled as "Happy Hippo" by the application. He decides to buy these two.

A.3.2 US020/Give direct feedback

Matt got a few eggs as present from his parents. This time he would like to help improving the analyzer, because last time one result turned out to be wrong. So he starts the SkillStore application again and analyzes the first surprise egg. The result is "Smurfs", he directly opens the egg to see if the result was correct. A "Cars" car emerges, so the result is wrong. He goes to the feedback section of the application and corrects the "Smurfs" result to "Cars" and sends this to the server. Then he continues to analyze the other eggs.

A.3.3 US021/Give indirect feedback

Matt bought again three eggs while grocery shopping. He already analyzed them in the shop. He could not open them in the shop so he opens the eggs at home. He memorized which egg he analyzed when. Again he wants to help improving the analyzer, so he starts the app after opening the first egg and navigates to the "History" section where all past analyses are stored. He selects the one matching the time when he recorded the first of the new eggs. He is moved again to the feedback section, which he sends this time without correction the result, because the promised "Happy Hippo" was a "Happy Hippo". He returns to the "History" and continues with the next egg.

A.4 Use Cases

A.4.1 UC010/Static menu navigation

Matt is at the grocery store and would like to analyze some eggs to buy one or two. In front of the shelf with the surprise eggs he pulls out his smartphone and starts the SkillStore application. He is confronted with a menu which fills the whole screen. In this menu he can choose between "Analyzer", "History", "Settings", "About", and "Help". He would like to analyze an egg, so he chooses "Analyzer". Next the analysis step follows, which is described in detail in A.4.4. After the analysis Matt returns to the menu page. Now he would like to see what version the currently installed application has, so he chooses "About" from the menu. In the "About"-page he is presented information about the developers, the applications content and the application version – it is 2.2. Then he returns back to the menu page to analyze more surprise eggs.

A.4.2 UC011/Swipe menu navigation

Matt is at the grocery store and would like to analyze some eggs to buy one or two. In front of the shelf with the surprise eggs he pulls out his smartphone and starts the SkillStore application. He is confronted with the start screen of the analyzer. But before he analyzes any eggs he would like to know the version of the currently installed application. In order to bring out the menu he swipes horizontally from left to right over the screen, a menu bar slides in from the left side. From this menu bar he chooses "About". In the "About"-page he is presented information about the developers, the applications content and the application version – it is 2.2. Then he presses the "Return"-button to get back to the analyzer start screen. The menu bar is hidden again.

A.4.3 UC012/Tab navigation

Matt is at home and would like to analyze some eggs he got as a present. He opens the SkillStore application and is confronted with two tabs. The analyzer tab is selected and its content fills the rest of the screen. The analyzer start screen can be seen. He first performs the analyses for all eggs he got. The analysis is described in detail in A.4.4. Then he opens them and decides to give feedback to the analyzer about its results. In order to do that he selects the "History"-tab and is presented with a list of all past analyses. He gives feedback for all eggs he just opened, as described in A.4.6. After that – he still is in the "History"-tab – he would like to see the applications version. Therefore he clicks the "Menu"-button in the top right corner of the screen, and a drop down menu appears with the entries: "Settings", "About", and "Help". He chooses "About". In the "About"-page he is presented information about the developers, the applications content and the application version – it is 2.2. Then he presses the "Return"-button to get back to the tabbed screen he came from. The "History"-tab is still selected.

A.4.4 UC020/Analysis

Matt is at the grocery store and would like to analyze some eggs to buy one or two. In front of the shelf with the surprise eggs he pulls out his smartphone and starts the SkillStore application. He is confronted with the start screen of the analyzer. The screen holds all instructions on how to perform the analysis and what to do with the surprise egg during the analysis. He then clicks the "Start"-button at the bottom of the screen. Then a countdown appears and begins at three. When it reaches zero it disappears and the surprise egg picture earlier obscured by the countdown now starts to shake. Which is a signal for Matt to start shaking the egg he picked up during the countdown. After a few seconds the egg on screen stops shaking and so does Matt with his egg. The countdown reappears. This procedure is repeated another two times, then Matt is presented with a results screen. It says "Happy Hippo". Matt is pleased with the result. He buys this surprise egg and leaves.

A.4.5 UC030/Feedback after Analysis

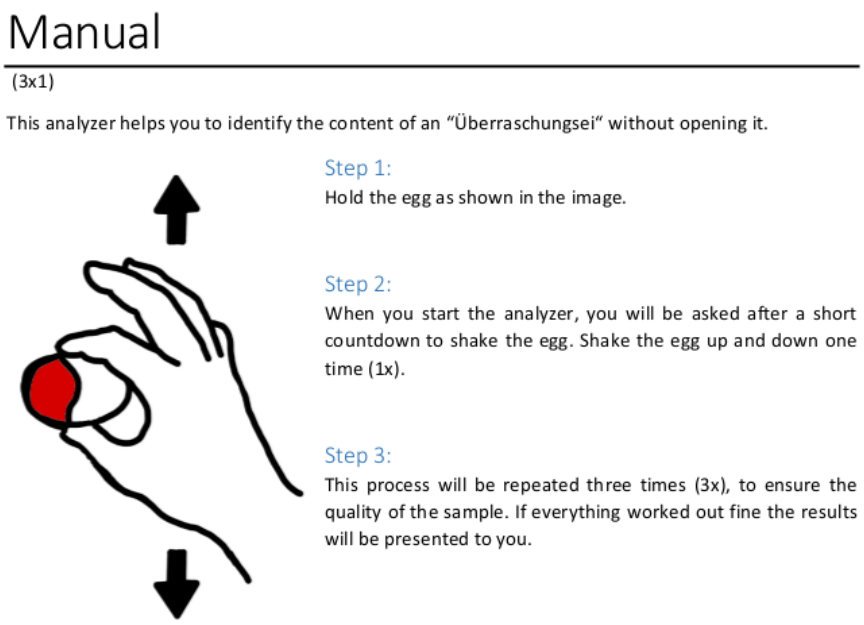
Matt just performed an analysis as described in A.4.4 and is currently on the result screen. He would like to give immediately feedback about the eggs content. Therefore he opens the egg and the promised "Happy Hippo" was a "Cars" car. He presses

the “Feedback”-button at the bottom of the result screen and is presented with a form page. In the first text edit field the given result is shown “Happy Hippo”. He changes the entry to “Cars”, in the next field he enters the exact model he received. Then he presses “Send” to correct the analyzer with this feedback. He is then returned to the analyzer start screen.

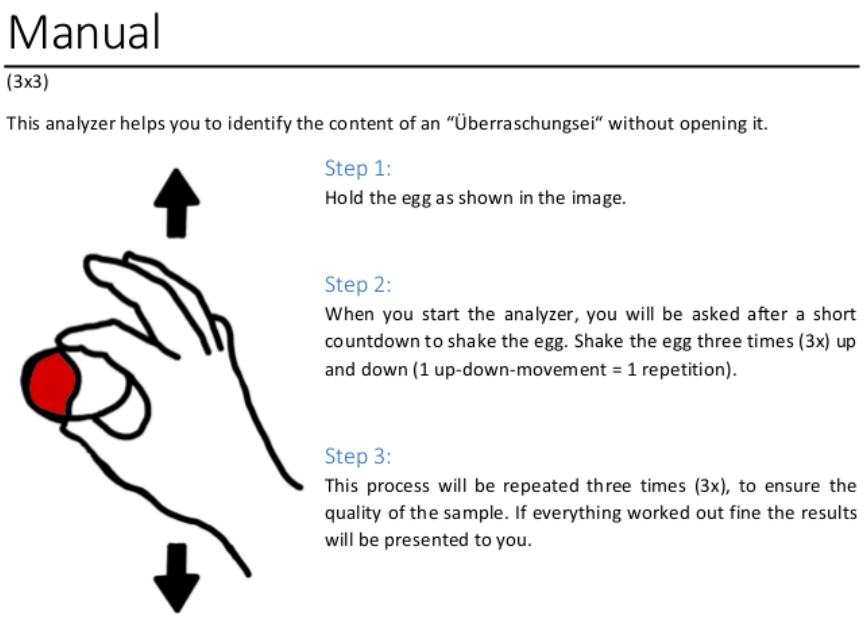
A.4.6 UC031/Feedback over History

Matt performed several analyses and would like now after opening the surprise eggs to correct the analyzer. For this purpose he opens the application and navigates to the “History”. On the page he is presented by a list of all past analyses. He chooses the one he would like to correct. He is moved to the same feedback form as described in A.4.5. There he performs the same tasks and sends his corrections by pressing “Send”. He is returned to the “History” page where he continues correcting the other eggs in the same fashion.

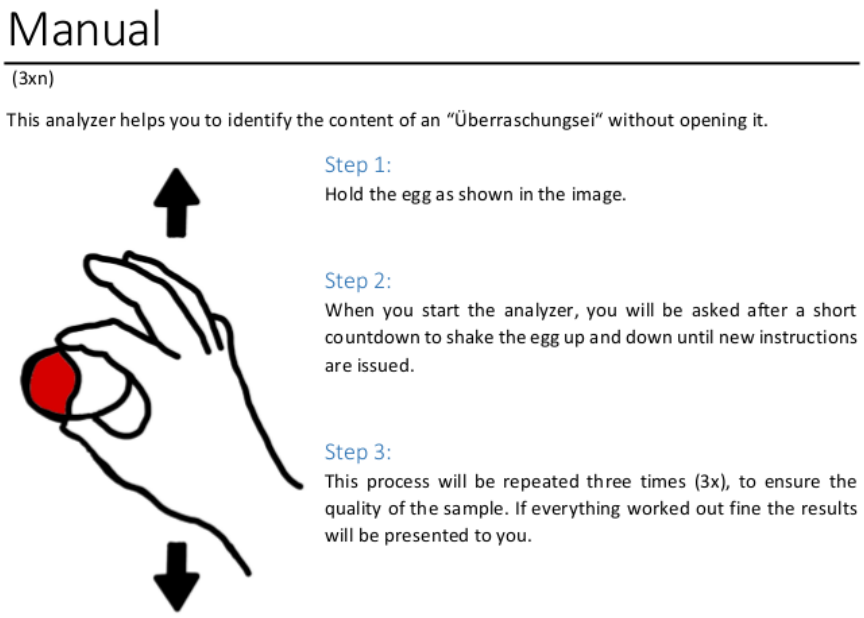
A.5 Paper Prototype Manual



(a)



(b)



(c)

Figure A.3: The three different manuals used during the formative study. (a) instructs to shake the egg one time during each recording part, (b) instructs to shake the egg three times during each recording part, (c) instructs to shake the egg continuously during the each recording part

A.6 Formative Study: Questionnaire

A.6.1 Study Survey

Formative Evaluation

System Usability Scale

	Strongly disagree				Strongly agree
1. I think that I would like to use this system frequently	1	2	3	4	5
2. I found the system unnecessarily complex	1	2	3	4	5
3. I thought the system was easy to use	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this system	1	2	3	4	5
5. I found the various functions in this system were well integrated	1	2	3	4	5
6. I thought there was too much inconsistency in this system	1	2	3	4	5
7. I would imagine that most people would learn to use this system very quickly	1	2	3	4	5
8. I found the system very cumbersome to use	1	2	3	4	5
9. I felt very confident using the system	1	2	3	4	5
10. I needed to learn a lot of things before I could get going with this system	1	2	3	4	5

Survey

	Strongly disagree				Strongly agree
1) I am sure I would have produced a good audio recording.					
	1	2	3	4	5

2) Did something in the App confuse you? When yes, what was it?

3) Which step in the manual have you understood best? And why?

4) Which step in the manual was hardest to understand? How could it be better?

5) How did you understand the eggs shaking?

A.6.2 Debriefing Survey

Debriefing

1) Please, order the three designs by difficulty:

2) Which design do you think will result in the best audio samples?

3) Where do you see Problems for the app in real life? (e.g. Supermarket)

4) Comments, opinions, suggestions:

A.7 High-fidelity Android prototype architecture

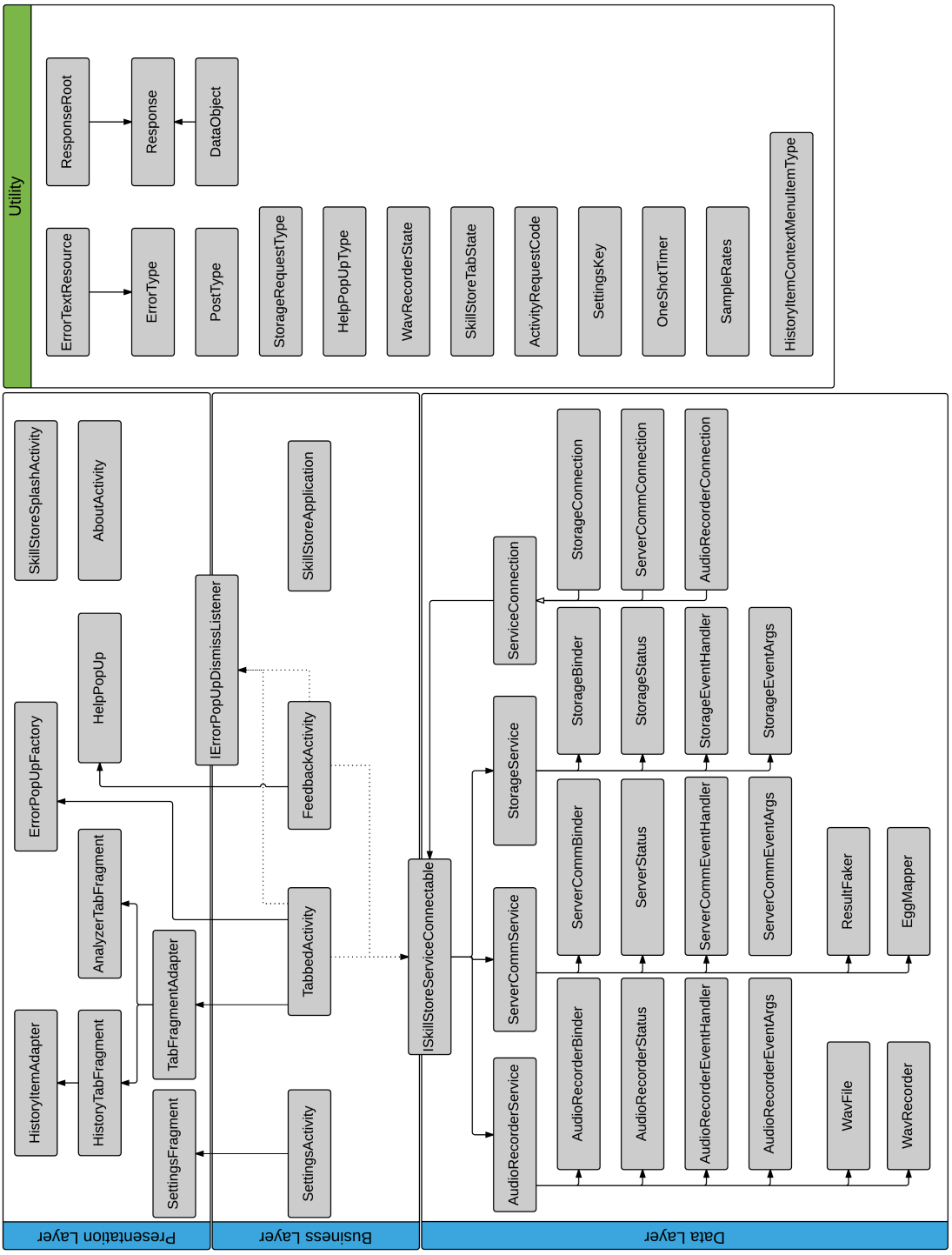


Figure A.4: Three layered architecture of the Android prototype. Utility module provides general functionality and status and type enumerators, used by several classes. The realations between utility classes and other classes are omitted for clarity.

A.8 Summative Evaluation: Questionnaire

A.8.1 Questionnaire: Group allocation

Fragebogen

Ihre Aufgabe während der Studie wird es sein möglichst saubere Tonaufnahmen von Überraschungseiern mit Ihrem Smartphone zu machen. Dieser Fragebogen bewertet Sie nicht, wie gut oder schlecht Sie für die kommende Studie geeignet sind; er hilft uns lediglich dabei, etwas mehr über Sie zu erfahren.

1. Probandennummer: _____

2. Haben Sie schon einmal Tonaufnahmen gemacht? (egal wie)

   ☐ Ja ☐ Nein (weiter bei Frage 7)

3. Falls Ja: Womit haben Sie schon Tonaufnahmen gemacht?

4. Falls Ja: Wie häufig haben sie in der Vergangenheit Tonaufnahmen gemacht?
   - ☐ Häufiger als 1x pro Woche
   - ☐ 1x pro Woche
   - ☐ 1x pro Monat
   - ☐ Seltener als 1x pro Monat

5. Falls Ja: Wie häufig machen Sie aktuell Tonaufnahmen?
   - ☐ Häufiger als 1x pro Woche
   - ☐ 1x pro Woche
   - ☐ 1x pro Monat
   - ☐ Seltener als 1x pro Monat

6. Falls Ja: Wie lange ist Ihre letzte Tonaufnahme her?
   - ☐ Länger als ein Jahr
   - ☐ ein Jahr
   - ☐ ein paar Monate
   - ☐ weniger als ein Monat
   - ☐ weniger als eine Woche

7. Wo in Relation zu Ihrem Smartphone würden Sie das Ei schütteln, um eine möglichst saubere Tonaufnahme zu erhalten? Und warum?

8. Wo ist Ihrer Meinung nach der beste Ort in einem Supermarkt, um Tonaufnahmen zu machen? Und warum?

9. Wenn Sie mit dem Videorecorder auf Ihrem Smartphone den Ton des Eies aufnehmen würden, wo würden Sie dann das Ei in Relation zu Ihrem Smartphone halten? Und warum?

A.8.1.1 Results

The survey was filled in by all 38 participants of the summative evaluation. The results of the multiple choice questions are presented below.

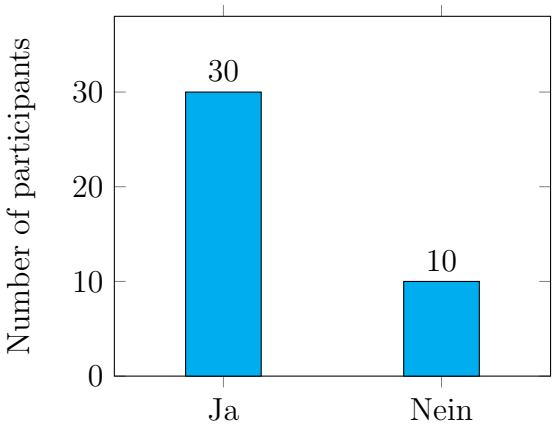


Figure A.5: Results of Q2: Have you ever recorded some audio?

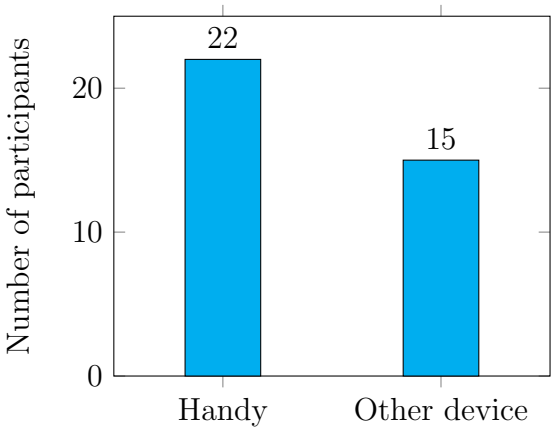


Figure A.6: Results of Q3: What did you use to record audio?

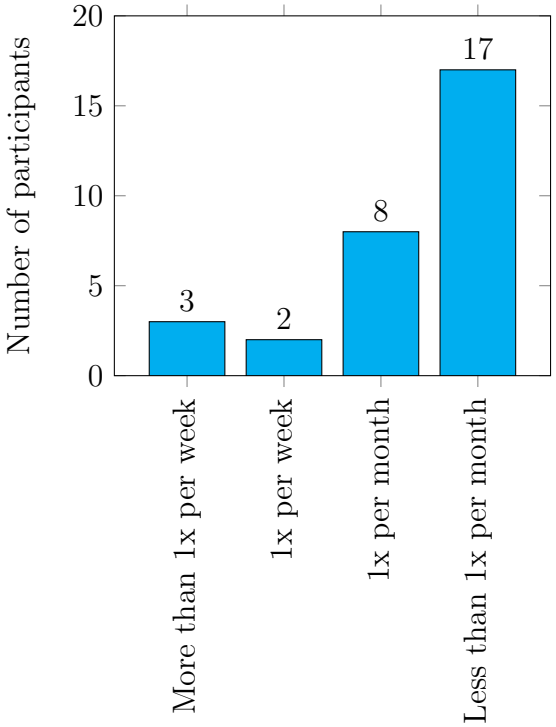


Figure A.7: Results of Q4: How often have you recorded audio in the past?

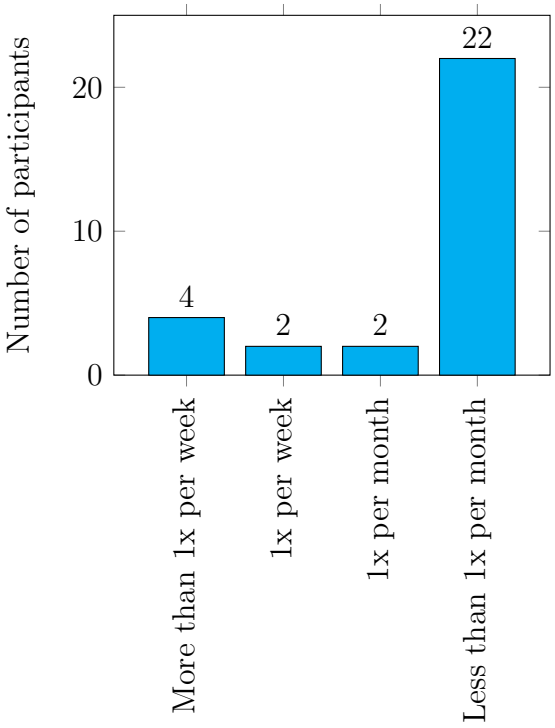


Figure A.8: Results of Q5: How often do you currently record audio?

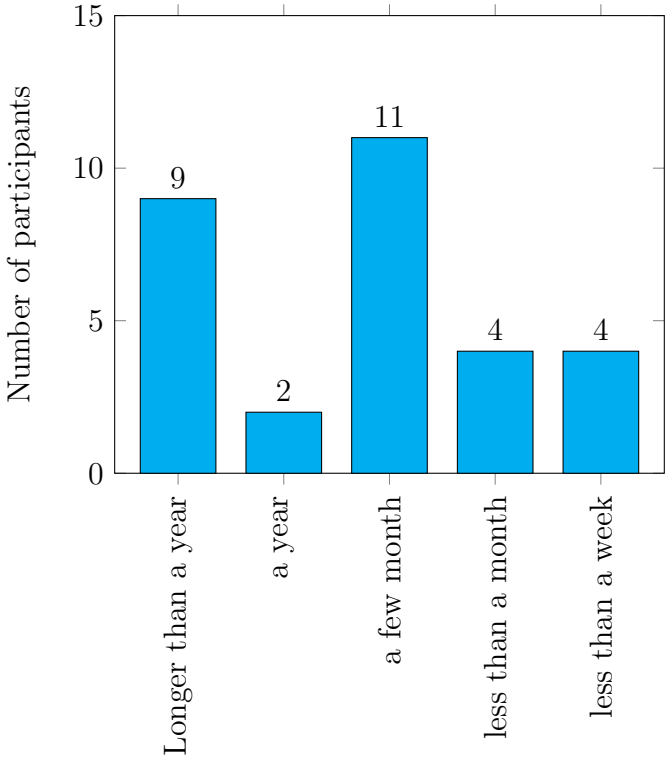


Figure A.9: Results of Q6: How long ago was your last audio recording?

A.8.2 SUS - System Usability Scale

System Usability Scale

	Stimme nicht zu				Stimme voll zu
	1	2	3	4	5
1. Ich denke ich würde dieses System häufig benutzen.					
2. Ich finde dieses System unnötig komplex.					
3. Ich denke das System war einfach zu benutzen.					
4. Ich denke ich würde die Hilfe einer technisch versierten Person benötigen um dieses System zu benutzen.					
5. Ich finde die verschiedenen Funktionen in diesem System waren gut integriert.					
6. Ich fand, dass in diesem System zu viel Inkonsistenz bestand.					
7. Ich könnte mir vorstellen, dass die meisten Leute die Bedienung dieses Systems sehr schnell erlernen.					
8. Ich fand das System war sehr umständlich zu bedienen.					
9. Ich habe mich sehr sicher bei der Bedienung des Systems gefühlt.					
10. Ich musste sehr viel lernen bevor ich das System benutzen konnte.					

Bibliography

[Act15a] Android ActionBar. http://developer.android.com/design/patterns/actionbar.html, 2015. Accessed: 2015-08-24.

[Act15b] Starting an Activity. http://developer.android.com/training/basics/activity-lifecycle/starting.html, 2015. Accessed: 2015-08-24.

[App15] Choosing a Membership. https://developer.apple.com/support/compare-memberships/, 2015. Accessed: 2015-08-24.

[BEHP+06] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy und M. B. Srivastava. Participatory sensing. *Center for Embedded Network Sensing*, 2006.

[Beny10] D. Benyon. *Designing interactive systems : a comprehensive guide to HCI and interaction design*. Addison Wesley, Harlow, England. 2. ed.. Auflage, 2010.

[BHSK+11] M. Böhmer, B. Hecht, J. Schöning, A. Krüger und G. Bauer. 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, S. 47–56.

[Che15] Chefkoch - Rezepte & Kochen. https://play.google.com/store/apps/details?id=de.pixelhouse&hl=en, 2015. Accessed: 2015-08-24.

[CSh15] Visual C#. https://msdn.microsoft.com/en-us/library/kx37x362.aspx, 2015. Accessed: 2015-08-24.

[DAKM+09] P. Dutta, P. M. Aoki, N. Kumar, A. Mainwaring, C. Myers, W. Willett, A. Woodruff und I. Design. Common Sense: Participatory Urban Sensing Using a Network of Handheld Air Quality Monitors, 2009.

[Ext15] Exit Points - Putting Down the Game - Extra Credits. https://www.youtube.com/watch?v=GqjkWec61gQ, April 2015. Accessed: 2015-08-24.

[Fac15] Facebook®. https://play.google.com/store/apps/details?id=com.facebook.katana&hl=en, 2015. Accessed: 2015-08-24.

[Foth08] M. Foth. *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City: The Practice and Promise of the Real-Time City*. IGI Global. 2008.

[Fra15] Fragments. http://developer.android.com/guide/components/fragments.html, 2015. Accessed: 2015-08-24.

[Fran] E. Frank. Class J48. http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/J48.html. Accessed: 2015-08-24.

[Gigg14] J. Giggs. A List of Alternative App Stores for Distributing your App or Mobile Game. http://www.mobyaffiliates.com/blog/a-list-of-alternative-app-stores-for-distributing-your-app-or-mobile-game/, Januar 2014. Accessed: 2015-08-24.

[Goo15] Prices, transaction fees, & currencies. https://support.google.com/googleplay/android-developer/topic/6075663?hl=en&ref_topic=3452890, 2015. Accessed: 2015-08-24.

[Hel15] Android Help. http://developer.android.com/design/patterns/help.html, 2015. Accessed: 2015-08-24.

[Inc.15a] A. Inc. Instant Heart Rate. https://play.google.com/store/apps/details?id=si.modula.android.instantheartrate, 2015. Accessed: 2015-08-24.

[Inc.15b] G. Inc. Google© Fit. https://play.google.com/store/apps/details?id=com.google.android.apps.fitness, 2015. Accessed: 2015-08-24.

[Inc.15c] G. Inc. Google© StreetView. http://www.google.com/maps/streetview/, 2015. Accessed: 2015-08-24.

[Inc.15d] M. Inc. Map My Run. https://play.google.com/store/apps/details?id=com.mapmyrun.android2, 2015. Accessed: 2015-08-24.

[Int15] Intents and Intent Filters. http://developer.android.com/guide/components/intents-filters.html, 2015. Accessed: 2015-08-24.

[Kell84] J. F. Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)* 2(1), 1984, S. 26–41.

[LEMM+08] N. D. Lane, S. B. Eisenman, M. Musolesi, E. Miluzzo und A. T. Campbell. Urban sensing systems: opportunistic or participatory? In *Proceedings of the 9th workshop on Mobile computing systems and applications*. ACM, 2008, S. 11–16.

[Map15] Google© Maps. https://play.google.com/store/apps/details?id=com.google.android.apps.maps&hl=en, 2015. Accessed: 2015-08-24.

[Mic15] Layered Application Guidelines. https://msdn.microsoft.com/en-us/library/ee658109.aspx, 2015. Accessed: 2015-08-24.

[MIT15] MIT SENSEable city lab. http://senseable.mit.edu/, 2015. Accessed: 2015-08-24.

[Nav15] Navigation with Back and Up. http://developer.android.com/design/patterns/navigation.html, 2015. Accessed: 2015-08-24.

[Oluw] H. S. Oluwatosin. Client-server model. *IOSR Journal of Computer Engineering (IOSR-JCE)* 16(1), S. 67.

[Pla15] Google© Play Store. https://play.google.com/store, 2015. Accessed: 2015-08-24.

[Rade03] K. Radeck. C# and Java: Comparing Programming Languages. https://msdn.microsoft.com/en-us/library/ms836794.aspx, Oktober 2003. Accessed: 2015-08-24.

[Rif15] Riff Wave Format. https://en.wikipedia.org/wiki/WAV, 2015. Accessed: 2015-08-24.

[RiRu08] L. Richardson und S. Ruby. *RESTful web services.* " O'Reilly Media, Inc.". 2008.

[RLMM09] R. Rogers, J. Lombardo, Z. Mednieks und B. Meike. *Android application development: Programming with the Google SDK.* O'Reilly Media, Inc. 2009.

[Saur11] J. Sauro. Measuring Usability with the System Usability Scale (SUS). http://www.measuringu.com/sus.php, Februar 2011. Accessed: 2015-08-24.

[Ser15] Services. http://developer.android.com/guide/components/services.html, 2015. Accessed: 2015-08-24.

[Sha15] Shazam. http://www.shazam.com/, 2015. Accessed: 2015-08-24.

[Stat13] Statista. Anzahl der Smartphone-Nutzer in Deutschland nach genutztem Betriebssystem im September 2013 (in Millionen). http://de.statista.com/statistik/daten/studie/176811/umfrage/verbreitung-mobiler-endgeraete-nach-betriebssystem-in-deutschland/, 2013. Accessed: 2015-08-24.

[Stat15] Statista. Number of smartphones sold to end users worldwide from 2007 to 2014 (in million units). http://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/, 2015. Accessed: 2015-08-24.

[Vis15] Visual Studio®. https://www.visualstudio.com/, 2015. Accessed: 2015-08-24.

[VSBSo94] M. W. Van Someren, Y. F. Barnard, J. A. Sandberg und andere. *The think aloud method: A practical guide to modelling cognitive processes*, Band 2. Academic Press London. 1994.

[Wang+03] A. Wang und andere. An Industrial Strength Audio Search Algorithm. In *ISMIR*, 2003, S. 7–13.

[Wha15] WhatsApp Messenger. https://play.google.com/store/apps/details?id=com.whatsapp&hl=en, 2015. Accessed: 2015-08-24.

[Wik15a] Software framework. https://en.wikipedia.org/wiki/Software_framework, 2015. Accessed: 2015-08-24.

[Wik15b] Comparison of C Sharp and Java. https://en.wikipedia.org/wiki/Comparison_of_C_Sharp_and_Java, 2015. Accessed: 2015-08-24.

[Wik15c] List of mobile software distribution platforms. https://en.wikipedia.org/wiki/List_of_mobile_software_distribution_platforms, 2015. Accessed: 2015-08-24.

[Worl15] K. Worldpanel. Smartphone OS sales market share. http://www.kantarworldpanel.com/global/smartphone-os-market-share/, Juni 2015. Accessed: 2015-08-24.

[Xam15] Xamarin. http://xamarin.com/, 2015. Accessed: 2015-08-24.