The Crowd as a Source of Knowledge - from User Feedback to Fulfilling Requirements

Anna Baj-Rogowska
Gdańsk University of Technology
Gdańsk, Poland *anna.baj-rogowska@pg.edu.pl*

Abstract

Crowd-based and data-intensive requirements engineering (RE) strategy is an approach for gathering and analyzing information from the general public or the so-called crowd to derive validated user requirements.

This study aims to conceptualize the process of analyzing information from a crowd to achieve the fulfillment of user requirements. The created model is based on the ADO framework (Antecedents-Decisions-Outcomes). In the empirical part, we chose the Instagram mobile app and user feedback on it as a source of data for the validation of our approach. For extracting antecedents from user feedback, we applied the Latent Dirichlet Allocation (LDA), and then sentiment analysis was performed for each topic to prioritize the most urgent tasks delegated by the crowd.

The main findings of our study reveal that using the wide spectrum of experience and knowledge of users (the wisdom of the crowd) from user opinions helps uncover different aspects that are helpful during software development. The conceptualization based on the ADO framework reflects and captures this process well. Thus, crowdsourcing is an alternative to traditional methods and techniques for requirements engineering.

Keywords: crowd engineering, crowdsourcing, wisdom of the crowd, crowdRE, ADO framework.

1. Introduction

Requirements are essential in software engineering since they ensure product quality and customer satisfaction [21]. Despite the constant striving for excellence during software development, some errors and bugs can be revealed at any stage of a product's life. They may result from an inaccurate analysis of requirements, disruptions in the communication process, or programming errors [45]. Undetected bugs are amplified and most often revealed by users. They can lead to software failures and cause a product to fail to meet user expectations. Traditional approaches used in requirements engineering (such as interviews, workshops and focus groups) are based on a limited number of users as stakeholders. However, now, thanks to the use of user opinions published on the Internet, it is also possible to gain knowledge about requirements and verify to what extent they keep up with the changing context. Their added value is that they are based on the involvement of a large group of potential users from around the world, including people from different cultural and geographical backgrounds [38]. Although users provide feedback on their experience with a given product in natural language, thanks to the use of text-mining techniques (e.g., by classifying, filtering and summarizing user comments, etc.), it is easy to monitor requirements and make effective decisions in the software development process [26]. In addition, user opinions extracted from online communities (i.e., social media platforms, app stores and many forums) capture the "wisdom of the crowd" and thus have begun to play an important role in requirements engineering processes [46].

The usage of text-mining techniques and user opinions for requirements engineering (RE) has been adopted in the scientific literature under various names, e.g., data-driven RE [26], crowd-centric RE [18; 40], crowd-based RE [8], [12], [23], [38], or crowdRE [22], [41]. Despite various names, all are focused on the same issue, namely, obtaining and analyzing user opinions about a product from a so-called “crowd” through automation, to obtain user requirements. Regardless of the adopted name, crowd based requirements engineering (or shorter, crowd engineering) is a promising approach to collecting and analyzing requirements from information generated by large and heterogeneous groups of users, often referred to as the “crowd”. Thanks to crowd use, by analyzing user feedback and ratings, it is possible to recognize which functions and features of an application users like or which are still lacking. In addition, behavioral patterns can also be detected, which allow us to learn more about user preferences [46].

In the digital world, every product has its “crowd” created by the users who utilize it. This crowd consists of huge, heterogeneous, and culturally and geographically dispersed groups of product users who interact with each other online. By using a software product (e.g., a mobile app, etc.) the crowd gathers practical user experience with that product. Crowd members eagerly share their experiences and opinions by posting reviews to praise and/or criticize apps or ask for functionalities that are currently not available. This allows product developers to understand which of the crowd's requirements are desirable and what should be omitted. There are plenty of mobile apps that are very useful and intuitive solutions to support everyday activities. The authors of [13] claim that Apple’s AppStore has around 1.2 million apps, and its competitor Google Play shares similar numbers. The number of mobile app downloads is massive, thus, the number of reviews is constantly growing. All of this information on the crowd’s experience provides valuable input that should be explored. Knowing how to benefit from this crowd gives a market advantage over all those competitors who fail to recognize this potential. However, the current state of research is still limited in providing a suitable framework for developing systematic user feedback acquisition. Therefore, the following research question is formulated:

RQ1. *How can the process of meeting user requirements be mapped and conceptualized based on the users' experience (as a crowd) received from user feedback?*

Bearing this in mind, this study aims to conceptualize the process of analyzing and managing information from a crowd to achieve the fulfillment of user requirements.

We propose an approach based on the ADO framework (Antecedents-Decisions-Outcomes) and text-mining of textual opinions obtained from users. In the created model, an important assumption is to profiteer the wisdom of the crowd flowing out from users. Thanks to the validation of the example of the Instagram (IG) application, we present the usefulness of this concept in meeting requirements from the recipients’ perspective. Importantly, our goal was not to evaluate the IG app, but only to test user reviews regarding this very popular app.

The novel contributions presented in this paper are three-fold: (i) conceptualization based on the ADO framework as a novelty in the crowd engineering field, (ii) the identification of antecedents from crowd feedback through the topic modeling (LDA) approach, (iii) the validation of the proposed approach in a case study.

The remainder of this paper has the following structure. In Section 2 the related works are presented. The proposed methodology is submitted in Section 3. We illustrate its practical applicability in the form of a case study in Section 4, while in Section 5 a discussion is conducted. Section 6 closes the paper with some conclusions.

2. Related works

Our approach encompasses the intersection of the wisdom of the crowd and engineering requirements for increasing app quality by adapting the app to the expectations of users. These domains are both rather complementary research challenges that create a new study area named crowd-based requirements engineering or shorter, crowd engineering.

2.1. Wisdom of the crowd

The first scientific exploration of the wisdom of the crowd (WoC) concept was described by the work of [7] regarding a weight-judging contest of a fat ox at a farmers’ fair. In [7] it was noted that the collective knowledge of the farmers' audience (the crowd) during the fair remarkably outperformed the accuracy of expert opinions (butchers). Later, similar findings were repeatedly confirmed by many scholars in various contexts, e.g.: real estate market price prediction [35], the improvement of disaster resilience [28], opinions influencing prices in the hospitality industry [34], and economic forecasting [5]. Thus, these findings have induced crowd leveraging for many challenging problems with very good results.

In [3] the authors define WoC as "the phenomenon wherein the combined judgment (i.e., the average judgment) of a group of individuals is, in most instances, superior to that of any one individual within the group". In turn, researchers in [6] suggested a definition of WoC - in terms of statistics sciences - as some linear combination of the crowd's estimates that should defeat the estimate of a randomly selected member of the crowd.

According to [43], members of a crowd should be: (i) diverse, so that individuals can offer various types of information, (ii) decentralized, so that no one at the top influences the collective outcome, (iii) independent, so that they are not guided by what others in the group think.

Diversity is crucial since the authors in [16] demonstrated that a group with both well-educated and not-so-well-educated members almost always defeats a group with just well-educated members. In turn, the study of [25] proved that even a little social impact can influence the effect of WoC in simple estimation tasks. Moreover, the findings of researchers in [15] confirmed that experience diversity, participant independence, and network decentralization are all positively related to crowd performance, which is consistent with the theory of crowd wisdom.

Thus, the wisdom of the crowd that emerges from user data is worth using in many areas, e.g., requirements engineering.

2.2. Crowd Engineering

Crowd engineering is based on the so-called wisdom of the crowd for product development, and assumes that groups can show an uncommon intelligence which exceeds that of the most intelligent individuals among them [43]. According to [11] the term is defined as "automated or semi-automated approaches to gather and analyze information from a crowd to derive validated user requirements". This term is rooted in the concepts of crowdsourcing and the wisdom of the crowd. However, crowd engineering delimits the application scope to engineering activities only.

With crowd-based engineering, in [12] the authors named the crowd members as 'informants of requirements', who usually, and in return for no payment, need not be in contact with the requesting team. These features distinguish this concept from the crowdsourcing phenomenon.

Prior studies related to online crowd engineering mainly concentrated on crowd-innovation tasks in terms of Open Source Software [44], [47]. In addition, there is a group of studies dedicated to the field of requirements engineering, and this stream is discussed in the next subsection.

2.3. Application of crowd engineering to extract requirements

Requirements are considered crucial in software engineering since their correct identification helps in ensuring product quality and customer satisfaction [21]. However, few works are conducted to extract crowd requirements. In [12] proposed "crowd based requirements engineering" as a way to integrate existing elicitation and analysis methods, as well as fill existing gaps by introducing new approaches, e.g., collecting feedback through direct interaction and social collaboration, and by deploying mining techniques. In [19] authors carried out a qualitative study that provides insight into the user-reported issues of iOS apps. They uncovered 12 complaint types from comments reported by users, which can help developers better prioritize their limited quality assurance resources. In [22] authors examined the requirements of smart home statements suggested by crowd users and required features from the given requirement statements dataset. Similarly, based on the smart home dataset, researchers in [29] identified key opportunities to develop automated crowd engineering techniques. In turn, the authors of [24] proposed a four-stage keyword-based machine approach to semi-automatically classify user requests in crowdsourcing scenarios. In [11] developed a novel crowd based RE approach for collecting user feedback which strongly focuses on a participatory vision. Intrinsically motivated users become crowd members because they benefit from software products that meet their needs.

The approach used in the present work is based on the principle of crowd-based requirements engineering presented by [11]. The dataset we used for the case study was collected considering the rules of the wisdom of the crowd and crowd engineering, thus it represents wide and diverse user requirements.

3. Methodology

In this section, the research approach adopted in this study was described. Subsection 3.1 discusses the model which was applied for conceptualization. Subsection 3.2 presents data collection for the case study while Subsection 3.3 describes the used approach for mining user feedback.

3.1. Antecedents-Decisions-Outcomes (ADO) Model

To achieve the goal of this study, the ADO framework was used for the conceptualization of the research approach. The ADO model was introduced by [32]. It is very flexible and can be used in many areas. To the best of our knowledge, we use this framework for the first time in crowd engineering to extract requirements, manage information from the crowd and finally meet the expectations of users. Antecedents constitute drivers which motivate future activity and can explain linkage or non-linkage to a particular task. An example of antecedents may be all factors hindering the use of the application, observed by users, e.g. technical problems after an update or suggestions on what is not working properly and should be fixed. They will trigger some decisions to be made.

Decisions represent certain kinds of tasks to do and are connected with the outcomes that occur after the performance or non-performance of an activity. In [32] the authors emphasize that the ADO dimensions are interrelated. An example of a decision can be actions taken by software developers, e.g. fix or delete, etc., which finally will trigger an outcome, e.g. full user satisfaction with the software.

To gain "structured insights" when creating antecedents, we decided to use user feedback (that is, opinions from the crowd) and process this using text-mining algorithms. This issue is presented in more detail in the subsequent subsections.

3.2. Data

Instagram is a Facebook-owned, free application created primarily for users of mobile devices. It allows users to share visually appealing content - mainly photos, but also posts and videos. After downloading the app, users are allowed to write online reviews about their experience with the app's usage. In this work, we treat these reviewers (IG app users) as the crowd and we intend to rely on their wisdom.

Data for this study cover Instagram mobile app user feedback (for Android = 391, and iOS = 176 users' opinions), scraped respectively from Google Play and the App Store. In total, 567 reviews, only in English, covering 1-31 March 2020 were downloaded.

3.3. Methods and Analysis

The R software was applied for the analysis of textual data. In our study, we decided not to create separate datasets for Android and iOS because (i) there were not a huge set of reviews and (ii) users' opinions for each group were not in comparable quantity. Thus, after collecting data from text reviews about the IG app, only one corpus consisting of

567 reviews was created. Then, it was subjected to several preprocessing steps according to standard text-mining procedures. First, punctuation marks (periods, commas, hyphens, etc.), numbers and white spaces were removed (library(tm)). Second, the characters in the entire corpus were converted to lowercase. Then, stopwords (e.g. "and", "or", "not", "in", "is", etc.) were removed. Finally, to ensure the terms in the corpus are uniquely identified, stemming was performed using Porter's stemming algorithm (library(SnowballC)).

This study focuses on uncovering antecedents as hidden sub-themes in the corpus. To achieve this, we applied the Latent Dirichlet Allocation (LDA) modeling approach (library(topicmodels)). To determine a suitable number of topics to extract from the corpus, we performed preparation-modeling-evaluation cycles. A 16-topic model was found to be optimal in terms of the average semantic coherence of the model [33] and [9].

Since we obtained topics described only by top-weighted keywords, a topic labeling process was needed. The result of this stage is presented in Table A (available on the GitHub repository). Using a lexicon for English opinion by [4] - to expand the context of this study with emotional information - we performed sentiment analysis (SA) for each topic at an extended binary level (positive or negative sentiment, i.e.: weak positive/negative sentiment, average positive/negative sentiment, high positive/negative sentiment, very high positive/negative sentiment, almost complete positive/negative sentiment). Such a gradation will determine the order for issues to be solved by developers. The most urgent topics are those with almost complete negative sentiment, contrary to the themes with weak negative sentiment. The Net Sentiment Rate (NSR) for each topic was calculated, according to the algorithm proposed by [1], including a standardized interpretation of the index result as well. The following calculation was used for the computation of the Net Sentiment Rate:

$$NSR = \frac{(PO-NO)}{(PO+NO)} \tag{1}$$

where: PO – positive opinions, NO – negative opinions.

NSR $\in$ (-1; +1), where: -1 opinions are totally negative, +1 opinions are totally positive. To be able to define the strength of the net sentiment precisely, the following classification was used [1]:

- $0.0 < | NSR | \leq 0.1$ - weak positive/negative sentiment,
- $0.1 < | NSR | \leq 0.3$ - average positive/negative sentiment,
- $0.3 < | NSR | \leq 0.5$ - high positive/negative sentiment,
- $0.5 < | NSR | \leq 0.8$ - very high positive/negative sentiment,
- $0.8 < | NSR | < 1.0$ - almost complete positive/negative sentiment.

The assignment of the sentiment of each extracted topic is shown in Tables B and Tables C (available on the GitHub repository).

Nowadays, fake data generated by artificial intelligence (AI) may pose a certain threat to scientific research. To check the quality of opinions from the dataset and finally avoid those generated, for example, by bots or fake accounts, a dozen users' feedback was selected from a random sample, and two issues were manually checked (i) whether it was a review or a short comment related to the topic and (ii) whether the sentiment score calculated by the algorithm is correct and raises no objections. No suspicious and questionable reviews or inconsistent sentiment scores were found. These activities provided the framework to evaluate the data quality of (i) the source data, (ii) results of algorithms, and finally (iii) output data.

4. Case study

A case study method is a popular approach in software engineering research [37] as it works well to research real-world phenomena, which should be studied with their practical context in mind [48].

For this study, we chose the Instagram mobile app and user feedback on it as a source of data for the case study. We created a crowd by joining its members by their common denominator, namely, that they use the IG mobile app and/or are interested in a well-functioning IG app. In addition, IG app users come from different cultural and geographical locations, thus, the set of users fits the definition of a "crowd".

According to [31], crowdsourcing and crowd engineering strategies may fail when applied to an inappropriate crowd or an insufficient number of participants who contribute to a project. To avoid these issues, we decided to use data covering Android and iOS feedback on the Instagram mobile app direct from Google Play and the App Store. Thus, we meet the key assumptions on which the wisdom of the crowd is based, namely, diversity, independence and decentralization [15].

In line with the presented methodology, we have extracted 16 topics from the corpus. Sentiment analysis for each of the recognized topics enabled expanding our study with an emotional diagnosis. In this way, we uncovered topics with negative sentiment as a foreground source of information reported by the crowd for improvement. We were guided by the findings of the authors [30] that unsatisfied users are willing to reveal many details about various aspects and app qualities that evoke frustration. Thus, they are a valuable source of information. In our model, the extracted topics become a group of antecedents. The conceptual model based on the ADO framework is presented in Fig. 1 (RQ1). The depicted conceptualization covers only antecedents with negative sentiment (see: Table C available on the GitHub repository) since they point out the most frustrating aspects for users, thus, they should be most urgently improved or removed.

The selected antecedents (as a result of LDA topic modeling) - based on the topic proportion and the value of the NSR indicator - are ranked in terms of the importance of their consideration and possible fulfillment. Thus, the first topic, *#13 Requests for adding some options to switch on and off*, most often appeared from the wisdom of the crowd and was at the same time the most negative (NSR = –0.7). By drilling down the opinions of application users on this topic, one can indicate what options /functionalities they lack. Table 1 in Annex contains some examples of original user feedback from topic #13.

Another topic with very high negative sentiment (NSR = –0.67) is *topic #15 Users' suggestions on what to fix*. User feedback from this topic provides ready-made tips work, and what should be improved so that the application better meets their expectations.

According to Figure 1, among the topics with the least negative sentiment is topic *#8 The app quality issues* (NSR = –0.01). By browsing the feedback related to this topic (Table 2 in Annex), you can easily get to the details of quality-related issues.

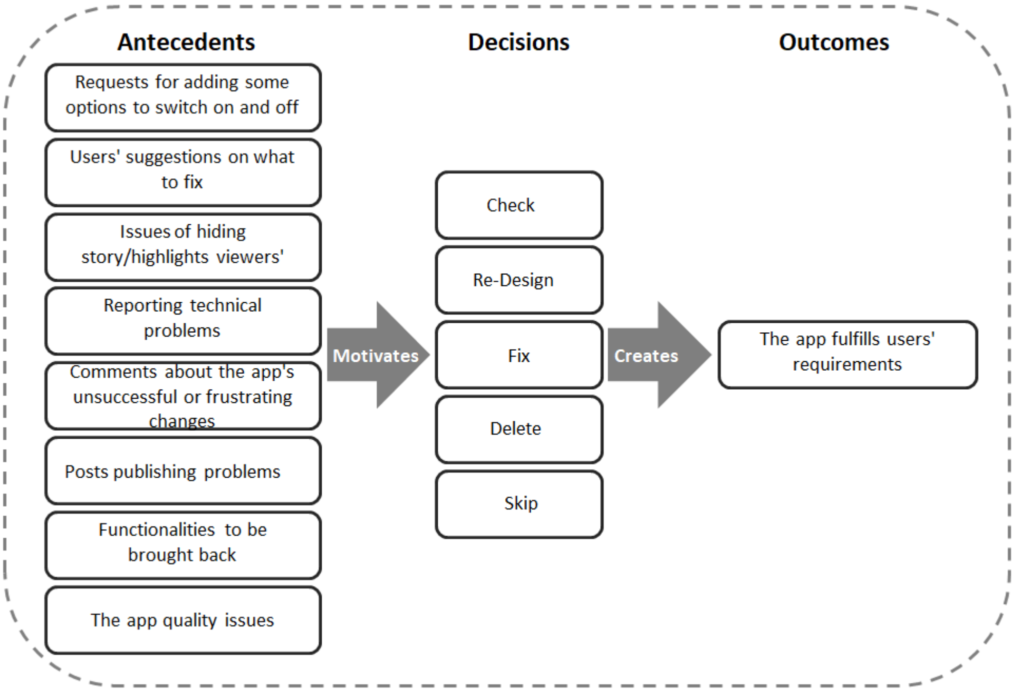

Fig. 1. Conceptual framework – antecedents, decisions and outcomes (ADO).

By extracting the knowledge discovered from opinions created in natural language (textual data), it is possible to uncover the most frustrating and problematic aspects indicated by app users. It should be emphasized that further inquiries can also consider topics with positive sentiment (Table B - available on the GitHub repository). These will certainly provide insight into additional user expectations.

The stages of collecting, analyzing and checking suggestions from the crowd would need to be repeated in an automated or semi-automated manner, preferably as iterative loops to verify the results of previous iterations in terms of monitoring aspects of the product. In this way, potential requirements can be identified early and continuously mapped based on the wisdom of the crowd.

Crowd wisdom is based on collective action in which only some part of the user population makes significant contributions to achieve the best version of the app. However, thanks to this approach, the desires and needs of potential end-users can really be more precisely understood and fulfilled. Thus, their implementation is a guarantee of achieving the quality desired by users.

5. Discussion

Quality is an abstract concept and, in principle, is difficult to measure. For this reason, the use of application users' opinions in the proposed approach, and their successive fulfillment is a guarantee that the quality desired by users can be achieved. Consequently, the app will fulfill the users' expectations and requirements.

By extracting topics and recognizing the type of sentiment associated with them, it becomes possible to garner all the main problems (*Antecedents*) that developers should deal with in order to obtain satisfaction from app users. Antecedents with negative sentiment are foreground themes that need to be addressed urgently. The next step is a diagnosis of areas associated with positive sentiment. All problematic issues can be considered and solved if they need improvement.

Users as crowd members report in their feedback on a variety of facets, which can be split into three main groups: (i) software problems, e.g., bugs, app crashes, (ii) extension ideas, e.g., feature requests, or (iii) ideas for improvements. Examples of user opinions in these three areas are presented in Table D (available on the GitHub repository). The aspects raised by users (the crowd) in our case study, identified in the table, are consistent with prior research. For example, the authors in [12], [41] identified software problems in terms of bugs and app crashes, in [17], [27] researchers noticed extension ideas encompassing feature requests, whereas in [23] researchers detected ideas for improvements.

In line with the authors of [22], we claim that the text mining approach delivers useful insights based on crowd data, which can be used by requirement engineers for bug fixing, software evolution planning, drawing ideas for improvements, etc. In the ADO model - in the *Decisions* part - we present general tasks to undertake, such as check, re-design, fix, delete and/or skip. The kind of decision is motivated by its *Antecedent*.

The key issue that initiates the whole process is that crowd members must have access to the places where submitting feedback is possible, e.g., app stores, social media sites, etc. Consequently, obtaining information from user opinions allows developers to take into consideration crowd members' requirements and needs, and finally initiates work to fulfill them. In terms of the ADO framework (RQ1), starting from *Antecedents* and subsequently performing the relevant *Decisions* results in getting the app in line with crowdsourcing requirements (as *Outcomes*, see Figure 1). Importantly, the obtained suggestions for improvement or the expressed requirements of the crowd are characterized by wisdom [15], thus their fulfillment can be very satisfying for both parties (users and app providers). When crowd members possess diverse knowledge and viewpoints (as is indisputable in our study), the crowd can be considered wise. This variety produces uncorrelated judgment errors that cancel out in aggregate [3]. Furthermore, a diverse group of individuals (arising from a heterogeneous population) brings different perspectives to user requests.

Beyond diversity, the characteristics of a wise crowd encompass independence and decentralization. Users' expressed opinions (the crowd) were not determined by others around them and we assume that they were written voluntarily and without pressure, thus the condition of feedback independence is fulfilled as well. Moreover, the crowd should be decentralized. This feature of a wise crowd is also maintained since our crowd members comprise a distributed conglomerate linked by the web and eventual social interactions [36]. The crowd members act freely of each other and there is no supervision over them. The user feedback regarding the Instagram app, therefore, includes the 'wisdom' of the crowd.

The communication covered by the conceptualization model (Figure 1) should complement traditional requirements elicitation approaches (interviews or workshops) in which only a limited number of users give feedback on their needs [11]. However, manually analyzing huge amounts of feedback has some flaws. Above all, it is extremely time-consuming and cognitively challenging. Therefore, techniques to automatically analyze user feedback are necessary to avoid these drawbacks. In the literature, there are studies on using automatic natural language processing (NLP) techniques in classifying, clustering and categorizing user opinions [13], [24], [39]. Similarly in our work, the text-mining method is employed along with sentiment analysis, which splits user opinions into negative and positive to easily find complaints and approvals regarding product features. Furthermore, this approach makes it possible to drill data down to obtain more detailed reports about bugs, feature requests, or other user needs. Developers can perform further specific analyses based on metadata such as time stamps to allow for the discovery of trends over time.

According to [2], topics with negative sentiment are more related to issues such as features, app crashes and fixes, whereas positive topics are more connected to general user perceptions. In [30] the authors revealed that disgruntled users are prone to provide details about various aspects that evoke frustration, while, when giving positive feedback, they usually present the overall qualities of the app. Having these findings in mind, we focused mainly on negative topics in our model. However, this does not mean that topics with positive sentiment should not be included for consideration in the next stage.

Based on a case study of crowdsourcing software development at a multinational corporation, the authors of [42] claimed that the cost was much greater than originally expected. Our approach ignores efforts to prepare specifications and avoids waiting for answers to queries of the IG app community. Similarly, the time to complete review mining and finally resolve quality issues is significantly shorter, due to the division of tasks related to negative and then the positive sentiment. Thus, the listed aspects speak in favor of our approach.

The presented approach brings *theoretical* (the conceptualization of the process of requirements acquisition based on the ADO model as a novelty in the crowd engineering field) and *practical contributions* (the procedure including tasks for developers captured in three stages of the ADO framework). An additional benefit of this approach is based on crowd wisdom usage.

6. Conclusions and future work

To entirely fulfill user needs (*Outcomes* in the ADO model), customers' demands should be constantly monitored and uncovered (*Antecedents*), and these demands should subsequently be satisfied (*Decisions*) in a short time period, with low cost and high efficiency (RQ1). Identified *Antecedents* motivate the choice and performance of relevant tasks from across a spectrum of decisions in order to create an app according to users' needs and requirements. Our conceptualization of this process based on the ADO model maps well onto these activities. What is more, the app manufacturing process and its improvement take into account continuous innovation with what comes from the wisdom of the crowd.

With the crowd engineering approach, user voices are captured and practically utilized very quickly. Thus, an important benefit is the reduction of development time and/or increased success of the developed product, as well as superior acceptance by customers. It is, however, necessary to note some flaws of this approach, namely, it can create unrealistic expectations and ambiguous requirements, which was also recognized by [8].

On the other hand, the dynamic development of AI technology in the researched field may enable scientists to use more advanced methods of text data analysis and sentiment analysis. Deep learning models can be better at understanding context and subtleties in expressions (i.e. irony or sarcasm). Advanced AI algorithms allow for analysis personalisation, adapting the results to the user's individual preferences and context. The proposed approach has the potential to develop and adapt to changes in terms of new possibilities. Thus, this is the universality of the presented conceptualization based on the ADO model.

The *practical implication* of our user feedback data based approach is the usage of the diversity and spectrum of experience and knowledge of users (wisdom of the crowd) from a large number of opinions to uncover different aspects which are helpful during product development and manufacturing. Our case study (for the Instagram mobile app) confirmed that crowdsourcing is an alternative to traditional methods and techniques for requirements engineering. Consequently, deep reflection on user perceptions and expectations within requirements (especially pointed out by negative sentiment) contributes to identifying priorities for action and developing an app that is in line with crowdsourcing requirements, i.e., entirely fulfills user expectations.

As with many other empirical studies, our study has some *limitations*. However, these limitations are minor and open up opportunities to pursue further research to generate actionable insights with value to the crowd-based and data-intensive requirements engineering area.

First, the study is based on a small number of user opinions subjected to text mining analyses. This approach resulted from the fact that this is a pilot study intended to test the developed concept.

Second, the focus on a single case study limits the generalizability of the findings. Extending research on additional apps and more user opinions could perhaps lead to better insights and conclusions. Thus, future research should be conducted on a much larger data set of user feedback and should cover other types of apps as well.

Third, in response to the research question, we provide only a framework for conceptualizing a process of analyzing and managing information from a crowd to fulfill user requirements. There is a need for further research on developing detailed principles (step by step) for using this approach. Additional research in the above area will provide developers with tools and more detailed practical guidelines on how to draw on this approach to achieve a high-quality application that fully meets the expectations of its users. In addition, in future research, we will investigate the different requirements groups, their flows, and their characteristics in a more direct way.

Despite these limitations, the findings of this study should provide a good starting point for future research on crowd-based and data-intensive requirements engineering.

References

1. Baj-Rogowska, A.: Evaluation of a company’s image on social media using the Net Sentiment Rate. In: Lechman E., Popowska M. (eds.), Society and Technology Opportunities and Challenges, pp. 202-218, Routledge (2020)
2. Baj-Rogowska, A., Sikorski, M.: Exploring the usability and user experience of social media apps through a text mining approach. Engineering Management in Production and Services, 15(1), pp. 86-105, DOI: 10.2478/emj-2023-0007 (2023)
3. Bhatt, S., Minnery, B., Nadella, S., Bullemer, B., Shalin, V., Sheth, A.: Enhancing crowd wisdom using measures of diversity computed from social media data. In: Proceedings of the international conference on web intelligence, pp. 907-913, (2017).
4. Breen J.: twitter-sentiment-analysis-tutorial-201107, https://github.com/jeffreybreen/twitter-sentiment-analysis-tutorial-

201107/tree/master/data/opinion-lexicon-English (2023)

5. Budescu, D.V., Chen, E.: Identifying expertise to extract the wisdom of crowds, Manage. Sci. 61, 267–280 (2014).
6. Davis-Stober C.P., Budescu D. V., Dana J., Broomell S.B.: When is a crowd wise? Decision 1 (2), 79–101 (2014).
7. Galton F.: Vox populi (the wisdom of crowds), https://scholar.google.com/scholar_lookup?title=&journal=Nature&doi=10.1038%2F075450a0&volume=75&publication_year=1907&author=Galton%2CF, Nature, 75, (1907)
8. Ghanyani, U.S., Murad, M., Mahmood, W.: Crowd-based requirement engineering. International Journal of Education and Management Engineering, 3, 43-53 (2018).
9. Griffiths T.L., Steyvers M.: Finding scientific topics. Proceedings of the National Academy of Sciences 101, suppl 1: 5228–5235, http://doi.org/10.1073/pnas.0307752101, (2004)
10. Groen E.C.: Crowd Out the Competition. 2015 IEEE 1st International Workshop on Crowd-Based Requirements Engineering (CrowdRE), Ottawa, ON, Canada, pp. 13-18, doi: 10.1109/CrowdRE.2015.7367583, (2015)
11. Groen E.C., Seyff N., Ali R., Dalpiaz F., Doerr J., Guzman E., Hosseini M., Marco J., Oriol M., Perini A., Stade M.: The Crowd in Requirements Engineering: The Landscape and Challenges. IEEE Software, 34 (2), 44-52, doi: 10.1109/MS.2017.33, (2017)
12. Groen, E.C., Doerr, J., Adam, S.; Towards crowd-based requirements engineering. In: Fricker, S.A., Schneider, K. (eds.) LNCS, vol. 9013, Springer, Cham (2015)
13. Guzman, E., El-Haliby, M., Bruegge, B.: Ensemble methods for app review classification: An approach for software evolution (n). 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), IEEE (2015)
14. Gu, Xiaodong & Sunghun Kim: What parts of your apps are loved by users?(T), 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), IEEE, (2015)
15. Hong, H., Ye, Q., Du, Q., Wang, G. A., Fan, W.: Crowd characteristics and crowd wisdom: Evidence from an online investment community, Journal of the Association for Information Science and Technology, 71(4), 423-435 (2020).
16. Hong, L., Page, S.E.: Groups of diverse problem solvers can outperform groups of high-ability problem solvers, Proceedings of the National Academy of Sciences of the United States of America, 101, 16385–16389 (2004)
17. Hosseini, M., Phalp, K., Taylor, J. Ali, R.: Towards crowdsourcing for requirement engineering, In. 20th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ), Springer, Heidelberg (2014)
18. Hu, W.-C., Hewijin C.J.: UCFrame: a use case framework for crowd-centric requirement acquisition. ACM SIGSOFT Software Engineering Notes 41 (2), 1-13 (2016)
19. Khalid H., Shihab E., Nagappan M., Hassan A. E.: What do mobile users complain about?, IEEE Software, 32(3), 70-77 (2015).
20. Khan, J. A., Liu, L., Wen, L., Ali, R.: Crowd intelligence in requirements engineering: Current status and future directions. In Requirements Engineering: Foundation for Software Quality: 25th International Working Conference, REFSQ 2019, Essen, Germany, Proceedings 25, Springer International Publishing (2019).
21. Khan, J.A., Liu, L., Wen, L., Ali, R.: Crowd intelligence in requirements engineering: Current status and future directions. In Requirements Engineering: Foundation for Software Quality: 25th International Working Conference, REFSQ 2019, Essen, Germany, Proceedings 25, pp. 245-261, Springer International Publishing, (2019).
22. Khan, J. A., Lin, L., Jia, Y., W, L.: Linguistic analysis of crowd requirements: an experimental study, In: Proceedings of 27th IEEE RE Workshop (Empri 2018), pp. 24–31, (2018)
23. Levy, M., Hadar, I., Teeni, D.: A gradual approach to crowd-base requirements engineering: the case of conference online social networks, In: IEEE 2nd CrowdRE, pp. 26–30 (2017)
24. Li, C., Huang, L., Luo, J. G. B., Ng, V.: Automatically classifying user requests in crowdsourcing requirements engineering, JSS 138, 108–123, (2018)

25. Lorenz J., Rauhutb H., Schweitzera F., Helbing D.: How social influence can undermine the wisdom of crowd effect, Proc. Natl. Acad. Sci. 108(2), 9020–9025 (2011)
26. Maalej, W., Nayebi, M., Johann, T., Ruhe, G.: Toward data-driven requirements engineering, IEEE Software, 33(1), 48-54 (2015)
27. Merten, T., Falis, M., Hübner, P., Quirchmayr, T., Bürsner, S., Paech, B.: Software feature request detection in issue tracking systems. In: Proceedings of RE 2016, pp. 166–175 (2016)
28. Moghadas, M., Fekete, A., Rajabifard, A. et al.,: The wisdom of crowds for improved disaster resilience: a near-real-time analysis of crowdsourced social media data on the 2021 flood in Germany, GeoJournal, (2023)
29. Murukannaiah P.K., Ajmeri N., Singh M.P.: Towards automating Crowd RE, IEEE 25th RE Workshops, pp. 512-515 (2017).
30. Nakamura, W.T., de Oliveira, E.C., de Oliveira, E.H.T., Redmiles, D., Conte, T.: What factors affect the UX in mobile apps? A systematic mapping study on the analysis of app store reviews. Journal of Systems and Software, 193, ISSN 0164-1212, https://doi.org/10.1016/j.jss.2022.111462, (2022)
31. Panchal J.H.: Using crowds in engineering design towards a holistic framework, Proceedings of the 20th International Conference on Engineering Design (ICED15), (2015)
32. Paul, J., Benito, G.R.: A review of research on outward foreign direct investment from emerging countries, including China: what do we know, how do we know and where should we be heading?, Asia Pacific Business Review, 24(1), 90-115 (2018)
33. Rajkumar, A.V., Suresh, C.E., Madhavan, V., Narasimha Murthy, M.N.: On finding the natural number of topics with Latent Dirichlet Allocation: Some observations. In Advances in knowledge discovery and data mining, Zaki, M.J., Yu, J.X., Ravindran, B., Pudi V. (eds.). Springer Berlin Heidelberg, 391–402, http://doi.org/10.1007/978-3-642-13657-3_43 (2010)
34. Ren, J., Raghupathi, V., Raghupathi, W.: Effect of crowd wisdom on pricing in the asset-based sharing platform: An attribute substitution perspective, International Journal of Hospitality Management, 94, (2021)
35. Rizun N., Baj-Rogowska A.: Can Web Search Queries Predict Prices Change on the Real Estate Market?, IEEE Access, 9, 70095-70117 (2021)
36. Rulke, D.L., Galaskiewicz, J.: Distribution of knowledge, group network structure, and group performance, Management Science, 46, 612–625 (2000)
37. Runeson, P., Höst, M., Rainer, A. Regnell, B.: Case Study Research in Software Engineering: Guidelines and Examples, Wiley (2012)
38. Sherief, N., Abdelmoez, W., Phalp, K., Ali, R.: Modelling Users Feedback in Crowd-Based Requirements Engineering: An Empirical Study, In: Ralyté, J., España, S., Pastor, Ó. (eds) The Practice of Enterprise Modeling. PoEM 2015. Lecture Notes in Business Information Processing, vol 235. Springer, Cham (2015)
39. Shi, L., Chen, C., Wang, Q., Boehm, B. M.: Is it a new feature or simply "Don't Know Yet"?: on automated redundant OSS feature requests identification, In: RE 2016, pp. 377–382 (2016)
40. Snijders R., Özüm A., Brinkkemper S., Dalpiaz F.: Crowd-Centric Requirements Engineering: A method based on crowdsourcing and gamification, Technical Report UU-CS-2015-004, Utrecht University (2015)
41. Stade, M., et al.,: Providing a user forum is not enough: first experiences of a software company with CrowdRE. In: IEEE 2nd International Workshop on CrowdRE, pp. 164–169 (2017).
42. Stol K.J., Fitzgerald B.: Two's Company, Three's a Crowd: A Case Study of Crowdsourcing Software Development. Proc. 36th Int'l Conf. Software Eng. (2014)
43. Surowiecki J.: The wisdom of crowds, 1st ed., Anchor Books, New York, NY (2005)
44. Temizkan, O., Kumar, R.L.: Exploitation and exploration networks in open source software development: An artifact-level analysis, Journal of Management Information Systems, 32, 116-150 (2015)
45. Wrycza S., (ed): Business informatics, PWE, Warszawa (2010)

46. Gu, X., Kim, S.: What parts of your apps are loved by users?, 30th IEEE/ACM International Conference on Auto-mated Software Engineering (ASE), IEEE (2015)
47. Yin, G., et al.: OSSEAN: Mining crowd wisdom in open source communities, 2015 IEEE Symposium on Service-Oriented System Engineering. IEEE (2015)
48. Yin, R.K.: Case Study Research, 3rd ed., SAGE (2003)

Annexes

Table 1. Examples of user feedback in topic #13 (original language).

User opinion ID	Sample Reviews
#123	"(...) However, I wish there was another way to remove followers other than block them (...)"
#166	"(...) when someone mention @ me in the insta highlight, there I still dont have the function for "add this to your story function" at my mesage. what is this? all my friends already gotten this feature and My account didnt see this option. PLEASE FIX THIS PROBLEM. and i dont like that doesn't allow me to see who viewed my stories the feature after 24 hours deadline. i want to know who is interested to see my stories. the numbers not important."
#511	"Would be nice if you add an option to switch on and off automatic video playback on mobile data. It cost me a lot of data when videos play by itself."

Table 2. Examples of user feedback in topic #8 (original language).

User opinion ID	Sample Reviews
#11	"Story video quality between IOS and Android is still far different. Especially at the night. IOS do much better in terms of quality and sound video story compare to Android. (...)"
#164	"Why does it lower the image quality when I record videos or take photos through the app? my internet connection is fine and there's nothing wrong with my camera. I asked my friends about it and it seems to be an occurring problem among Android devices. Please fix it immediately!"
#215	"when uploading a video on my story, the quality of the video turns out very bad and blurry. same issue when i add gif to a pic story. please fix this problem! however, everything was fine if i only upload a pic story without gifs."
#224	"This app sucks you'll insta people better fix the video quality. Oh us we don't mind moving to FB so solve this video quality thing. This App deserves 1 star for now. Insta won't really do anything to help us fix the video quality (...)"