Enabling Non-Technical Users to Shape their own Automation Experience

Carmelo Ardito
University of Bari Aldo Moro
via Orabona, 4 – 70125 Bari, Italy
carmelo.ardito@uniba.it

Paolo Buono
University of Bari Aldo Moro
via Orabona, 4 – 70125 Bari, Italy
paolo.buono @uniba.it

Giuseppe Desolda
University of Bari Aldo Moro
via Orabona, 4 – 70125 Bari, Italy
giuseppe.desolda@uniba.it

Rosa Lanzilotti
University of Bari Aldo Moro
via Orabona, 4 – 70125 Bari, Italy
rosa.lanzilotti@uniba.it

Maristella Matera
Politecnico di Milano
piazza Leonardo Da Vinci – 20134 Milano, Italy
maristella.matera@polimi.it

ABSTRACT

Internet of Things (IoT) technology offers many opportunities for users interested in automatizing tasks and activities in various domains. Unfortunately, most Task Automation Systems for orchestrating smart devices are scarcely adopted as they are based on interaction metaphors not suitable for non-technical users or because they permit only trivial synchronizations of smart-device behaviors. Our approach aims at overcoming the above limitations by enabling people without programming skills to exploit the abundance of resources (object functionality, produced data, related applications), and allow them to personalize the behavior of the system, so that they can be directly involved in the creation of Smart Interactive Experiences (SIEs), i.e. usage situations created by synchronizing the behavior of multiple smart devices in order to accommodate their everyday needs.

1 INTRODUCTION

The Internet of things (IoT) creates a network of objects that can communicate, interact and cooperate together to reach a common goal [3]; each device stops acting as a single device and becomes part of an entire full connected system. Because of these characteristics, IoT devices are often referred to as smart objects. They are increasingly pervading the environments we live in and can enhance our daily lives.

KEYWORDS

Internet of Things; Smart Object Modelling;
Smart Interactive Experience.

Workshop proceedings *Everyday Automation Experience'19*
In conjunction with CHI'19, May 5th, 2019, Glasgow, UK
Website: http://everyday-automation.tech-experience.at

Figure 1: An example of IFTTT applet.

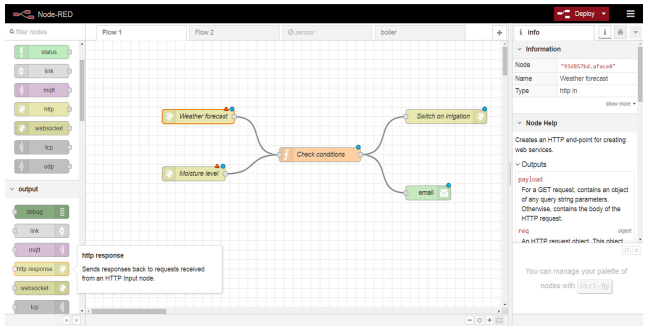

Figure 2. An example of a graph representation of a rule in Node-RED.

Programming the behavior of smart objects is currently a prerogative reserved for professional developers, as it requires the use of scripting languages and tools that can also vary depending on the underlying hardware. Allowing non-technical users to master the complexity of defining the behavior of smart objects - even when they are part of an ecology of devices, as for example in a smart home or in a museum - is a challenging application field for the End-User Development (EUD). Indeed, EUD is a research area whose goal is to support non-technical users in the creation of products and services tailored to their needs and desires [4, 10, 12, 17].

Some approaches have been proposed to support non-technical users to configure the behavior of smart objects. In particular, Task-Automation Systems (TAS) have become popular as they offer easy and intuitive paradigms to synchronize the behavior of objects and applications [13]. Through Web editors, users can synchronize the behavior of smart objects by defining event-condition-action (ECA) rules [16], which specify chains of conditional statements for triggering actions that change the status of coupled resources. An example of TAS based on ECA-rules is IFTTT (If This Then That), a popular Web platform that, by means of a wizard-based composition paradigm, guides users to create simple chains of conditional statements called "applets" [11]. As in the example of Figure 1, each applet consists of (1) a service that IFTTT tracks to detect if a specific event is triggered (e.g., the soil moisture level measured by a smart hygrometer is lower than 50%) and (2) another service that reacts to the triggered event by executing a specific action (e.g., send an email). Another class of TAS allows users to synchronize the behavior of smart objects by graphically sketching the interaction among the objects, for example by means of graphs that represent how events and data parameters propagate among the different objects to achieve their synchronization. An example of a graph-based tool is Node-RED, an open-source Web platform to compose both smart objects and Web services [15] by wiring nodes representing smart objects, control statements, functions, and debug procedures (see Figure 2).

Despite the popularity of TASs, their adoption is still limited. Indeed, on one hand, TASs implementing ECA rules, which are typically suitable for non-technical users, allow a trivial synchronization of smart-objects behaviors, without the possibility to define powerful constraints on events activation and actions execution. On the other hand, TASs based on graph metaphor do not match the mental model of most users because they do not think about "connecting" services, as demonstrated in different works on visual service composition [14, 18]. Moreover, TASs are typically conceived as general-purpose systems, claiming that one single design might satisfy the requirements of many domains. However, by observing people adopting our EUD tools during field studies, we realized that their generality often implies a scarce adoption by specific communities of end users [5].

Our position is that, in order to directly involve non-technical users in configuring the behavior of their smart objects, new approaches, based on high-level abstractions and adequate interaction paradigms, have to be developed that implement an EUD approach to customize and synchronize the behavior of resources, like smart objects and Web services, through ECA rules [8]. We are confident that our work will provide a contribution and will be able to arouse discussion at the workshop, as it directly addresses one of its research questions, namely "*How to allow people without programming skills to personalize the behavior of a system?*". Our research is also strictly related to another workshop question, i.e "*How to design for an efficient and enjoyable interplay of non-expert users and automated system?*".

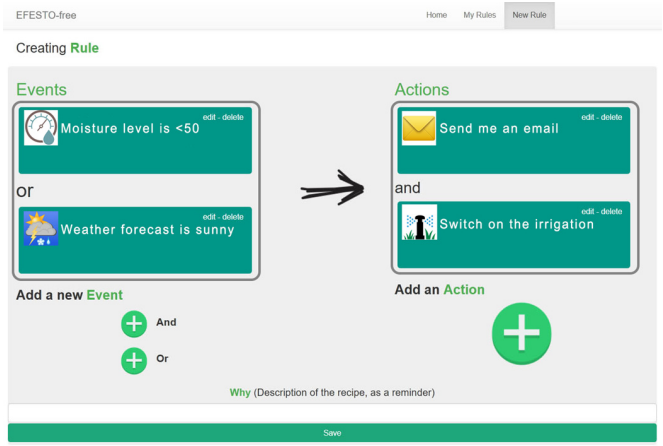

Figure 3. An example of ECA rule in EFESTO.

Next section describes the EFESTO platform and, with the help of a usage scenario, the challenges of defining a smart interactive experience and how our approach addresses them. The last section concludes the paper and suggests some research issues worth discussing at the workshop.

2. THE EFESTO PLATFORM

The EFESTO platform offers a visual design environment where the capabilities exposed by IoT devices (i.e., events and actions) can be combined by means of visual mechanisms that avoid writing code [8]. It capitalizes on a set of studies we performed to elicit and design visual composition metaphors adequate for non-technical users for web service composition [7, 9]. With respect to other TASs, the EFESTO approach promotes a richer set of high-level abstractions and operators to define rules and a visual notation that, despite the intrinsic complexity related to managing events and actions, is affordable by non-programmers. Figure 3 shows how the rule example of Figure 1 and 2 is finally displayed in EFESTO.

As already reported above, another limitation of TASs is that they are typically conceived as general-purpose systems. The recent version of EFESTO proposes an alternative visual framework to empower non-technical end users to build a semantic layer for TASs based on the definition of *custom attributes*. Similar to ontology concepts (e.g., see [6]), custom attributes are meant to add knowledge that can simplify the definition of ECA rules. The peculiarity of custom attributes is that they are *usage-driven terms*, specific to the application domain, that help the SIE designers make sense of digital resources, putting them in context with respect to the actual usage situations to be addressed. For example, in Cultural Heritage (CH), guides and curators are non-professional in Computer Science who might propose an SIE by creating objects that visitors of CH sites can bring with themselves, touch and manipulate for experiencing the site by receiving personalized information.

Thus, even though EFESTO is also general purpose it can be customized to several application domains. It proposes higher-level abstractions that allow end users themselves to define custom properties to characterize the semantics of smart objects, thus helping them to make sense of the available smart devices and digital resources and facilitate the definition of their cross interactions. In order to clarify how custom attributes and ECA rules can be exploited to define an SIE, let us consider the following usage scenario.

2.1 A usage scenario

Maria is a professional guide who wants to create a serious game for schoolchildren. This game can be played exploring the “smart” exhibition rooms of an archaeological museum. The game goal is to identify the display cases containing artifacts with a specific characteristic, for example belonging to the “Roman age”. Using an app available in her tablet, Maria sets the quest, i.e. the game current goal, to “Age – Roman”. Visitors explore the museum, identify the display case corresponding to Maria’s request and put the lantern (given them by Maria) close to it. If the answer is correct (namely, the display case exhibits artifacts of the Roman age) the lights of the display case turn green and the visitor’s score is increased. The game continues with Maria asking other questions and setting new quests.

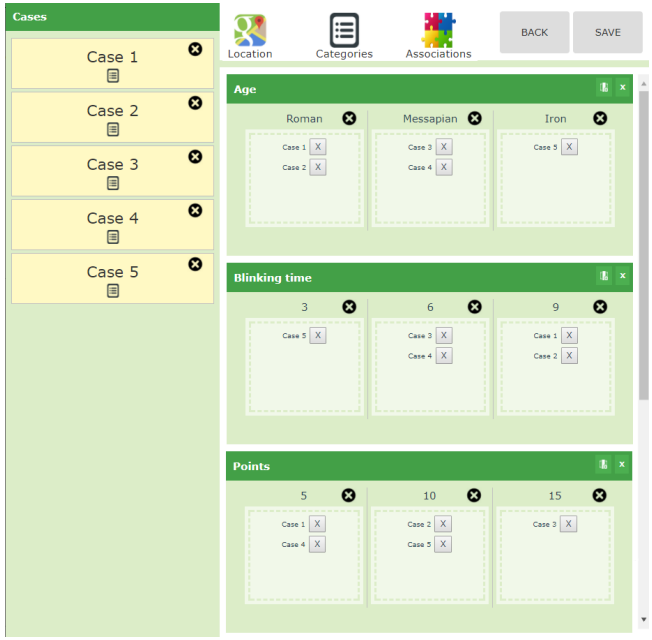

Figure 4. EFESTO tool for defining and assigning custom attributes.

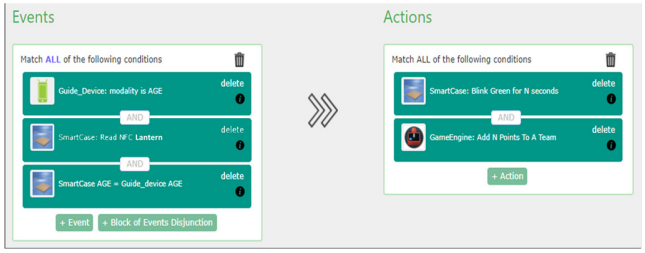

Figure 5. A single rule determining the behavior of multiple cases and lanterns.

Maria has a number of display cases and lanterns available. Cases are equipped with sensors to detect lanterns and "dialogue" with them. How can she organize the game by synchronizing the behavior of such objects? For each of them, she must define properties (i.e., attributes) that are useful for the game semantics; based on such properties she can then define the object behaviors.

For example, in the game a lantern is used to identify the visitor who brings it; thus, it has an "Owner" attribute that can take values such as "John Doe, Joe Bloggs, etc.". Each display case, depending on the contained objects, has attributes such as "Age" (Roman, Greek, etc..), "Worship" (of the dead, of the divinities, etc.), "Audio" (with a specific audio track for each case). It is worth remarking that both attributes and values are freely defined by Maria, depending on the goals she wants to pursue in the visit, thus we call them *custom attributes*.

After defining the custom attributes of the smart objects included in the visit, Maria specifies the Event-Condition-Action (ECA) rules to determine their behaviors. An example of a rule is: "*IF* the visitor puts his lantern close to a case with Case_Age = Quest_Age *THEN* turn on the Case_light and play the Case_audio file". Giving attributes to objects has two main advantages when creating ECA rules: 1) the language adopted in the rules is closer to the domain expert's vocabulary, 2) rules referring to custom attributes fit a class of rules, while without attributes several rules have to be created to determine the behavior of each lantern when it is close to each case. For example: "If the *lantern_012* is put close to the *display_case_032*, and the current quest is *Age – Roman*, then the display_case_032 turns on its light and display_case_032 plays the soundtrack <audio01.mp3>". This example shows that the definition of the SIE narrative without using custom attributes forces Maria to create a large number of rules including technical terms she might not be able to manage. Such rules refer to technical terminology (e.g., the NFC code of the lantern) that does not match the language adopted by the domain experts.

In our proposal, before creating ECA rules, Maria interacts with a visual tool offered by EFESTO, which allows her to assign *custom attributes* to each object/device by manipulating widget interfaces, without the need of coding. Custom attributes can be seen as conceptual tools that can allow designers to characterize the basic elements of a smart experience (i.e., smart objects and rules) with a semantics related to the content to be conveyed during the smart experience. In the example of Figure 4, she defines and assigns the attributes *Age*, representing the age of the artifacts contained by the cases, *Points*, representing the number of points the visitor gains if the answer is correct, *Blinking time*, indicating for how many seconds the case has to blink. From now on, the creation of ECA rules can exploit this terminology (see for example Figure 5). In addition, more general rules, i.e., parametric, can be created. In Maria's scenario, she does not need to define a multitude of very similar rules for coupling every single case and lantern, since they are all encompassed by the single rule shown in Figure 5.

3. CONCLUSIONS AND FUTURE WORK

Introducing adequate abstraction mechanisms facilitates smart object programming for an efficient and enjoyable interplay of non-expert users and automated systems [2]. Our ongoing work is about the definition of interaction paradigms that support the custom attribute creation phase in a more creative way, with the aim to improve the design of more expressive and richer SIEs (early results are reported in [1]). Furthermore, in

order to shape SIEs in more engaging and articulated narratives, we are investigating how to help designers framing ECA rules in 'scenes'. Finally, capitalizing on all our previous and ongoing experiences and on literature evidences, we are modelling a framework that includes dimensions related to user experience of interacting with smart environments. The framework aims at driving both the design and evaluation of SIEs.

We are confident that the work presented in this paper, as well as the future directions of our research outlined above, have the potential of generating interesting discussions around the theme of automating the interaction with smart environments.

REFERENCES

1. Ardito, C., Malizia, A., Desolda, G., Matera, M. and Lanzilotti, R. Advanced interaction paradigms to define smart visit experiences in the internet of things era. In *Proc. Doctoral Consortium, Posters and Demos at Biannual Conference of the Italian SIGCHI Chapter* CEUR Workshop proceedings (2017), 148-152.
2. Ardito, C., Buono, P., Desolda, G. and Matera, M. 2018. From smart objects to smart experiences: An end-user development approach. *INT J HUM-COMPUT ST 114* (2018), 51-68.
3. Atzori, L., Iera, A. and Morabito, G. 2010. The Internet of Things: A survey. *Computer Networks 54*, 15 (2010), 2787-2805.
4. Barricelli, B. R., Cassano, F., Fogli, D. and Piccinno, A. 2019. End-user development, end-user programming and end-user software engineering: A systematic mapping study. *Journal of Systems and Software 149* (2019), 101-137.
5. Casati, F. How End-User Development Will Save Composition Technologies from Their Continuing Failures. In *End-User Development - IS-EUD 2011*, Springer (2011), 4-6.
6. Corno, F., De Russis, L. and Roffarello, A. M. 2017. A Semantic Web Approach to Simplifying Trigger-Action Programming in the IoT. *Computer 50*, 11 (2017), 18-24.
7. Desolda, G., Ardito, C. and Matera, M. EFESTO: A Platform for the End-User Development of Interactive Workspaces for Data Exploration. In *Rapid Mashup Development Tools: First International Rapid Mashup Challenge, RMC 2015, Rotterdam, The Netherlands, June 23, 2015, Revised Selected Papers - ICWE '15*, Springer International Publishing (2016), 63-81.
8. Desolda, G., Ardito, C. and Matera, M. 2017. Empowering End Users to Customize their Smart Environments: Model, Composition Paradigms, and Domain-Specific Tools. *ACM Trans. Comput.-Hum. Interact. 24*, 2 (2017), Article N. 12 - 11-52.
9. Desolda, G., Ardito, C., Jetter, H.-C. and Lanzilotti, R. 2019. Exploring spatially-aware cross-device interaction techniques for mobile collaborative sensemaking. *International Journal of Human-Computer Studies 122* (2019), 1-20.
10. Fischer, G., Fogli, D. and Piccinno, A. Revisiting and Broadening the Meta-Design Framework for End-User Development. In *New Perspectives in End-User Development*, Springer International Publishing (2017), 61-97.
11. IFTTT, IFTTT. https://ifttt.com/.
12. Lieberman, H., Paternò, F., Klann, M. and Wulf, V. End-User Development: An Emerging Paradigm. In *End User Development*, Springer Netherlands (2006), 1-8.
13. Lucci, G. and Paternò, F. Analysing How Users Prefer to Model Contextual Event-Action Behaviours in Their Smartphones. In *End-User Development - IS-EUD 2015*, Springer International Publishing (2015), 186-191.
14. Namoun, A., Nestler, T. and Angeli, A. Conceptual and Usability Issues in the Composable Web of Software Services. In *International Conference on Web Engineering - ICWE 2010 Workshops - Revised Selected Papers*, Springer (2010), 396-407.
15. Node-RED, Node-RED. http://nodered.org/.

16. Pane, J. F., Ratanamahatana, C. A. and Myers, B. A. 2001. Studying the language and structure in non-programmers' solutions to programming problems. *INT J HUM-COMPUT ST 54*, 2 (2001), 237-264.

17. Paterno, F. and Wulf, V. (eds.) *New Perspectives in End-User Development*. Springer International Publishing, Cham, Switzerland, 2017.

18. Zang, N. and Rosson, M. B. What's in a mashup? And why? Studying the perceptions of web-active end users. In *Proc. IEEE Symposium on Visual Languages and Human-Centric Computing*, IEEE Computer Society (2008), 31-38.