Journal of Ubiquitous Systems & Pervasive Networks
Volume 17, No. 1 (2022) pp. 27-34

The way it made me feel – Creating and evaluating an in-app feedback tool for mobile apps

Simon André Scherr*[a], Mher Ter-Tovmasyan[a], Frauke Neugebauer[a], Steffen Hupp[a], Frank Elberzhager[a]

[a]*Fraunhofer IESE, Kaiserslautern, Germany, 67663*

Abstract

Mobile apps are becoming increasingly important in everyone's daily life. The success of an app is linked to high user acceptance. Therefore, it is necessary to capture users' expectations, needs, and problems regarding an app in any situation. By continuously capturing and analyzing user feedback, developers can evaluate the level of user acceptance. There are various feedback channels, such as app stores, social networks, and within the app, which can be used to capture user feedback. As we already have experience with feedback from app stores and social networks, we wanted to investigate in-app feedback approaches and thus conducted a mapping study to understand the state of the art of these approaches. We analyzed 36 publications and derived requirements for in-app feedback tools. Based on that, we defined requirements for an in-app feedback tool to describe its prototypical realization. Then we performed an evaluation regarding user acceptance of our tool with 33 participants. The evaluation showed a high rate of acceptance for the tool among the participants. The results also highlighted improvement areas for our tool, such as optimizing the rate of requests for feedback. We plan to address these aspects in future work and to continue improving our tool.

Keywords: *Mobile Apps; Software Quality; User Feedback; User Experience; In-App Feedback*

1. Introduction

The importance of mobile apps in our life is increasing every day. There are more than seven million apps available in Google Play and in the Apple App Store combined [1]. Users must be satisfied with an app to use it in the long term. An unintuitive user interface, slow performance, crashes, missing features, as well as other factors are often the reason for an app being uninstalled [2]. A high level of user acceptance is an appropriate means to avoid unsuccessful apps. Therefore, it is relevant to continuously capture and understand the expectations, needs, and problems of the users, which are likely to change often, and to address them. One way to do this is to collect and analyze user feedback during the usage of an app [3] [4]. This analysis can be used to identify missing features and problem areas of an app. However, for apps with many users, using traditional approaches to get insights from users (e.g., interviews) can be a cumbersome process [3]. An alternative is to collect feedback, e.g., from app stores, software product forums, and social networks, or to integrate feedback mechanisms into the software app itself [4].
In contrast to feedback from app stores and social networks, in-app feedback, which is provided during the use of an app, can lead to more detailed feedback about an encountered problem as well as about the app itself [5]. The integration of an in-app feedback tool allows combining the opinion of the user with additional information about the app as well as the device [6] and can be acquired in a structured manner, making "it easier to aggregate, process, analyze, and evaluate" [7]. This makes in-app feedback a valuable source for software improvement.
Since 2016, we have been working on User Echo Service (UES) our crowdsourcing tool as a feedback acquisition and analysis solution. Crowdsourcing has been used successfully in past studies for services, e.g. Rescuer [8]. This is why we want to apply the crowd idea to quality assurance as well and help developers understand feedback trends in their products without spending much effort. This means user feedback should be understood as a continuous stream of data, as users are constantly changing their feedback and developers are steadily changing their products while capturing all available feedback data [9]. Currently, UES allows adding single feedback entries [4], but so far, an API for integrating in-app feedback in an automated way has been missing.
This work focuses on the following research questions:

RQ1. What are concrete requirements for state-of-the-art existing in-app feedback tools?
RQ2. How are such feedback tools accepted by end users?

The structure of this article is as follows: Section 2 provides some background on our project and on feedback-driven development. In Section 3, we describe our systematic mapping

[*] Corresponding author. Tel.: +49 631 6800- 2294
E-mail: simon.scherr@iese.fraunhofer.de

DOI: 10.5383/JUSPN.17.01.004

study (SMS) on in-app feedback. Section 4 presents the requirements of an in-app feedback tool and a prototypical realization, whose evaluation is described in Section 5. Section 6 discusses our results. Section 7 describes the threats to validity. Section 8 concludes with a summary and future work.

2. Foundations

2.1. Project Background

The need for an in-app feedback tool emerged in the research project "EnStadt:Pfaff". The goal of this project is to develop a climate-neutral sustainable smart city district [10]. Its core will be a platform with various services [11]. To create services that fulfill the needs of the users, it is essential to center the solution around the users. Besides following concepts that enable user-centered design, we perform workshops and design sprints to get the right starting point. This also includes having a prototyping platform for smart city service development. Additionally, permanent validation and verification of the implemented requirements must be done. Therefore, we decided to include the collection of user feedback within these services to permanently improve them. These decisions not only enable the creation of an urban district that supports its residents in living sustainably in the sense of climate neutrality, but also allows the construction of sustainable software services that can be permanently adapted to the needs of the users.

2.2. Capturing Feedback to Improve the Quality of Apps

In the context of crowd requirements engineering (CrowdRE) [12], researchers have started to investigate the potential benefits of performing feedback analysis to enable validating, verifying, or identifying requirements for a product and identifying potential bugs. Stade et al. [13] presented the FAME framework for collecting feedback and monitoring data to support requirements elicitation. Scherr et al. [9] introduced a common data model for textual feedback from multiple data sources and an architecture for feedback collection. In this work, they considered several feedback data sources, e.g., Apple App Store, Google Play, Facebook, and Twitter. Aljannan et al. [6] discusses the shortcomings of online feedback channels and proposes the collection of feedback from the software in use.
We plan to consider the app itself as a feedback source. Therefore, we collected data regarding in-app feedback to specify requirements for an in-app feedback tool. Such a tool will allow us to collect feedback during the use of an app.

3. Systematic Mapping Study (SMS)

3.1. Methodology

To answer RQ1, we conducted an SMS to gain insights into the state of the art of in-app feedback and in-app feedback collection approaches. Following Petersen et al. [14], the process we applied consisted of five steps.
In the **planning** phase, we defined the study questions, selected digital libraries, created a query term, and defined exclusion and inclusion criteria. In the **conducting** phase, we searched these libraries for relevant studies. In the **selection** phase, we screened the studies we had found and applied exclusion and inclusion criteria. In the next phase, we **extracted** relevant data from the selected studies. Finally, in the **reporting** phase, we answered our study questions with the data extracted from the selected studies. To ensure reproducibility of this study and avoid potential bias, all the steps we took and the decisions we made in each phase of the SMS will be described in the following.

3.1.1. Planning the Systematic Mapping Study

To define the scope of this SMS, we derived three study questions:

MSQ1. Which data is collected with in-app feedback?
With the first study question, we aimed at eliciting information about the possible content of in-app feedback and the data types that can be collected during the use of an app.

MSQ2. How to collect in-app feedback during the use of an app?
The second study question aimed to highlight the recommendations to follow when collecting in-app feedback during the use of an app, in particular the necessary steps and guidelines that need to be followed.

MSQ3. What are the objectives of collecting in-app feedback?
The third study question was about investigating the underlying intentions and objectives of collecting in-app feedback.

The digital libraries we selected to search for relevant publications were Scopus, IEEE Xplore, ACM Digital Library, Base, Google Scholar, MS Academic, and SpringerLink. Last, we derived a query term from the study questions. The query was prototyped and revised multiple times. The query term[1] used for each library was then adapted according to the guidelines and limitations of the respective search engine. In the end, we came up with the following query term:

```
(software OR application OR app OR "mobile app")
AND
("in app feedback" OR "in-app feedback" OR "user feedback" OR
"user input" OR "feedback approach" OR "feedback tool" OR
"feedback method" OR "feedback form" OR "feedback collection" OR
"feedback input" OR "feedback channel" OR "feedback integration")
AND
("requirements gathering" OR "requirements elicitation" OR "quality
assurance" OR evolution OR maintenance)
```

3.1.2. Performing the Systematic Mapping Study

For this systematic mapping study, the query term was used to check the keyword, title, and abstract fields. We conducted the initial search in November 2020. After this search, we assessed the results to improve the query term further, iteratively updating the query to increase the quality. The final search was conducted in December 2020. An overview of the results of the final search per source can be seen in Table 1.

Table 1. Digital libraries, fields, and number of results

Digital Library	Fields of the Digital Library	Results
Scopus	Title, Abstract, Keywords	441
IEEE Xplore	All Meta Data	157
ACM Digital Library	Title, Abstract, Author Keyword	92
BASE	Title, Subject Headings	215
Google Scholar	Title	80
MS Academic	All	54
SpringerLink	All	348

[1] The decisions made about the definition of the query term and the changes made for each digital library are available for download from https://figshare.com/s/a2d72ee8ae62af4c2cd7 and are not listed here.

	Total Number 1,387

As a result of the search phase, a total of 1,387 publications were considered for the selection phase. These were selected based on the inclusion criteria (Table 2). For a publication to be selected for our mapping study, it had to meet the criteria ToP, L, PT, A, and iaF and, additionally, at least one of the following: iaF.ID or iaF.A. During this selection process, we checked each publication against the negations of the defined inclusion criteria, e.g. n.ToP for publications prior to the year 2000.

Table 2. Inclusion criteria for study selection

Acronym	Criterion	Definition
ToP	Time of Publication	The publication was published in or after the year 2000.
L	Language	The publication is in English or German.
PT	Publication Type	The type is Article, Paper, Conference Proceeding and Paper, Journal Paper and Article, Book Chapter.
A	Access	The full text of the publication is accessible.
iaF	In-App Feedback	The publication discusses user feedback (besides bug reporting or usage-mining feedback) that can be collected directly from within the app.
iaF.ID	In-App Feedback Input/Data	The publication provides an overview of input/data for in-app feedback, apart from bug reporting or usage-mining feedback.
iaF.A	In-App Feedback Approach	The publication presents an overview of in-app feedback collection approaches.

The selection was carried out in multiple phases (Fig. 1). In the first phase, all publications were checked for duplicates, which were removed. After excluding the duplicates, the criteria n.ToP, n.L, and n.PT were checked in the second phase. Publications meeting these criteria were excluded. In the following phases, the title, the abstract, the introduction, the conclusion, and then the entire text were checked successively and excluded if one of the criteria n.A, and n.iaF was met.

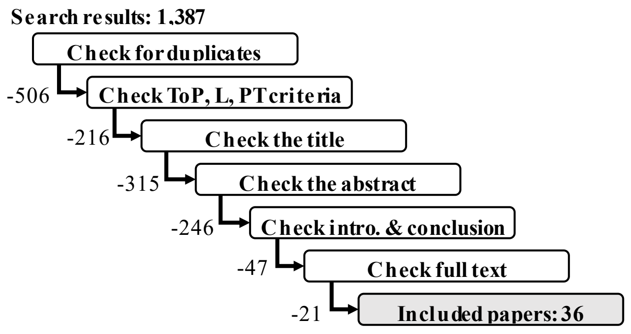


Fig. 1. Reading strategy for the selection and number of studies excluded per phase

For the extraction phase, we created an extraction form to gather all relevant data from the identified articles. This mainly included data about the content of any in-app feedback and what to use for this content (MSQ1), steps and recommendations for collecting in-app feedback (MSQ2), and the objectives of collecting in-app feedback (MSQ3). Overall, the extraction form comprised eight fields to be completed for each article.
During the extraction, we marked each of the selected publications with a unique ID and collected the extracted data into a single datasheet [2]. We also included the list of all publications as well as the selection decisions.

3.2. Results

3.2.1. Which data is collected with in-app feedback?

Overall, 32 of the included publications provide details on how the content of in-app feedback is represented. The selected publications address a variety of topics that can make up the content of in-app feedback. Typically, this content often includes problem descriptions, feature requests, and an overall opinion about an app. Table 3 lists the data used to represent this content of in-app feedback. In the case of images, different scenarios are described, such as user-selected images from the device gallery or screenshots created automatically by the app during feedback initiation. Furthermore, it is mentioned that these images can be annotated or sketched during the feedback process to add additional information, e.g., indicating UI flaws [15] [16]. Note the feedback type "Emotions and Behavior" with six mentions. Users' emotions are, for example, captured with the help of emojis, while the front camera can be used to capture behavior, e.g., a user's eye movements.

Table 3. Data used for the content of in-app feedback (paper IDs as defined in the datasheet)

Data Type	#	Paper ID
Text	25	P01-P06 P09 P11 P13-P17 P20-P22 P25-P29 P32 P33 P35 P36
Image	15	P01 P03-P06 P09 P13 P16-P17 P19 P21 P25 P28 P29 P33
Audio	9	P03 P04 P07 P09 P13 P16 P25 P28 P29
Rating	8	P01 P03 P05 P09 P16 P32 P33 P36
Emotions & Behavior	6	P01 P10 P16 P25 P26 P30
Video	1	P33
Attachments	1	P16

Furthermore, 17 publications point out that in-app feedback should be enriched with log data such as monitoring data, stack traces, device information, as well as with information about the user, user-assigned feedback priority, the view where the feedback emerged, and much more. This complementary information can provide a better understanding of the user's situation and consequently improve the overall analysis of the provided feedback [17] [18]. Furthermore, there are suggestions for using different feedback data together when collecting in-app feedback. For example, app users can be asked to fill in a questionnaire where the answers to the question can be given with different types of data, such as text, ratings, and yes/no selections [6]. Such questionnaires can also be extended to include a category selection for the feedback and the possibility to attach different file types [13].

3.2.2. How to collect in-app feedback during the use of an app?

30 of the 36 publications discuss recommendations for in-app feedback collection, including information about when and how in-app feedback can be collected and what to consider during the collection process. Regarding the timing, i.e., when to start in-app feedback collection, two options are discussed in the publications (Table 4). The first one is to trigger the collection automatically when predefined conditions of an app are met. Such triggers include reaching a specific goal, restarting a

[2] The results of the selection and extraction are available for download from https://figshare.com/s/6c453b4a46f97e788d2e and are not listed here.

crashed app, recognizing a particular emotion, using a certain feature, and others. The second option is for the initiation process to be straightforward, with only a few steps.

Table 4. Initiation of in-app feedback collection (paper IDs as defined in the datasheet)

Initiation	#	Paper ID
At predefined conditions (Pull)	16	P05 P07 P10 P14 P16 P17 P19 P21 P25-P27 P30-P33 P35
At user request (Push)	9	P05 P06 P16 P19 P20 P22 P25 P31 P32
Both	6	P05 P16 P19 P25 P31 P32

Apart from the question of the best time for collecting in-app feedback, it is also important to know how to collect it and what to consider in the process. In total, 30 publications discuss various aspects in this regard, which are listed in Table 5. It is worth mentioning that different data can be used to enrich feedback. To motivate users to give feedback, different recommendations were found. Commonly suggested is the application of various digital motivation techniques, e.g., gamification techniques. In addition, it is mentioned that the way feedback is requested, as well as the simplicity and usability of the feedback process, are important to motivate users to provide more feedback [19]. Moreover, a user's motivation is strongly influenced by system improvements based on their feedback [12] [19] [20].

Table 5. Recommendations regarding in-app feedback collection (paper IDs as defined in the datasheet)

Recommendations regarding Collection	#	Paper ID
Enrich in-app feedback	17	P02 P04-P06 P10 P11 P14 P16 P17 P20 P21 P27 P30 P32 P34-P36
Motivate users	9	P01 P08 P09 P18 P19 P28 P29 P31 P32
Enable category selection	8	P03 P06 P11 P13 P16 P19 P20 P29
Respect user's privacy	7	P04 P05 P20 P29 P30 P31 P32
Defined feedback input	6	P03 P04 P05 P14 P25 P33
Provide information about the use of feedback	5	P09 P28 P29 P32 P35
Allow user to decide about data to be sent	5	P04 P05 P19 P20 P31
Enable alternative documentation type	4	P04 P13 P16 P31
Enable communication after sending feedback	4	P04 P09 P13 P25
Show available feedback for a given context	3	P06 P19 P32
Allow users to opt out	3	P07 P31 P32
Provide explanatory information upon request	3	P03 P07 P32
Prevent user interruptions	3	P06 P19 P27

3.2.3. What are the objectives of collecting in-app feedback?

All publications discuss objectives for collecting in-app feedback. Table 6 lists the mentioned objectives and shows their distribution across the publications. It can be observed that the discussed objectives for collecting in-app feedback hardly differ from those for collecting feedback from other sources. One noteworthy objective goes one step further than in-app feedback collection, yet it assumes such a collection: the possibility of enabling communication between users and developers within an app. This can be started, e.g., after in-app feedback has been given by a user. Such communication can have a positive impact on user satisfaction and motivate users to provide more feedback, which turns out to be beneficial for developers [6].

Table 6. Objectives for collecting and integrating/combining in-app feedback (paper IDs as defined in the datasheet)

Objective	#	Paper ID
Prioritization and decision-making for future development	25	P01-P04 P06 P08-P16 P18-P21 P23 P24 P26 P27 P30 P33 P34
Identify new requirements	22	P01-P05 P08-P11 P13 P14 P16 P18 P20 P21 P26 P28-P31 P33 P35
Improve the acceptance of the app	15	P02 P04-P09 P13 P14 P20 P21 P23 P25 P27 P34
Detect problem areas in the app	14	P02 P05 P07 P11 P12 P17 P21 P24 P26 P28-P30 P33 P36
Enable feedback with additional context	12	P04-P07 P10 P20 P21 P24 P27 P28 P34 P35
Improve feedback analysis through combination of different channels	9	P05 P09 P10 P14 P15 P18 P27 P28 P35
Capture overall user satisfaction	9	P02 P05 P06 P08 P16 P17 P20 P21 P27
Capture app-related trends/topics	4	P02 P11 P21 P22
Enable in-app communication between users and developers	4	P04 P14 P16 P20
Create feedback traceability	2	P06 P14
Increase QA effectiveness	2	P02 P12

3.3. Threats to Validity

In our SMS, we identified and mitigated several possible threats to validity. To capture the relevant sources, we iteratively prototyped our search term to keep the result set relatively broad, and included several cross-publisher databases. During the selection, we made sure that in the title step, we only removed results that obviously did not match our search. Another identified threat is that errors might occur while searching, e.g., by entering a malformed search term, using the wrong field codes, or not copying all the results. Therefore, we checked the search results with our trial searches to ensure that no wrong term was entered.
As the extraction phase is a very important phase of an SMS, we performed quality assurance on the extraction form. In addition, we focused on copying citations from our results into our forms. This should prevent the addition of personal interpretation or bias regarding the primary research.

3.4. Summary

The results of our SMS revealed that in-app feedback should consist of linguistic and nonlinguistic data, such as textual feedback, ratings, media files, and more. For the initiation of the feedback collection process, three different options were identified: push, pull, and a combination of both. Our findings also yielded multiple recommendations on what to respect during in-app feedback collection. Finally, the answers revealed that multiple objectives exist for collecting feedback during the use of an app. Yet, these do not differ from the objective of incorporating feedback from other channels. Overall, this mapping study revealed valuable findings that can be used as a baseline for creating an in-app feedback tool.

4. Building an In-App Feedback Tool

4.1. Requirements

With the insights gained from the SMS, we derived requirements for our in-app feedback tool. The initiation of the in-app feedback collection can be triggered by the app user (push feedback) or by the app itself (pull feedback). Using both

initiation types within an app will allow reaching all app users. We decided to use all data types found in the SMS (Table 3) for expressing feedback. An app user should be able to express textual feedback as well as give a rating, while also being able to append attachments, e.g., images, audios, and videos, to their feedback. Furthermore, it should be possible to use multiple data types in combination and to include app log data as well as specific feedback categories to enhance given feedback so that more information can be derived from it during the analysis. Finally, it is important for the app user to be aware of the data that is collected along with the feedback and to know how the included data and the feedback will be used.
With these insights, we specified the requirements for an in-app feedback tool by facilitating user stories[3]. We used two roles in describing the user stories: Product Owner, who is responsible for an app, and App User, who will use an app that includes an in-app feedback tool. To get a better overview, we grouped related user stories together and categorized them into three different groups. *Initiation of Collection* (Table 7) defines the options for starting the feedback collection process; *Collection Enhancements* (Table 8) specifies the data to be collected as in-app feedback; while *Transparency* (Table 9) points out that it is important to inform the app user about which data is being collected and how the data will be used. Below, the user stories for each category are presented.

Table 7. Initiation of collection

User Story	Description
User initiated feedback	As an App User, I want to be able to provide feedback at any time and stay in the current context so that I can express myself without leaving the application.
Requesting feedback for defined conditions	As a Product Owner, I want to be able to request feedback in a certain usage scenario and/or app state so that I can collect targeted feedback for the usage scenario/state.

Table 8. Collection enhancements

User Story	Description
Multiple data types for feedback expression	As a Product Owner, I want to be able to collect feedback using different data types as well as multiple data of the same type so that I can provide multimodal feedback collection and choose the appropriate type depending on the use case to collect more information with one feedback.
Feedback with attachments	As an App User, I want to be able to attach a file to my feedback so that I can better visualize my feedback.
Feedback enriched with additional information	As a Product Owner, I want to be able to gain more information from the App User regarding their feedback and further understand the conditions under which the in-app feedback occurred so that I can better understand the context of the given feedback and the feedback itself.
Category selection	As a Product Owner, I want to get information about a feedback category so that I can easily find feedback for a specific category.

Table 9. Transparency

User Story	Description
Transparency about sent data	As an App User, I want to know which data is being sent along with my feedback so that I can be sure that no personal data is collected without my knowledge.
Transparency about data usage	As an App User, I want to know how my data and feedback are being used so that I can be sure that my data will not be used for malicious purposes.

[3] The acceptance criteria of the defined user stories are available for download from https://figshare.com/s/9b18e46b8e68c43454 22 and are not listed in this paper.

4.2. Prototype Realization

For the prototype, we first developed an in-app feedback data model. As we intend to use the tool in UES, the model is based on the feedback data model defined for UES as presented in [9]. According to our model, in-app feedback can include multiple ratings, textual feedback, and attachments. We also include the name of the app to identify from which app the feedback was sent. Also, a feedback reference is included to understand the source of the feedback within the app and to provide further information about what the feedback refers to, e.g., a specific screen or feature of the app. The data model also includes the type of the initiation (push or pull) as well as a selected category that allows categorizing the feedback before sending it.
Next, we created the prototype consisting of a frontend and a backend to receive, process, and store the feedback. The frontend is implemented as a library that can be easily integrated and adapted into apps. We implemented different views and pages to fulfill the user stories, which can be seen integrated into an existing app in Fig. 2. For push feedback, we used a full page, which can combine different implemented views (e.g., views for title, rating). For pull feedback, we implemented a popup page. This page is used to request a quick rating feedback from an app user after a specified user interaction. The quick rating view can use different kinds of ratings with predefined views, e.g., likes and dislikes, five-star ratings, emotions, and reactions. After selecting a rating value, the app user is further provided with an option to give more detailed feedback using the full page (same as for push feedback) and, e.g., include attachments, if desired. To reduce the number of feedback requests and prevent annoying the users, we implemented an increasing delay for requesting feedback after certain user interactions.

5. Evaluation

5.1. Evaluation Design

For the evaluation of the tool, we integrated its frontend into the app PfaffFunk. PfaffFunk provides a communication service for citizens, communicating news to them and informing them about the smart city district. We included options for manual initiation of the feedback process from different pages of the app as well as automatic initiation after the use of various features, like commenting on a post. The goal was to measure the user acceptance of the created in-app feedback tool.
33 participants, who were mostly work colleagues or acquaintances, agreed to test the app and provide feedback within this app over a period of one week. Before the participants started to test the app, they had to answer a questionnaire containing some socio-demographic questions as well as questions about their experience and expectations regarding in-app feedback and feedback in general. After the testing phase, the participants had to answer a second questionnaire aimed at evaluating the acceptance of the current implementation of the in-app feedback concept.

5.2. Integrating our Tool into PfaffFunk

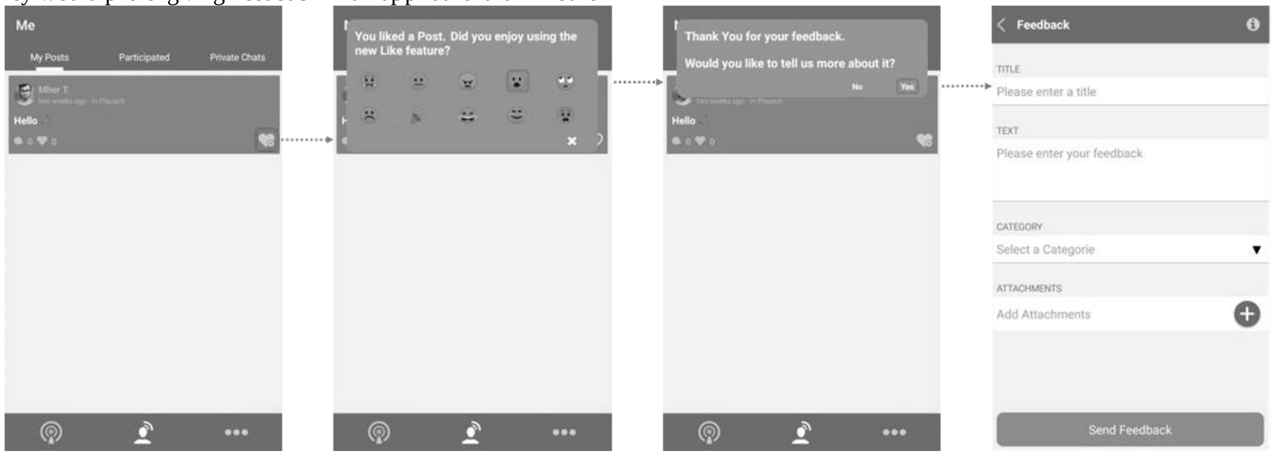
To integrate the frontend of the tool, we customized it to match the styles used in PfaffFunk. On several pages, we added options for manual feedback initiation. For the pull feedback quick rating, we used a rating based on emotion. To represent the emotions of an app user, a base set of emojis was used. We also added an option to disable/enable automatic feedback requests. Overall, PfaffFunk requested feedback from app users after thirteen different interactions involving various features (Fig. 2). These feedback requests were initiated after a built-in delay counter reached a predefined delay threshold. These thresholds were then increased after each request. For the follow-up evaluation, we selected low initial thresholds so that the participants could experience the feedback request scenarios.

5.3. Structure of the Questionnaires

Both questionnaires [4] were created in German and every participant got a token to match the first and second questionnaires of the participants anonymously. Both surveys together comprised a total of eight question groups.
The first questionnaire started with general socio-demographic information, including a question about the participant's occupation. Afterwards, questions about the participant's general experience with apps and the app PfaffFunk were asked, followed by questions about their current feedback behavior, to check whether their prior feedback behavior might influence their willingness to provide built-in feedback. The last question group of the first questionnaire asked about the participant's motivation to provide feedback during the use of an application.
The second questionnaire contained not only closed questions, but also free text questions, which are disregarded in this work. The survey started with questions about the participants' overall satisfaction and opinion regarding the current realization and continued with their concerns about in-app feedback. Such concerns might influence giving feedback directly in the app in the long run and even lead to abandoning the built-in feedback options. The third question group was aimed at getting feedback for the current realization in order to find improvement areas for an improved tool in the future. At the end of the second questionnaire, the participants were given the opportunity to name general problems they experienced while using the app and to add comments.

5.4. Results

In the first survey, 28 participants (strongly) agreed that giving feedback directly in an app would be easier, and 27 stated that they would prefer giving feedback in an app rather than in other feedback channels. Furthermore, two thirds of the participants (21) suggested that the opportunity to give feedback in an app would increase their willingness to give feedback.
Table 10 compares the questions from the first questionnaire with the questions from the second questionnaire to spot how the attitude towards our feedback tool and in-app feedback in general changed. The answers from questionnaire 1 show that in general, the participants stated that they are good at handling apps (30 participants) and getting familiar fast with new apps (31) and their functions (29). In questionnaire 2, all but one participant stated that giving feedback in the app is easy, while only 27 of the 33 participants answered that the handling of the feedback options is easy. This comparison shows that in general, giving feedback in the app is easy, even for people who at first stated that they were not that familiar with apps.
A question aimed at finding out how satisfied the participants were with the idea of in-app feedback was asked in both surveys. After testing the in-app feedback tool in our app for one week, all participants agreed that in-app feedback is a good idea. In contrast, before the testing phase, only 28 participants agreed with this. The interest in using in-app feedback tools in the future grew by more than 15%, and more than 75% (N=27) of our participants stated that they would use a similar implementation in the future. In addition to that, 27 participants said that providing feedback did not require too much time for them.
In questionnaire 2, the participants had the option of providing a free text answer regarding what they liked about the current implementation. Some answered that they liked the selection of emojis or that the interaction with emojis was easy and made providing feedback more fun, while other participants stated that providing feedback within the app was very easy and fast. Nevertheless, the participants were also asked to name possible changes to the current realization of requesting feedback. While some of the participants felt that there were too many emojis, others believed that more emojis should be available to choose from. As we also observed people reporting that the selection of emojis was well suited for them, we should investigate this topic further in future research.
To classify the sample, we analyzed the socio-demographic data. 14 participants (42.4%) were female, and 19 participants (57.6%) were male. While most participants (N=23; 69.7%) stated their age as between 25 and 34 years, five people marked their age as between 15 and 24 years (15.2%), three between 35 and 44 years (9.1%), and two between 45 and 54 years (6.1%).



You liked a Post. Did you enjoy using the new Like feature?

Thank You for your feedback.

Would you like to tell us more about it?

Fig. 2.PfaffFunk integration

[4] The English translation of the questionnaires is available for download from https://figshare.com/s/2fbc1bf2d061f75b8514 and is not included in this paper.

Table 10. Evaluation of questionnaire 1 and questionnaire 2

Questionnaire 1			Questionnaire 2		
Variable	%	SD	Variable	%	SD
Good at handling apps	90.9	0.3	Giving in-app feedback is easy	97.0	0.2
Getting familiar fast with new apps	93.9	0.2	Handling of feedback options is easy	81.8	0.4
Getting familiar fast with functions of new apps	87.9	0.3			
In-app feedback is a good idea	84.8	0.4	In-app feedback is a good idea	100.0	0.0
Interested in using in-app feedback in the future	66.7	0.5	Would use feedback options in the future	81.8	0.4
Not sure about giving in-app feedback	12.1	0.3	Would use a similar implementation in the future	78.8	0.4

Notes: % = percentages; SD = standard deviation

6. Discussion

The findings of the mapping study provide a good overview of what feedback data should be collected and what collection mechanisms are commonly used. This information is a good basis for determining basic requirements for an in-app feedback tool (RQ1).
Concerning the acceptance of such a tool (RQ2), we prototyped the defined requirements and integrated the frontend of the tool into the PfaffFunk app. We conducted a survey to analyze the participants' attitudes towards in-app feedback before testing the tool within PfaffFunk. A second survey was conducted after the testing phase to check the acceptance of the tool after the participants had been using it for one week.
The results of the evaluation indicate that after the evaluation, all participants were convinced that in-app feedback is a good idea, which is an increase of about 15%. Similarly, around 15% more participants got convinced to use in-app feedback in general in the future, while nearly 80% stated that they would use implementations of an in-app feedback tool similar to the one in PfaffFunk. This implies that the current realization generally had a high level of acceptance among the participants of this evaluation. This statement can be supported by the high percentage of participants who said that the handling of the feedback options is easy, and by the even higher percentage of people who stated that giving in-app feedback is easy. However, as only 33 participants tested the tool and not all shared these views, we assume that our in-app feedback tool needs to be tested in more detail and should be improved in the future.
Even before the testing phase, the acceptance regarding in-app feedback was very high. After testing the realization of our in-app feedback tool for one week, the participants reported an even higher rate of acceptance (see Table 10, row 5) regarding feedback tools. One of the benefits of in-app feedback is that users can give their feedback while using the app in parallel, so they do not have to switch to another app or site to provide their feedback. Because of that, barriers to giving feedback can be minimized and the frequency of feedback will grow.
The high acceptance of the feedback request and the free text answers given in the second questionnaire also indicate that using emojis for this purpose is a good idea. Obviously, the participants understood that they should use emojis to express their emotions in the feedback. Which emojis are suitable for expressing feedback is subject to further research.
The second questionnaire highlighted improvement areas for the specified requirements. Feedback requests must be defined independently and not as a subset of pull feedback. The type of rating used in the feedback request is also an important aspect that must be specified separately.
To obtain a better picture of the acceptance of the realization, another study with a larger and more diverse group of participants needs to be conducted over a longer period to assess acceptance.
With this in-app feedback tool embedded into our UES framework, we can continuously collect user feedback regarding specific functionality. This eases the overall feedback analysis process, as we already know to which feature and component of the observed app the data belongs. We can also expect this to have a positive influence on product quality, as developers can get much more fine-grained feedback than if they must rely solely on app store feedback. Another possibility for the use of the in-app feedback tool would be to conduct A/B testing of new features and alternative implementations or beta-functionality, allowing the developers and product managers to shape the app more to the users' liking.
Together with our previously presented app store crawling infrastructure, developers can get a more complete picture regarding the user acceptance of their app and its features. In combination, we can gather a great amount of feedback data on which our analysis infrastructure can work and gather more insights that would not be found when using one source alone. This extension of UES allows developers to better monitor the perceived quality of their app, gather new features for it, and derive subsequent development steps more easily.

7. Threats to Validity

In this section, we list potential issues threatening the validity of our evaluation results.
Some of the participants were familiar with PfaffFunk before the evaluation, which might have compromised the conclusions drawn from the data. We mitigated this threat to internal validity by selecting a mixed set of participants in terms of their familiarity with PfaffFunk and added this as a question to the questionnaire.
At the beginning of the evaluation of the frontend of the in-app feedback tool in PfaffFunk, we asked for feedback on some of the more frequent use cases, like closing an image view, too often. We minimized this threat to internal validity by informing the participants prior to the evaluation that feedback would be requested more often than would normally done in a real app. Furthermore, we updated the app during the evaluation on the fourth day to limit the rate of feedback requests.
As we picked the participants ourselves and only had a relatively small number of testers (n=33), external validity is currently limited. To address this threat to external validity, an evaluation with a larger population should be conducted to improve the significance of the results.

8. Conclusion and Future Work

As collecting in-app feedback is a valuable method for understanding the user and improving software, we identified the need to extend User Echo Service to support such kind of feedback. We conducted a systematic mapping study to get an overview of the state of the art of in-app feedback collection and integration approaches. Following our findings, we defined eight user stories for an in-app feedback tool that can be used to collect feedback during the use of an app (RQ1). Based on these requirements, we designed and developed a feedback tool. The tool includes front- and backend components and was integrated into UES. To address RQ2, we conducted an evaluation of the realized tool and presented the results in this article. We integrated the frontend into our PfaffFunk app. We performed our evaluation with 33 participants over the period of one week. The results of the evaluation showed that most participants were

satisfied with the current realization. Their interest in using in-app feedback tools in the future grew by more than 15% and more than 75% stated they would use a similar implementation in the future. Furthermore, the participants found giving feedback with emojis to be non-disturbing and easy.
Besides the overall positive acceptance among the participants, the answers revealed many areas of improvement for the realization. The two most important improvement areas are the selection of emojis for the feedback request and the number of requests. In the future, we will explore different digital motivation techniques that can be applied to our tool. Furthermore, we intend to improve the tool based on the feedback we will collect from its use within PfaffFunk.

Acknowledgments

This research was performed in the context of the “EnStadt: Pfaff” project (grant no. 03SBE112D and 03SBE112G) of the German Federal Ministry for Economic Affairs and Energy (BMWi) and the Federal Ministry of Education and Research (BMBF). We would like to thank all participants for their contribution to this study. We thank Sonnhild Namingha and Yannika Egler for proofreading.

References

[1] "28 Mobile App Statistics To Know In 2020," MindSea Team, [Online]. Available: https://mindsea.com/app-stats/. [Accessed 25 11 2020].

[2] S. Ickin, K. Petersen and J. Gonzalez-Huerta, "Why Do Users Install and Delete Apps? A Survey Study," in *Software Business. ICSOB 2017. Lecture Notes in Business Information Processing, vol 304,* Springer, Cham, 2017, pp. 186-191.

[3] S. Norbert, S. Melanie, F. Farnaz, G. Martin, G. Emitza, K.-H. Martina, M. Denisse, O. Marc and S. Ronnie, "End-user Driven Feedback Prioritization," in *International Workshop on Requirements Prioritization and Enactment,* Essen, Germany, CEUR-WS.org, 2017, pp. 1-7.

[4] S. A. Scherr, F. Elberzhager and L. Müller, "Quality Improvement of Mobile Apps - Tool-Supported Lightweight Feedback Analyses," in *PROFES 2018,* Wolfsburg, Germany, 2018.

[5] N. Seyff, G. Ollmann and M. Bortenschlager, "AppEcho: a user-driven, in situ feedback approach for mobile platforms and applications," in *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems (MOBILESoft 2014),* ACM, 2014, p. 99–108.

[6] M. Aljannan, M. A. Ismail and A. Salah, "Enhancing Software Evolution Requirements Engineering Based on User Feedback," in *Computer and Information Science, Canadian Center of Science and Education, vol. 13(3),* 2020, pp. 1-16.

[7] I. Morales-Ramirez, A. Perini and R. S. Guizzardi, "An ontology of online user feedback in software engineering," in *Applied Ontology 10 (2015),* IOS Press, 2015, p. 297–330.

[8] K. Holl, N. Claudia, V. Vaninha and V. Karina, "Safety-Critical Mobile Systems – The RESCUER Interaction Evaluation Approach," *Journal of Ubiquitous Systems & Pervasive Networks,* vol. 9, no. 1, pp. 01-10, 2017.

[9] S. A. Scherr, S. Hupp and F. Elberzhager, "Establishing Continuous App Improvement by Considering Heterogenous Data Sources," *International Journal of Interactive Mobile Technologies (iJIM),* vol. 15, no. 10, pp. 66-86, 2021.

[10] F. Elberzhager, P. Mennig, S. Polst, S. A. Scherr and P. Stüpfert, "Towards a Digital Ecosystem for a Smart City District: Procedure, Results, and Lessons Learned," *Smart Cities,* vol. 4, no. 2, pp. 686-716, 2021.

[11] F. Elberzhager, M. Koch and B. Weitzel, "Towards a Digital Ecosystem for Rural Areas: Experiences from Three Years of Development," *Product-Focused Software Process Improvement (PROFES) Lecture Notes in Computer Science,* vol. 11271, pp. 98-105, 2018.

[12] E. C. Groen, N. Seyff, R. Ali, F. Dalpiaz, J. Doerr, E. Guzman, M. Hosseini, J. Marco Gómez, M. Oriol Hilari, A. Perini and M. Stade, "The Crowd in Requirements Engineering: The Landscape and Challenges," in *IEEE Software, vol. 34, no. 2,*, IEEE, 2017, pp. 44-52.

[13] M. Oriol, M. Stade, F. Fotrousi, S. Nadal, J. Varga, N. Seyff, A. Abello, X. Franch, J. Marco and O. Schmidt, "FAME: Supporting Continuous Requirements Elicitation by Combining User Feedback and Monitoring," in *2018 IEEE 26th International Requirements Engineering Conference (RE),* Banff, IEEE, 2018, pp. 217-227.

[14] K. Petersen, S. Vakkalanka and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," in *Information and Software Technology, Volume 64,* Department of Software Engineering, Blekinge Institute of Technology, Sweden, Elsevier, 2015, pp. 1-18.

[15] D. Wüest, F. Fotrousi and S. Fricker, "Combining Monitoring and Autonomous Feedback Requests to Elicit Actionable Knowledge of System Use," in *Requirements Engineering: Foundation for Software Quality. REFSQ 2019. Lecture Notes in Computer Science, vol 11412,* Cham, Springer, 2019, pp. 209-225.

[16] M. Stade and H. Indervoort, "Feedback Gathering for Truck Parking Europe: A Pilot Study with the AppEcho Feedback Tool," in *Product-Focused Software Process Improvement (PROFES) Lecture Notes in Computer Science, vol 10611,* Cham, Springer, 2017, pp. 255-262.

[17] D. Dzvonyar, S. Krusche, R. Alkadhi and B. Bruegge, "Context-Aware User Feedback in Continuous Software Evolution," in *2016 IEEE/ACM International Workshop on Continuous Software Evolution and Delivery (CSED),* Austin, IEEE, 2016, pp. 12-18.

[18] D. Pagano and B. Bruegge, "User involvement in software evolution practice: A case study," in *3013 35th International Conference on Software Engineering (ICSE),* San Francisco, IEEE, 2013, pp. 953-962.

[19] M. Almaliki, C. Ncube and R. Ali, "The design of adaptive acquisition of users feedback: An empirical study," in *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS),* Marrakech, IEEE, 2014, pp. 1-12.

[20] L. Steffen and R. Asarnusch, "Fostering Remote User Participation and Integration of User Feedback into Software Development," in *Conference: Proceedings of the First Workshop on the Interplay between Usability Evaluation and Software Development,* Pisa, I-USED, 2008.