Journal of Visual Languages and Computing 42 (2017) 60–75

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

ImAtHome: Making trigger-action programming easy and fun

Daniela Fogli[*], Matteo Peroni, Claudia Stefini

Department of Information Engineering, University of Brescia, Via Branze 28, 25123 Brescia, Italy

A R T I C L E   I N F O

Article history:
Received 9 January 2017
Revised 17 July 2017
Accepted 13 August 2017
Available online 18 August 2017

Keywords:
End-user development
Smart home
Trigger-action programming
Event-condition-action rules

A B S T R A C T

ImAtHome is an iOS application for smart home configuration and management built over Apple HomeKit, a framework for communicating with and controlling home automation accessories. This paper describes the design and development of the visual interaction language made available in ImAtHome for empowering end users, without programming skills, to create event-condition-action rules that control home behavior. It can be regarded as an alternative approach to traditional trigger-action programming interfaces, where the user must define such rules by means of "if-then" constructs. Last but not least, attention has been put to make the interaction style as much coherent as possible with other iOS applications. The paper finally presents a user experiment, carried out with 30 participants according to a between-subject protocol, to evaluate the usability of ImAtHome and compare it with the official app for home automation recently released by Apple.

1. Introduction

Human-Computer Interaction (HCI) and visual languages are becoming more and more important for research fields of Ambient Intelligence (AmI) [41] and Internet of Things (IoT) [2]. Indeed, user interfaces are fundamental for making AmI environments and IoT systems easy to use, by possibly empowering their inhabitants or consumers to install, configure and modify them over time.

Therefore, End-User Programming (EUP) [39] and End-User Development (EUD) [36] approaches are being proposed in these fields. EUP and EUD are traditionally oriented at empowering non-technical users with methods and tools that allow them to tailor and extend their software artifacts at use time. In AmI and IoT, these approaches may lead to transform end users from passive consumers of sensors, robots and smart devices scattered in the physical environment to active producers of complex and coordinated behaviors of such hardware/software components [4,10].

These approaches have been applied in particular to the smart home, where the idea is to provide the house inhabitants with methods and tools to modify and adapt home behaviors to their needs, in order to cope with the continuous request of user-system co-evolution [12]. EUP and EUD tools can indeed support users in creating simple commands for direct activation (e.g. "I am at home, please switch the oven on") or automatic and repeated execution (e.g. "At 7 a.m. raise shutters and play classic music").

From the analysis of the literature and commercial products [11,23] it emerges the event-condition-action (ECA) rules represent the most used paradigm in user interfaces for configuration and adaptation of smart homes by users without programming skills. In other terms, rule-based programming appears as a promising solution to create EUD tools in this field. Such tools usually allow users to perform trigger-action ("if, then") programming [42], by setting up the "if" and "then" parts with the help of available lists of events, conditions and actions (filtered-list metaphor), virtual puzzle pieces (jigsaw composition) or components to be put in a network (wired composition) [16].

This paper proposes an alternative interaction metaphor for trigger-action programming by helping end users create antecedent and consequent parts of the rules, without requiring them to think in terms of "if-then" constructs like computer scientists naturally do. In this sense, we talked about "unwitting" trigger-action programming [24]. To achieve this goal, the proposed metaphor splits rule creation in two steps, namely defining *scenes* followed by defining *rules* on the basis of available scenes. Scenes are sets of actions that operate simultaneously on smart devices, which can be also manually activated by the user, thus becoming high-level commands for the home. Rules are defined to make the smart home able of activating itself some given scenes on the basis of the occurrence of an event, possibly combined with one or more conditions. The metaphor also encompasses a more guided way of defining events and conditions for triggering rules, and does not impose to the user the constraint of defining the action only after the definition of the trigger, often present in other similar applications (e.g. IFTTT, Atooma, Tasker, and others). These ideas have been implemented in ImAtHome, an iOS mobile application.

* Corresponding author.
E-mail address: daniela.fogli@unibs.it (D. Fogli).

http://dx.doi.org/10.1016/j.jvlc.2017.08.003
1045-926X/© 2017 Elsevier Ltd. All rights reserved.

Another important issue, often neglected in scientific literature, is the cost of transforming a traditional house into a smart home. Current solutions usually require buying global services from companies that take care of smart home installation and maintenance or, alternatively, acquiring home automation boxes (e.g., Zipabox, Zibase, Vera, and eeDomus). The latter, in turn, requires someone in the family, called "guru" in [18], who is capable of taking care of system installation and personalization. As described in a field study carried out in France with 10 households using different home automation systems for a long period of time, in most cases, the guru is a software expert or a technician able to perform software programming activities and take care of hardware aspects [18]. Therefore, the pure end user, neither expert in software programming nor interested in it, seems practically excluded from the use of such kinds of tools.

To overcome this problem, ImAtHome allows end users to add gradually smart devices to their house, on the basis of their emerging needs and economic possibilities, according to a smooth approach to home automation. To this aim, ImAtHome has been built over the framework for home automation made available in iOS, namely HomeKit.[1] HomeKit is a framework for communicating with and controlling the automation accessories available in a house. It provides the developer with a way to create apps that automatically discover accessories in a house to easily associate them with rooms. It also makes available functionalities for executing scenes, by possibly activating them using Siri – the voice-controlled virtual assistant available in iOS. Finally, it foresees the trigger concept to allow rule-based programming, by exploiting conditional relationships concerning time, position and accessory states.

ImAtHome proposes itself as a hub application that manages all the devices currently available in a smart home, as well as those that will be acquired and included in the future. Recently, Apple has released, in the last version of iOS, its own app for controlling smart home, called Home. The interaction metaphor of this app is similar to ours since it supports trigger-action programming without the explicit use of "if-then" constructs; however, its visual language appears as not consistent with other iOS apps, thus affecting app usability.

This paper presents the design and implementation of ImAtHome, and a user experiment with 30 participants organized according to a between-subject protocol to compare the usability of ImAtHome with Apple Home app. More precisely, 14 participants were involved to evaluate the usability of ImAtHome (first user study); subsequently, further 16 users participated in the evaluation of Apple Home (second user study). Experiment results allowed us to assess the validity of the interaction metaphor (both applications were used successfully by non-tech savvy participants), and to demonstrate that ImAtHome is perceived as easier to learn and to use than Apple Home. Another interesting outcome of the user study with Apple Home is concerned with the different judgments provided by younger participants with respect to middle-age participants; the former appreciated much more this app, whilst the latter expressed several complaints, even though they should be the real target users of an app for smart home control. These differences did not emerge in the user study with ImAtHome.

The paper is organized as follows: Section 2 discusses related works in the IoT and AmI fields, with particular reference to the smart home and user interfaces for their configuration and management based on ECA rule programming. Section 3 describes the Apple HomeKit framework, including available guidelines for the development of user interfaces of applications based on HomeKit. Section 4 illustrates the design and development of the app ImAtHome, with examples of its operation. Section 5 presents the results of a first user study aimed at evaluating the usability and acceptability of ImAtHome. Section 6 presents the results of the comparison between ImAtHome and Apple Home app after the execution of a second user study. Section 7 discusses the main features of our application and its current limitations. Section 8 draws some conclusions and proposes hints for future work.

2. Related works

Smart home control and configuration by end users have been discussed in literature for several years - e.g., [6,38]. For example, the "Media Cubes" programming language is proposed in [6]: it is based on the physical arrangement of infrared remote cubes, which represent abstract functions whose combination allows one creating complex behaviors of a smart home. In the e-Gadgets project [38], instead, a visual editor is presented, where end users may define "synaptic associations" (cause-effect relationships) among different smart appliances available in a home.

ECA rule-based programing is at the basis of most of the approaches to EUD that have been proposed in IoT and AmI fields. One of the first frameworks for AmI based on a rule-based approach is described in [28]. Three different graphical user interfaces for rule creation were proposed for the framework [27], even though no usability study was reported in the paper. Barricelli and Valtolina [3], on the other hand, have delineated an extension of the ECA paradigm pairing it with the use of formula languages. Recently, Coutaz and Crowley [15] have presented AppsGate, a EUD environment for smart homes, which combines rule-based and imperative programming. This system consists of a server and a set of web-based clients for user interaction to be used on a PC or a tablet, as well as a set of devices compatible with the AppsGate Core World, which operates as an integration middleware. AppsGate has been qualitatively evaluated through an external field experiment with five families for three weeks, and through a "first-person experience" by the authors (a married couple) for more than one year in their own house. Even though AppsGate provides a high expressive power in terms of programming facilities, it offers a complex visual language and requires a wide screen to use it (the same authors admitted that they preferred to edit programs with their laptops). Furthermore, AppsGate also includes debugging aids and state monitoring features. In this way, its scope and goals are wider than ours, even though the authors highlighted that "user engagement in programming takes time and effort" ([15], p. 37). On the other hand, we are more interested in designing a mobile app characterized by a user experience that allows a natural and playful approach to ECA rule creation, without requiring users to think in terms of "programming activities". Another study carried out "in the wild" confirms these results: the professional installer configured rules, whilst most of the households did not use rules as they definitely represented an additional level of automation complexity [9].

Dahl and Svendsen [16] carried out a preliminary comparison among three paradigms for rule composition, namely filtered lists, wiring composition and jigsaw puzzle composition. Filtered lists, where condition-action compositions are obtained by selecting conditions and actions from respective lists, resulted to be the most intuitive; whilst, participants considered jigsaw puzzle composition the most playful and engaging type of interaction. The filtered lists metaphor has been recently adopted by several commercial and research applications, such as Tasker, Locale, Atooma, IFTTT and others. Tasker, Locale and Atooma applications have been compared in [37]: Tasker resulted to be the best tool in terms of expressiveness and Atooma resulted to be the easiest to use by end users. Ur and colleagues [42] have proved that IFTTT is easy

[1] https://developer.apple.com/homekit/.