Rodrigo de Andrade Maués

Keep Doing What I Just Did: Automating Smartphones by Demonstration

DISSERTAÇÃO DE MESTRADO

Dissertation presented to the Programa de Pós-Graduação em Informática of the Departamento de Informática do Centro Técnico Científico da PUC-Rio, as partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor: Simone Diniz Junqueira Barbosa

Rio de Janeiro
March 2014

Rodrigo de Andrade Maués

Keep Doing What I Just Did: Automating Smartphones by Demonstration

Dissertation presented to the Programa de Pós-Graduação em Informática of the Departamento de Informática do Centro Técnico Científico da PUC-Rio, as partial fulfillment of the requirements for the degree of Mestre em Informática.

Rio de Janeiro, March 14th, 2014

Rodrigo de Andrade Maués

Rodrigo Maués is a graduate student in the Pontifical Catholic University of Rio de Janeiro. His research interests include human-computer interaction, cross-platform applications, mobile interaction, and ubiquitous and context-aware computing. Currently he is investigating how to evaluate cross-platform applications under the perspective of Semiotic Engineering and how to ease end-user automation of smartphones using a programming-by-demonstration approach. He received his BSc in Computer Science from the Federal University of Amazonas.

Bibliographic data

Maués, Rodrigo de Andrade

Keep Doing What I Just Did: Automating Smartphones by Demonstration / Rodrigo de Andrade Maués; advisor: Simone Diniz Junqueira Barbosa – 2014.

97 f. : il. (color.); 30 cm

Dissertação (Mestrado em Informática) – Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2014.

1. Informática – Teses. 2. Programação por Demonstração. 3. Interação Humano-Computador. 4. Computação Móvel. I. Barbosa, Simone Diniz Junqueira. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Acknowledgments

First and foremost, I would like to thank my advisor, Simone, who has been a role model for me and has provided endless resources that enabled me to carry out not one, but two different researches throughout my graduate studies. Countless times I felt better and motivated after talking to her. Thank you for always keeping your door open, Simone.

I would like to thank my friends, the new ones and specially the old ones, who gave me words of wisdom or at least helped me to relax and laugh during this journey.

I would like to thank all the participants of the studies. The research presented in this dissertation would have been impossible without them.

I cannot forget to thank my friends and colleagues at Petrobras, who gave me the opportunity to learn HCI beyond the theory, and who supported me in my graduate studies whenever I needed.

I would also like to thank the National Council for Scientific and Technological Development (CNPq) for the support to this research work.

Finally, I cannot even begin to express enough gratitude towards my parents, for their continuous love, care, support and endless encouragement to pursue my dreams.

Abstract

Maués, Rodrigo de Andrade; Barbosa, Simone Diniz Junqueira. **Keep Doing What I Just Did: Automating Smartphones by Demonstration.** Rio de Janeiro. 2014. 97p. MSc. Dissertation – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Smartphones have become an integral part of many people's lives. We can use these powerful devices to perform a great variety of tasks, ranging from making phone calls to connecting to the Internet. However, sometimes we would like some tasks to be performed automatically. These tasks can be automated by using automation applications, which continuously monitor the smartphone's context to execute a sequence of actions when an event happens under certain conditions. These automations are starting to get popular with end users, since they can make their phones easier to use and even more battery efficient. However, little work has been done on empowering end users to create such automations. We propose an approach for automating smartphone tasks by retrospective demonstration. Succinctly, we consider the logic behind the approach as "keep doing what I just did": the automation application continuously records the users' interactions with their phones, and after users perform a task that they would like to automate, they can ask the application to create an automation rule based on their latest recorded actions. Since users only have to use their smartphones, as they would naturally do, to demonstrate the actions, we believe that our approach can lower the barrier for creating smartphone automations. To evaluate our approach, we developed prototypes of an application called Keep Doing It, which supports automating tasks by demonstration. We conducted a lab user study with the first prototype to gather participants' first impressions. The participants created automation rules using our application based on given scenarios. Based on their feedback and on our observations, we refined the prototype and conducted a five-day remote user study with new participants, who could then create which and how many rules they wanted. Overall, the findings of both studies suggest that, although there were some occasional inaccuracies (especially when demonstrating rules that contain conditions), participants would be willing to automate smartphone tasks by demonstration due to its ease of use. We concluded that this approach has much potential to aid end users to automate their smartphones, but there are still issues that need to be addressed by further research.

Keywords

Mobile computing; context-aware systems; ubiquitous computing; human-computer interaction; end-user development; programming by demonstration; smartphone automation

Resumo

Maués, Rodrigo de Andrade; Barbosa, Simone Diniz Junqueira. **Keep Doing What I Just Did: Automatizando Smartphones por demonstração.** Rio de Janeiro. 2014. 97p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Smartphones se tornaram uma parte integrante da vida de muitas pessoas. Podemos usar esses dispositivos para executar uma grande variedade de tarefas, que vão desde dar telefonemas a acessar a Internet. No entanto, às vezes gostaríamos que algumas tarefas fossem executadas automaticamente. Estas tarefas podem ser automatizadas usando aplicativos de automação, os quais monitoram continuamente o contexto do *smartphone* para realizar um conjunto de ações quando um evento acontece sob certas condições. Estas automações estão começando a ficar popular entre os usuários finais, uma vez que elas podem tornar os celulares mais fáceis de usar e ainda mais eficientes no uso da bateria. No entanto, pouco foi feito para capacitar os usuários finais a criar tais automações. Propomos uma abordagem para automatizar tarefas de *smartphones* por demonstração retrospectiva. Sucintamente, consideramos a lógica por trás da abordagem como sendo "continue fazendo o que eu acabo de fazer" : a aplicação de automação grava continuamente as interações dos usuários com seus telefones, e depois que os usuários realizaram uma tarefa que eles gostariam de automatizar, basta eles pedirem à aplicação para criar uma regra de automação com base em suas últimas ações gravadas. Como os usuários só têm que usar seus *smartphones*, como eles naturalmente fariam, para demonstrar automações, acreditamos que a nossa abordagem pode reduzir a dificuldade na criação de automações de *smartphones*. Para avaliar a nossa abordagem, desenvolvemos protótipos de um aplicativo chamado Keep Doing It, que permite automatizar tarefas por demonstração. Realizamos um estudo em laboratório com o primeiro protótipo para coletar as primeiras impressões dos participantes. Os participantes tiveram que criar regras de automação usando o nosso aplicativo com base em cenários pré-determinados. Baseado no *feedback* dos participantes e na nossa observação, nós refinamos o protótipo e realizamos um estudo remoto de cinco dias com novos participantes, os quais desta vez podiam criar quais e quantas regras quisessem. No geral, os resultados de ambos os estudos sugerem que, embora houvesse algumas ocasionais imprecisões (especialmente ao demonstrar regras que contêm condições), os participantes gostariam de automatizar tarefas de *smartphones* por demonstração devido à facilidade de uso. Concluiu-se que esta abordagem tem muito potencial para ajudar os usuários finais a automatizar seus smartphones, mas ainda há questões que precisam ser abordadas em pesquisas futuras.

Palavras-chave

Computação móvel; sistemas sensíveis ao contexto; computação ubíqua; interação humano-computador; desenvolvimento por usuário final; programação por demonstração; automação de *smartphones*

Table of Contents

List of Figures

List of Tables

1 Introduction

Smartphones have become part of our everyday life. They are rapidly evolving and nowadays we can use these powerful devices to perform a great variety of tasks, ranging from making phone calls to connecting to the Internet. Task automation, *i.e.* letting the device perform some repetitive tasks automatically for their owners, is a good way to improve the user experience (Antila et al., 2012; Ravindranath et al., 2012) and to reduce the effort required to use a device. These tasks can be automated by using automation applications, context-aware applications that continuously monitor the devices' context to perform a set of actions whenever a predefined event happens under certain conditions. An example of an automated task would be to launch the music application (action) when leaving home (event) if a wired headset is connected (condition).

Smartphone automation applications are starting to get popular with end users (Ravindranath et al., 2012), since they can make their phones easier to use, smarter, and even more battery efficient (e.g., automatically turning off the GPS or reducing the brightness of the screen when running out of battery). However, little work has been done on empowering end users to create automations for their smartphones. Most of the existing work makes such useful applications accessible only for users with programming skills. In general, there are ready-made rules available either on the application itself or on the web, but according to Dey et al. (2006), for the successful development of ubiquitous and context-aware applications, end users should be empowered to build their own applications. Dey argues that end users should be empowered because they have more in-depth knowledge about their individual activities and environments than any developer (Dey et al., 2004). Besides, if only a developer can control system behavior, the user will be unable to evolve the system when her environments, activities or needs change.

Currently, aside from manually writing the code, which is not interesting for end users who have little or no programming experience, the most common (although inefficient) option is to manually select predefined conditions and actions to configure the automation rules (Häkkilä et al., 2005; Ravindranath et al., 2012). Most of the commercial smartphone automation solutions nowadays rely on the latter approach. However, this “mixing and matching” approach is inefficient and frustrating, since it requires users to create rules from scratch, it involves a number of steps during the setup process, and the user has to find and select items from long lists of options in a small screen size.

A less common approach, unlike the aforementioned ones, takes advantage of context monitoring not only to detect and run the automations but also to assist the user in creating them in the first place. This approach consists of the analysis of long-term behavior of individuals from smartphone usage data to recommend automations based on discovered usage patterns (Antila et al., 2012). Therefore, it assists the user to automatically create rules by analyzing how the user actually uses her or his smartphone. Although helpful, this approach does not allow users to create any automation they want in a certain moment, since it focuses on recommending only automations of tasks done on a regular basis by the user. However, we believe there are many occasional tasks (which are not detected by this approach because they are not performed often enough) that users may wish to automate. Moreover, we believe it is exactly because they often do not remember or do not want to invest time in doing such tasks that they wish these tasks were automatic in the first place.

In this dissertation, we explore an approach for programming smartphone automations by retrospective demonstration. In this approach we concentrate in analyzing a given sequence of user actions to identify possible relations between them that may constitute a desired automation rule, regardless of whether these relations happen frequently during the sequence, *i.e.* if they represent a pattern. A large number of users are familiar with smartphones today, while still most of these people still do not have any programming skills. So we believe that our approach can lower the barrier for end users to create smartphone automations since users only have to use their smartphones, as they would naturally do, to demonstrate actions to be automated.

In order to explore these ideas we have developed an Android application called Keep Doing It, which enables end users to automate their smartphones in situ by recommending automation rules based on their own smartphone’s recent usage history. Keep Doing It continuously collects a user’s interactions with her or his smartphone. After users have performed a task that they consider worthy of automation, they can ask our application to look back in the history and to analyze a sequence of their recorded actions as a demonstration of what they intend to automate. Succinctly, we consider our approach as “keep doing what I just did.” We focus mainly on standalone automation applications, but we believe that the technique and lessons presented here can be readily extended to the end-user development of more complex context-aware mobile applications.

1.1 Research Goals and Methodology

In this dissertation we investigate the feasibility of using programming by demonstration to create smartphone automation rules. Our goal is to support end users to automate tasks in their smartphones. Given this scenario, we first posed the following research question:

- RQ1: What do users think about automating their smartphones by demonstration after using our first prototype of Keep Doing It?

To answer this research question, we conducted a lab user study with 10 participants to investigate their first impressions and whether they could understand our approach and create rules by demonstration. Participants were instructed to create a set of specific automation rules using the first Keep Doing It prototype version. This study served as early feedback and an opportunity to alter the user interface or other aspects of the prototype before conducting a study with a larger audience. From this first study, two additional research questions emerged:

- RQ2: How easily and well can users build context-aware rules with Keep Doing It?
- RQ3: What are the expected benefits and limitations of automating smartphones with Keep Doing It?

To address these two research questions, and taking into account that user needs and real context of use may have a large impact on the experience of using a mobile context-aware application, we decided to evaluate our approach in a more realistic setting by conducting a remote user study. We asked 20 Android users to create as many and whatever rules they wanted for five days using Keep Doing It installed on their personal smartphones. After each session with the tool, they had an opportunity to provide structured feedback via an in-app questionnaire. Their usage of the tool was also logged and analyzed. At the end of the study, we asked them to fill out another survey that directly addressed the research questions.

1.2 Dissertation Outline

This dissertation is organized as follows. In Chapter 2, we give an overview about context-awareness, end-user development and programming-by-demonstration, and we discuss related work in end-user development of context-aware applications and mobile applications. In Chapter 3, we describe the user interface and the implementation details behind the Keep Doing It application. In Chapter 4, we discuss the preliminary lab user study we conducted with the first prototype version of the Keep Doing It application and our findings. In Chapter 5, we discuss the refinements made to Keep Doing It and the findings of the remote user study conducted with this new prototype. Finally, in Chapter 6, we wrap up this dissertation discussing its main contributions and opportunities for future work.

1.3 Publications

The first prototype version of the Keep Doing It application and its lab evaluation, which are discussed in this dissertation, have been published as a long paper at the 15th ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI 2013) (Maués & Barbosa, 2013b). We also presented a discussion about how the automation by demonstration approach may as well make smartphone automations more accessible for end-users with disabilities (as the elderly and the blind) at IFIP INTERACT 2013 Workshop on Rethinking Universal Accessibility: A broader approach considering the digital gap (Maués & Barbosa, 2013a).

2 Background

2.1 Context-Awareness

Context-awareness has many different definitions according to the area of study. In the definition we adopt in this study, which is used in ubiquitous or pervasive computing (Robles & Kim, 2010), context-awareness refers to a general class of systems that can sense their physical environment or infer a user's current situation, and adapt their behavior accordingly. Dey (2001) defines context as "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves."

Context-awareness in mobile computing has in the past been widely exploited to provide more usable mobile devices and services by automatically adapting content and presentation of services to be more relevant or accessible for each individual user and his or her current context of use. Location-based services are one of the most popular context-aware applications.

In this dissertation, we focus on smartphone automation or tasking applications, which enable the user to create context-aware tasks, *i.e.* tasks that will automatically be executed according to the current context.

2.2 End-User Development

End-user development (EUD) or End-user programming (EUP) is a research topic within the field of computer science and human-computer interaction (Lieberman et al., 2006), describing approaches or techniques that make programming computers easier for end users (*i.e.* who are not expert in programming).

Certainly we should not expect the average user to write programs with the same ease as professional programmers. In addition to the knowledge a programmer must have about details and notations of a particular programming language before even starting to code, an aspect that makes programming hard for end users is that they are not used to creating abstract plans without feedback (Blackwell, 2002). The programmer does not see the effects of the program he is writing until he runs the program; he must mentally simulate them instead.

Therefore, in order to make end-user programming easier, we must reduce the amount of detail the user must know before programming, and we must reduce the mental overload necessary to create the plan that is the program. These problems have been addressed with different kinds of EUD techniques (Paternò, 2013). Because there is always some effort to learn a EUD technique, the users' motivation depends on their confidence that it will empower their work, save time on the job or raise productivity. Next we present the programming-by-demonstration approach, since our work is based on this EUD technique.

2.2.1. Programming-By-Demonstration

Programming-by-demonstration (PbD) or Programming-by-example is an EUD technique in which the user creates a program by demonstrating what it should do with a concrete example (Cypher & Halbert, 1993; Lieberman, 2001). The system records the sequence of actions or commands performed by the user and then it tries to infer what kind of script or program the user intended to create. Succinctly, programming by example is “do what I did.” This method is a simple and very natural programing interface for humans since they often transfer skills between themselves by showing how something is done.

The learning curve of this technique is very low since the user works with the system that he is already familiar in the similar way as when she or he is normally/ordinarily performing a task. Besides, since users are programming directly in the user interface of the system, they can visualize more easily the outcome of their actions as if they were debugging while coding.

The simplest form of PbD is macro programming, where the generated script corresponds exactly to the recorded sequence of actions. However, usually through an inference process a PbD system generalizes from the demonstrated actions to a robust program that is more likely to work in different situations.

The inference process of PbD systems may rely either on AI techniques (such as machine learning algorithms) or on a set of heuristics. Either way, this inference is subject to errors and the created program may not reflect the user's true intentions (Chen & Weld, 2008; Dey & Mankoff, 2005). That is why usually such systems allow users to edit the created programs, if necessary, in order to correct any mistakes made by the system or to supply any missing information.

2.3 Related Work

The following related work mainly encompasses research in EUD of context-aware applications and EUD of mobile applications. So far little research has been dedicated to the latter, which is not surprising as mobile devices had until recently limited capabilities and supported few advanced functionalities.

One of the major goals identified by Dey et al. (2006) for the successful development of ubiquitous and context-aware applications is that both designers and end users should be empowered to build their own applications. End users should be empowered because they have more in-depth knowledge about their individual activities and environments than any developer (Dey et al., 2004). Besides, if only a developer can control system behavior, the user will be unable to evolve the system when her or his environments, activities or needs change. Still, so far most of the research on context-aware applications focuses on hardware and infrastructure instead of supporting the rule creation process.

One option to support automation rules creation is using visual languages, in which users need to design the logical structure of the rule using graphical instead of textual notation. One example is the iCAP project (Dey et al., 2006), which allows end users to configure context-aware applications in a fully visual environment. Although this technique does not require actual programming or syntax knowledge, users need to understand the programming paradigm of a tool and the logic of how rules are created. Another approach is based on a metaphor called Magnetic Poetry (Truong et al., 2004): Users can see all possible elements of the rule (objects, actors, actions, and conditions) and then construct a verbal rule using those elements, similar to magnets on a fridge. Although their evaluations have shown that their approach is considered intuitive and easy to use, the scope of rules it allows to create is rather narrow. More critically, it is inefficient since the larger the amount of possible elements to select, the more difficult it is to locate the desired ones to compose the rule. When it comes to mobile devices, even lists with few items can be a burden due to the limited size of the screen.

Regarding the mobile environment, Locale[1] and Tasker,[2] as most of the commercial smartphone automation solutions nowadays, rely on an inefficient approach of manually selecting predefined conditions and actions to configure the automation rules. This is the same approach used in one of the first researches to support users in specifying context-action rules for automating mobile phone tasks (Häkkilä et al., 2005). Both the CITA system (Ravindranath et al., 2012) and Microsoft's on{x} application[3] for Android work with high-level triggers (e.g., "is running") that end users can easily understand, and allow developers to remotely program complex smartphones automations in JavaScript. However, while CITA also allows end users to create automations by manually composing them, the only alternative to coding that on{x} provides to end users are ready-made rules called "recipes" that are available on its website. Aside from smartphone automation, IFTTT[4] focuses on automating Internet services and also uses the term recipes to refer to the automation rules. Unlike the previously discussed applications, IFTTT allows users to select very application-specific triggers (e.g. "If I check in on Foursquare") and actions (e.g. "then create a status message on Facebook") to compose their automation rules.

Some researches and tools target the development of more generic mobile applications by end-users. Microsoft TouchDevelop (Tillmann et al., 2011) represents an emerging development model for mobile applications. It aims to lower the end-user programming barriers by enabling on-device app development and providing a fairly simple scripting language. With TouchDevelop, anyone should be able to program his or her mobile device directly on this device. MicroApp (De Lucia et al., 2012) and Puzzle (Danado & Paternò, 2012) follow the same premise of allowing users to program applications directly on their phones, but using jigsaw programming to assist them to compose their applications. It is a visual language based technique where program constructs are

[1] http://www.twofortyfouram.com/
[2] http://tasker.dinglisch.net/
[3] http://onx.ms/
[4] http://ifttt.com/

represented using icons that look like jigsaw pieces, and only icons that fit together can be composed to form legal programs.

Our work is based on the programming-by-demonstration (PbD) end-user development technique. The idea of using PbD to create automation rules in context-aware system is not new and has been mentioned in the past as an interesting solution to investigate (Häkkilä et al., 2005). It has also been implemented in a smart environment prototyping tool named a *CAPpella* (Dey et al., 2004), which employs machine learning for generalizing rules from user actions and requires small amounts of time and data to produce reasonable activity recognizers. In contrast, our application inference process is based on heuristics, and therefore it requires only one demonstration of a task in order to infer automation options.

However, as far as we know, Keep Doing It is the first smartphone automation application to use the ideas underlying PbD. Moreover, despite the great amount and variety of systems and platforms that use PbD approaches (Cypher & Halbert, 1993; Lieberman, 2001), we believe our research is the first to use this approach on a smartphone and one of the few that is not restricted to automating tasks within one application but instead it is placed at the operating system level.

Although there has been a body of research on discovering patterns from smartphone usage data, we are aware of only one work that focuses on using the detected patterns to automate smartphones (Antila et al., 2012). However, the Android application that they developed, called RoutineMaker, was capable only of launching applications according to location and time, while our application takes into account a wider range of triggers and actions.

Regarding home automation, a system, called PUBS (Patterns of User Behavior System), implements an algorithm named APUBS to discover user's common behaviors and habits from data recorded by sensors in Ambient Intelligent environments (Aztiria et al., 2008). Although our application does not focus on discovering patterns, our inference algorithm is partially based on APUBS. PUBS also contains an interaction system, which allows end users to verbally mediate the detected patterns prior to transforming them in automations by accepting, refining, or deleting them. Support for an even more natural and flexible speech interaction is presented yet in (Lucas Cuesta et al., 2010).

3 Keep Doing It Application

In this chapter we describe the major aspects of the design and architecture of *Keep Doing It*. First, we explain the recording of the user's interactions with her or his smartphone. Next, we describe the inference process that recommends task automations based on the part of the recorded data chosen by the end user as the demonstration. Then we discuss how our application presents the automation rules, the visual strategies implemented in order to reduce choice effort and enhance the system intelligibility. Finally, we describe the strategies implemented in order to handle the uncertainty of the recommendations.

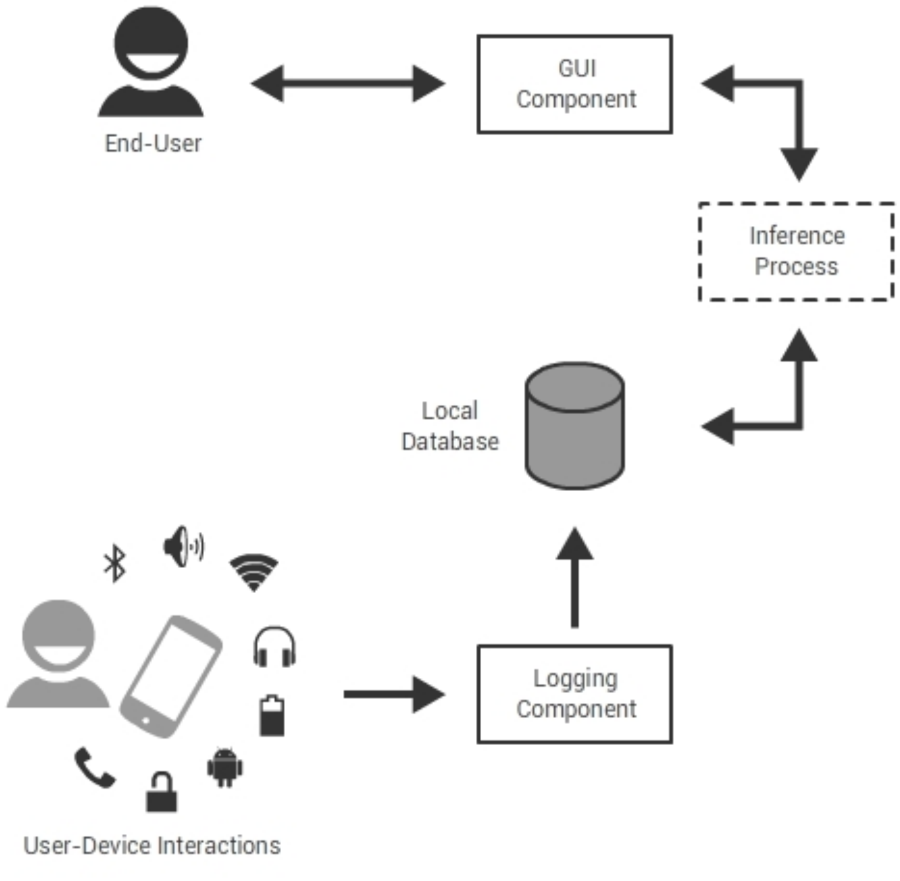


Figure 1: The architecture of Keep Doing It

Keep Doing It has been designed to run on any Android operating system version 2.3 or above. The application architecture is illustrated in Figure 1 and Figure 2 shows the application user interface. During the course of our research we refined the first prototype version after our initial findings (discussed in Chapter 4), so the version presented in this chapter is the last one developed, known as Keep Doing It 2.0.

3.1 Recording User-Device Interactions

According to Lieberman (2001), an ideal recording mechanism for a programming by demonstration system will be unobtrusive, so that users can demonstrate tasks under conditions identical to their standard use, and detailed enough to support reasoning about the user's intent. Automation applications, by their very nature, always have a component continuously running in the background of a device, listening for events that might trigger an automation rule. In our application, however, we use an Android service not only to listen to events but also to record them for further analysis (the Logging Component in Figure 1).

An event can be related either to a user-device interaction or to context information (Aztiria et al., 2008), where context is "any information that can be used to characterize the situation of an entity" (Dey, 2001). The user has little or no control over context information (e.g., time, location, battery status). Thus, we focus on recording events related to interactions (direct manipulations), given they are more closely linked with the user's explicit and intended actions. Still, it is important to save the context information associated with each interaction, since the user might implicitly be considering it as important to compose the automation rule he demonstrated (*e.g.*, the user may want his current location to trigger the action that he executed). Keep Doing It supports so far a limited variety of simple interactions (low granularity) that can be recorded: changing the ringer mode, toggling Wi-Fi and Bluetooth, making a call, unlocking the phone, connecting a wired headset or a charger to the phone, and opening an application. Regarding context information, so far Keep Doing It supports only a specific location (not a range) and an instant of time (not a period of time or a specific day of the week).

Mentioning context-aware computing raises almost automatically serious privacy issues (Lane et al., 2010). Therefore Keep Doing It does not take into account the actions performed within an application (e.g., checking in a place on foursquare), since users may have privacy concerns regarding this kind of fine granularity information. Also, in order to respect users' privacy, no information regarding their identity or personal information is stored, and all the stored data is processed locally on the device, is private to the application and is erased automatically when the application is uninstalled.

As database we used SQLite, an Open Source Database that is embedded into Android by default. The interaction events are stored in a specific table within the database called INTERACTIONS_TABLE. Every record in this table contains the type of interaction (e.g., outgoing call), a timestamp, a value (e.g., the phone number called) or two (e.g., the contact's name) corresponding to the interaction event. The device's current context information (e.g., location) is stored in a similar table called CONTEXTS_TABLE with an extra field that references the interaction in the INTERACTIONS_TABLE to which each context is related. We decided that the fields in both tables should be as generic as possible to make supporting new interaction and context information types easier in the future.

3.2 Inferring Automation Rules from Interactions

In this section, we detail the Keep Doing It inference process (Figure 1), which is based on a set of heuristics instead of machine learning.

As stated in the previous section, data about the user's interactions are continuously collected, creating a log of sequences of interactions. When users decide to automate a task that they just performed (either naturally or forcing a demonstration), they simply need to launch the Keep Doing It application (GUI Component in Figure 1), press the button with the + symbol on the top of the screen, and then a list of recommendations will be generated (Figure 2). In case the application is already open on the recommendations list screen, they only need to press a button on the top of the screen to refresh the list. Alternatively, the list will be refreshed in case they select a time window different from the last one selected. The time window defines from how long ago the application should take into account the actions performed by the user. Available options are from 15, 30, 45 or 60 seconds ago. No recommendation will be shown in case the user did not perform any action that could be automated within the chosen time window.

The sequence of user actions taken from the log does not necessarily tell Keep Doing It what the user is really trying to automate. So we developed an algorithm to organize and add meaning to this sequence in order to try to infer the user's intentions and derive the desired task automation rule.

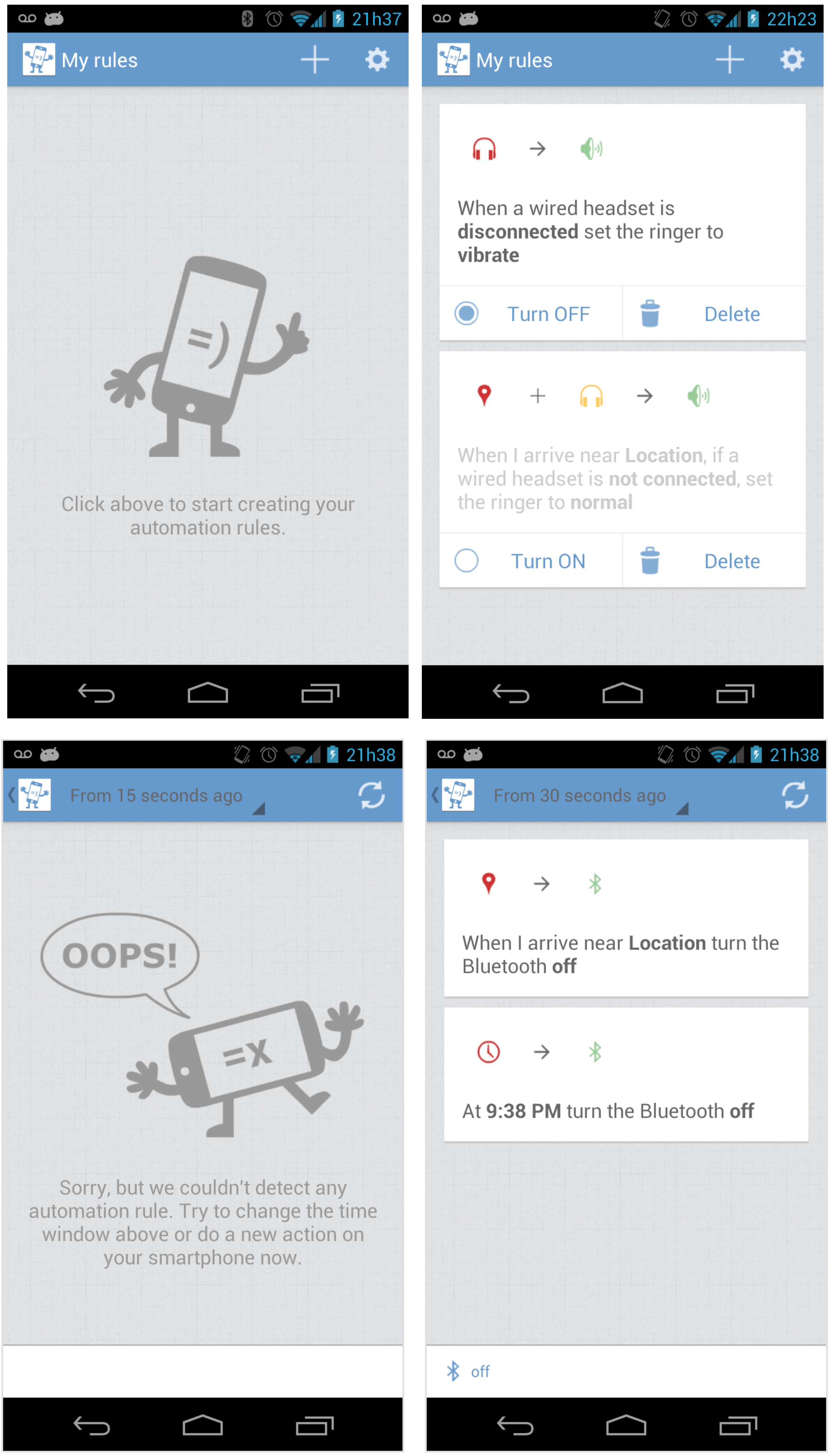
The generated rules follow the well-known ECA (Event-Condition-Action) rules format (Aztiria et al., 2008), where each part of the rule has its role: there is an event, the main factor responsible for triggering an action; zero or multiple conditions, which will determine whether the action should be triggered once the triggering event has happened; and one or more actions, which compose the task being automated.

Our inference algorithm is split in two main steps: grouping interactions and generating possible combinations.

3.2.1. Grouping Interactions

This first step of the inference algorithm creates subsets of the input sequence of user-device interactions, from which the next step will later generate possible rules. First, the algorithm selects from the input sequence interaction events that can be classified as actions in the ECA rule format, *i.e.*, events that can be reproduced by the system without user intervention. For instance, application launch, ringer mode switching, Bluetooth and Wi-Fi toggle and outgoing calls are all eligible events, while connecting a wired headset or a charger to the phone are not, since the action of connecting peripherals to the phone cannot be automated. We define these eligible interactions as main actions. If no main action is found in the input sequence, then the inference process stops and the user is informed that there are no recommendations for the given time window (Figure 2).

Although we believe the task being automated is related to the most recent main action in the original input sequence, we cannot be sure of that. That is why we consider other main actions as well. However, when assembling the final recommendation list, we give priority to the rules generated for each main action in reverse chronological order.



Click above to start creating your automation rules.

When a wired headset is **disconnected** set the ringer to **vibrate**

When I arrive near **Location**, if a wired headset is **not connected**, set the ringer to **normal**

OOPS!

Sorry, but we couldn't detect any automation rule. Try to change the time window above or do a new action on your smartphone now.

When I arrive near **Location** turn the Bluetooth **off**

At **9:38 PM** turn the Bluetooth **off**

Figure 2: Screenshots of Keep Doing It 2.0, showing (from left to right, top to bottom): initial screen with and without saved rules; internal screen with possible results of a user's request to automate a task

After defining the main actions, the application associates for each one of them a limited number of interactions (we empirically defined it to be 3) from the input sequence that are not from the same type of the main action and that happened immediately before it within the given time frame. Also, in each group there cannot be two associated interactions of the same type; only the closest one to the main action is included, since this one defines the latest state of the system regarding that feature. For instance, let us consider a main action whose type is not toggling the Wi-Fi. If in the sequence of interactions that happened immediately before this main action there is an event of turning the Wi-Fi off followed (not necessarily immediately after) by another event of turning the Wi-Fi on, only the latter will be included in the list of associated interactions for that main action.

Once each group of main actions and its list of associated interactions are created, they are all sent to the second step of our algorithm, where the role of each interaction in the associated list will be defined and some of the possible ECA rules will be generated.

3.2.2. Generating Possible Combinations

Inferring a user's intended automation rule is prone to uncertainty (Chen & Weld, 2008; Dey & Mankoff, 2005). A sequence of interactions may be misinterpreted. For instance, when the user toggles the Wi-Fi on and then launches the music player, we could infer a rule where the former interaction is the triggering event and the latter is the action. However, for some users this interpretation might not hold — for example, the user might actually be considering that both interactions constitute the task, and that some context information (e.g., time or place) is the triggering event for both of them. So the goal of this second step is to generate some of these possible interpretations to let the user choose the one she or he believes is the most adequate later. The generated possibilities do not intend to be exhaustive, since a long list would impair the user to find the correct one even if it was among the available options. When the algorithm fails to provide the desired option, we must employ some remedial mechanisms (Chen & Weld, 2008; Dey & Mankoff, 2005), as we describe in section 3.4.

The amount of suggestions generated for each main action depends mainly on the size of the associated interactions list. We may suggest up to four rule variations, which are: event + action, event + actions, event + condition(s) + action, and event + condition(s) + actions. In case of combinations from different groups of interactions resulting in duplicated automation rule recommendations, these duplicates are deleted prior to presenting the recommendations to the user.

Unconditional automation rules should be preferred over conditional ones (Chen & Weld, 2008). Therefore, given a main action, at least the most basic variation (event + action rule) is always generated. The main action is the action itself and, if there is at least one associated interaction, then the one with the smaller time distance from the main action will be considered the triggering event. If no associated event is found within the given time frame, the system considers that the triggering event must be some context data and up to two default rules are generated (so far), one where the current time is the event and the other where the event is the location (if this information is available). If there is more than one interaction in the associated list, then the application can also generate an event + condition(s) + action rule. In this case, the changes in the system state caused by the interactions that precede the triggering event are considered conditions (since they were valid at the time the triggering event occurred). By default, if there is more than one condition, they will be joined by an AND logical operator.

There are two situations when rules with more than one action may also be generated. The first situation is when the interaction defined as the triggering event is also eligible to be an action. In this case, it is added to the action clause along with the main action, and one of the conditions is transformed into the new triggering event. The second situation is when the associated interaction with smaller distance from the main action is only eligible to be an action (e.g., in some automation applications opening an application may be considered only as an action, never as a triggering event) and then the next interaction must be considered as the triggering event (or in case there is none left, the time and location are considered the possible triggering events, as explained earlier). In either way, in those situations it is possible to generate event + actions rules or event + condition(s) + actions (in case there was an interaction left to be considered as a condition).

3.3 Presenting The Rule Recommendations to the User

An important consideration in PbD systems is how the inferred behavior should be represented (Meskens et al., 2010). Providing good feedback about what has been learned from the sample values can facilitate users to understand and eventually correct undesired system behavior (Lieberman, 2001). Therefore, in Keep Doing It, every interaction or context is shown as a combination of a human-readable textual description and a representative icon. The color of the icon changes according to the role of the interaction/context in the rule: the role of a triggering event is represented by a red color, the role of a condition by a yellow color, and the role of an action by a green color. The main reason why we decided to employ this color scheme is to help users to differentiate and identify the rules in the recommendations list. In (Chen & Weld, 2008) a color scheme is also used, but its purpose was to provide feedback to user about the confidence of the system in the predictions.

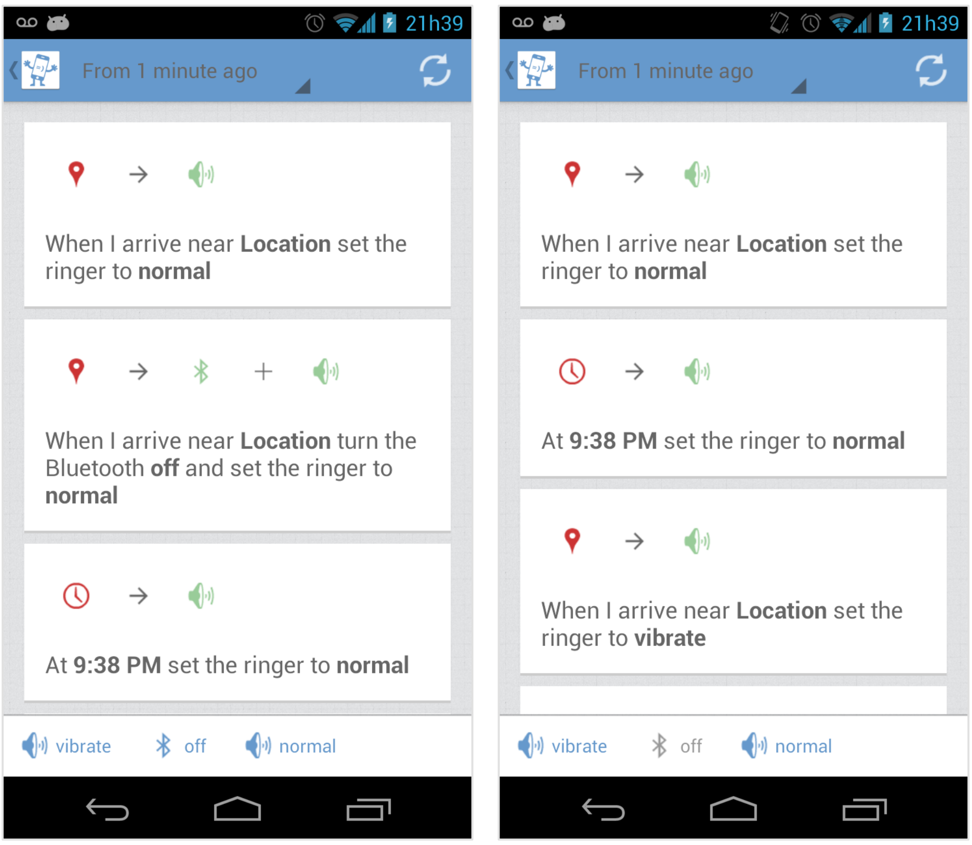
Once the list of automation rules recommendations is ready, Keep Doing It selects the necessary icons from a collection and generates the textual description following the ECA format (WHEN event IF condition THEN action) for each one of them using a template-based approach (Van Deemter et al., 2005). The ECA format is straightforward and easy to understand, even by end users completely unfamiliar with programming (Dey et al., 2006; Holloway & Julien, 2010). Since the user will mainly rely on such descriptions to understand the recommended rules, it is important that they are clear, *i.e.*, free of ambiguity. We manually authored specific templates for each type of interaction or context information. Each variable may have up to three different templates according to its role in the automation rule. For example, if changing the ringer mode is an event, condition or action, the application respectively chooses "(when) I set the ringer to [ringer mode]", "(if) the ringer is set to [ringer mode]" or "set the ringer to [ringer mode]" as the template (parameters on the templates are highlighted in bold to give more emphasis). Our application also coordinates phrases in case there is more than one condition or action (e.g., "(if) the ringer is set to vibrate and a wired headset is connected"). These pieces are then put together in a final template to form the description. For example, for an automation rule where unlocking the phone is the event, a wired headset being connected to the phone is the condition and changing the ringer mode is the action, our application produces the following description: “When I unlock my phone, if a wired headset is connected, set the ringer to vibrate.”

3.4 Handling Uncertainty

As mentioned before, an important part of our application is how it will handle the uncertainties of the inference process (Chen & Weld, 2008) to ensure the task that will be automated is in accordance with end-user’s expectations. One common strategy for dealing with uncertainty in context-aware systems involves user mediation, where the user resolves the uncertainty (Chen & Weld, 2008; Dey & Mankoff, 2005). Only when the mediation process is complete may the system proceed with an action. In our case, the action that proceeds after the mediation is finally saving the demonstrated automating rule.

Our application enables different user mediation strategies according to our inference algorithm's accuracy. The most basic mediation consists of letting the user select one rule from an n-best list (Dey & Mankoff, 2005) of automation rules. This mechanism allows users to resolve the inaccuracy of which one among the recommendations is the one they actually intended to demonstrate.

Eventually the user may perform some unintended interactions along with the intend ones within the chosen time window, which can negatively affect the accuracy as well as increase the number of recommendations. In Keep Doing It, the detected interactions are shown in a bottom bar in the recommendations list screen. By pressing the unintended interactions the user tells the application to regenerate the last recommendations but now ignoring these interactions from the sequence of detected interactions. This way the user does not have to waste time and effort to demonstrate a task again only to remove the unintended interactions. Figure 3 shows an example of a user ignoring the unintended interaction of switching off the Bluetooth. Displaying the performed interactions has the additional benefit of increasing the system’s intelligibility by making clear to the users what was indeed detected by the system.



21h39

From 1 minute ago

When I arrive near **Location** set the ringer to **normal**

When I arrive near **Location** turn the Bluetooth **off** and set the ringer to **normal**

At **9:38 PM** set the ringer to **normal**

vibrate off normal

21h39

From 1 minute ago

When I arrive near **Location** set the ringer to **normal**

At **9:38 PM** set the ringer to **normal**

When I arrive near **Location** set the ringer to **vibrate**

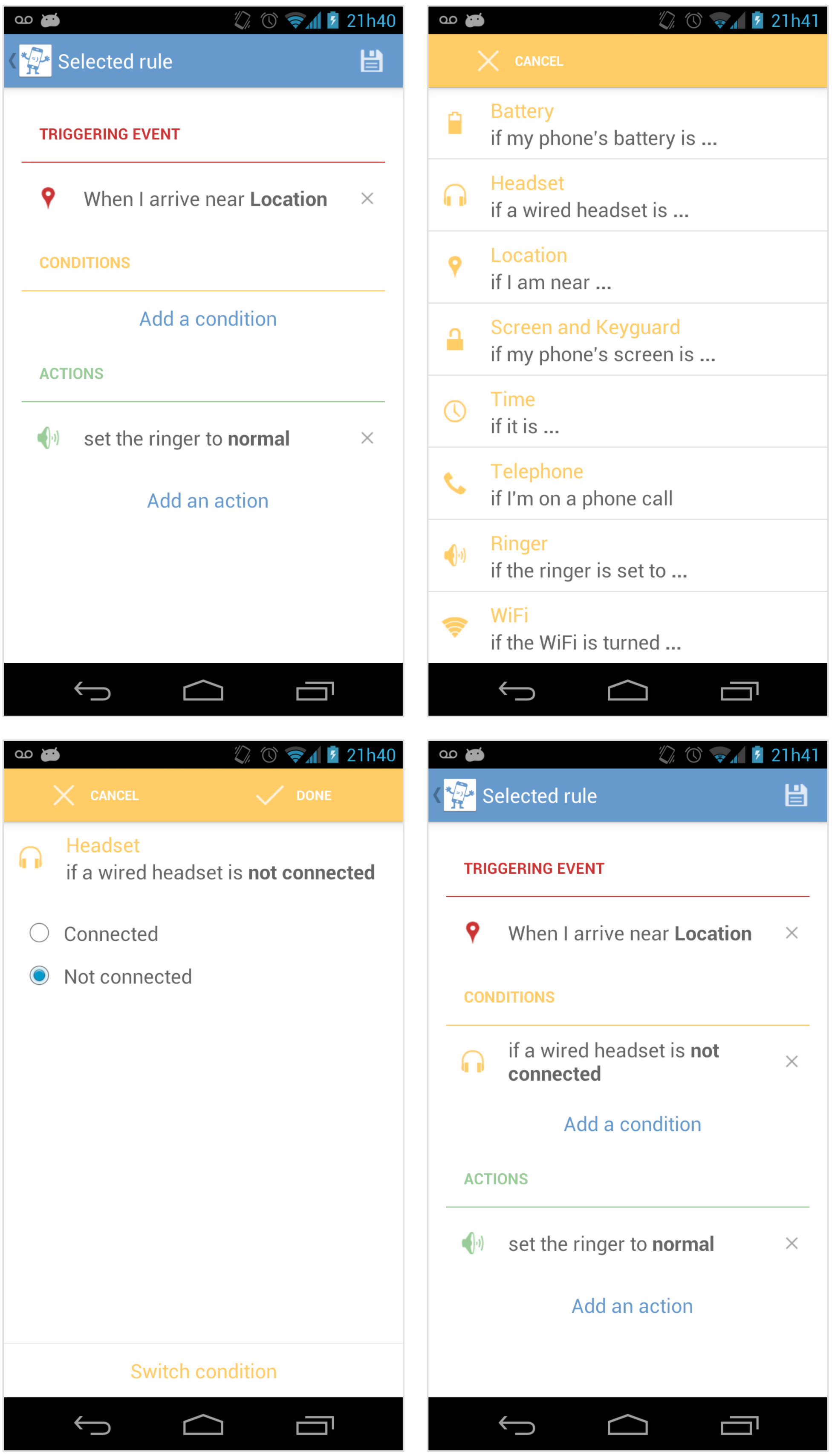
vibrate off normal

Figure 3: Screenshots of Keep Doing It 2.0, showing the possible results of a user's request to automate a task before (left) and after (right) ignoring a detected unintended interaction

Alternatively, the user may also select the recommendation most similar to the intended rule and then manually remove from it the unnecessary additional parts.

Even if our application correctly infers the intended automation rule, users might still need to correct some details prior to automating it. For instance, the GPS inherent inaccuracy may lead to incorrect user locations, so we must allow the user to fix it by choosing a location directly from a map component. Thus, besides the location, in our application the user can change parameters as she or he pleases: time, if a feature is "on" or "off", the ringer mode, the contact whom she or he called, etc. In the future this editing could also include changing the default AND logical operator between conditions to OR if necessary.

The worst-case scenario is when our application fails to infer one or more elements that were necessary for the intended automation rule. In this case, the user must be able to complete the automation rule by adding the missing piece of information, which might be another condition, another action or even a new event. Figure 4 shows the steps taken to edit a selected rule in order to add a missing condition.


21h40

Selected rule

TRIGGERING EVENT

When I arrive near **Location** ×

CONDITIONS

Add a condition

ACTIONS

set the ringer to **normal** ×

Add an action

21h41

CANCEL

Battery
if my phone's battery is ...

Headset
if a wired headset is ...

Location
if I am near ...

Screen and Keyguard
if my phone's screen is ...

Time
if it is ...

Telephone
if I'm on a phone call

Ringer
if the ringer is set to ...

WiFi
if the WiFi is turned ...

21h40

CANCEL DONE

Headset
if a wired headset is **not connected**

Connected

[✓ Not connected]

Switch condition

21h41

Selected rule

TRIGGERING EVENT

When I arrive near **Location** ×

CONDITIONS

if a wired headset is **not connected** ×

Add a condition

ACTIONS

set the ringer to **normal** ×

Add an action

Figure 4: Screenshots of Keep Doing It 2.0, showing the user adding a missing condition to a selected rule being edited

A similar scenario is when our application selects the correct user interactions and context information to compose the intended automation rule, but it fails to define their role properly (e.g., mistakenly switches which interaction or context is the condition and which is the event). In this case the user has to remove the switched interactions and add them again with the correct roles. In the future the user could only switch directly the interactions to their correct role instead of removing and adding them again.

Our application ensures that a rule has up to one event, three conditions and three actions. While per rule indeed there can only be at most one event, the application could support more than three conditions and three actions if we wanted too. In fact, the conditions and actions per rule could be unlimited. However, defining a maximum number made the implementation easier; and we believe that this amount of actions and conditions is more than enough for users, since they usually keep the rules simple (Chen & Weld, 2008). Our application also ensures that the edited rule must have at least the minimal ECA format (*i.e.*, one event and one action) before the user is able to save it. Once the rule is recorded, it can be seen on the initial main screen, where it may be toggled active or inactive, or it may be edited or deleted (Figure 2).

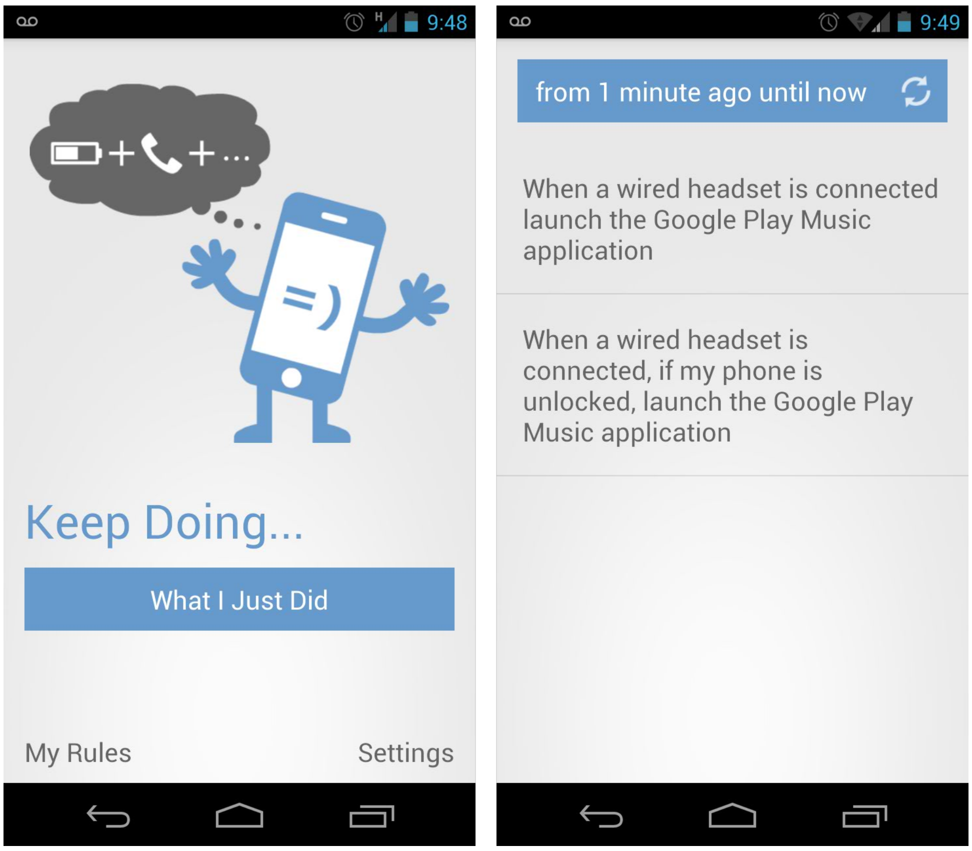
4 Lab User Study

This chapter presents and discusses the findings of the lab user study conducted on the initial prototype at the first stage of our work. The aim of this study was to examine how users would respond to the proposed smartphone automation by demonstration approach and to gain their feedback for improving the Keep Doing It application.

It is worth mentioning that we have opted for a predominantly qualitative study to help identify the more evident problems and opportunities for improvement that could not be anticipated by the researchers themselves by inspection alone. This way, the second stage study with a larger audience could be conducted with a higher-quality prototype, and so we avoided distracting many users over a longer period of time with problems that could easily be identified by in-depth qualitative research with fewer users.

4.1 Keep Doing It 1.0

When we developed the first prototype of Keep Doing It, we wanted to build only the necessary functionalities for it to serve as a proof of concept of the automating by demonstration approach. There were also no concerns with device compatibility since the prototype would run only in the evaluator's smartphone during the lab study. So we focused on the inference process and the logging component of the prototype, which already worked as depicted in chapter 3, and we implemented only a simple user interface to present the rule recommendations (Figure 5). The overall architecture (Figure 1) and the supported interactions and context information were the same. The rule recommendations presentation was fairly simple, displaying only the human-readable textual descriptions. The user already had the same control over the time window as presented in chapter 3. None of the mediation strategies to handle uncertainty were implemented and the user could not save the recommended rules.



When a wired headset is connected launch the Google Play Music application

When a wired headset is connected, if my phone is unlocked, launch the Google Play Music application

Figure 5: Screenshots of Keep Doing It 1.0, showing the initial screen (left) and possible results of a user's request to automate a task (right)

4.2 Participants

The evaluation study involved a convenience sample of 10 volunteers (Table 1). The median age of the sample was 29 years, with ages ranging from 22 to 37 years old. The selected participants were predominantly computer science graduate students and were active smartphone users (*i.e.*, all of them used smartphone applications a couple of times a day). The participants were not required to have a suitable mobile phone supported by the application since the evaluation was conducted using a provided Android smartphone with the application installed. Only 5 participants had previous experience with automation: P2, P4, P5 and P6 had already automated some sort of task in a computer and P10 had already automated a smartphone.

Table 1: Lab user study participants' profiles

Participant	Gender	Age	Phone
P1	Female	22	Android
P2	Female	26	iPhone
P3	Female	29	iPhone
P4	Female	33	iPhone
P5	Male	24	iPhone
P6	Male	26	Android
P7	Male	29	iPhone
P8	Male	29	Android
P9	Male	31	iPhone
P10	Male	37	Android

4.3 Tasks

Each task corresponded to a different automation rule that the participants had to configure by demonstration:

- Task 1 (event + action): "When I set the ringer to normal, turn the Bluetooth on"
- Task 2 (event + actions): "When a wired headset is connected, turn the Wi-Fi on and launch the music application"
- Task 3 (event + condition + action): "When a wired headset is connected, if the Wi-Fi is turned on, launch the music application"
- Task 4 (event + condition + actions): "When a wired headset is connected, if the ringer is set to vibrate, turn the Wi-Fi on and turn the Bluetooth on"
- Task 5 (spatial event + action): "When you arrive at LOCATION, turn the Wi-Fi off"
- Task 6 (temporal event + action): "At HH:MM set the ringer to silent"

While accuracy was of course desirable, one of the goals of this study was the process participants went through to create rules, and the (potentially creative or misunderstood) solutions they may come up with, in order for us to refine the prototype. Thus, we were particularly interested in the rules that the participants would fail to demonstrate, as these would point to areas or concepts that participants found difficult to comprehend or input, allowing future improvement.

4.4 Procedure

We conducted the study with each of the 10 participants on an individual basis in a laboratory at PUC-Rio. The only people who were present in the lab were the participant and the researcher. Each session took approximately 30 minutes.

The researcher gave the participants a brief explanation about what smartphone automations are and how to use the Keep Doing It application before starting the first task. The researcher also showed each participant how to create a simple automation rule as an example: "When I unlock my phone, launch the music application".

We used a think-aloud protocol (Ericsson & Simon, 1993), asking participants to verbalize their thought processes and feelings as they performed each task. During the experiment, a laptop was used to record the participants' verbal reports and their strategies to demonstrate the automation rules. The researcher also conducted a post-task and a post-experiment semi-structured interview with open questions to gather more detailed data (Appendix A).

4.5 Findings

4.5.1. Summary of Successes and Failures

Table 2 shows, for each participant and task, when the application was able to generate the designated automation rule. Four participants, who failed at first, asked to demonstrate the automation again, and all of them succeeded in their second attempt (these retries are represented as "No/Yes" in Table 2). All the participants failed to demonstrate the designated automation at least in one task and at most in three tasks. All the participants were able to successfully demonstrate the automation in the first task. In tasks 2, 5 and 6, few participants failed to demonstrate the automation, while tasks 3 and 4, both of which had a condition in the automation rule, had the higher number of demonstration failures.

Table 2: Successes and failures of each participant (Pi) to create the automation rule in each task

	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
P1	Yes	No	No	No	Yes	Yes
P2	Yes	No	No	No	Yes	Yes
P3	Yes	Yes	No	Yes	Yes	No
P4	Yes	Yes	No/Yes	No	Yes	Yes
P5	Yes	Yes	No	Yes	Yes	Yes
P6	Yes	Yes	No/Yes	Yes	Yes	Yes
P7	Yes	Yes	Yes	No	Yes	Yes
P8	Yes	Yes	Yes	Yes	No	Yes
P9	Yes	No	No/Yes	No	Yes	Yes
P10	Yes	Yes	No/Yes	Yes	No/Yes	Yes

We noted that the kind of mobile phone owned by the participant (iPhone or Android) did not affect the results. Figure 6 summarizes the task success rate.

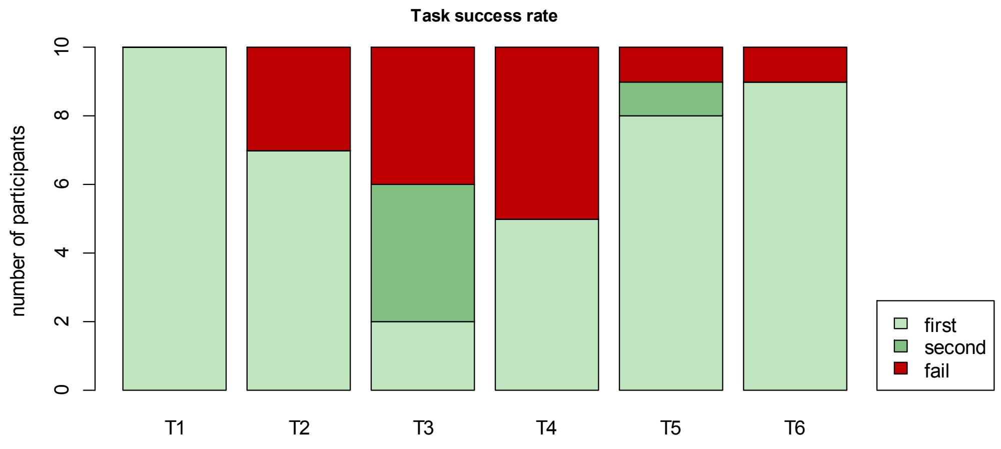


Figure 6: Task success rate

The number of generated recommendations was very particular to each task. In general, when the application successfully generated the automation, the total of recommendations ranged from 2 to 6. However, when the application did not generate the designated automation, this number was either very large, ranging from 6 to 12 recommendations, or it was a single recommendation. Figure 7 illustrates the range of recommendations per task.

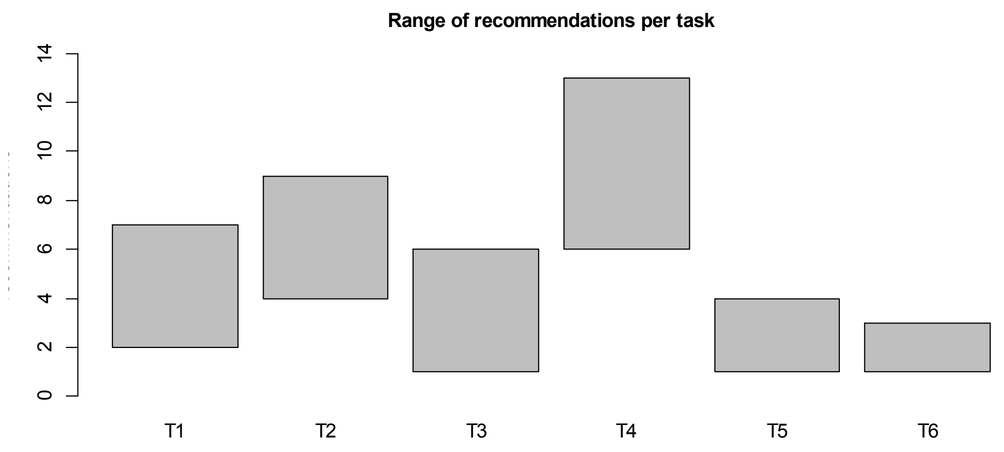


Figure 7: Range of recommendations per task

Even when the application failed to generate the intended automation, the participants were asked to indicate whether there was any other option that could, in their opinion, be easily adjusted into the correct automation. In all the tasks, regardless whether the application generated the intended automation or the user indicated the closest one, the chosen recommendation was at most among the top 4 recommendations.

4.5.2. Summary of Interviews

The interviews were conducted in Portuguese and the snippets shown here were translated to English. The Keep Doing It assessment by the participants was very positive overall. All 10 participants found the application potentially useful. P1, P5, P8, P9 and P10 expressed interest in using it once it becomes available for general use.

P2, P4, P7, P8, P9 and P10 said that the application was simple and easy to use:

> "You don't even have to think too much about how create the automation, you just have to use your phone as you normally do and that's it." (P4)

> "It (Keep Doing It) is very easy to use and convenient because you can ask it to automate what you just naturally did on your phone." (P8)

P10 affirmed that Keep Doing It is easier to use than some existing automation applications:

> "I found Locale [11] easier to use than Tasker [15], but this application (Keep Doing It) is even easier to use because what you have to do is basically to press a button to automate something you already did on your phone." (P10)

P10 pointed out, however, that it is important that Keep Doing It somehow provides to the user information about which smartphone features are being captured. Otherwise, the user might fail to demonstrate the automation rule just because he or she did not know that a certain feature was not being captured by the application.

P1, P3, P4, P5, P6, P8, P9 and P10 enjoyed the fact that the application automatically assembled the automation rule for them:

> "I don't want to create it (the automation) from scratch (…) this way (automating by demonstration) saves me time and effort." (P6)

> "I like the recommendations because I wouldn't have the patience to create automations if I had to configure each single part of them." (P9)

Most participants thought that the automation rule descriptions were clear, with the exception of P6:

> "Complex automations descriptions are difficult to read and sometimes it is hard to tell which part is the event and which one is the condition." (P6)

Most users reported having trouble when demonstrating conditions. Except from P7 and P8, all the other participants thought at first that they had to do nothing to demonstrate a condition if it was already true:

> "The Wi-Fi is already on so I don't have to toggle it." (P5, during task 3)

Although they were confused for a while about why it had not worked, later all of them guessed that they should have interacted with the phone to demonstrate the condition:

> "I believe I have to turn the Wi-Fi off and then on again to tell the application to pay attention to the Wi-Fi as a condition." (P1, after task 3)

Some participants also reported an issue regarding the location. No automation rule involving the location was recommended to P4 during task 5 and 6, and to P8 during task 6. Both were confused about why the rule with the location was not shown. P4 mentioned that time and location are basic triggers and therefore our application should always recommend one rule for each one of them. We hypothesize that a problem occurred with the GPS signal, responsible for providing the location information.

Regarding the amount of recommendations, according to P1, P2, P3, P4, P5 and P7, the maximum number of recommendations provided by the application should be around 4 or 5, in order to fit in the screen and to not hinder the selection of the desired option:

> “It would be hard to read all the recommendations and to find the right one if there were too many of them.” (P2)

However, P1, P3, P4, P5 and P8 stated that this limit could be higher, ranging from 7 to 10, if the user had performed many actions during the demonstration. P3 and P5 suggested that there should be a lower bound of at least 3 options as well. P7 suggested that the application should show the maximum of 3 recommendations at first, and let the user decide if she or he would like to see further options.

There were mixed opinions about how to sort the recommendations. P4, P5 and P10 enjoyed that they were in reverse chronological order. P2, P4, P7, P8 and P9 would rather see the desired option among the top 3 recommendations. P1, P3, P10 said that the less complex recommendations should be shown first, whereas, P2 and P5 would prefer to see the more complex ones first. P4 and P10 said that they would like to see recommendations that involve time or place at the end of the list, while P1 affirmed they should be in the beginning.

4.6 Discussion

It can be seen by the results that most of the automations (68%) could be created by demonstration without any problems. Even when the application failed to provide an accurate recommendation, according to the participants, there was at least another one that could easily be corrected using any of the discussed mediation strategies if they had already been implemented. Having to correct a recommendation or not, either way the user would not have to start configuring the automation rule from scratch, which most participants pointed out as one of the main benefits of the automating by demonstration approach. The other benefit was being easy to use. Both of these positive points met our expectations about enabling end users to easily create automations.

Although participants often used features that were irrelevant to the creation of the rule (like locking and unlocking the phone), most of the times the correct answer was still generated and displayed at the top of the list, but in this case the number of recommendations naturally increased. Besides, although almost all the users pointed out that they would rather have a small number of recommendations, we hypothesize that showing more rules than the correct one allow users to also find new useful and unintended automations by serendipity.

Two major issues were uncovered in this initial study. First, it is clear from the results that almost all the participants had problems to some extent with expressing conditions. Tasks 3 and 4, the only ones involving conditions, were exactly the tasks that had a high number of failures. Participants expected that information regarding Wi-Fi, Bluetooth, and other features of the system would be included automatically as conditions in some of the recommendations. However, this is not a simple task, since it increases the number of possible combinations, and, as the participants also stated, they would rather see the intended answer among the first recommendations and they also expect the number of recommendations to be low. So we believe that this is an issue that probably is best resolved by putting users in the loop and letting them decide which conditions are important, either before or after generating the recommendations list.

The second issue, although only one participant pointed it out, is likely to happen to most of the users in a real scenario: how does the user know which features are being observed? Our application must provide enough clues to answer this question, otherwise it may cause the user: (a) to get frustrated because she or he thinks she or he is doing something wrong, although the reason why the feature is not included in a recommendation is only because it is not supported yet; or (b) to mistakenly assume some supported feature is actually not supported whenever she or he fails to demonstrate it for any other reason, doubting the capacity of the application. The situations mentioned in the results when the location was not considered because it was outdated fall in the second case: the user is completely unaware why the location was not considered in the recommendations; therefore he might presume it is just because the application is not capable of detecting it.

5 Remote User Study

This chapter describes the five-day remote user study conducted after we developed an improved prototype version based on the findings from the lab study. Our main goal with this remote user study was to find out how people would react to the Keep Doing It application and use it in their daily lives. We also were interested in gathering quantitative data this time in order to understand more about their user experience and the application performance.

Individual needs and real context of use may have a huge impact on the overall user experience of a mobile context-aware application, especially one that is used for personal purposes. Therefore, we were aware that some issues could only be discovered through an *in situ* evaluation of Keep Doing It's use in everyday life. Also data is collected without presence of the evaluator, which decreases effects of users feeling observed and could lead to more reliable data. We wanted the study to be more ecologically valid. Consolvo et al. (2007) further described the value of in situ deployments for computing technologies.

5.1 Keep Doing It 2.0

The lab study conducted with Keep Doing It 1.0 showed that the inference process performed quite well and that participants had overall a positive impression on the approach of automating by demonstration. However, a couple of refinements to the prototype were necessary before conducting the remote study. The functionalities of this final version were already described in details in chapter 3. Hence, in this section we describe only the reasons that drove their implementation.

Some changes were already planned even before the lab study, as implementing the mediation strategies and enabling the user to save the demonstrated rules. Unfortunately, due to time constraints, we could not implement the rule engine, which is the component responsible for executing the saved rules. However, the absence of this component did not affect our study since the focus of our research was on the rule creation process.

Some other changes were made taking into account the problems identified during the lab study with Keep Doing It 1.0 and the feedback from the participants. For instance, in order to address situations similar to when participants accidentally demonstrated additional unintended interactions and also to address the issue of knowing which interactions that are indeed being monitored, we added the bottom bar with the detected interactions (Figure 3). This solution was inspired in a similar strategy of removing unintended actions from a demonstration sequence discussed in the work of Chen & Weld (2008).

We did not change the templates to create the rule descriptions since participants thought they were clear. However, as pointed out by one participant, at times it may be difficult to read the descriptions and identify each composing part of the rule. Hence, we introduced the colors, icons and the parameters highlighted in bold to increase the recommendations readability. We believe this change would be worth even though it reduces the space available to show the recommendations on the screen as a consequence.

Despite receiving some feedbacks concerning them, we decided to not make any changes yet to either the number or the order of the recommendations displayed. We decided to investigate further the appropriate maximum and minimum amounts of recommendations displayed before making any changes to it, and regarding the order there was no consensus. We also decided to investigate further the issue related to expressing conditions during the remote study before addressing it.

Finally, two main concerns of Keep Doing It 2.0 were stability and device compatibility, since for the remote study the application would have to run on participants' personal smartphones.

5.2 Participants

The remote study involved a new convenience sample of 20 volunteers (Table 3), recruited by e-mail. None of the participants were familiar with the application prior to the study, and therefore they were able to provide their first impressions.

All are Android users with a suitable mobile phone supported by the application (Android v2.3 or higher). 14 of the users have programming knowledge and either study (predominantly graduate Computer Science students) or have a career in technology, but the rest covers a diverse set of occupations including two administrators, one business consultant, one lawyer, one international buyer and one social assistant. The median age of the sample was 26 years, with ages ranging from 21 to 35 years old. Only 5 participants were female.

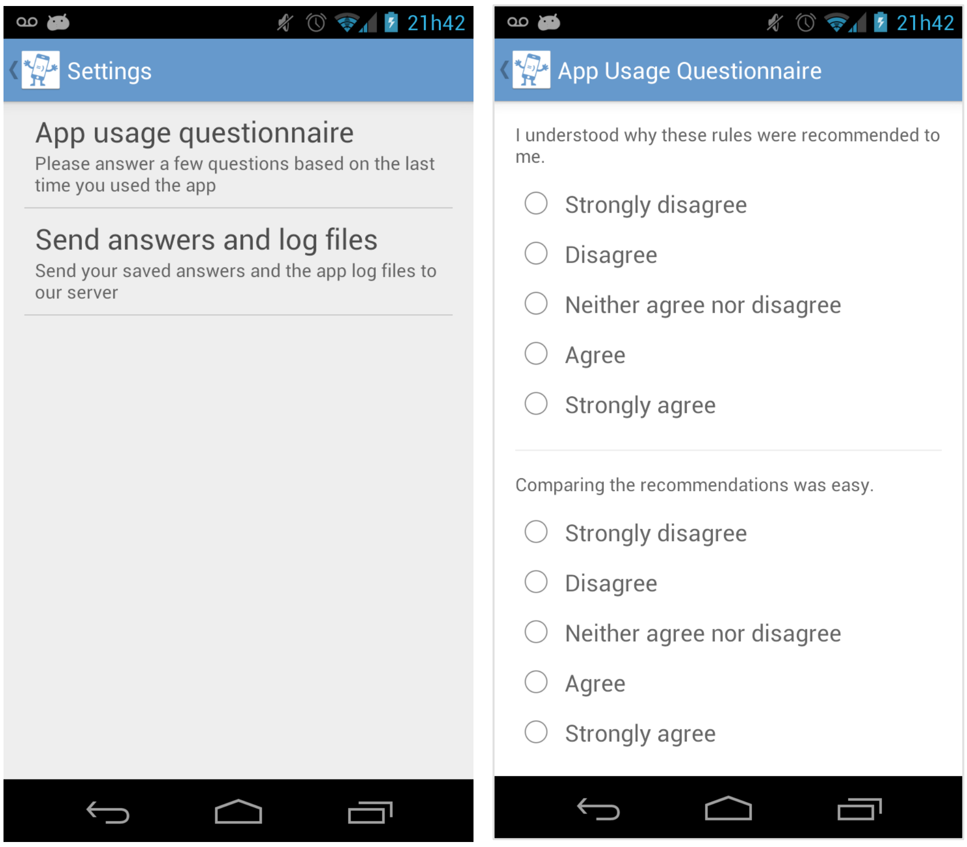
Table 3: Remote user study participants' profiles

	Gender	**Age**	**Occupation**	**Android Device**	**Version**
P1	Male	24	CS Student	Nexus S	4.1 - 4.3
P2	Male	26	CS Student	Sony Xperia E Dual	4.1 - 4.3
P3	Male	23	Software Developer	Galaxy S3	4.1 - 4.3
P4	Male	27	System Analyst	Galaxy Note 2	4.1 - 4.3
P5	Male	26	Software Developer	Alcatel One Touch HD	4.1 - 4.3
P6	Male	24	Engineer	Samsung Galaxy GT	4.1 - 4.3
P7	Male	21	CS Student	Samsung Galaxy Nexus	4.1 - 4.3
P8	Male	27	HCI Student	Samsung Galaxy Nexus	4.4
P9	Male	27	Business Consultant	Samsung Ace duos	2.3.3 - 2.3.7
P10	Male	26	Lawyer	Motorola	4.0.3 - 4.0.4
P11	Male	27	Administrator	Samsung Galaxy S4	4.1 - 4.3
P12	Female	29	IHC Analyst	Sony Ericsson	2.3.3 - 2.3.7
P13	Female	32	HCI Student	Samsung Ace	2.3.3 - 2.3.7
P14	Male	25	Web Developer	Moto X	4.4
P15	Female	24	International Buyer	Samsung Galaxy SII	4.0.3 - 4.0.4
P16	Female	24	Social Assistant	Nexus 4	4.4
P17	Male	30	System Analyst	Samsung Galaxy S4	4.1 - 4.3
P18	Male	23	CS Student	Galaxy S3	4.1 - 4.3
P19	Female	23	Administrator	Samsung Acer GT	2.3.3 - 2.3.7
P20	Male	35	Software Developer	Samsung Galaxy S4	4.1 - 4.3

5.3 Procedure

The study lasted for 5 days and, prior to the first day, participants received an email explaining the study and containing instructions on how to install the application (Appendix B). Each participant used her or his personal mobile phone for the study.

We encouraged participants to explore the application daily and to create and save as many rules as they wished. We did not prescribe tasks with specific rules to create in this study; rather we wanted participants to use the application for creating rules that would naturally occur in their daily life or that they would really want to automate.



21h42

Settings

App usage questionnaire

Please answer a few questions based on the last time you used the app

Send answers and log files

Send your saved answers and the app log files to our server

21h42

App Usage Questionnaire

I understood why these rules were recommended to me.

- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly agree

Comparing the recommendations was easy.

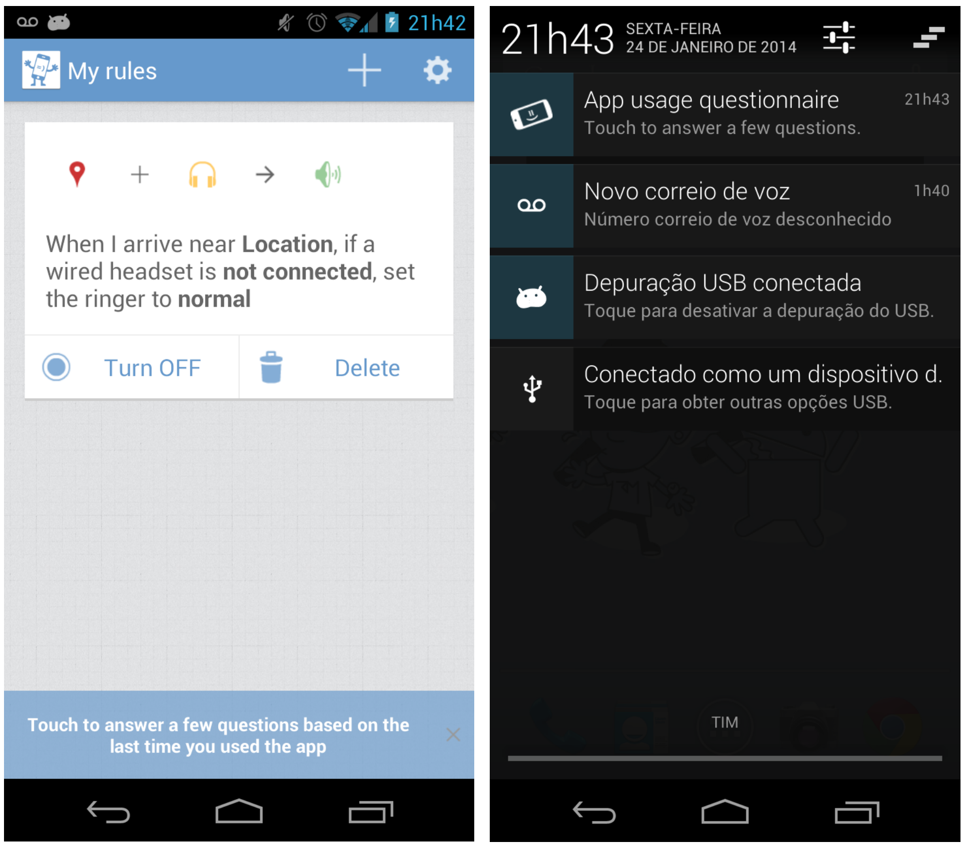
- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly agree

Figure 8: Screenshots of Keep Doing It 2.0 settings screen (left) and in-app questionnaire (right)

At the end of each session,[5] users were asked to answer a brief post-session in-app questionnaire (Appendix B), composed mostly of five-point Likert scale questions in order to get their feedback about their experience with the application. This functionality was included in the prototype (Figure 8) to make the post-session questionnaire itself feasible; otherwise, if the participants had to fill an online survey they probably would fell less inclined to answer it after every

[5] In this study, a session is the period of interaction between the start of an attempt to create an automation and the end of the attempt by going back to the initial screen, either having achieved success or not (*i.e.*, with or without saving a rule). It may comprise one or several user actions.

session, and they could also take longer to start filling the questionnaire, which would make it harder for them to recall how their experience during each particular session was. Participants could access the questionnaire anytime through the application settings screen (Figure 8), but we also implemented a notification system in order to remind and encourage them to answer the questionnaire right after each session (Figure 9). Participants had only to touch the notification in order to access the questionnaire. A notification would appear in the initial screen in case the user either saved a rule or went back to the initial screen after demonstrating a couple of rules. In case participants closed the app after using it without returning to the initial screen, a notification would appear after two minutes in the Android system notification drawer, which is visible for them by the operating system.


21h42

My rules

When I arrive near **Location**, if a wired headset is **not connected**, set the ringer to **normal**

Turn OFF | Delete

Touch to answer a few questions based on the last time you used the app

21h43 SEXTA-FEIRA 24 DE JANEIRO DE 2014

App usage questionnaire 21h43
Touch to answer a few questions.

Novo correio de voz 1h40
Número correio de voz desconhecido

Depuração USB conectada
Toque para desativar a depuração do USB.

Conectado como um dispositivo d.
Toque para obter outras opções USB.

TIM

Figure 9: Notifications reminding the user to answer the post-session in-app questionnaire in the Keep Doing It 2.0 initial screen (left) and in the Android notification drawer (right)

When a participant answered a post-session questionnaire, the answers were automatically sent to our central server. The application kept a record of the post-session questionnaires answered by the participants locally on their phone, allowing us to retrieve any data not received at the end of the study (e.g., if the participant sent an answer when her phone was not connected to the Internet). Participants could tell the application to send the data locally stored by pressing a button at the settings screen (Figure 8).

In addition, we also maintained a time-stamped log of the participants' interaction with the system in order to gather a record of the number of attempts to create automations, how many of them were indeed saved, the size of the recommendation sets and which of the mediation strategies available they used and how often. The log entries were also stored locally and then sent to a server whenever the user sent post-session questionnaire answers.

After the five days of study, we sent an email thanking the participants and asking them to fill a post-study questionnaire (Appendix B) about their overall experience with the application.

5.4 Findings

5.4.1. Summary of Session Logs

Overall, participants had a good compliance over the five days of the study; on average the participants interacted with the app during three out of the five days, with P8 and P9 interacting throughout the five days, P10 for four, and P7, P16 and P15 for only one day. Table 4 shows, for each participant and day of study, the number of interaction sessions they had. The peaks of interaction with the application were on day two and three. Half of the participants had more than 10 interaction sessions in total, with a minimum of 4 sessions (P2 and P7) and a maximum of 28, 31 and 32 sessions (P3, P9 and P13, respectively).

The application did not crash during any point of the study. To discover what each participant did in Keep Doing It during the interaction sessions, we logged five main actions:

- **Generate recommendations**: the participant asked the application to generate a list of recommendations based on his latest interactions;
- **Regenerate recommendations**: the participant enabled or disabled an interaction from the sequence of detected interactions (by clicking on it in

the bottom bar, as seen in Figure 3), causing the application to regenerate the recommendations with or without considering this interaction;

- **Select rule**: the participant selected one of the recommended rules;
- **Edit rule**: the participant altered a selected rule by removing, adding or switching an event, condition or action; this action was logged only if the editing happened to a rule that had not been recorded yet.
- **Save rule**: the participant recorded a selected rule; this action was not logged if the user was only updating a rule that had already been recorded.

Table 4: Number of sessions each participant had per day of study

Participants	Day of Study					Total of Sessions	Days of use
	1	2	3	4	5		
P13	0	0	18	13	1	32	3
P9	8	10	9	3	1	31	5
P3	0	21	1	6	0	28	3
P5	15	3	0	0	2	20	3
P17	0	0	0	12	5	17	2
P14	0	11	2	1	0	14	3
P10	1	4	1	0	7	13	4
P8	3	6	1	1	2	13	5
P19	0	7	3	1	0	11	3
P6	3	1	7	0	0	11	3
P20	0	3	1	0	5	9	3
P4	0	6	3	0	0	9	2
P12	0	0	7	1	0	8	2
P18	0	0	0	7	1	8	2
P15	0	0	7	0	0	7	1
P1	3	2	1	0	0	6	3
P11	0	0	4	0	1	5	2
P16	0	5	0	0	0	5	1
P2	2	1	1	0	0	4	3
P7	0	4	0	0	0	4	1

Table 5 shows, for each participant, how many times each action was detected. On average, around 50 actions were recorded per participant, with P9, P3, P13 and P5 performing over 100 actions each, and P7 only 6. Only 7 participants performed all the five different actions, 2 participants (P4, P14) performed only the actions of generating and regenerating recommendations, and 2 participants (P7, P19) only performed the action of generating recommendations. Each type of action was performed by most of the participants: all the participants performed the action of generating recommendations; the actions of regenerating recommendations, editing and saving a rule were performed by 12 participants each; and only 4 participants (P4, P7, P14, P19) did not perform the action of selecting a rule.

Table 5: Number of each action type logged per participant

	EDIT	**GENERATE**	**REGENERATE**	**SAVE**	**SELECT**	**Total**
P5	30	59	10	10	16	125
P13	7	86	18	5	5	121
P3	33	43	8	14	17	115
P9	1	94	9	1	1	106
P14	0	78	7	0	0	85
P12	4	60	9	2	3	78
P19	0	58	0	0	0	58
P17	7	23	4	4	6	44
P1	4	23	6	3	6	42
P4	0	18	22	0	0	40
P6	7	18	0	3	3	31
P20	2	23	0	2	3	30
P8	1	27	0	0	1	29
P18	0	17	0	3	3	23
P10	0	20	1	0	1	22
P16	5	12	0	1	2	20
P15	0	8	2	0	2	12
P2	1	7	0	2	2	12
P11	0	7	1	0	1	9
P7	0	6	0	0	0	6
Total	102	687	97	50	72	1008

68% of all the actions performed by all the participants together were actions of generating recommendations, while only 5% of them were actions of saving a rule. This was expected, since participants were not encouraged to save any rule at all, unless they really wanted to, and also due to the fact that we warned the participants that we had not yet implemented the rule engine to execute the saved rules anyway. Editing was the second most often performed action, followed by the actions of regenerating recommendations and selecting a rule. In the next sections, we discuss each logged action type.

5.4.1.1. Generate Recommendations Action

Table 6 shows, for each participant, how many actions of generating recommendations returned no rule recommendations, while Table 7 shows how many actions returned one or more rule recommendations at each session.

By looking into Table 7 we can see that, with the exception of P8, none of the participants could obtain any rule recommendation during the first few sessions. On average, participants started to see recommendations only after the fourth interaction session. Regarding the number of actions, by looking into Table 6 we can see that it took participants, on average, around 13 actions to start seeing any recommendation. For most of them (14 participants) it took at most 8 actions, but there were some outliers for whom it took more than 46 (P12), 53 (P9) and 59 (P14) actions to start seeing recommendations. By analyzing both Table 6 and Table 7, we can see that in general participants indeed got the gist of the application after their first successful attempt to generate recommendations, even P12 and P14. The only exceptions to this pattern were P9 and P13, based on the high number of unsuccessful attempts that they had even after their first successful attempt.

All the participants used the 15 seconds time window to generate the recommendations at first, which was expected since it was the default. However, the most often used time window was 60 seconds, and participants barely used the 30 or 45 seconds time windows. As we can see in Table 8, which shows only the number of successful attempts to generate recommendations for each participant and available time window, the application barely generated automation rule recommendations when using any time window other than 60 seconds. Moreover, participants usually only started to see recommendations after they changed the default 15 seconds time window to 60 seconds, which partially explains why it took a while for most participants to start getting recommendations.

Table 6: Number of actions that returned no recommendations per participants at each session

Session	Participant																			
	1	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**2**	**20**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	1	1	1	1	4	4	1	1	2	2	2	3	1	1	1	4	1	1	0	1
2	1	1	1	5	2	13	1	5	1	1	1	1	5	2	1	1	1	2	1	6
3	2	1		5	1	1	2	3	1	1	6	1	5	0	0	1	2	2	1	2
4	11	3		12	5	5	1	1	5	0	1		0	7	4	0	1		1	3
5		0		23	0	14	0	0	1	0	11		0	0	1	0	1		0	2
6		0		8	1	12	1	0	1	4	0		0	1	3	0	0		0	1
7		1		1	0	10			2		10		1	0	0	0	1		0	1
8		3		2	5	5			1		10		0	0	1	0	6		0	1
9		1			5	5			0		3		2	1		2	0		0	5
10		1			7	0			0		8			1		0	0		2	2
11		0			2	0			1		1			0		0	1		4	2
12		3			1	1			0					0		2			0	1
13		1			7				1					1		17			3	4
14					1				0					0		2				1
15					2				0					1		0				1
16					1				1					0		2				5
17					2									0		0				2
18					1									1		0				1
19					5									1		0				1
20					1									0		2				2
21					5									0						1
22					1									0						2
23					1									1						1
24					0									2						5
25					2									2						
26					6									0						10
27					6									0						3
28					1									1						23
29					0															1
30					1															1
31					3															
32					1															
Total	15	16	2	57	80	70	6	10	17	8	53	5	14	23	11	33	14	5	12	91

Table 7: Number of successful attempts to generate recommendations per participants at each session

Session	Participant																			
	1	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**2**	**20**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0
3	3	0	2	0	0	0	0	0	0	1	0	1	1	2	1	1	1	0	0	0
4	2	0	1	0	1	0	0	0	0	1	0	1	3	0	0	1	0	1	0	0
5	1	1	2	0	1	0	1	2	0	4	4		2	1	1	2	0		5	0
6	2	1		1	0	0	0		0	0	1		2	0	0	2	1		2	0
7		0		0	1	0	1		0	1			0	4	3	1	0		2	0
8		0		2	0	1			0	1			1	1	0	5	0		1	0
9		0			0	0			1					1	1	1	1		1	0
10		1			0	1			1					0		2	1		1	0
11		1			0	1			0					1		1			0	0
12					0	0			1					1		1			1	0
13					0	1			0					0		0				0
14					0	4			1					1		1				0
15					0				1					0		2				0
16					0				0					1		1				0
17					1				1					1		2				0
18					0									1		2				0
19					0									0		1				0
20					0									1						0
21					0									1						0
22					0									1						0
23					0									0						0
24					1									0						0
25					0									0						1
26					0									1						0
27					0									1						0
28					0															0
29					1															0
30																				0
31																				2
32																				
Total	8	4	5	3	6	8	2	2	6	9	5	2	9	20	7	26	4	1	15	3

Table 8: Number of successful attempts to generate recommendations for each participant per available time window

Participants	Time Window (in seconds)				Total
	15	30	45	60	
P9	0	0	0	3	3
P13	0	0	1	5	6
P14	0	0	0	8	8
P12	0	0	1	2	3
P5	0	0	0	26	26
P19	1	0	0	4	5
P3	1	3	0	16	20
P8	1	2	0	12	15
P1	1	0	2	5	8
P17	0	0	0	6	6
P20	0	0	1	8	9
P10	2	0	2	0	4
P4	0	0	0	7	7
P6	2	0	2	0	4
P18	9	0	0	0	9
P16	0	0	1	1	2
P15	0	0	0	2	2
P11	1	0	0	4	5
P2	0	0	0	2	2
P7	0	0	0	1	1
Total	18	5	10	112	145

Table 9 shows, for each participant, the typical recommendation set sizes and the number of times they obtained a set with each one of those sizes during the study. In general, aside from the zero size sets, the most common generated set sizes were 1 and 2. In third place were the sets with 4 recommendations. Only P5 and P17 saw a set with 10 recommendations, which was the higher size of a set logged during the study.

Table 10 shows how many times each set size was generated according to the time window used. As expected, sets with more than 3 recommendations were generated almost only when a 60 seconds time window was used.

Table 9: Number of times a recommendations set size occurred per participant

Participants	**Recommendations Set Size**											**Total**
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
P9	91	2	1	0	0	0	0	0	0	0	0	94
P13	80	2	2	0	1	0	1	0	0	0	0	86
P14	70	5	3	0	0	0	0	0	0	0	0	78
P12	57	0	3	0	0	0	0	0	0	0	0	60
P5	33	0	20	0	5	0	0	0	0	0	1	59
P19	53	0	5	0	0	0	0	0	0	0	0	58
P3	23	14	5	0	0	1	0	0	0	0	0	43
P8	12	3	6	2	2	0	1	0	0	1	0	27
P1	15	0	4	0	3	0	0	0	1	0	0	23
P17	17	1	3	1	0	0	0	0	0	0	1	23
P20	14	3	5	0	0	0	0	1	0	0	0	23
P10	16	4	0	0	0	0	0	0	0	0	0	20
P4	11	4	3	0	0	0	0	0	0	0	0	18
P6	14	3	1	0	0	0	0	0	0	0	0	18
P18	8	2	4	0	3	0	0	0	0	0	0	17
P16	10	2	0	0	0	0	0	0	0	0	0	12
P15	6	2	0	0	0	0	0	0	0	0	0	8
P11	2	2	2	0	1	0	0	0	0	0	0	7
P2	5	0	1	0	1	0	0	0	0	0	0	7
P7	5	0	1	0	0	0	0	0	0	0	0	6
Total	542	49	69	3	16	1	2	1	1	1	2	687

Table 10: Number of times that each recommendations set size occurred according to the time window used for the demonstration

Recommendations Set Size	**Time Window (in seconds)**				**Total**
	15	**30**	**45**	**60**	
0	159	46	45	292	542
1	6	1	6	36	49
2	8	4	3	54	69
3	0	0	0	3	3
4	4	0	0	12	16
5	0	0	0	1	1
6	0	0	0	2	2
7	0	0	0	1	1
8	0	0	1	0	1
9	0	0	0	1	1
10	0	0	0	2	2
Total	177	51	55	404	687

Table 11: Number of times an interaction type was performed by each participant

	Screen	App	Ringer	Wi-Fi	Bluetooth	Charging	Headset	Call
P1	0	4	6	0	3	0	0	0
P10	2	3	0	1	0	0	0	0
P11	2	6	0	0	0	0	0	0
P12	15	1	0	2	0	0	0	0
P13	39	4	3	0	0	1	0	1
P14	24	4	4	0	0	0	0	0
P15	6	2	0	0	0	0	0	0
P16	0	2	0	0	0	0	0	0
P17	18	5	7	0	0	0	1	0
P18	2	0	8	4	0	0	0	0
P19	0	5	0	0	0	0	0	0
P2	0	0	3	0	0	0	0	0
P20	4	6	6	0	0	0	0	0
P3	20	7	3	17	0	1	0	0
P4	6	8	0	0	0	0	0	0
P5	7	32	0	0	2	0	0	0
P6	1	0	3	1	1	0	0	0
P7	0	1	0	0	0	0	0	0
P8	20	21	11	2	0	0	0	0
P9	5	3	0	0	0	3	0	0
Total	171	114	54	27	6	5	1	1

Table 11 shows the number of times each interaction type was performed by each participant during the study (only the interactions performed within a demonstration time window were considered). All the 8 supported interaction types were detected. However, none of the participants performed the interaction of disconnecting the phone from charging or disconnecting the headset. The interaction types related to making a phone call and to connecting a headset were performed only once. Charging the phone's battery and toggling the Bluetooth were performed 5 and 6 times respectively, by 3 different participants each. All these aforementioned interactions represent all together only 4% of the detected interactions. 45% of all the interactions were related to the phone's screen status, 30% to opening applications, 14% to changing the ringer mode and 7% to toggling the Wi-Fi. Almost all the participants performed interactions related either to the phone's screen status (except for P1, P2, P7, P16, P19) or to opening applications (except for P2, P6, P18). This was expected since quite often users lock (and then unlock) the screen (even if by accident, as seen during the lab study), and since opening applications is one of the most common interactions with smartphones. In general, participants opened the same application around 2 or 3 times; however, P8 opened 19 times an application called Nova Launcher.[6] This application is a replacement for the default Android's home screen, which is considered an application itself (also known as Launcher). In our implementation, we did not consider custom home screens, therefore every time P8 went to the home screen to open an application or perform any other action, the interaction of "opening" the home screen was also (mistakenly) detected.

Table 12: Number of times that each interaction and context type appeared in a rule of all the recommendations set generated during the study

Interaction Type	**Position of the recommendation on a set**									
	1	2	3	4	5	6	7	8	9	10
App	84	64	13	13	5	6	3	3	2	2
Bluetooth	5	4	2	2	1	1	1	1	0	0
Charging	0	0	0	0	0	0	0	0	0	0
Headset	1	1	0	0	0	0	0	0	0	0
Location	72	5	14	0	4	2	2	0	2	0
Call	1	1	1	1	0	0	0	0	0	0
Ringer	39	27	14	12	2	1	1	0	0	0
Screen	38	12	4	3	2	1	1	1	1	0
Time	30	79	8	21	0	4	2	3	0	2
Wi-Fi	20	9	3	3	2	1	1	1	1	0

Table 12 shows, for each interaction and context type, how many times it appeared in a rule recommendation and in which position on the list the corresponding recommendation was. Charging the phone's battery was the only interaction that did not appear in any recommendation. Opening an application and changing the ringer mode were the interactions that most often appeared in recommendations, while the time and location context information appeared almost as often. Although, as mentioned before, most of the interactions detected were related to the phone's screen status, they did not appear that often in recommendations because they cannot have the role of an action. Interestingly

[6] http://novalauncher.com

though, this was the only type of interaction that also appeared with the role of a condition. Going through the log files we observed that only 6 of all the rule recommendations were conditional (event + condition + action), and that they only appeared on the second, third or sixth position of a recommendation set.

5.4.1.2. Regenerate Recommendations Action

As we mentioned before, regenerating recommendations was the third overall most performed action. By taking a look at Table 13, we can see that it was only used with five different types of detected interaction, and most of the times it was used to disable/ignore a detected interaction that was related to the phone's screen status. However, by looking at the context when each of these actions were logged, we observed that almost always the recommendations set size was already zero even before a detected interaction was ignored (for instance, this was the case in 88% of the times when this action was used to ignore an detected screen interaction). Therefore, this clearly indicates that, in general, the participants who used this action were not using it to regenerate the list of recommendations without a detected unintended interaction (which was the original purpose of this functionality). We believe they were only trying to figure out if this action would somehow make a recommendation appear whenever the attempt to generate recommendations was unsuccessful.

Table 13: Number of an interaction of a certain type was disabled (ignored) from or enabled back in the sequence of detected interactions

Interaction Type	**Status on the sequence of detected interactions**	
	Disabled	**Enabled**
App	10	6
Bluetooth	5	4
Charging	4	3
Screen	35	25
Wi-Fi	4	1
Total	58	39

5.4.1.3. Select Rule Action

Table 14 shows in which positions on the list of recommendations were the recommendations selected by the participants, how frequently the selected recommendations were in each list position, and which were the typical sizes of the selected rules. 72% of the selected recommendations were on the top of the list and 22% were in the second position. Only twice a selected recommendation was in the third position of the list, and only once it was in the fourth and fifth positions. 96% of the selected rules were of size 2 (event + action) and only 4% were of size 3 (two "event + condition + action" rules and one "event + 2 actions" rule).

Table 14: Number of times a selected recommendation was in a position and had a certain rule size

Position on the recommendations set	**Rule size**		**Total**
	2	**3**	
1	52	0	52
2	14	2	16
3	2	0	2
4	0	1	1
5	1	0	1
Total	69	3	72

Table 15: Number of times that each interaction and context type appeared with a certain role in the selected rule

Interaction Type	**Role**			**Total**
	Event	**Condition**	**Action**	
App	0	0	39	39
Bluetooth	0	0	2	2
Headset	1	0	0	1
Location	24	0	0	24
Call	0	0	1	1
Ringer	0	0	20	20
Screen	15	2	0	17
Time	31	0	0	31
Wi-Fi	1	0	11	12
Total	72	2	73	147

Table 15 shows the typical interaction types present on the selected recommendation and how frequently they were in each role. Almost all the interaction types were present on at least one of selected rules, with the exception of course of the interaction of charging the phone’s battery, which, as aforementioned, was the only interaction that did not appear in any generated recommendation. The most commonly selected rules were the ones where either the event was related to a location or to the time, and where the action was to open an application.

5.4.1.4. Edit Rule Action

Table 16 shows how many times the participants edited the selected recommendations and which modification was made. 60% of all the modifications were related to adding an interaction or context information to the rule, 20% were related to removing an interaction or context information, and 20% were related to switching an interaction or context information for another one (either one from a different type or one of the same type but with a different parameter).

Table 16: Number of times a certain modification was made to a selected recommendation per modified interaction and context information

Interaction Type	**Modification**			**Total**
	Add	**Remove**	**Switch**	
App	4	5	1	10
Bluetooth	2	0	0	2
Charging	11	0	0	11
Headset	5	0	0	5
Location	5	8	1	14
Call	3	1	0	4
Ringer	4	0	1	5
Screen	2	1	1	4
Time	7	4	13	24
Wi-Fi	19	1	3	23
Total	62	20	20	102

Adding a condition to the rule was the most common change. Participants added conditions related usually to the Wi-Fi or the phone’s battery. This is probably due to the fact that participants have difficulty to demonstrate conditions, as noticed during the lab study: participants expect the application to consider a specific phone feature (as the Wi-Fi or the phone's battery) as relevant without the need for them to interact with it first. Moreover, probably due to the same reason, some participants also felt the need to add missing events and actions related to the Wi-Fi (although they could demonstrate it just by toggling the Wi-Fi).

The most common switch was with events related to time. This is probably due to the fact that participants did not want the time event to be as accurate as it was being recommended (for instance, instead of 12:49 PM they would rather have the time set to 13:00 PM).

5.4.1.5. Save Rule Action

Table 17 shows the sizes of the saved rules. Not surprisingly the most frequent sizes were 2 and 3, but there were also saved rules with sizes 4, 5 and even 7, which is the biggest size a rule can get in our application (event + 3 conditions + 3 actions).

Table 17: Number of rules saved per rule size

Rule Size	**Total**
2	21
3	20
4	5
5	3
7	1
Total of rules saved	50

Table 18 shows how many times an interaction or context information appeared in the saved rules and in which role. The most common events to appear in a saved rule were related to time, arriving at a given location, or unlocking the phone's screen. The most common conditions were the phone's battery discharging or to the Wi-Fi being on. Finally, the most common actions were opening an application, turning the Wi-Fi on or changing the ringer to normal.

Table 18: Number of times that each interaction and context type appeared in a certain role in the saved rule

Interaction Type	Role			Total
	Event	Condition	Action	
App			25	25
Bluetooth			4	4
Charging	1	9		10
Headset	2	4		6
Location	13	2		15
Call	1	1	1	3
Ringer		3	13	16
Screen	10	3		13
Time	19	5		24
Wi-Fi	4	8	16	28
Total	50	35	59	144

5.4.2. Summary of Post-Session Questionnaire Answers

At the end of the study we were able to gather 60 post-session questionnaires filled out; on average, each participant answered around 3 times the questionnaire for different sessions or groups of sessions. P3 answered 8 times, which was the higher number of questionnaires per participant, while P6, P7, P8, P15 and P16 answered only once. We asked the participants to rate the recommendations received during a given session on intelligibility (*i.e.*, how clear it was to understand why a particular rule was recommended), ease of choice (*i.e.*, how easy it was to compare the recommendations in order to choose one), relevance (*i.e.*, how relevant the recommendations seemed according to the context), perceived number of recommendations, and accuracy. These questions, as well as the ones used in the post-study questionnaire, were based on the ones used for user-centered evaluations of recommender systems (Knijnenburg et al., 2012; Pu et al., 2011). To analyze the answers on intelligibility, ease of choice and relevance, we first associated discrete scores ranging from -2 to 2 to each of the items on the 5-point Likert scales and then we displayed these data in a box plot[7] with whiskers from minimum to maximum. In Figure 10 we can note that in general the recommendations were very well evaluated on these three criteria.

[7] We shifted the data based on the minimum value, so that all the scores are positive and better displayed on the chart.

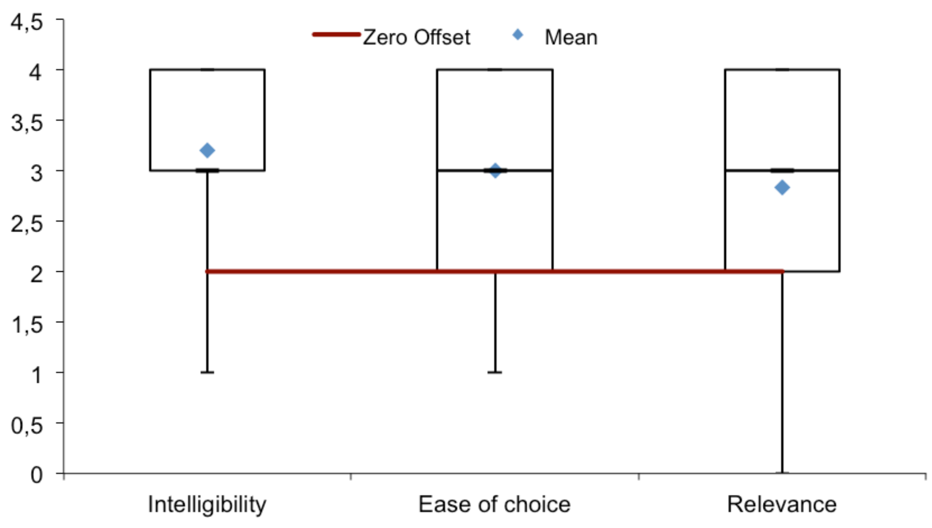


Figure 10: Box plot of the scores given by the participants for the evaluated sessions on intelligibility, ease of choice and relevance.

In recommender systems, it is usually unclear how many recommendations the system should provide (Knijnenburg et al., 2012). In conventional choice situations, too few items may restrict the users' freedom of choice, whereas too many items may lead to choice overload. Hence, it was important not only to ask how hard it was to compare and choose among the recommendations, but also if the participant perceived them as not enough (or too many). Figure 11 shows how the participants perceived the number of recommendations received during the evaluated sessions. Overall, most of the participants thought that the number of recommendations was reasonable; while very few thought that there were too many. This makes sense, since, as aforementioned, the non-zero recommendations set sizes ranged from 1 to 10 recommendations, but most of the sets were size 1 or 2. But there was even a case where a set with 10 recommendations was considered reasonable for P17 and another where P20 thought that 7 recommendations were too few. This indicates that the amount itself is not the best indicator if there were too many, too few or a reasonable number of recommendations.

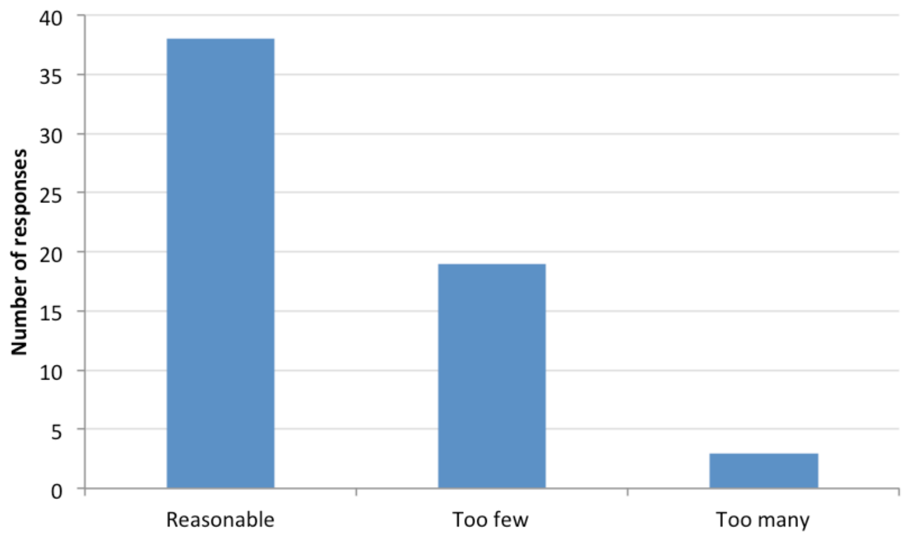


Figure 11: Number of responses that classified the number of recommendations as reasonable, too few or too many

However, according to Figure 12, 68% of the times that participants classified a session as having too many or a reasonable number of recommendations, they also rated the accuracy positively (“Close enough” or “Perfect match”); while 79% of the times that they classified a session as having too few recommendations, they also rated the accuracy negatively (“Not so close” or “Not even close”). Hence, the number of recommendations is perceived usually as too few whenever the participant cannot find a single accurate recommendation (even if 7 recommendations were presented, as happened to P20). However, if the participant can find at least one accurate response, it does not really matter much how many recommendations there were. This indicates that we could possibly add more recommendations without bothering the user if it contributes to increase the chances of suggesting an accurate automation rule.

Figure 13 shows in detail how the recommendations were rated regarding their accuracy. Overall, 52% of the time they were considered accurate (37%) or very accurate (15%); 37% of the time they were somewhat inaccurate (34%) or very inaccurate (3%); and 11% of the time the participant was not trying to demonstrate any particular rule and therefore the accuracy could not be rated (neutral).

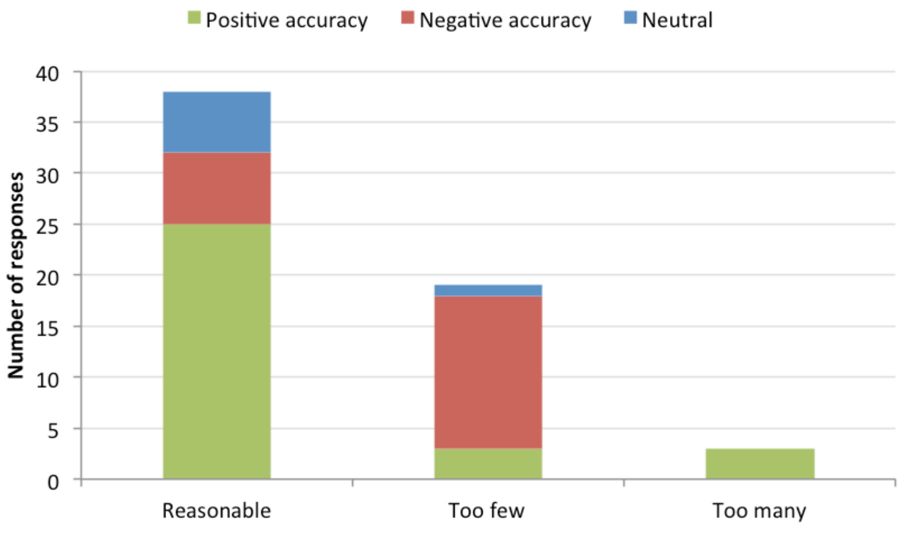


Figure 12: Number of responses that classified the number of recommendations as reasonable, too few or too many and that also rated the accuracy as positive, negative or neutral

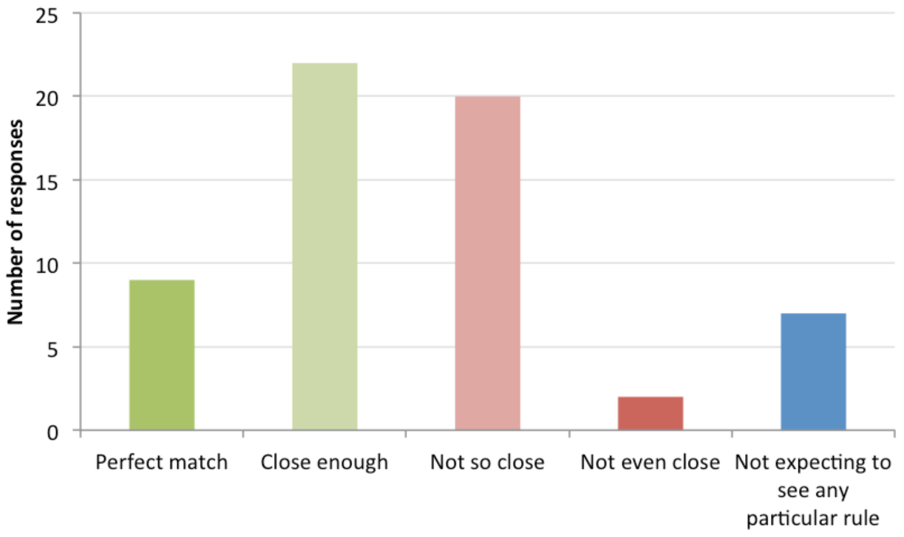


Figure 13: Accuracy of the recommendations received during the evaluated sessions. The green color indicates a positive accuracy, the red color a negative accuracy, and the blue color is neutral.

By investigating further only the evaluated sessions whose recommendations were rated accurate, we discovered that: 45% of the events were time-based; 50% of the actions were related to opening an application; 41% of the “Close enough” recommendations were classified as such because the participant had still to either edit the time or add an interaction related to the Wi-Fi. To investigate why the recommendations in the other evaluated sessions were considered inaccurate, we took into account the responses to an optional question in the post-session questionnaire about which automation rule the participant expected to see. Based on the 6 responses we obtained, we noticed that some of the participants were expecting more flexible parameters for location and time: leaving a location instead of only arriving or being at a location; setting the rule to trigger when it is right after or right before a given time, or inside a time interval, instead of only when it is equal to a specific point in time. Others were expecting the application to take into account application-specific information (fine granularity) as scheduled appointments (in a calendar application, for instance).

5.4.3. Summary of Post-Study Questionnaire Answers

At the end of the study we asked each participant to rate the application and the proposed approach on ease of learning, ease of use, perceived usefulness (*i.e.*, how helpful the application to create automation rules was), serendipity, respect for privacy, overall satisfaction, interaction and interface adequacy, system intelligibility (*i.e.*, how clear it was to understand how the application worked), and use intentions. The post-study questionnaire also included four open questions at the end in order to gather comments and recommendations regarding future developments of features and to obtain more insights about the advantages and disadvantages of the application.

To analyze the first five attributes, we once again associated all the related 5-point Likert answers to discrete scores ranging from -2 to 2, and then displayed these data in a box plot with whiskers from minimum to maximum. In Figure 14 we can see that Keep Doing It received mostly positive ratings regarding all of these five attributes. One of the highest rated was the overall satisfaction: almost all the participants agreed that they were overall satisfied with Keep Doing It and the approach of creating automation rules by demonstration (13 agreed and 4 strongly agreed).

The second highest-rated criterion was the respect for privacy. Only two users (P4 and P17) expressed concerns for their privacy. P4 recommended that the application should clearly inform which smartphone data are collected and how they are used. P17 concerns went further, reaching the point in which he said that he would not use the application unless he clearly knew the privacy of his data was being assured. However, except for P7, all the other participants expressed interest in using the application if it were available for general use, and some would use it even if they had to pay for it (Figure 15).

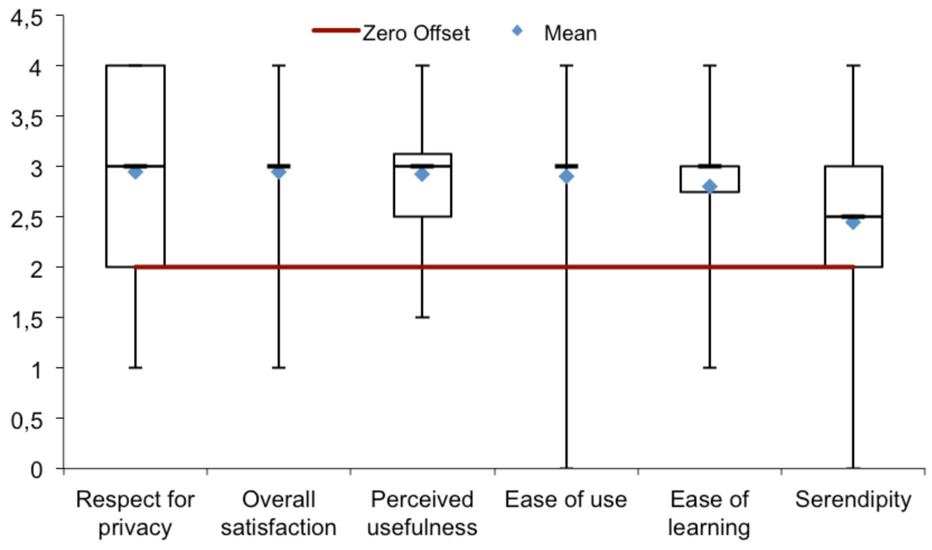


Figure 14: Box plot of the scores given by the participants for the application on ease of learning, ease of use, perceived usefulness, serendipity, respect for privacy and overall satisfaction.

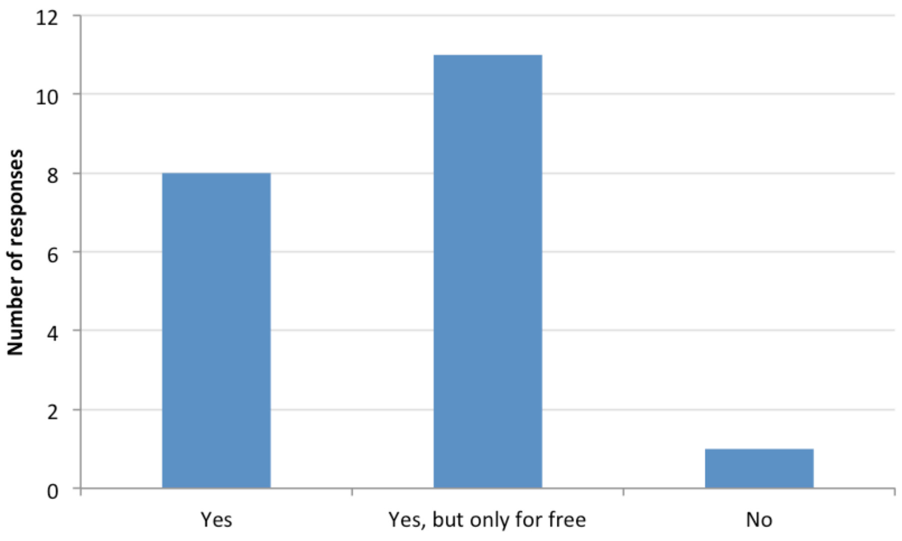


Figure 15: Responses regarding future use intentions of Keep Doing It

Despite the apparent struggle (according to the log files) to create recommendations in the first sessions, even the ease of learning and ease of use ratings were overall very positive: only P11, P12 and P13 disagreed (but not strongly) regarding the ease of learning, and only P18 disagreed (strongly) regarding the ease of use (although he agreed that it was easy to learn). Some participants even mentioned those aspects when asked about the main advantages of the approach:

"It was very simple to understand." *(P18)*

"It is very easy to define new rules without a lot effort. If you open the app you will have recommendations that are simply to modify and save." *(P5)*

"I think it was pretty efficient, fast and intuitive." *(P10)*

Despite agreeing that the application is easy to learn, P19 (who took the longest to make the application generate successful recommendations) suggested adding a brief guidance about how the application works in order to help users when they open the application for the first time.

The perceived usefulness was measured by two questions related to two possible benefits of the proposed approach: making the rule creation easier and faster than setting the rules by scratch. 70% of the participants agreed that the approach made the rule creation process easier (Figure 16), and 95% of the participants believed that the approach made it faster (Figure 17). However, P5, P16 and P17 would like to be able to manually create some rules as well, instead of only demonstrating them.

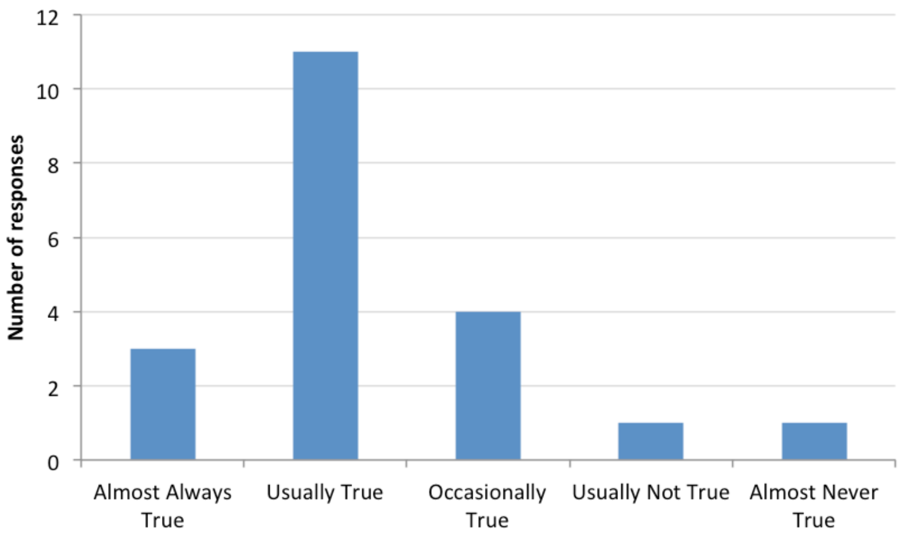


Figure 16: Responses regarding if our approach made the rule creation process easier

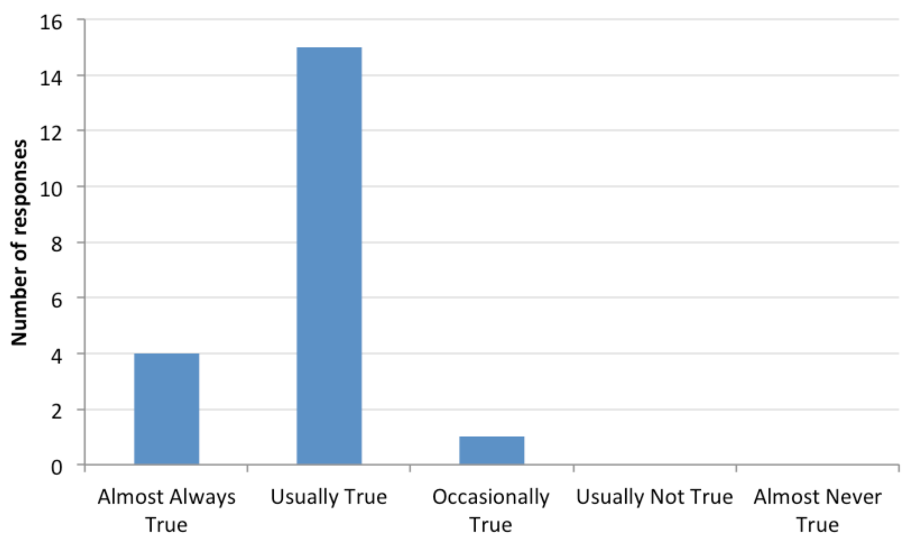


Figure 17: Responses regarding if our approach made the rule creation process faster

Although the focus of our approach is to recommend rules that the user expects to see, we were curious to know if among the recommendations the participants would discover unintended automations by serendipity as well. Although this criteria was rated lower compared to the others, its mean score was still positive, and some participants commented in the open questions at the end of the questionnaire that the application indeed somewhat also helped them to discover new rule possibilities.

> “Sometimes, the features shown are relevant as well, even though we have not thought about them.” *(P3)*

> “I liked to receive recommendations since in general there were rules based on features that I did not even know that could be automated by the app” *(P12)*

> “The suggestions were interesting because I didn’t have much knowledge about smartphone automations before, so I did not know exactly what I could automate or not.” *(P9)*

However, P4 and P18 think that some of the rules (as opening a specific application after unlocking the screen, for instance) did not make sense and that they should not be recommended because they were related to things that (in their opinion) no one would normally want to automate.

Regarding the interaction adequacy, 55% of the participants considered that demonstrating the rules was not annoying most of the time, while 30% answered that occasionally it was, and 15% answered that it usually was annoying to demonstrate the automations (Figure 18). One of the reasons why participants sometimes thought that demonstrating the automation rules was annoying was due to the demonstration interval.

> "I think it is a little annoying that sometimes the app only shows apps that were opened a minute ago. Sometimes in order to have a good recommendation you have to open the app immediately after you open the app you want to create the rule." *(P5)*

> "I didn't like the given time intervals, they made me feel a little desperate" *(P12)*

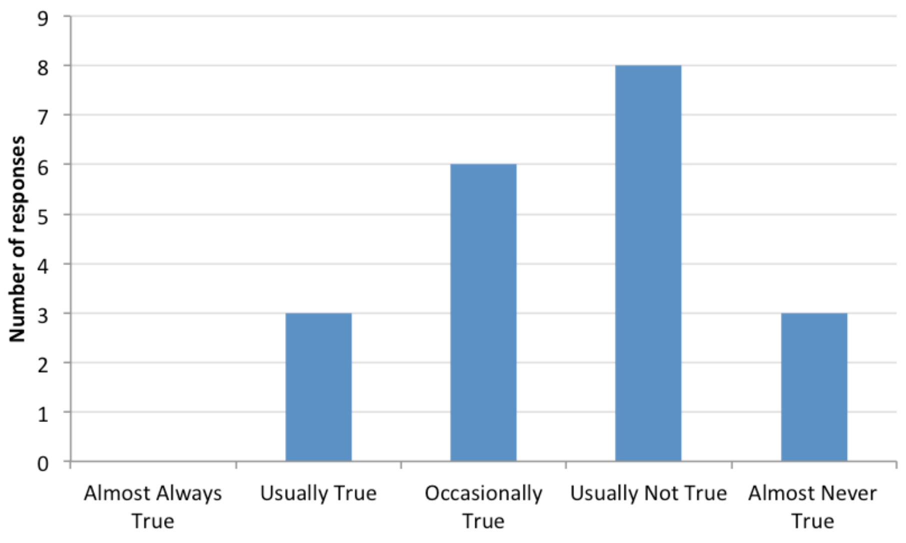


Figure 18: Responses regarding if participants found performing the demonstrations annoying

In fact, almost all the participants were unsatisfied with the available time windows (Figure 19) and wanted to be able to customize it (Figure 20).

> "I wasn't satisfied with the available time intervals." *(P3)*

> "I didn't like the fact that the time (interval) was fixed, I think it would be better if the user could define it freely." *(P9)*

> "I think extending a bit the time frames would be very useful" *(P10)*

> "The time (interval) could be bigger." *(P19)*

Also some interesting situations related to the demonstration time window happened during the first time using the application: P9 and P12 thought that they should start the demonstration only after setting the time window; P14 and P20 thought that the application would start monitoring what they were doing from time to time and then it would send a notification recommending some rules. The latter situation was actually recommended later as a complement for automating by demonstration approach by P1, P8 and P18.

“The app should keep a log of the detect rules over time and recommend the ones that happened more frequently.” *(P1)*

“I think that the app could silently detect some patterns, and when the app had some confidence about the pattern, it would suggest me to add the automation rule.” *(P8)*

“It is important to consider some behavioral aspects besides the simple momentary use of the smartphone.” *(P18)*

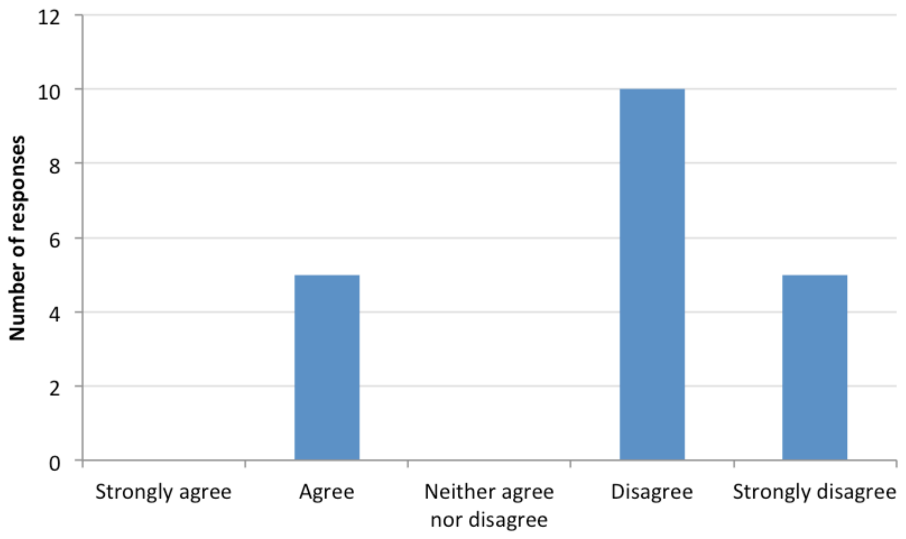


Figure 19: Responses regarding if the participants were satisfied with the available demonstration time windows

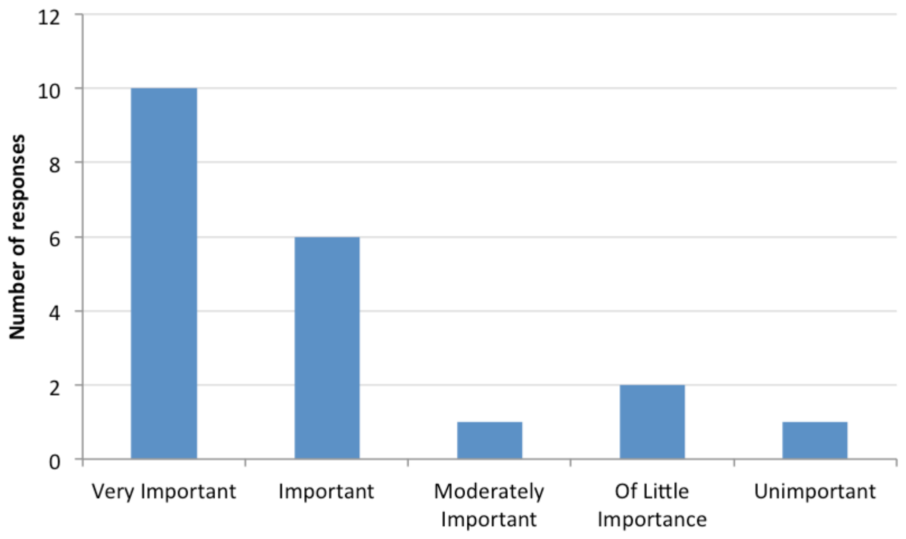


Figure 20: How important the participants considered being able to fully customize the demonstration time window

We were interested to investigate the interface adequacy of this new prototype as well. One of the improvements of Keep Doing It 2.0 over Keep Doing It 1.0 was the visual presentation of the rule recommendations. Hence, we asked the participants if the use of different colors as well as the icons helped them in the process of comparing the rule recommendations. As we can see in Figure 21 and Figure 22, most of the participants thought these changes helped them to compare the recommendations, specially the use of icons.

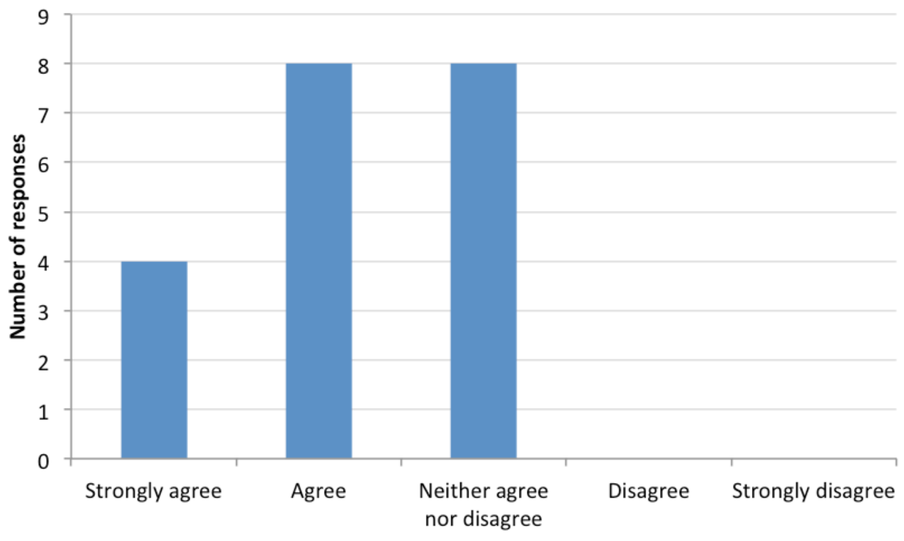


Figure 21: Responses regarding if the colors helped to compare the rules

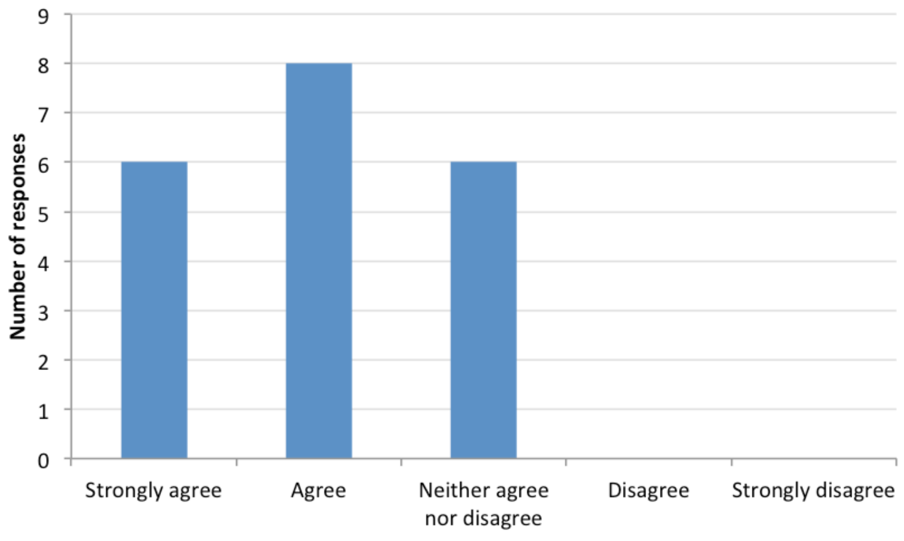


Figure 22: Responses regarding if the icons helped to compare the rules

Another improvement that the participants enjoyed was the bottom bar that displays the detected interactions. P8 even commented that this increased the system intelligibility:

> “The strip at the bottom of the app allowed me to see the recent detected system events, so I could understand why the automation rules were suggested” *(P8)*

However, P8 was unaware of the functionality to ignore a detected interaction from the inference process by tapping its icon on the bottom bar. In fact, only 35% of the participants were aware of this functionality (which intentionally was not described in the email sent to the participants), and P11, although aware, did not see much benefit on it (Figure 23). This reinforces the aforementioned assumption that, despite the great number of actions on the log files related to regenerating the recommendations, the participants were clicking on the detect interactions in the bottom bar mostly because there were no recommendations and they wanted to see if it would somehow make a recommendation appear.

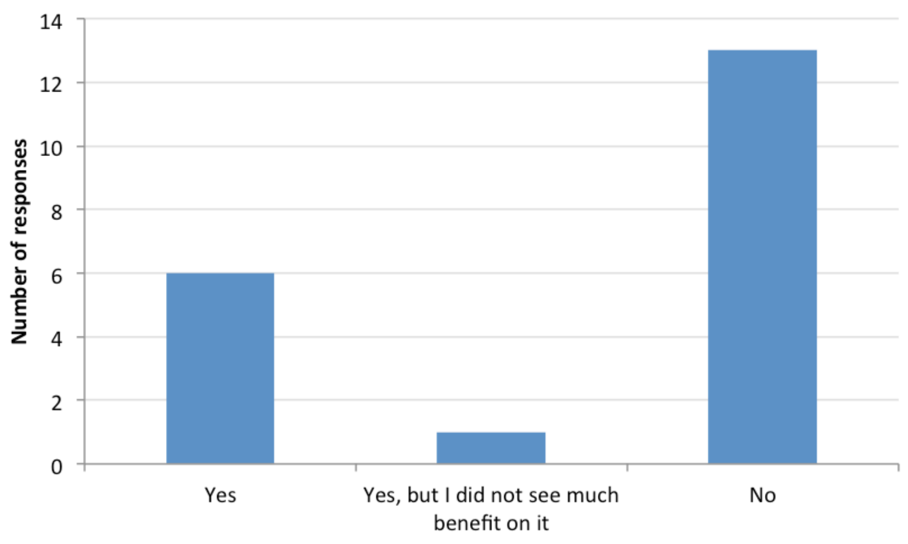


Figure 23: Number of participants aware of Keep Doing It 2.0 functionality to ignore a detected interaction from the inference process by tapping its icon on the bottom bar

Finally, regarding the system intelligibility, we also wanted to know if the participants were aware that the application showed the recommendations in a reverse chronological order, since the order in which recommendations are presented may have an effect on the users’ experience and interaction (Knijnenburg et al., 2012). 70% of the participants were aware of the order;

however, 43% of these did not care about the order of the recommendations (Figure 24). Probably the order did not matter much since most of the recommendations sets generated during the study had either only 1 or 2 recommendations, which in any case fits the screen.

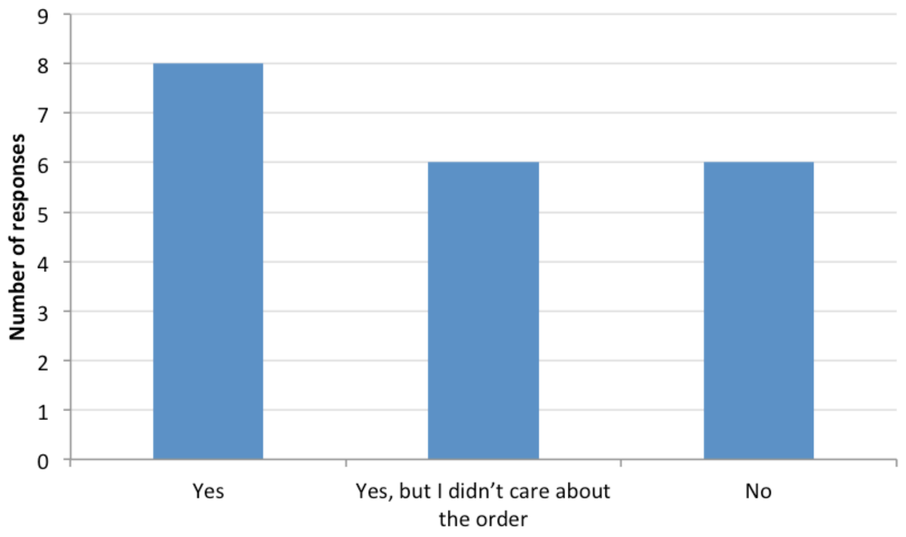


Figure 24: Number of participants aware that the recommendations were shown in reverse chronological order

5.5 Discussion

Overall the participants had little difficulty understanding the basic concept of automating smartphone tasks by demonstration, regardless of their previous technological knowledge. Given surprisingly little instruction, after some sessions all of them were able to comprehend and create rules that covered all the supported interactions and context information. Most participants rated the application positively regarding ease of use and ease of learning, although during the first uses of the application, most participants could not make the application generate any recommendation.

Whenever the list of recommendations was empty, there was a brief message explaining to the participant that there was no recommendation because the application did not detect any automation within the selected time window; however, this message probably was not clear enough for them. Perhaps the message would be more effective for first time users if it included an example of how to demonstrate a simple automation rule. One of the main reasons why they could not see any recommendations during the first uses of the application was due to the limited demonstration time window. The default time of 15 seconds was clearly not enough time to demonstrate automations for most of them, and to some extent not even the 60 seconds time window. Participants would rather have a more extended interval or freely set a custom time window.

Despite the fact that most of the participants rated the approach has being an easier and faster way to create automation rules than setting rules from scratch, some participants pointed out that it would be nice to be able to eventually create rules from scratch, too. Some participants would also like to see recommendations regarding long-term behavior patterns. One of the benefits of our approach is that it can be used seamlessly along with both the approaches of setting rules from scratch or based on usage patterns. The former can be used whenever no rules are available or none of the available ones is close to the user intention. Conversely, the latter can be used when the user decides to generate recommendations based on longer periods of time (a whole day, a week or even several months), since the patterns are extracted from nothing more than a series of continuous demonstrations.

The recommendations set sizes were generally small. This indicates that the participants do not tend to perform many different interactions at once in a short period of time. This behavior also affected the size of the rules recommended: the rules were mostly simple (event + action), since our inference process only adds conditions to the recommendations if one or more interactions were detected before the one that was considered the event. Consequently, in general the application would not deliver a good accuracy in case the participant was expecting a rule recommendation to contain one or more conditions related to smartphone features that the participant did not interact with during the demonstration period. This issue regarding the conditions demonstration is equivalent to the one observed during the lab study. It is clear from the logged data that the participants were indeed expecting the application to add conditions automatically based on the status of the features on the smartphone. For instance, the conditions that the participants usually added to the rules being edited were either related to the Wi-Fi or the phone’s battery.

Overall, contradicting the opinion of the participants of the lab study, the remote study participants seemed much more concerned whether there was a reasonable accurate recommendation in a set than with the set size or with how the recommendations were sorted. Hence, probably we could mitigate the issue to demonstrate conditions by including some additional recommendations that contained conditions based on some of the system features current statuses. Adding more recommendations could also contribute to the serendipity, which some participants pointed out as something good about the system, although not the focus of the approach.

However, the number of possible feature statuses and interaction combinations is very high, and therefore the inference process would need some additional intelligence in order to recommend only the additional rules with conditions that the participants were probably expecting given the demonstration scenario. An extended use of the application, as well as feedback from users, could give the application enough data to associate weights to the relation between different interactions and context information. For instance, for some users, unlocking the device and then opening an application might not make as much sense as unlocking the device and then turning the Wi-Fi on, and vice-versa. This weight system could also help to increase the generated recommendations perceived relevance, which (although it had positive ratings) was the least rated criterion in the post-session, compared to intelligibility and ease of choice.

Regarding the ease of choice, although the participants indicated that the colors and icons in the rules presentation helped them to compare the recommendations, we cannot be sure to which extent, since only few recommendations were generated at the same time. Hence, it is important to evaluate this criterion again after altering the application to show the additional recommendations. The same goes for evaluating the order of the recommendations and checking if the users would still most of the times select the first position of the list as they did during this study.

The participants in general could not get the gist of the functionality that allows the user to ignore an interaction from the sequence of detected interactions in which the recommendations are based. Probably because the number of recommendations generated and of interactions detected during a demonstration period was usually low, they could not see any benefit on it. And most did not even realize that this functionality was available. For most of the participants the sole purpose of the bottom bar was to confirm to the user which interactions were detected during the demonstration period, which indeed increased the participants understanding about why some rules were recommended.

Finally, although we did not strive for accuracy, overall the participants thought that the recommendations were pretty accurate. Based on our findings, aside from the issue to demonstrate conditions, one of the reasons why some recommendations were not very accurate was because participants expected the application to collect and use more information about their interactions with the smartphone and its context than the study had allowed them. Some of the participants even expected support for application-specific rule recommendations, which we avoided to respect the users' data privacy. But apparently, privacy was not a big concern among the participants. However, as mentioned by one of the participants, it is important to include in a final version of the application an explanation for the user about which data is collected and how it is used. Further research into privacy concerns would be interesting once more detailed data about the user is used for the inference process.

6 Conclusion

The overarching goal of this dissertation was to propose an approach that supports end users to automate their smartphones. In particular, this dissertation explored an approach in which end users can program their own intelligent smartphone behaviors by demonstration. Briefly, this dissertation has contributed with the following:

- It proposed an approach to automate smartphone tasks by demonstration, which makes it easier and faster for end users to create the automation rules compared to the usual approach of setting rules from the scratch.
- It demonstrated the feasibility of the approach by implementing it on a working prototype called Keep Doing It, an automation application that continuously collects a user's interactions with her or his smartphone to (in case the user asked for) use them shortly afterwards in an inference process to recommend an automation rule that the user may want. Succinctly, the inner logic to create automation rules with the application is "keep doing what I just did."
- It described different user correction strategies that an application using the proposed approach should support in order to make it more feasible, since inferring a user's intent is prone to uncertainty.
- It showed the positive attitude users had towards using the proposed approach. Participants of both conducted studies thought that in general the approach was an interesting, easy, fast and simple way to create automation rules; and they were excited to use Keep Doing It if it became publicly available. Keep Doing It also eventually helped the participants to discover new and interesting automation rule possibilities, which they probably would not have discovered if they were setting the rules from scratch.
- It investigated how well users can create the desired automation rules using demonstration and it identified the difficulties they experience during the process. On the one hand, in general, even with barely guidance

on how to use the application, participants of the studies were able to create accurate rules that in sum covered all the supported interactions so far. On the other hand, most participants had issues with the limited demonstration time windows available; and they also had issues with understanding how to demonstrate rules with conditions, since they often expected the current status of smartphone features with which they did not interact to be used as conditions.

- It presented lessons that can help designers and developers build automation tools that are easy to use and efficient. Moreover, we believe that the proposed approach and these lessons can be readily extended to apply to the end-user development of more complex context-aware applications, mobile or not.

6.1 Future Work

There are several ways to continue the work presented in this dissertation. First, longer remote studies could be conducted. The remote studies in this research were conducted in less than a week and participants were instructed to use the prototype daily. However, running longer studies would allow us to investigate how people would use the approach more naturally and under which circumstances. Also, we could compare creating rules by demonstration with the manual setting of rules from scratch. Pools of subjects would be instructed to use both approaches. Their speed of learning, ability to create specified task automations, and so forth would be compared. Performing such an experiment was beyond the scope of this research, since at the time the number of subjects needed to produce statistically useful results, as well as the time to run the study would be considerably higher than what we had available for this dissertation. That is why, instead of attempting such an experiment, we elected to evaluate our approach in this dissertation in a more qualitative and subjective way. Discovering whether users perceived automating by demonstration to be easy to use was just as interesting as finding out whether it was actually better than the traditional approach of setting rules from scratch. Nonetheless, we have collected a wealth of data that can be further triangulated and analyzed before conducting these studies.

Second, research could be conducted involving more complex automations in which the demonstration is based on the interaction with other mediums, such as wearable devices, cars or smart environments.

Third, we could conduct a study to investigate how our approach could make smartphone automation more accessible to blind people, who would benefit greatly from the automatic behavior.

Lastly, Keep Doing It presents several opportunities for improvement before conducting new studies, such as:

- Implement a rule engine to execute the saved automation rules;
- Support the detection of more interactions and context information, which would allow the users to demonstrate more diverse rules;
- Allow users to customize the demonstration time window, or even implement a record mode, so they can more easily inform the application the period of demonstration;
- Implement and evaluate new rule correction strategies, as allowing users to merge, combine and chain different recommendations in one to create more complex automations;
- Improve the inference process, perhaps by using individual and group patterns from usage to allow attributing levels of priority and confidence to each recommendation, to decide which combinations of interactions are more likely to form a rule; and consequently decide which conditions could be automatically included even if the user did not interacted with a feature to demonstrate them.

7 References

Antila, V., Polet, J., Lamsa, A., & Liikka, J. (2012). RoutineMaker: Towards end-user automation of daily routines using smartphones. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)* (pp. 399–402). doi:10.1109/PerComW.2012.6197519

Aztiria, A., Augusto, J. C., & Izaguirre, A. (2008). Autonomous Learning of User's Preferences Improved through User Feedback. *BMI*, *396*, 72–86.

Blackwell, A. (2002). What is programming. In *14th workshop of the Psychology of Programming Interest Group* (pp. 204–218).

Chen, J.-H., & Weld, D. S. (2008). Recovering from Errors During Programming by Demonstration. In *Proceedings of the 13th International Conference on Intelligent User Interfaces* (pp. 159–168). New York, NY, USA: ACM. doi:10.1145/1378773.1378794

Consolvo, S., Harrison, B., Smith, I., Chen, M. Y., Everitt, K., Froehlich, J., & Landay, J. A. (2007). Conducting In Situ Evaluations for and With Ubiquitous Computing Technologies. *International Journal of Human-Computer Interaction*, *22*(1-2), 103–118. doi:10.1080/10447310709336957

Cypher, A., & Halbert, D. C. (1993). *Watch what I Do: Programming by Demonstration*. MIT Press.

Danado, J., & Paternò, F. (2012). Puzzle: A Visual-Based Environment for End User Development in Touch-Based Mobile Phones. In M. Winckler, P.

Forbrig, & R. Bernhaupt (Eds.), *Human-Centered Software Engineering* (pp. 199–216). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-34347-6_12

De Lucia, A., Francese, R., Risi, M., & Tortora, G. (2012). Generating Applications Directly on the Mobile Device: An Empirical Evaluation. In *Proceedings of the International Working Conference on Advanced Visual Interfaces* (pp. 640–647). New York, NY, USA: ACM. doi:10.1145/2254556.2254674

Dey, A. K. (2001). Understanding and Using Context. *Personal Ubiquitous Comput.*, *5*(1), 4–7. doi:10.1007/s007790170019

Dey, A. K., Hamid, R., Beckmann, C., Li, I., & Hsu, D. (2004). A CAPpella: Programming by Demonstration of Context-aware Applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 33–40). New York, NY, USA: ACM. doi:10.1145/985692.985697

Dey, A. K., & Mankoff, J. (2005). Designing Mediation for Context-aware Applications. *ACM Trans. Comput.-Hum. Interact.*, *12*(1), 53–80. doi:10.1145/1057237.1057241

Dey, A. K., Sohn, T., Streng, S., & Kodama, J. (2006). iCAP: Interactive Prototyping of Context-Aware Applications. In K. P. Fishkin, B. Schiele, P. Nixon, & A. Quigley (Eds.), *Pervasive Computing* (pp. 254–271). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/11748625_16

Ericsson, K., & Simon, H. (1993). *Protocol analysis; Verbal reports as data (revised edition). Bradford Books*. MIT Press, Cambridge, MA.

Häkkilä, J., Korpipää, P., Ronkainen, S., & Tuomela, U. (2005). Interaction and End-User Programming with a Context-Aware Mobile Application. In M. F. Costabile & F. Paternò (Eds.), *Human-Computer Interaction - INTERACT 2005* (pp. 927–937). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/11555261_73

Holloway, S., & Julien, C. (2010). The Case for End-user Programming of Ubiquitous Computing Environments. In *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research* (pp. 167–172). New York, NY, USA: ACM. doi:10.1145/1882362.1882398

Knijnenburg, B. P., Willemsen, M. C., Gantner, Z., Soncu, H., & Newell, C. (2012). Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction*, *22*(4-5), 441–504. doi:10.1007/s11257-011-9118-4

Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., & Campbell, A. T. (2010). A survey of mobile phone sensing. *IEEE Communications Magazine*, *48*(9), 140–150. doi:10.1109/MCOM.2010.5560598

Lieberman, H. (2001). *Your Wish is My Command: Programming By Example*. Morgan Kaufmann.

Lieberman, H., Paternò, F., Klann, M., & Wulf, V. (2006). End-User Development: An Emerging Paradigm. In H. Lieberman, F. Paternò, & V. Wulf (Eds.), *End User Development* (pp. 1–8). Springer Netherlands. Retrieved from http://link.springer.com/chapter/10.1007/1-4020-5386-X_1

Lucas Cuesta, J. M., Ferreiros López, J., Aztiria, A., Augusto, J. C., & McTear, M. (2010). Dialogue-based management of user feedback in an autonomous preference learning system.

Maués, R. de A., & Barbosa, S. D. J. (2013a). Automating by demonstration: making smartphone automations accessible for end-users, the elderly and the blind. *INTERACT 2013 Workshop on Rethinking Universal Accessibility: A Broader Approach Considering the Digital Gap*. Retrieved from http://sipt07.si.ehu.es/WS/RethinkingUA/Papers/wsrua2013_submission_Maues.pdf

Maués, R. de A., & Barbosa, S. D. J. (2013b). Keep Doing What I Just Did: Automating Smartphones by Demonstration. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services* (pp. 295–303). New York, NY, USA: ACM. doi:10.1145/2493190.2493216

Meskens, J., Luyten, K., & Coninx, K. (2010). D-Macs: Building Multi-device User Interfaces by Demonstrating, Sharing and Replaying Design Actions. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology* (pp. 129–138). New York, NY, USA: ACM. doi:10.1145/1866029.1866051

Paternò, F. (2013). End User Development: Survey of an Emerging Field for Empowering People. *ISRN Software Engineering*, *2013*. doi:10.1155/2013/532659

Pu, P., Chen, L., & Hu, R. (2011). A User-centric Evaluation Framework for Recommender Systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems* (pp. 157–164). New York, NY, USA: ACM. doi:10.1145/2043932.2043962

Ravindranath, L., Thiagarajan, A., Balakrishnan, H., & Madden, S. (2012). Code in the Air: Simplifying Sensing and Coordination Tasks on Smartphones. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems &#38; Applications* (pp. 4:1–4:6). New York, NY, USA: ACM. doi:10.1145/2162081.2162087

Robles, R. J., & Kim, T. (2010). Review: context aware tools for smart home development. *International Journal of Smart Home*, *4*(1).

Tillmann, N., Moskal, M., de Halleux, J., & Fahndrich, M. (2011). TouchDevelop: Programming Cloud-connected Mobile Devices via Touchscreen. In *Proceedings of the 10th SIGPLAN Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software* (pp. 49–60). New York, NY, USA: ACM. doi:10.1145/2048237.2048245

Truong, K. N., Huang, E. M., & Abowd, G. D. (2004). CAMP: A magnetic poetry interface for end-user programming of capture applications for the home. In *UbiComp 2004: Ubiquitous Computing* (pp. 143–160). Springer.

Van Deemter, K., Krahmer, E., & Theune, M. (2005). Real Versus Template-Based Natural Language Generation: A False Opposition? *Comput. Linguist.*, *31*(1), 15–24. doi:10.1162/0891201053630291

Appendix A: Lab User Study Material

This appendix displays the material used on the lab user study conducted at the first stage of our research: the questions that guided the post-task and the post-study interviews.

A.1 Post-Task Interview Questions

A.1.1. In Portuguese

1. A automatização que você queria lhe foi sugerida? Caso negativo, por que você acredita que a automatização desejada não lhe foi sugerida?
2. Caso houve sugestões que não fizeram sentido para você, cite quais foram e por quê.
3. O que você achou da quantidade de sugestões? Qual seria o limite de sugestões que você aceitaria receber simultaneamente?
4. O que você achou da ordem das sugestões? Como você preferia ou esperava que esta ordem fosse?

A.1.2. In English

1. Did the application suggest the automation you wanted? If not, why do you believe that the desired automation was not suggested?
2. In case there were suggestions that did not make sense to you, cite which were them and why.
3. What did you think of the amount of suggestions? What would be the limit of suggestions that you would accept to receive simultaneously?
4. What did you think of the order of the suggestions? How would you prefer or expect this order to be?

A.2 Post-Study Interview Questions

A.2.1. In Portuguese

1) O que você achou da descrição de cada automatização gerada? Quais foram a mais clara e a mais confusa e por quê?
2) Sem se limitar à lista de eventos atualmente suportados pelo aplicativo, quais automatizações você gostaria de programar (ou já programou, caso já tenha tido contato com automatizações) no seu celular?
3) Quais as vantagens que você acredita que esta abordagem de automatizar o celular possui e por quê?
4) Quais as desvantagens que você acredita que esta abordagem de automatizar o celular possui e por quê?
5) O que você achou da janela de tempo disponível para demonstrar a regra de automação?
6) O que você acharia de poder olhar o histórico das suas ações para definir um período como sua demonstração?
7) Você prefere buscar uma tarefa feita há muito tempo atrás no histórico ou realizar uma nova demonstração na hora? Por quê?
8) O que esse aplicativo precisaria ter para você querer utilizá-lo? E o que ele precisaria ter para você querer comprá-lo por cinco reais?

A.2.2. In English

1) What did you think about the description of each automation rule? What were the clearest one and most confusing one and why?
2) Without limiting yourself to the list of events currently supported by the app, which rules you would like to automate (or already automated, if you have already had contact with smartphone automation) on your phone?
3) What advantages do you believe that this smartphone automation approach has and why?
4) What disadvantages do you believe that this smartphone automation approach has and why?
5) What did you think of the time window available to demonstrate the automation rule?

6) Would you like to be able to look at your actions history file to define the demonstration period?
7) Would you rather look for a task done long ago in the history file or demonstrate the task again? Why?
8) What would this application need in order for you to want to use it? And what would it need in order for you to want to buy it for 2 dollars?

Appendix B: Remote User Study Material

This appendix includes the email invitation sent to the participants in the beginning of the remote user study (in Portuguese), and also the questionnaires given to the participants to complete at the end of each app usage session (in-app questionnaire) and at the end of the five-day study (email survey).

B.1 Email Invitation

"Prezado(a),

Venho por meio deste lhe convidar a participar de um estudo remoto referente à minha pesquisa de mestrado. Para participar basta instalar e utilizar o aplicativo enviado em anexo pelos próximos 5 dias. Os horários e o número de vezes em que irá utilizar o aplicativo é de sua escolha, no entanto recomendamos que você procure explorar o aplicativo o máximo que puder ao longo de cada dia.
O aplicativo tratado no estudo é um protótipo de aplicativo de automação de tarefas de Smartphone. Ou seja, ele é um aplicativo que monitora continuamente o contexto e as atividades realizadas no seu Smartphone, a fim de realizar automaticamente alguma ação ou tarefa sempre que algum evento for detectado e certa condição for atendida. Este evento, condição e ação constituem uma regra de automação (automation rule), a qual é definida por você. Um exemplo de regra seria: Ao chegar no trabalho (evento) coloque o celular no modo vibratório (ação).
O objetivo deste estudo não é avaliar o desempenho do protótipo em monitorar e executar as regras definidas por você (funcionalidade que nesta versão não está habilitada), mas sim avaliar a criação das regras de automação em si. O protótipo segue uma abordagem de criar as regras por demonstração, ou seja, ele procura criar uma regra de automação baseada na própria interação momentânea que você teve com seu Smartphone.
Você então apenas tem que utilizar o aplicativo para criar quantas regras quiser. Ao final de cada uso do aplicativo para criar uma regra (tendo salvo ela ou não), o

aplicativo irá lhe enviar uma notificação para que você preencha (no próprio aplicativo) um pequeno questionário sobre o uso. Caso o aplicativo não lhe envie uma notificação após certo uso, ou caso você sem querer ignore a notificação, basta ir nas configurações do aplicativo para responder ao questionário.

O aplicativo não precisa estar conectado a Internet para funcionar, porém é importante que ao menos ao final de cada dia (ou na pior das hipóteses ao final dos 5 dias) você conecte seu Smartphone a Internet para que o aplicativo possa enviar os questionários previamente preenchidos, assim como enviar um pequeno registro (log) coletado do uso do aplicativo.

O registro não conterá informações privadas do seu Smartphone, e sim apenas dados referentes ao uso do aplicativo. Além disso, tanto o registro quanto os questionários serão analisados de forma anônima durante a pesquisa. O aplicativo buscará enviar estes dados pela Internet a cada vez que você preencher o questionário. Porém, caso você não esteja conectado, as respostas serão salvas e enviadas apenas da próxima vez que você preencher o questionário. Caso precise enviar manualmente estes dados, basta acessar novamente uma opção para isto nas configurações do aplicativo.

Ao final dos 5 dias um questionário final sobre o estudo lhe será enviado por e-mail.

Em suma, sua participação consiste em:

- Instalar e utilizar o aplicativo para criar regras de automação
- Responder um breve questionário após cada uso do aplicativo
- Enviar os questionários respondidos e o registro (log) pela Internet
- Responder um questionário final que lhe será enviado por e-mail ao final do estudo

Caso possa participar, por favor responda este e-mail confirmando se conseguiu instalar o aplicativo sem problemas. Agradeço desde já pela sua atenção. Se houverem quaisquer outras dúvidas sobre o aplicativo, a instalação ou o estudo em si, favor me contatar pelos e-mails maues.rodrigo@gmail.com ou rmaues@inf.puc-rio.br.

OBS: Para instalar o aplicativo em anexo (.apk) você precisa primeiramente permitir a instalação de aplicativos a partir de outras fontes que não o Google Play Store. Para fazer isto no seu Smartphone vá em Configurações > Segurança >

Habilitar fontes desconhecidas. Depois disto basta passar o .apk em anexo para o seu Smartphone ou o baixar diretamente abrindo este e-mail pelo seu Smartphone. Ao clicar no .apk o próprio Smartphone irá lhe guiar com a instalação. Caso prefira, outra opção após passar o .apk para seu Smartphone é utilizar um instalador de aplicativos.

Atenciosamente,”

B.2 Post-Session Questionnaire

1) I understood why these rules were recommended to me.
    a) Strongly disagree
    b) Disagree
    c) Neither agree nor disagree
    d) Agree
    e) Strongly agree
2) Comparing the recommendations was easy.
    a) Strongly disagree
    b) Disagree
    c) Neither agree nor disagree
    d) Agree
    e) Strongly agree
3) Based on my previous actions and context, the recommended rules seem relevant.
    a) Strongly disagree
    b) Disagree
    c) Neither agree nor disagree
    d) Agree
    e) Strongly agree
4) The app showed:
    a) Too few recommendations
    b) A reasonable number of recommendations
    c) Too many recommendations

5) How close to the rule that you were expecting to see was the best-recommended rule?
    a) Not even close
    b) Not so close
    c) Close enough
    d) Perfect match
    e) I was not expecting to see any particular rule
6) *[If the participant was expecting to see a particular rule]* What was the automation rule that you were expecting to see (optional)?

B.3 Post-Study Questionnaire

1) It is easy to learn to use the app to create an automation rule.
    a) Strongly disagree
    b) Disagree
    c) Neither agree nor disagree
    d) Agree
    e) Strongly agree
2) Using the app takes little effort.
    a) Strongly disagree
    b) Disagree
    c) Neither agree nor disagree
    d) Agree
    e) Strongly agree
3) The approach of setting rules by demonstration makes it easier to create automation rules than to set them from scratch.
    a) Almost Never True
    b) Usually Not True
    c) Occasionally True
    d) Usually True
    e) Almost Always True
4) The approach of setting rules by demonstration makes it faster to create automation rules than to set them from scratch.

   a) Almost Never True
   b) Usually Not True
   c) Occasionally True
   d) Usually True
   e) Almost Always True

5) The recommendations made me aware of interesting automation rule options that I hadn't previously considered.
   a) Never
   b) Rarely
   c) Occasionally
   d) Frequently
   e) Very Frequently
6) I feel confident that the app respects my privacy.
   a) Strongly disagree
   b) Disagree
   c) Neither agree nor disagree
   d) Agree
   e) Strongly agree
7) I found annoying to demonstrate a rule.
   a) Almost Never True
   b) Usually Not True
   c) Occasionally True
   d) Usually True
   e) Almost Always True
8) I was satisfied with the available time intervals (15, 30, 45 and 60 seconds).
   a) Strongly disagree
   b) Disagree
   c) Neither agree nor disagree
   d) Agree
   e) Strongly agree
9) How important do you think it would for you to be able to fully customize the time intervals?
   a) Unimportant

b) Of Little Importance
c) Moderately Important
d) Important
e) Very Important

10) The icons used in the rules presentation helped me to compare the recommendations.
a) Strongly disagree
b) Disagree
c) Neither agree nor disagree
d) Agree
e) Strongly agree

11) The colors used in the rules presentation helped me to compare the recommendations.
a) Strongly disagree
b) Disagree
c) Neither agree nor disagree
d) Agree
e) Strongly agree

12) Were you aware that the system showed the recommendations in an order beginning from the most recent recorded interaction?
a) No
b) Yes
c) Yes, but I didn’t care about the order

13) Were you aware that that by clicking on a recorded interaction you could ignore it when generating the recommendations?
a) No
b) Yes
c) Yes, but I did not see much benefit on it

14) Would you use a full version of the app?
a) No
b) Yes
c) Yes, but only if it was available for free

15) Overall, I am satisfied with the approach of setting rules by demonstration.
a) Strongly disagree

b) Disagree

c) Neither agree nor disagree

d) Agree

e) Strongly agree

16) What rules would you like to create? Were you able to create any of those with the app?

17) What did you like about the approach of setting rules by demonstration and the app?

18) What did you dislike about the approach of setting rules by demonstration and the app?

19) What are your general comments and suggestions about the approach of setting rules by demonstration and the app?