End-User Development for the Internet of Things: EFESTO and the 5Ws composition paradigm

Giuseppe Desolda[1], Carmelo Ardito[1], Maristella Matera[2]

[1]Dipartimento di Informatica, Università degli Studi di Bari Aldo Moro
Via Orabona, 4 – 70125 – Bari, Italy
{name.surname}@uniba.it
[2]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano
Piazza Leonardo da Vinci, 32 – 20134 – Milano, Italy
maristella.matera@polimi.it

Abstract. This paper illustrates a composition paradigm and a related tool to express rules for *smart object composition*. The composition paradigm is characterized by operators for coupling multiple events and conditions exposed by smart objects, and for defining temporal and spatial constraints on rule activation. The composition paradigm has been designed based on the results of an elicitation study involving 25 participants.

Keywords: End-User Development of Mashups, Visual Paradigms for ECA Rule Expression, Internet of Thing

1 Introduction

In the last years, researchers have been devoting many efforts to improve technological aspects of the Internet of Things (IoT) with the result that it is now possible to program the behaviour of smart objects. Little attention has been however dedicated to social and practical aspects and, despite all the advances in the IoT field, end users still encounter many difficulties when trying to make sense of such technology. The research community agrees on the fact that the opportunities offered by IoT can be amplified if high-level abstractions and adequate interaction paradigms are devised to enable also **non-programmers** to program and synchronize the behaviour of smart objects [2]. This paper illustrates an approach that goes in this direction as it offers a visual interaction paradigm that allows end users to express rules for the composition of smart objects. The paradigm is based on a model, called 5Ws for the elements (*Which, What, When, Where, Why*) to be specified to build a rule. This model includes new operators for defining rules coupling multiple events and conditions exposed by smart objects, and for defining temporal and spatial constraints on rule activation. The paradigm has been designed during an elicitation study with 25 participants. It has then been implemented in a Web composition environment that extends the capability of EFESTO, a platform for the End-User Development (EUD) of Web mashups through which data provided by Web APIs can be integrated into unified visualizations [1].

adfa, p. 1, 2011.

2 EFESTO and its 5Ws Composition Paradigm

In this section we illustrate through an example the main features of the 5Ws composition paradigm. A user, who we suppose is a female, creates a rule to automatically turn on the coffee machine and roll-up the shutters when her smart bracelet detects that she has just woken up or the smart alarm clock rings. To create this rule, the user clicks the "New Rule" button in the navigation bar (Figure 1, circle 1) and the "Creating Rule" interface appears. The UI shows the main area in which a rule is defined. The left side is for specifying the triggering events, and the right side is to define the actions to be activated by the selected services.

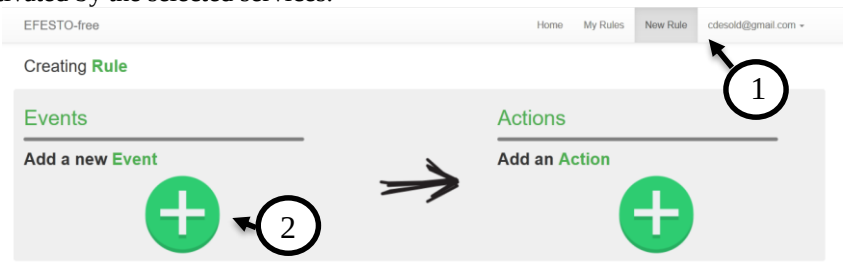

Figure 1. EFESTO: the interface for rule creation.

A **wizard procedure**, activated by the green "+" button highlighted by circle 2 in Figure 1, guides the users in defining the events in a **full-automated** fashion. The wizard sequentially shows some pop-up windows in which the service, the events and the conditions are specified. According to **WYSIWYG** approach, the wizard steps allow the user to define an event in terms of *Which* is the service to be monitored for detecting the triggering event (**Figure 2**a), *What* service event has to be monitored (**Figure 2**b), *When* and *Where* the event has to occur (**Figure 2**c). The specification of When and Where conditions is optional. At the end of the wizard procedure, the event is defined and its summary appears under the "Events" area (**Figure 2**d, circle 1). In the example of **Figure 2**d, the user has specified that the triggering event is the "Just Awake" condition of her "Bracelet" object.

Actions can be defined by clicking on the green "+" button highlighted (circle 3 in **Figure 2**d). The button activates a wizard that helps the user define an action in terms of *Which* service will execute the action as a consequence of the event(s), *What* action the service has to perform and *When* and *Where* the action can be performed.

In EFESTO, users may either define first all the events and then the actions, or define first a basic rule with one event and one action and then include new events and new actions. Events and actions can be added or removed at any time fostering a **dynamic modification of the running mashup**. Further events can be added by clicking one of the two green "+" buttons labeled And / Or (d, circle 2). Choosing the "And" button starts the definition of a new event that will cause the execution of the rule action(s) if all conditions of all events are satisfied. The "Or" button determines the definition of a new event that will cause the execution of the rule action(s), if the conditions of at least one event are satisfied. Once the rule is created, it can be saved by entering a short description of the rule (the *Why* in the 5W model).

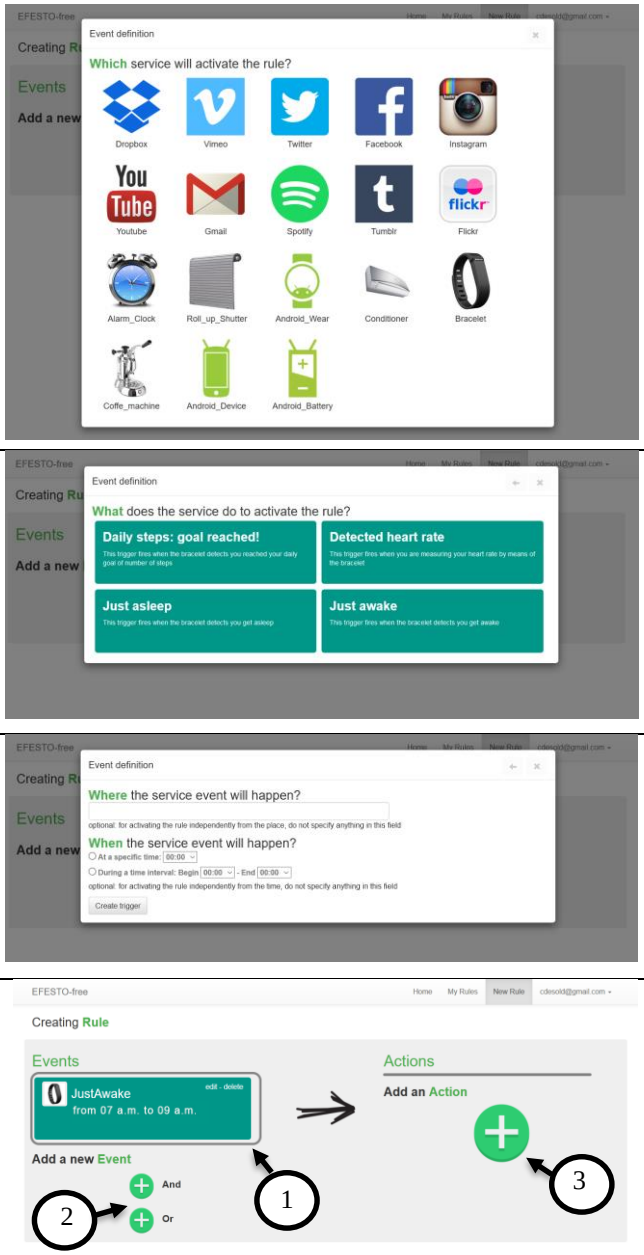

Figure 2. EFESTO wizard procedure for specifying events: a) the wizard first asks to select the service that will activate the event; b) as second step, the event is selected among those offered by the chosen service; c) temporal and spatial constraints are defined; d) the event has been defined and the user can define further events or actions.

3 Architecture and Feature Checklist

The proposed tool was designed starting from some modules already developed in the EFESTO mashup framework [1]. The core part of the new extended architecture, depicted in Figure 3, consists of the *Service Handler* and *Rule Engine* modules. The first one provides a JSON abstract representation of Web Services and Smart Objects to the *Rule Generator*, which is in charge to create a visual representation of services and objects to the users. The Rule Engine, starting from the rule defined by the users, **orchestrates the integration logic** of the involved services by instantiating a rule object representing a **logic mashup.** Such object is monitored by the Rule Engine every N minutes (3 minutes in our current implementation) to check triggers activation and, in case, to activate consequent actions (the **logic components**). The rule object is active on the platform even if the user is not logged-in, and until it is removed by the users (**Long-living Instantiation Lifecycle**).

EFESTO **run-time location** is on the **server**, which includes the Logic Layer and the Service Layer (see Figure 3). The client is used as a front-end to allow the user to access the representation of services and synchronize them according to the 5Ws composition paradigm.

In its current version, EFESTO is used only for private and academic purposes. However, we aim at building a **public community** to promote the sharing of rules for smart space configurations.

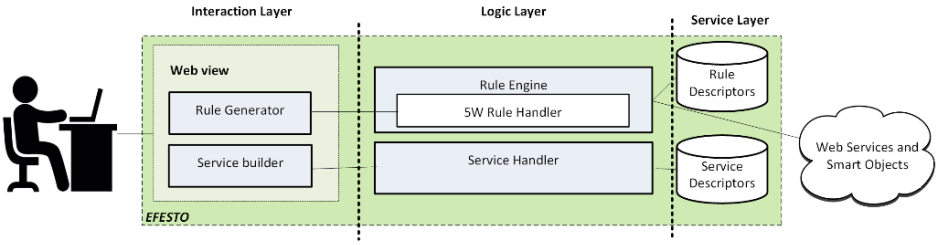

Figure 3. The EFESTO extensions supporting the definition and execution of ECA rules for smart object synchronization.

References

1. Desolda, G., Ardito, C., Matera, M. (2015). EFESTO: A platform for the End-User Development of Interactive Workspaces for Data Exploration. In Daniel, F., Pautasso, C. Rapid Mashup Development Tools - Rapid Mashup Challenge in ICWE 2015. (Vol. 591, pp. 63 - 81)
2. Tetteroo, D., Markopoulos, P., Valtolina, S., Paternò, F., Pipek, V., Burnett, M. (2015). End-User Development in the Internet of Things Era. In: Proc. of CHI '15. Seoul, Republic of Korea. pp. 2405-2408