Improving the Elicitation of Delightful Context-Aware Features: a Data-Based Approach

Authors:
Rodrigo Falcão

IESE-Report No. 035.17/E
Version 1.0
November 2017

A publication by Fraunhofer IESE
Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft. The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competitive market position.

Fraunhofer IESE is directed by Prof. Dr.-Ing. Peter Liggesmeyer (Executive Director) Prof. Dr. Dieter Rombach (Director Business Development) Fraunhofer-Platz 1 67663 Kaiserslautern Germany
Improving the Elicitation of Delightful Context-Aware Features: a Data-Based Approach

Rodrigo Falcão
Department of User Experience and Requirements Engineering
Fraunhofer Institute for Experimental Software Engineering IESE
Kaiserslautern (Germany)
rodrigo.falcao@iese-extern.fraunhofer.de

Abstract—The pursuit of innovation in the mobile software industry frequently requires coming up with new features – and not just any feature, but startling and unexpected delightful features. Despite the potential of context awareness to provide a system with delightfulfulness, current requirements elicitation techniques do not cope with an essential aspect: comprehension of the relationships among the numerous contextual elements of a certain domain and how they can positively influence the user task. As a result, solution providers continuously miss the opportunity to add more value to their software solutions by identifying context-aware behaviors that will lead to better recommendations or adaptations. This paper discusses this problem and proposes a process for tackling it by taking the task to be improved as input and semi-automatically generating (1) a conceptual context model that reveals and represents both the relevance of contextual elements and the relationships among them, and (2) candidate requirements. The expected scientific contributions of the ongoing research are delineated as well.

Index Terms—software engineering; requirements engineering; requirements elicitation; context awareness; context modeling; context meta-modeling;

I. INTRODUCTION

We live in a rapidly changing world where computers are increasingly ubiquitous and where part of our experience in terms of mobility and pervasiveness is related to the broad adoption of smartphones. Through them, people have ready access to literally millions of apps in a market with numerous players, where new apps and features are deployed on a daily basis.

In order to attract and keep users, competitors in the mobile software industry yearn to reduce the time to market needed to come up with new features in their apps. However, their aim is not to develop just any feature: ideally, they want to come up with delightful features. According to [1], a delighter is a surprising, innovative feature that amazes users by surpassing their expectations – but the delightfulfulness of any feature decreases over time and it becomes a satisifier, which refers to features that are consciously expected and demanded by stakeholders. Therefore, in order to continue to offer delightful features, app providers have to continuously discover them by performing fast iterations in their development cycles. However, delighters are a particular type of requirement that has been hardly elicited because it deals with exceeding stakeholders conscious as well as their subconscious expectations [2].

There are different ways to promote delightfulfulness. One possibility is to improve the system with context awareness [3] [4] [5], which refers to the systems ability to take into account the context in order to adapt its behavior [6] [7]. Context awareness is a key enabling characteristic of ubiquitous computing [8] [9]. The ability to properly sense, reason about and, finally, act before the user promotes a better experience in the sense the system is able to do more without explicit user interaction. In other words, it saves user attention and time.

The elicitation of context-aware features starts with a selected user task that is to be improved via context awareness. Requirements engineers identify the contextual elements that are pertinent to the domain and, for each of them, analyze their relevance for the selected task and how they could influence (either improve or impair) the task. Finally, requirements engineers have to identify possible existing meaningful relationships among the contextual elements and whether such relationships can positively influence the user task (i.e., whether they are relevant for the task). The importance of relevance and relationships in context analysis is consistently present in the literature (e.g. in [7] [11] [12] [10] [13] [14] [15]), but the question remains of how to discover them.

So far, traditional elicitation techniques such as questionnaires and interviews have not proven to be the best way to unveil the unknown relevance of contextual elements, let alone the unknown relationships among them. Furthermore, it is already well known that elicitation techniques do not lead to delighters [2]. In contrast, there are creative techniques, which are suited very well for the elicitation of delighters [16] [17] [2], but which are still not effective for the identification of relevance and relationships, which is necessary for full comprehension of the context – which in turn would enable the elicitation of new, unexpected context-aware features. Regardless of which creative technique is chosen, practitioners still would have to figure out how the contextual elements might relate to each other. This is impractical for two reasons: First, it is not intuitive to discover meaningful relationships among

1This work adopts the concept of contextual elements presented in [10]: “A contextual element (CE) is any piece of data or information that enables to characterize an entity in a domain.” Common examples of contextual elements for an entity user could be “age” and “gender”; for an entity mobile device, “battery level” and “network”; for an entity environment, “weather condition” and “luminosity”; and so on.
contextual elements, which usually refer to implicit, sensed data or information. Second, even if requirements engineers were to carefully try to analyze each possible combination in search of meaningful relationships that could influence the targeted user task, this would not be viable because of the high number of possible combinations. In practice, the analysis of the relationships among contextual elements has been overlooked because with the current techniques it is not feasible to investigate all possible combinations in a scenario with dozens of contextual elements.

This paper addresses the complexity of eliciting contextual behaviors due to the lack of understanding of the relationships among contextual elements when the user performs a certain task. It is structured as follows: The problem is detailed in Section II; related work is discussed in Section III; the proposed solution, the expected contributions, and a discussion on open issues are presented in Section IV; the research method is outlined in Section V; and conclusions are drawn in Section VI.

II. THE PROBLEM

The global competition pushes competing companies to run the development cycle of delightful features faster and faster, and the elicitation step is at the core of the complex task of inventing unexpected functionalities. Since traditional elicitation techniques are not suitable for eliciting delightful features, requirements engineers have employed creativity techniques [2], which are numerous and have been successfully utilized to trigger the discovery of innovative, surprising features [17]. However they are not well suited for coping with the specific characteristics of context awareness. Providing a system with context awareness does, from a requirements engineering perspective, call for:

1) the identification of contextual elements [10], i.e., clear comprehension of what the context is for the given application domain [18];
2) a technical understanding of what is actually accessible (i.e., is feasible to sense) among the identified contextual elements [15];
3) comprehension of the relevance of each contextual element when an actor is performing a particular user task [19] [10] [15]; and
4) the identification of meaningful relationships (among the contextual elements) that can positively influence a particular user task [15].

The relevance of contextual elements (item 3) and particularly the relationships among contextual elements (item 4) are hard to capture, even by creativity techniques, for they rely on the experience of the participants running the creativity sessions, and because such comprehension goes beyond human capability, as exemplified in the following illustration.

Consider a given mobile app that was deployed in the past and had some delightful features that were appreciated by its users. Over time, the delightfulness of the features decreased until it disappeared, because the users got used to them. Later the app providers decide to come up with new surprising features by improving the app with context awareness. They apply creative techniques. If they succeed, new context-aware behaviors will be discovered and expressed as validated requirements, which will serve as input to the development team, who, in turn, will deliver a new version of the app to be deployed with new context-aware features. Again, these features will become satisfiers over time. Then the app owners have to run the cycle over and over again, if they want to constantly provide their users with innovative context-aware experiences. Figure 1 depicts this cycle.

When creative techniques are applied, the relevances of previously known contextual elements is analyzed by a number of stakeholders who are brought together in a creativity session. As long as this is done offline, i.e., before the system is running, these techniques just systematize an improved guess about relevance. For example, Morphological Forced Connections could be used to find out how certain contextual elements would support a chosen task, but the technique itself does not state whether the contextual element is de facto relevant for this user task or not. Similarly, the 6-3-5 Method can be used to push practitioners to think “out of the box” on how the contextual elements can support the chosen user task, but it would still be a conjecture, a feeling about the relevance of each contextual element for the task. Even more difficult would be to analyze the relationships among contextual elements. Imagine a scenario where the mobile app has 20 contextual elements: If requirement engineers tried, they would have to analyze 190 possible combinations of two elements just for starters2, and envisage whether each

2The number of possible 2-by-2 combinations among 20 elements is 190. This is a simplification that considers only one possible relationship between two contextual elements. For instance, consider two contextual elements, time and location. Time could be labeled as morning, noon, afternoon, and evening; while location could be home, school, and work. In this example, there are two contextual elements but the single relationship among them is not exactly “single”: there are $4 \times 3$ possible combinations to be analyzed.
combination would be relevant for each user task or not. Moreover, context sources frequently offer raw sensor data, which makes the requirements engineer’s task of reasoning on the relationships near to impossible.

Because it is simply impractical to evaluate such large numbers of combinations, requirement engineers barely analyze the relationships among contextual elements at all and do not investigate the relevance of such relationships. Thus practitioners continue to rely on stakeholders’ experience regarding context awareness (based on previous projects or known applications) to declare context-aware behaviors. As a result, solution providers keep on missing the opportunity to add more value to their products by identifying unexpected context-aware behaviors that would lead to better recommendations or adaptations. The research question is this: How to improve the requirements engineers’ task of eliciting delightful context-aware behaviors?

III. RELATED WORK

The starting point for improving a system with context awareness is to define what should be acquired as contextual data. In [18], a process called context conceptualization is presented as a way to address this issue. It consists of a sequence of brainstorming sessions that look at the context domain in both a bottom-up and a top-down manner. The result is a set of contextual elements classified into a valid taxonomy for the application domain: a “conceptualized context model”. Thus, this work provides a process for coming up with contextual elements and a conceptual model to represent it.

Conceptual models are handy for analyzing and better understanding requirements in general. Several works have proposed different ways to model context, such as those presented in [20], [12], [13] and [21], which are usually grouped according to their characteristics: key-value models, markup scheme models, graphical models, object-oriented models, spatial models, logic-based models, and ontology-based models. Besides the logic-based models, which actually describe contextual behaviors – and do not support the elicitation of contextual behaviors, but rather their documentation –, these designed models and meta-models support representing what context is, but do not offer clues on how to use it. Such an activity is expected to be performed creatively by requirement engineers. In addition, such models are built manually, which means that even when they provide assistance for describing relationships (as graphical models or ontological models do), the task of discovering them is still up to the requirement engineers.

Depending on the domain, there may be a large collection of contextual elements, and some of them may be in evidence when a user is performing a specific task. This notion was explored in [19]: To model a context-aware system, identifying the user’s “attention focus” is mandatory and involves recognizing an actor performing a task that can be enriched through context usage. This means that the relevance of contextual elements can vary depending on the user’s focus (or task). The reason for being concerned with the relevance of contextual elements is also stated, for instance, in [10]: Such relevance is important for identifying behavior patterns for contextual processing rules. In [13], the authors argue that relationships and dependencies are requirements for modeling context-aware systems. Thus, the next step is to figure out how contextual elements are supposed to be combined in order to trigger a change of the system behavior.

This step is covered in a process for designing context-aware systems presented in [15], where the authors surveyed state-of-the-practice design methods and elaborated an ideal process, which reiterates the importance of discovering the relevance and the relationships (referred to as “combinations”) of contextual elements, and also postulated that these activities are often done together. However, the process suggests that such activities should be done manually, which is not feasible when there are many contextual elements involved, as discussed in Section II.

The proposed solution differs from existing work in many ways. With regard to conceptual modeling, the major contrast refers to the automatic generation of the model. Among current modeling techniques, ontology models are the most similar, for “ontologies are essentially descriptions of concepts and their relationships” [13], and the proposed solution emphasizes the concept of relationships. Nevertheless, the idea stands out for two reasons. Despite the expressiveness of ontologies for describing relationships, ontology-based models are not built automatically – creative human effort is necessary to elaborate them. Once they are available, reasoning techniques can be used to make inferences, like identifying a user’s activity [13]. The other motive is that ontology-models are task-independent, whereas the relevance of contextual elements and their relationships are task-dependent. The proposed solution intends to express that the relevance of contextual elements and their relationships specifically concern tasks. Furthermore, its automatic approach improves the systematic, though manual, process for discovering the relationships presented by [15].

IV. PROPOSED SOLUTION

Instead of trying to discover how the contextual elements relate to each other and whether they influence the task to be improved via context awareness, the idea is to provide requirements engineers beforehand with an automatically built conceptual context model that expresses the relevant contextual elements and the relationships among them by analyzing data collected from accessible context sources, including the usage data. These sources will be evaluated in order to identify how the data behaves when a certain task is performed. From that information, the conceptual context model will be built automatically, expressing which contextual elements and relationships are relevant for the task.

Automatic building of a conceptual context model requires formal definition of a meta-model. In addition, such a meta-model, in combination to requirements templates, is expected to provide support to an algorithm for coming up with candidate requirements (to be validated by requirements engineers). Figure 2 illustrates the to-be situation.
This research proposes a process to address the specific discovery of relevant contextual elements and the relationships among them that are significant for the user task. In a high-level description, it comprises three steps: decompose task into metrics, collect and store contextual data, and process data.

The activity of the first step, decompose task into metrics, refers to extracting from the user task one or more metrics that indicate the frequency of success\(^3\) of performing this task. This is a fundamental step. The user task is expressed in natural language; however, if we want to know how the context influences the task, this task must have measurable indicators. Thereby, the task can be observed in quantifiable terms regarding how it varies depending on the context. The input of this activity will be the task to be improved with context awareness; the output, a set of metrics that represents the task. It will be performed manually by requirements engineers.

The set of metrics will serve as input for the next step, collect and store contextual data. When an event triggers the metric, contextual data from all available context sources will be read and stored in association with the referred metric. If the system has synchronous access to all involved context sources, a snapshot of the context data could be taken when the event happens; otherwise, historical, time-stamped data must be filtered when it coincides with the occurrence of the event. The output of this activity will be a dataset containing all contextual data that refers to the input metrics from all context sources.

The final step is called process data. It will take the dataset containing the contextual data as input. This step will comprise two sub-steps. The first will build up a conceptual context model (its first output) that represents the contextual elements and relationships among them that influence the (metrics of the) task to be improved. This step will make use of a conceptual context meta-model, which must be expressive regarding how the contextual elements relate to each other and to the task. The second sub-step will take the conceptual context model and utilize requirements templates to elaborate candidate requirements to be validated by requirements engineers. It is important to emphasize that the activity process data will be carried out automatically; therefore, at the end it is expected that requirements engineers will, near without any effort, benefit from a novel model to improve the elicitation technique to be employed in order to discover new context-aware features.

Each of these three steps composing the process are more likely to be sub-processes (to be narrowed down). All in all, the process as a whole will have one external input (the task) and two external outputs (the conceptual context model and the candidate requirements). Figure 3 illustrates these. The external input and outputs are depicted outside the dashed box.

The step decompose task into metrics triggered a question about the level of abstraction at which the input task must be.

---

\(^3\)Actually, the metric can indicate not only success but rather anything. The requirements engineering strategy could be, for instance, to find out how the context influences the non-fulfillment of a certain user task. The approach will work as well, because the meaning of the metrics does not matter, since the analysis will be done automatically.
Should it really be a task, subtask or even a system functionality? This input must somehow be clearly quantifiable; thus its level of abstraction must comply with this constraint. Existing work on task measurement will be considered.

Concerning the step collect and store contextual data, there are tools for this purpose, such as Google’s Snapshot API\(^4\) and Funf\(^5\). Up to now, however, it cannot be determined yet whether these tools will be enough to deal with multi-sourced contextual data. In smart ecosystems, for instance, numerous applications cooperate and it is expected that a single standalone solution might not be suitable. Additional investigation will be carried out.

Finally, there are some issues concerning the step process data. With regard to the conceptual context meta-model, no meta-model with the desired expressiveness has been found to date. However, it is still not clear whether an entirely new meta-model should be proposed or whether an existing one could be extended or adapted. Whatever the solution will be, it is of fundamental importance that it be extensible and formal, thus it will provide basis for the automation. Concerning the implementation of process data, the employment of machine learning approaches (such as classification and regression trees) seems to be promising. None-the-less, if further investigation reveals that these techniques are not suitable for this problem-solving task, a new algorithm might be considered to analyze whether each contextual element (and the relationships among them as well) influences the task or not.

**D. A Word about the Identification of Contextual Elements**

Notice that analyzing the relationships among contextual elements requires previous identification of contextual elements. Such identification might not be task-specific, but rather domain-specific.

There is a whole universe of existing contextual elements. Within this universe, there is a set of contextual elements that concern a certain domain of interest. This set is called identified context, whereas the excluded contextual elements are called undiscovered or unvalued context. Within the identified context set, there is a subset of contextual elements that are feasible to sense, which is named accessible context. Within this subset, there is a subset called relevant context, which contains the contextual elements that are considered relevant for a specific task of interest. Figure 4 illustrates these sets of contextual elements and exemplifies possible meaningful relationships among them.

This proposal argues that all accessible contextual elements must be taken into account during the analysis, due the difficult in clearly determining the relevance of all contextual elements with regard to a certain task – although the relevance of the contextual elements depends on the task. Thus, relationships will be sought among all accessible contextual elements.

---

\(^4\)“You can use the Snapshot API to get information about the user’s current environment.” [https://developers.google.com/awareness/android-api/snapshot-api-overview](https://developers.google.com/awareness/android-api/snapshot-api-overview)

\(^5\)“The Funf Open Sensing Framework is an extensible sensing and data processing framework for mobile devices.” [http://funf.org](http://funf.org)

---

**V. RESEARCH METHOD**

The research method to be applied is *Design Science Research*, which centers on the incremental production and assessment of artifacts [22], so the solution idea will evolve until it can be properly evaluated.

The research hypotheses are:

**H1:** Requirements engineers discover a larger number of delightful context-aware features if they know in advance the relevance of the relationships among contextual elements and tasks.

**H2:** Requirements engineers discover delightful context-aware features faster if they know in advance the relevance of the relationships among contextual elements and tasks.

**H3:** Novice requirement engineers discover new delightful context-aware features as often as experienced requirement engineers if they know in advance the relevance of the relationships among contextual elements and tasks.

**H4:** Automatically formulated candidate requirements are perceived as delighters by the stakeholders.

The action plan covers the following steps: (1) defining the problem, (2) designing the solution, and (3) evaluating the solution. So far, the literature review has supported the problem definition and ongoing research has helped to outline the solution. Additional investigation is still necessary to determine whether existing conceptual context meta-models can be exploited. Similarly, the continuing literature review is expected to confirm whether machine learning techniques might be suitable for the intended conceptual context model or whether a tailored algorithm should be developed. Once these steps have been accomplished, the actual solution will be built. Figure 5 summarizes the timetable.

As of now, a two stage evaluation plan is foreseen. First, a controlled experiment will be performed where the participants will be asked to come up with context-aware features for a task performed with the support of a certain application. There will be one factor – the availability of the proposed conceptual
context model – and two treatments – using the model or not. This experiment will quantitatively evaluate whether the requirements engineers’ task of discovering unexpected context-aware features was improved in terms of efficiency and effectiveness (see H1, H2 and H3).

Second, a real industry case study will be carried out to double-check the results of the controlled experiment results in a qualitative manner. Furthermore, as this research claims to provide stakeholders with delightful context-aware features, a qualitative evaluation shall be employed to check whether such context-aware features (which will emerge either from automatically formulated candidate requirements or from those discovered by requirements engineers triggered by the conceptual context model) are actually perceived as delights by stakeholders (see H4).

VI. CONCLUSION

This paper presented the ongoing research regarding the problematic elicitation of delightful context-aware features. Coming up with delights is a continuous goal for competitors in the mobile software industry in order to attract and keep users, but although context awareness is perceived as a way to surprise and amaze users, the complexity of understanding the relevance of contextual elements and the relationships among them has prevented requirements engineers from succeeding in this endeavor.

The solution idea addresses this issue with a data-based approach: A process is proposed that will take the task to be improved, decompose it into metrics, collect and store contextual data from all accessible context sources, and process this data; this will be followed by the automatic generation of a model and on requirement templates, an algorithm will formulate candidate requirements to be validated by requirements engineers.

The next steps are aimed at the formalization of the meta-model and the design of the step process data in the proposed process. After that, the evaluation plan will be fine-tuned.

ACCEPTED PAPERS


ACKNOWLEDGMENT

This work has been done with the support of CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico), Brazil.

REFERENCES
