Empowering End Users to Customize their Smart Environments: Model, Composition Paradigms, and Domain-Specific Tools

Empowering end users to customize their smart environments: model, composition paradigms and domain-specific tools

GIUSEPPE DESOLDA, University of Bari
CARMELO ARDITO, University of Bari
MARISTELLA MATERA, Politecnico di Milano

Research on the Internet of Things (IoT) has devoted many efforts to technological aspects. Little social and practical benefits have emerged so far. IoT devices, so-called *smart objects*, are becoming even more pervasive and social, leading to the need to provide non-technical users with innovative interaction strategies for controlling their behavior. In other words, the opportunities offered by IoT can be amplified if new approaches are conceived to enable non-technical users to be directly involved in "composing" their smart objects by synchronizing their behavior. To fulfil this goal, this article introduces a model that 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. Results of an elicitation study that was conducted to identify possible visual paradigms for expressing composition rules are presented. Prototypes implementing the resulting visual paradigms were compared during a controlled experiment and the one that resulted most relevant for our goals was used in a study that involved home automation experts. Finally, the article discusses some design implications that came out from the performed studies and presents the architecture of a platform supporting rule definition and execution.

• **Information systems~Mashups** • **Software and its engineering~Integrated and visual development environments** • *Software and its engineering~Visual languages* • *Human-centered computing~User studies.*

Additional Key Words and Phrases: End-User Development (EUD), Internet of Things (IoT), End-User Empowerment, Task-Automation Platforms.

1. INTRODUCTION

Technological advances we are confronting today are influencing society. End users can now exploit powerful, pervasive devices that offer several features, especially connectivity and sensors, and host the execution of multiple applications that until a few years ago were totally out of their reach. In addition, the Internet of Things (IoT) facilitates the creation of ecosystems of heterogeneous and distributed services that enable the access through the Internet to functionality and data provided by physical devices equipped with electronics, sensors and actuators, and embedded software, the so-called *smart objects* [Atzori et al. 2010].

Smart objects can foster important changes in our lives as they are increasingly pervading the environments we live in. If enabled to exploit the abundance of resources (the object functionality, the produced data, the related applications), end users could compose the "behavior" of the surrounding environment to accommodate their everyday needs. However, while research on IoT has devoted many efforts to the technological aspects characterizing smart objects, little social and practical benefits have emerged so far. Programming the behavior of smart objects is currently a prerogative reserved for professional developers, as it requires the use of scripting languages that can also vary depending on the underlying hardware. Another aspect is that often the available objects expose a very specific functionality that does not result in useful services able to accommodate users' needs.

The opportunities offered by IoT can be amplified if new approaches, based on high-level abstractions and adequate interaction paradigms, are conceived to involve directly non-technical users in configuring the behavior of their smart objects. End users possess the domain knowledge required to build applications that can support their tasks. Also, the new technology scenario increases the end users' attitude towards the new devices and applications, and fuels their desire to participate in the creation of applications to satisfy the "long tail" of specific - and sometimes unexpected – needs [Daniel et al. 2011; Fischer 2009]. End users are therefore the most suitable stakeholders to specify how the available resources should be exploited

to create new valuable services. To accommodate this vision, *End-User Development (EUD) paradigms supporting smart object composition* are needed. Indeed, as largely recognized in the literature [Ardito et al. 2012a; Costabile et al. 2007; Fischer et al. 2004; Lieberman et al. 2006a], EUD methodologies fit very well the requirement of letting users customize their systems to support personal, situational needs.

By *smart object composition* we mean *synchronizing the behavior of multiple objects* to create *new, added-value services*. Flexibility must be privileged to accommodate the variability of users' needs. In other words, we believe it is not relevant to provide end users with very specific applications governing the behavior of single objects. Rather, end users should be empowered to take advantage of *ecosystems of interoperable smart objects and services* [Barricelli and Valtolina 2015], by letting them combine flexibly, i.e., according to their situational needs, the behavior of different smart objects.

Recently, some approaches have been proposed to support non-technical users to configure smart object behavior. Many such tools, however, provide pre-packaged solutions, e.g., vendor- and device-specific apps for remotely controlling single smart objects that cannot be easily adapted to the requirements deriving from specific domains and contexts of use. So-called *Task-Automation* (TA) tools [Coronado and Iglesias 2016], to combine social services, data sources and sensors, are also gaining momentum. They have become popular as they offer very easy and intuitive paradigms to synchronize the behavior of objects and applications [Lucci and Paternò 2015]. Through Web editors, users can specify object behavior by either: a) 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, or b) defining *event-condition-action* (ECA) rules [Pane et al. 2001], a paradigm largely used for the specification of active systems (see for example [Ceri et al. 2007; Daniel et al. 2008]), that in the IoT domain can be fruitfully exploited to express how and when some object behaviors have to be activated in reaction to detected events. However, very often the adopted graphical notations for rule specification do not match the mental model of most users [Wajid et al. 2011]. Research on Web mashup composition paradigms [Daniel and Matera 2014] – a field that has many aspects in common with smart object composition – showed that graph-based notations are suitable for programmers; instead, fundamental issues concerning the conceptual understanding of such notations arise with laypeople, as they do not think about "connecting" services [Namoun et al. 2010b; Zang and Rosson 2008].

Another aspect is that the expressive power of the ECA rules that can be specified through current tools is limited to very simple synchronized behaviors. For example, in [Barricelli and Valtolina 2015] authors discuss the importance of temporal and spatial conditions to create ECA rules to better satisfy users' needs. Specifying temporal conditions also emerged as an important requirement in home automation to schedule rule for appliance activation [Rode et al. 2004]. Some of the available TA tools allow the definition of such conditions only by means of workarounds, for example by considering additional events to monitor the system time, or by creating filters on smart device data (e.g., in Zapier). Obviously, such workarounds complicate the rule creation, thus resulting into a scarce adoption of the available tools, especially by non-technical users, or in their adoption only for very simple tasks.

We believe there are numerous opportunities for research and development in the area of *EUD for smart object composition*. Such opportunities can spread the adoption of the new programmable, pervasive objects, as they can help people make sense of the smart environments they live in. Many challenges are related to technical aspects, concerning for example the interoperability of different smart objects and devices, the heterogeneity of data formats, the necessity to connect in

real-time to network interfaces and the scalability of the proposed systems. Our research recognizes all these technical challenges, but purposely focuses on a different facet, which is more related to the possibility for the end users, especially the ones not skilled in programming, to make sense of the advances of IoT technology. The ultimate goal of our research is indeed to provide end users with interactive tools that could allow them to customize smart spaces without being forced to get acquainted with technology issues. This does not mean that our research neglects the technology features to be faced to achieve a full-fledged, running IoT system. We however believe that a user-centered design of IoT platforms, as the one that we conducted and that is described in this article, is fundamental to identify abstractions and metaphors that can help even non-technical users to take advantage of the available IoT capabilities. This paper concentrates on this specific aspect.

1.1 Contributions

In our previous work, we defined paradigms and related platforms for the EUD of Web mashups. In particular, we investigated how data provided by Web APIs could be integrated into unified visualizations [Desolda 2015; Desolda et al. 2016] and how such visualizations could be synchronized at the presentation level by means of an event-driven technique [Cappiello et al. 2015; Cappiello et al. 2011]. Through visual editors, end users were enabled to build interactive Web applications synchronizing the behavior of different visual components. Inspired by our previous work, in this article we concentrate on a special class of mashup tools, the TA tools [Coronado and Iglesias 2016], which revisit the mashup paradigm to synchronize even smart objects, and in general any resources that can generate events or can be activated by the occurrence of events. Such tools exploit an event-driven, publish-subscribe composition of the involved resources, a technique that already proved successful for the composition of Web APIs within Web mashups [Cappiello et al. 2015; Daniel et al. 2007]. As a new contribution, we thus show how, thanks to adequate notations and usable user interfaces designed by involving a sample of users, the event-driven synchronization of services and smart objects can be mastered by non-technical users. The contribution of this article is therefore articulated along the following issues:

- *Specification of Task-Automation (TA) rules*. We analyzed the most common TA tools that allow users to define the event-driven synchronization of objects and services. We identified the pros and cons of their composition paradigms; thus, we conceived some new elements for rule specification that can extend the expressive power of currently supported ECA rules, still being adequate with respect to the background of non-programmers. In this article, we present a model that specifies how these new elements can be exploited in the creation of ECA rules.
- *EUD paradigms for task automation*. With the help of end users involved in an elicitation study, we identified how ECA rules, extended with new operators, can be specified through visual notations that were considered adequate by the end users themselves. Three different visual paradigms emerged, that we then compared in a controlled experiment conducted to investigate their usability. We further validated the best-evaluated paradigm with the help of a group of experts in home automation. We report on the results of the studies and discuss some design implications.
- *Reference architecture*. In order to foster the replicability of our approach, we present the architecture of the platform that we exploited to create the different prototypes implementing the visual paradigms defined through the elicitation study. Illustrating the platform architecture also allows us to show how we solved the problem of supporting *domain-specificity*, i.e., the capability of the platform to adapt to the requirements characterizing specific

domains [Casati 2011]. In fact, given the intrinsic flexibility of the resulting tool, which is favored by the decoupling of the User Interface (UI) from the other layers, different user skills and competences can be easily accommodated by "plugging-in" different UIs with different composition metaphors. In particular, we exploited the flexibility of the platform to generate the three prototypes, each one offering a different composition metaphor, which we used in the comparative and validation studies.

- *Lightweight integration of resources*. The main challenge of the proposed composition paradigm is not the definition of a new service synchronization technique; rather, based on existing integration and synchronization technologies, our approach especially aims to promote abstractions that: (1) capture and simplify the most salient technology aspects of smart objects, making them suitable for end users; and (2) can be handled by lightweight Web architectures, making them easily accessible in any environment where they are needed and through different devices. This lightweight paradigm could have a limited coverage with respect to the immense capability offered by IoT technology. We however deem this is not a weakness with respect to the goals of our research, as we purposely tried to filter out, through a user-centered design, those aspects that can really help end users make sense of IoT technology.

1.2 Article organization

The article is organized as follows. Section 2 introduces some background concepts by referring to related work and clarifies the rationale of our research. Section 3 illustrates the main elements of the model we propose for the specification of ECA rules. Section 4 and Section 5, respectively, describe the elicitation study, conducted to identify visual paradigms for ECA rule specification, and the three prototypes that we implemented on the basis of the identified visual paradigms. Section 6 reports the comparative study conducted to assess both user performance and satisfaction of the three prototypes; the results are discussed against threats to the validity of the comparative study, also highlighting under which conditions study results can be exploited in other contexts. Section 7 describes a further validation study in which experts in IoT and home automation were asked to use the prototype that in the comparative study resulted as the most promising. Section 8 discusses the design implications resulting from the overall study. Section 9 illustrates a reference architecture that we adopted for the development of a platform prototype. Finally, Section 10 draws our conclusions and outlines our future work.

2. RATIONALE AND BACKGROUND

The Internet of Things (IoT) idea emerged in 2009 when Kevin Ashton and his team at MIT's Auto-ID Center coined this term to describe a system where the Internet is connected to the physical world via ubiquitous sensors[1]. In the following years, thanks to a wide spreading of low-cost integrated technologies with sensors and actuators, it became possible to easily build the so-called *smart objects*, i.e., the building blocks of the IoT.

A *smart object* is a device equipped with embedded software that is typically connected to the Internet [Atzori et al. 2010]. It exploits sensors to "feel" the environment and/or actuators to communicate with the environment. Examples of sensors are those measuring light intensity, the physical pressure of an object, and also air humidity. Actuators can be light and sound emitters, electric valves, motor servos, relays, etc. According to the classification proposed in [Barricelli and

[1] http://www.rfidjournal.com/articles/view?4986

Valtolina 2015], smart objects can be classified as *settled*, if they are installed in a fixed position (e.g., an IP Camera in a room) or *mobile* if their position can change as in the case of wearable devices (e.g., smart bracelets that measure steps and heart rate); as *asynchronous* or *synchronous* depending on the modality for sending and receiving data; for either *individual* or *collective* use, if they communicate data only to the owner, for example, the alcohol gas sensors that alert home owners in case of danger, or if they provide data to groups of users, for example, the air pollution stations that provide air quality data to the citizens of a given geographical area.

Many domains are taking advantage of IoT technology, for example, healthcare, smart homes, industry, smart cities, agriculture, vehicles, smart buildings, retail, factories, oil&gas. Each one exploits different types of smart objects [Atzori et al. 2010]. About 18.2 billion smart objects were produced by the end of 2015. Cisco's forecasting estimates that more than 50 billion smart objects will be deployed by 2020[2].

Given the rapid spread of such technology, there is a great interest in effective solutions in this field; a number of research lines are emerging to address this problem space, which differ for the strategies used for considering end users and technologies according to the sphere of activity and users' skills. Among the most prominent, *workflow design tools* and *business rules engines* address the problem of ECA rule design and management from a technical perspective. Such platforms have traditionally supported the definition of ECA rules for the design of active systems [Casati et al. 2001; Jennifer and Stefano 1996]. They are now receiving interest also in relation to the design of IoT systems. Platforms like JBoss Drools [Red Hat 2016], OpenRules [OpenRules 2016], and IBM WebSphereJRules [IBM 2016] support the management of rules. In these platforms, programming the behavior of IoT systems consists in specifying the flow of data produced by different devices, and how events captured from this flow have to trigger actions that progressively change the state of the whole system. In some cases[3], such platforms offer mechanisms to ease rule programming, such as diagrammatic notations, decision tables, or helper codes to avoid syntactic errors. Still, these paradigms are effective for expert information system designers and programmers, while they require technical competences that are not generally mastered by non-programmers. For example, graph-based notations ask the designers to deal with concepts like data flow, parameter passing, or even to identify the sequence of actions. Certainly such engines can cover complex, critical requirements. However, we believe that an innovative aspect in IoT would be to empower non-expert users to make sense of IoT. With this respect, an intuitive rule specification paradigm needs to be identified. Also, the multiple, complex features supported by business rule engines need to be filtered out, as very often they go well beyond the actual needs of "casual" programmers and non-technical users. We do not want to argue that complexity, and all the related technical requirements, have to be discarded. We instead aim to promote the right level of simplicity and expressiveness, which can allow the non-technical end users to operate on the IoT.

More on the side of this class of end users, the composition paradigms adopted in many *mashup tools* break new ground by offering easy-to-use control metaphors and lightweight platforms, mainly deployed on the Web, for the composition of heterogeneous resources (e.g., Web APIs, data sources, interactive widgets) and the execution of the resulting applications [Daniel and Matera 2014]. Recently, also the mashup paradigm has been revisited to move the focus from pure service composition to task automation (TA): the integration addresses not only Web APIs but also services controlling smart objects, which can be synchronized to define active

[2] http://newsroom.cisco.com/feature-content?type=webcontent&articleId=1208342

[3] See for example tools like Talend Open Studio (https://www.talend.com/products/talend-open-studio), and Waylay.io (http:// http://www.waylay.io/)

behaviors in smart spaces. Some of such tools adopt wired notations, thus keeping a composition paradigm similar to the tools dedicated to workflow designers. Other tools then follow the trend to simplify as much as possible the definition of automation tasks, in some cases proposing collections of ready-to-use "recipes". These tools are gaining momentum, as their simplicity and immediateness look appealing for the masses [Coronado and Iglesias 2016].

Our research addresses this specific segment of systems, which includes popular tools, such as IFTTT, Zapier and Node-RED, that currently best encounters the skills of inexperienced users [Lucci and Paternò 2015]. In the sequel of this section, after a brief explanation of the motivations to adopt a mashup approach for the EUD of IoT systems, we will review and classify works that fall within the category of *mashup tools for task automation* – TA tools for short.

2.1 EUD for IoT by means of the mashup paradigm

The rapid spread of smart objects and their strong heterogeneity are presenting important challenges, which are not exclusively related to the technology involved, but rather to methodologies and systems that let the end users make sense of this promising technology [Tetteroo et al. 2013]. IoT is intrinsically about people [Burnett and Kulesza 2015], as its ultimate goal is to improve people experiences in their living and working environments. IoT can achieve deeper, more meaningful and faster insights if the user is placed at the center of systems that include ambient and personal sensors, pervasive devices and communicative tools [Tetteroo et al. 2015]. One relevant challenge, scarcely approached so far in the literature, is therefore related to the definition of adequate paradigms to compose different resources, i.e., based on adequate interfaces that allow end users to customize technology to their needs.

EUD fits very well the problem of customizing systems to support the user's personal, context-specific and situational needs [Lieberman et al. 2006b]. EUD goes beyond conventional methodologies for the design of interactive systems since its goal is to provide end users with tools to compose the applications they use or to create brand new ones. Some works in the literature propose composition techniques based on EUD practices, that allow non-technical users to compose smart objects [Bellucci et al. 2014a; Bellucci et al. 2014b]. Thanks to IoT technology, smart objects can indeed be accessed as services, as they are often provided with a URI that identifies them on the Internet and are published by exploiting traditional service technologies (e.g., RESTful) that enable capturing events and running actions remotely. The problems of composing smart objects can be therefore considered as a special case of *Web service mashup*.

In the case of service mashups, the platforms implementing an event-driven, publish-subscribe approach, such as the one described in [Cappiello et al. 2011], synchronize Web APIs, so that the events produced by/on a service (e.g., selection of a displayed data item) trigger operations of other services (e.g., a search based on the selected word). This paradigm suits very well the need to synchronize the behavior of smart objects [Ghiani et al. 2015]. However, very often tools exploiting it are oriented to developers, while there is a lack of approaches exploiting notations and metaphors adequate for non-technical users.

The approach proposed in this article takes advantage of our previous experience in the design of mashup platforms [Ardito et al. 2014a; Ardito et al. 2014c; Desolda et al. 2016]. The composition paradigm previously defined, which supported the synchronization of service functionality and the integration of data, has been purposely extended to cover also the synchronization of smart object behaviors. As smart objects are accessible and programmable through Web services, the new extensions mainly privilege event-driven synchronization by means of ECA rules, a

composition technique that we previously experimented for the composition of Web APIs through client-side, light-weight mashup approaches [Ardito et al. 2012b; Cappiello et al. 2015; Cappiello et al. 2011]. The new extensions are also in line with recently proposed TA tools (see next section). However, as illustrated next in this article, these extensions propose a richer set of operators for the definition of ECA rules, and especially introduce a visual paradigm that tries to accommodate the user mental model, as it was elicited with the help of a sample of end users.

2.2 Task-Automation Tools

Event-driven architectures have been studied since many years to address the design of active systems in different fields, from active databases, to workflow design and context-aware applications [Eisenhauer et al. 2009]. They can be applied anytime the involved components of a system have to be orchestrated depending on the production, detection and consumption of some events.

In this architectural pattern the design and management of Event-Condition-Action (ECA) rules has a fundamental role. These rules allow the specification of active behaviour by means of events indicating a signal triggering the invocation of the rule, a condition that can determine the activation of an action, and the action that consists in an operation acting on data or functions also exposed.

Recently, different Web tools have revisited the ECA rule paradigm to address the problem of Task Automation (TA). In particular, they support the definition of ECA rules to synchronize the behavior of smart objects and services [Coronado and Iglesias 2016; Fogli et al. 2016a]. Table 1 summarizes the main characteristics of the most popular tools for ECA rule specification. Many of these tools are designed for non-technical people and offer wizard procedures that guide users during the composition process. One of the most popular is *IFTTT* (If This Then That), a free Web platform that, by means of a wizard-based composition paradigm, allows end users to create simple chains of conditional statements called "recipes" [IFTTT 2016]. Each recipe consists of 1) a service that IFTTT tracks to detect if a specific event is triggered (e.g., the position of an *Android Device* is within a specific area) and 2) another service that reacts to the triggered event by executing a specific action (e.g., switch on the smart *Coffee Maker*).

Other tools also exploit wizard procedures. *elastic.io* provides a catalogue of Web services primarily oriented to business aspects (e.g., Magento, SAP) and allows the registration of custom services, for example to access and control smart objects; wizard procedures assist the users in the definition of data-flow chains among both predefined and custom services [GMBH 2016]. *Zapier* enables the composition of both Web services and smart objects. It proposes a wizard to specify one event and one action in a "basic rule", which can be later extended with further events and actions [Inc. 2016b]. The addition of *filters* on the triggering event to further control rule activation is also possible. For example, if in a rule the event is the creation of a new tweet, a filter on the tweet text can limit the activation of the rule to the occurrence of the only tweets that contain specific words. *itDuzzit* is a Web tool with a composition paradigm very similar to Zapier, but its rules can contain only one trigger and one action [LLC 2016]. The wizard approach is also exploited by *WigWag*, a commercial Web tool specifically designed for automation of smart environments [Inc. 2016a]. WigWag is also available as mobile app.

Other types of composition paradigms have been proposed. For example, in *We Wired Web* rule creation occurs in a Web page, vertically divided into two panels dedicated to the specification of the elements composing a rule: triggering events can be defined in the left-hand panel and actions in the right-hand panel [Apiant 2016]. As in Zapier, filters can be used in the specification of triggering events.

Table 1. Task-automation tools and their characteristics.

Name	Service Type	Composition Paradigm	License	Execution Device	Target Users	Rule Type
Atooma	Device functions Web services Smart Objects	Wizard	Free Paying	Mobile	Non-tech.	IF TriggerS DO ActionS
AutomateIt	Devices functions	Item selection	Free Paying	Mobile	Non-tech.	IF TriggerS THEN ActionS
Bip.io	Web services Custom	Wired	Free Paying	PC	Non-tech. technical	Complex process
Context-Dependent Authoring	Web services Smart Objects	Item selection	Research project	PC	Non-tech. technical	IF TriggerS THEN ActionS
Elastic.io	Web services Custom	Wizard	Paying	PC	Non-tech. technical	Data flow chain service -> service
IFTTT	Web services Smart Objects	Wizard	Free	PC Mobile	Non-tech.	IF Trigger THEN Action
itDuzzit	Web services	Wizard	Free Paying	PC	Non-tech.	IF Trigger THEN Action
Node-RED	Web services Smart Objects Custom	Wired Programming	Free	PC	Technical	Complex process
Spacebrew	Web services Smart Objects Custom	Wired Programming	Free	PC	Non-tech.	Publishers -> Subscribers
Tasker	Devices functions	Wizard	Free Paying	Mobile	Technical	IF TriggerS THEN ActionS
We Wired Web	Web services Smart Objects	Item selection	Free Paying	PC	Non-tech.	When TriggerS THEN ActionS
WigWag	Smart Objects	Wizard	Paying	PC Mobile	Non-tech	When TriggerS THEN ActionS
Zapier	Web service Custom	Wizard	Free Paying	PC	Non-tech.	IF Trigger THEN ActionS
Zipato	Smart Objects	Building blocks	Paying	PC Mobile	Technical	Complex process

A different approach is provided by *Bip.io*, which exploits the graph metaphor for wiring Web services represented as nodes [wot.io 2016]. When nodes representing Web services are connected by means of arrows, assisting procedures guide users in defining trigger and action properties. A graph-based representation is also adopted in *Node-RED*, a Web tool for composing both smart objects and Web services [JS_Foundation 2016]. Unlike Bip.io, Node-RED is meant for professional users since it also supports advanced rule customization by means of nodes representing control statements, functions written in JavaScript and debug procedures.

Spacebrew offers a mix of the paradigms implemented in We Wired Web and Node-RED [Group 2016]. It supports rule creation in a workspace vertically divided

in two parts, the left-hand panel for the configuration of the services publishing the events and the right-hand panel for the configuration of services providing actions in response to events. Such services can be connected in a wired fashion. A paradigm more suitable for non-technical users is the one implemented in Zipato, a web tool that provides typical programming language constructs (e.g., *when-then*, *while*, *if*, logical operators) in form of graphical widgets that can be combined to create task-automation rules [Zipato 2016].

Some TA services can also be run on mobile devices, as in the case of *Atooma*, an Android app for the composition of device functions, Web services and smart objects connected to a mobile device [Atooma 2016]. The rule composition follows an *If-Do* paradigm. Multiple triggers and actions can be included in each rule. Moreover, filters can be defined as in Zapier. Other similar Android apps are *AutomateIt* and *Tasker*, which support the creation of rules limited to the composition of apps and functions available on mobile devices [EU 2016; Ltd 2016].

In line with our goals, the ECCE toolkit [Bellucci et al. 2014b] aims to support the definition of "ecologies" of smart objects. This work especially focuses on the way smart objects can connect to a server, so that an ecology of such objects can be set-up and managed. An XML-based language is used for describing the properties of single smart objects. Based on this language, the corresponding code to handle the smart objects behavior is generated on the platform server. Then the behavior of single "connected" devices can be synchronized by means of a Web interface addressing the end user expertise. The synchronization consists in coupling some events generated by one object to the operations exposed by other objects. The specification of further conditions to constrain the rule activation is not supported.

Manipulation of physical objects was also investigated as a programming paradigm to define smart object behaviors. For example, *AutoHAN* implements a new paradigm for interaction with abstract functions of home appliances through special cubes that act as "one-button remote controls". Each cube is devoted to a function, for example Play/Pause, and the users can associate such functions by holding one face of a cube against the front of an appliance (e.g., the Play/Pause cube can be associated with a VCR or a DVD player). The expressive power of the language is realized through the composition of these functions. It is achieved by placing two or more cubes next to each other, and instructing AutoHAN to store the configuration of Cubes. Such configuration can then be used to schedule home automation processes. Another system, *SiteView*, exploits the manipulation of physical objects for creating ECA rules [Beckmann and Dey 2003]. Conditions are expressed by locating physical objects corresponding to rule conditions in a "condition composer area"; actions are programmed by placing physical objects, representing for example appliances, in a "world-in-miniature area", which is a small-scale picture of the active environment. In addition, the users can see a rule representation in a "rule display" and can simulate the rule results in an "environment display".

A diverse use of user-defined event-action rules is illustrated in [Ghiani et al. 2015]. An authoring tool supports the development of context-aware cross-device user interfaces through the creation of rules in which different types of events can activate actions indicating how the user interface should adapt to the detected context.

The tools described above cover several relevant aspects of task automation. However, in relation to the composition paradigm, which represents the main aspect addressed by this work, their potential benefits are still limited [Barricelli and Valtolina 2015]. On the one hand, tools like IFTTT, elastic.io, Zapier and itDuzzit address the skills of non-technical users, but they only assist users in the creation of "basic" rules, i.e., rules that synchronize one event with an action and do not include any additional conditions for rule activation. This is also true for some research works illustrated in literature. For example, the approaches reported in [Kubitza and

Schmidt 2015; Zancanaro et al. 2015] allow the reuse of pre-defined "recipes" or "schemas of digital experiences". New rules can be defined by only modifying and adapting pre-defined rules. Thus, end users are not allowed to define their own rules, which can be needed if the predefined ones do not accommodate their situational, even unexpected needs. On the other hand, tools like Node-RED allow one to create more complex rules, but they also require advanced skills.

Finding a trade-off to mediate these two extremes is the challenge that we address in this article by focusing on the following aspects:

- Rule specification must include *logical operators* to combine triggering events and actions exposed by multiple services and objects. This would accommodate the need for programming the composition of multiple resources. Some platforms (for example Atooma) already support the OR combination of multiple events. In line with these approaches, we also think that, in order to express composite behaviors, end users must be enabled to define combinations of multiple events and actions. Logical operators can allow them to express such combinations.
- The specification of *temporal and spatial constraints* must be possible. As discussed in [Barricelli and Valtolina 2015], existing tools (e.g., itDuzzit) do support the specification of such constraints, but only as filters on the data generated by the services that trigger events. This could not be simple for non-technical users, who are forced to understand what output is generated by the services. In addition, if the services involved in the rule do not return timestamp data, temporal constraints cannot be expressed by the existing tools. These are the reasons why we aim to propose *mechanisms for temporal and spatial constraints* to be defined on parameters characterizing the context in which the rule is executed (spatial constraint) and regardless of the service timestamp (temporal constraint).
- If on the one hand the use of logical operators and the specification of temporal and spatial conditions would introduce some cognitive overhead, on the other hand *usable composition paradigms*, increasing the users' awareness of what the surrounding environment offers and how composite behaviors of the available objects and services can be achieved, could alleviate the resulting complexity. The following sections will illustrate how, by means of user studies, we identified some design implications leading to the definition of usable composition paradigms.

3. MODEL FOR RULE CREATION

TA platforms allow one to synchronize the behavior of Web services and smart objects by specifying chains of conditional statements for triggering actions that change the status of coupled resources. Such chains are usually called "rules" as they are actually based on ECA rules, but different names are also used, e.g., "recipes" in IFTTT, "Zaps" in Zapier, "Duzzit" in itDuzzit. Although specified according to a different syntax, the rules adhere to the common schema:

$$cause(s) \Rightarrow effect(s)$$

where the causes are the events triggered by some services and the effects are the actions performed by the same or by other services (see Figure 1).

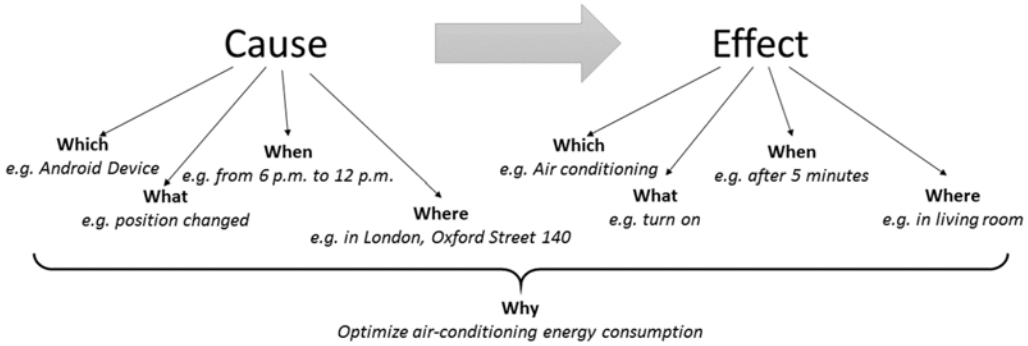


Figure 1. Example of causes and effects in a rule.

In relation to the specification of rules, our research aims to identify the right trade-off between simplicity and expressiveness, which can allow the end users to operate on the IoT. By analyzing the available TA platforms, we observed that they effectively cover several aspects of IoT. However, their composition paradigms are very complex and with an abundant expressiveness that goes well beyond the needs of end users, or very simple but not expressive enough. Most of such platforms do not permit (for example in IFTTT) or make it difficult (for example in Node-RED) to specify multiple events and actions. Similar difficulties arise when specifying temporal or spatial constraints, e.g., to define the following behavior: "If *I'm in Rome* and I post an image on Instagram *between 8.00 and 11.00 a.m.*, post the same picture also on Twitter and Tumblr". The need to define temporal and spatial conditions in ECA rules was recently discussed in [Barricelli and Valtolina 2015], and also emerged as requirements in home-automation systems to schedule rule for appliance activation [Rode et al. 2004].

In order to identify composition paradigms able to guide users in the definition of articulated rules, we wanted to elicit the end-user mental model, which is an aspect scarcely explored in the field of task automation [Ur et al. 2014]. As the seed of our investigation, we were inspired by the *5W* model, which is adopted in several domains, such as journalism and customer analysis, and more in general in problem solving, to analyze the complete story about a fact. It suggests describing a fact by answering the following questions:

- *Who* did it?
- *What* happened?
- *When* did it take place?
- *Where* did it take place?
- *Why* did it happen?

These questions can help formulate rules with rich conditions; especially they can guide the users to express temporal and spatial elements. We therefore adopted the 5W model in an elicitation study aimed at identifying, with the help of users, a notation for the specification of task-automation rules that would feature the trade-off between simplicity and expressiveness that we mentioned above. In particular, the model, which we called *Rule_5W*, would guide users in identifying the elements that are essential for creating complete sensible rules for smart object composition. In the Rule_5W model (see Figure 1), "Who" is replaced by *"Which"* for specifying the services involved in a rule. *"What"* indicates the triggered events, as well as the actions to be activated. *"When"* and *"Where"* refer to the specification of, respectively, temporal and spatial conditions for triggering events and performing actions. Finally, *"Why"* is used for reporting a short description to explain the rule behavior to a human reader, e.g., other users with whom the rule is possibly shared.

4. ELICITATION STUDY

Driven by the main goal of our research, i.e., providing end users, even without technical skills in computer programming, with interactive tools to customize smart spaces, we carried out an elicitation study to identify adequate visual composition mechanisms. In this study, starting from the identified Rule_5W model, we asked participants to propose, in terms of Which, What, When, Where and Why, how they would specify in a rule event-action relationships between different services. This study, and the consequent identification of composition mechanisms, can be considered the first step to design an EUD platform. Indeed, we wanted to assess in which measure and under which assumptions the IoT technology could be mastered by non-technical users. We wanted in particular to understand whether and how the definition of synchronization rules involving the invocation of web services, the use of logic operator and the specification of temporal and spatial constraints, could be mastered by users who do not have any knowledge about all these technical concepts. The leading question of the study, therefore, was:

> "*How to specify events and actions in a rule by answering to the Which, What, When, Where and Why questions?*"

Different techniques can be used to elicit system solutions but few were those investigated to understand programming in domestic environments. An example was the "Fuzzy Felt Ethnography" technique proposed in [Rode et al. 2004] aiming to elicit the programming patterns of domestic appliance. In particular, a felt board, divided into 4 sections with a set of felt icons representing appliances, is used as data gathering tool. Such board support designers to understand programming of domestic appliances, distinguishing between program actions at future times and macro creation to facilitate repeated tasks. Since the goal of this phase in our research was to elicit techniques to create ECA rules for smart environments, which is more general than the domestic environments, we exploited a different technique. In particular, the study followed a rigorous procedure, based on carefully selected materials, questionnaires and tasks to be performed. In order to collect as much as possible significant ideas properly discussed among participants, we required them to work in groups, according to the *partners* technique presented in [Morris et al. 2014]. This consists of performing several focus groups involving participants, i.e., partners, enabling them to fruitfully build upon one another's ideas, carefully analyzing the proposed ideas, to stimulate more reflection and discussion and elicit diverse opinions about possible designs. Therefore, our focus groups went beyond pure elicitation, already including some co-creation activities, similar to what was done in [Marquardt et al. 2012; Voida et al. 2005]. However, even if our participants (i.e., Computer Science students) had some experience in system design, we could not expect that they would came out with a complete and successful design proposal. Thus, we adopted a scenario-based design. Scenarios help designers to maintain an orientation towards perspective users and their actual needs [Rosson and Carroll 2003]. As described later in this paper, the results of the study showed that this organization enabled participants to provide novel and elaborated indications about user interface and interaction mechanisms. In order to avoid participants being forced to "tell what they don't have to tell", we did not require groups to produce a minimum number of proposals for accomplishing the assigned tasks or to reach a goal. Finally, participants were encouraged to demonstrate their ideas by sketching paper prototypes. At the end of the focus groups, participants filled in online questionnaires. The study protocol was preliminarily assessed by involving two groups of four participants each.

It is worth noting that we purposely did not contextualize the identification of the visual notation in a specific domain. First of all, we wanted to identify "generic"

interactive mechanisms that would be adequate for non-programmers and especially ensure a reasonable level of expressiveness. Our approach to the definition of EUD paradigms and tools indeed promotes the definition of generic platforms that however can be easily adapted to be domain-specific. As we will discuss in Section 9, it is possible to define a platform organization so that visual notations and interactive paradigms can be easily adapted to the requirements characterizing specific end-users' communities, identified through dedicated ethnographic studies to be conducted after the deployment of the generic platform.

4.1 Participants

The elicitation study involved a total of 25 participants (6 female), randomly divided into 6 groups (from 3 to 5 participants), aged between 20-43 years ($\bar{x}$ = 23.7, SD = 5.06). The participants were recruited from the third-year students of the Bachelor degree in Computer Science. They had no experience with tools for defining TA rules or for managing IoT elements. All of them had a PC and a mobile device; only six participants had one or more smart objects.

4.2 Procedure

The study was performed by two HCI researchers on two consecutive days in a university laboratory. It consisted of six sessions, one for each group. Three sessions took place the first day. In every session, the group sat around a table. One of the two HCI researchers gave a 10-minute presentation to introduce participants to the addressed domain, by illustrating daily-life and working situations in which a tool for defining the behavior of Web services and objects could be useful. To avoid any bias in the participants' proposals, possible solutions or tools were not shown.

The group was provided with blank paper sheets and markers for sketching their proposals. Each participant was also provided with a paper sheet reporting the two tasks to be performed. The two following tasks asked participants to propose a user interface and interaction mechanisms as a response to the leading question of the study:

1. How to define a task-automation rule by using the system that you want to propose? For example, how to program the system so that each post published on my Facebook wall is also posted on Twitter? Or so that every time the alarm clock is switched-off the roll-up shutters are opened?
2. How to include further conditions on rule activation? For example, how to program the system so that each post published on my Facebook *or* Instagram wall is also posted on Twitter? Or so that every time the alarm clock is switched off the shutters are opened *and* the coffee machine is switched on?

After reading aloud and commenting the first task, the researcher asked participants to reason about and discuss how they would perform the activities reported in the example. He also highlighted that the elements involved in the rules (services, things, conditions) can be described by specifying four aspects, i.e., Which, What, Where and When, while the purpose of the rule can be described by answering to the Why statement. The researcher stimulated participants to elaborate new interaction ideas, which were also expressed by sketching new paper prototypes. He also encouraged the discussion of positive and negative aspects of the suggested solutions. The same procedure was repeated for the second task, during which participants revised the sketches produced for the first task, in order to address the new requirements. The second researcher took notes. Each session was also audio-video taped.

At the end of the session, participants filled in a questionnaire composed of 18 questions. Thirteen questions aimed to collect participants' demographic data, and determine their expertise with programming, mobile devices, smart objects and Web services. Two questions investigated participants' understanding of and comfort with study procedure and proposed tasks. One question addressed the perceived usefulness of TA tools. The last two questions addressed the pros and cons of the ideas the participants suggested during the study.

4.3 Data Collection

The data analyzed in the study were collected by reviewing: 1) the set of notes taken by the researchers in the study sessions; 2) the video recorded during the sessions 3) the sketches drawn during the sessions; 4) the answers participants gave to the online questionnaire.

The two researchers transcribed their notes and the audios, and independently double-checked 65% of the material. The initial reliability value was 81%; thus, the researchers discussed the differences and reached a full agreement. The transcripts were analyzed through a thematic analysis following a semantic approach. Themes were identified within the explicit or surface meaning of the data [Braun and Clarke 2006a]. The two open questions of the online questionnaire, related to the advantages and disadvantages of participants' proposals, were analyzed through the affinity diagram technique proposed in [Rogers et al. 2015].

4.4 Results from the proposed user interface sketches

The six groups elaborated a total of nine user interface sketches, complemented with comments explaining how to interact with them. A design team, composed of the two HCI researchers who participated in the study and an interaction designer, analyzed the sketches. In order to understand important details on how the sketches were created, video recordings were analyzed. Some sketches had several features in common, thus the design team managed their integration in a unique design proposal also taking into account interaction principles and usability criteria. Eventually, the three different proposals described below emerged and gave rise to the three prototypes described in Section 5.

The first design proposal came from the contributions by groups G1, G3, G4 and G6. G1 envisioned the workspace for defining and representing the rule vertically divided in two sides: events on the left, actions on the right. Services representing events and actions are displayed in a menu available at the most left side of the screen and are dragged in the appropriate workspace side. Every time a service is dropped in the workspace, a wizard procedure is presented in a pop-up window to guide users in defining service details by answering the What, When and Where questions. At the end of the wizard, the new event or action is displayed as a box, labelled with an identifying name and icon, and placed at the bottom of other events/actions previously created, if any. A similar proposal also originated from G3; the main difference is that when a service is added in the workspace, it is immediately represented as a large box divided in three areas for specifying the type of event/action, and the temporal and the spatial constraints (see Figure 2). After the parameterization, the box can be minimized.

The second design proposal was conceived according to the input of G4. They conceived the rule creation into two main steps: the first one is the creation of a basic rule with only an event and an action; the second step consists in adding further events and actions. The rule is visualized in a workspace where the events are placed on the left side and the actions on the right side. Two buttons, "Add an event" and "Add an action", allow users to add further events and actions, as also suggested by G6 participants.

The third design proposal is completely different from the previous ones. It is based on a graph metaphor and it was proposed by G3 and G5. Nodes represent services and arrows between nodes indicate event/action relationships. In particular, given two nodes connected by an arrow, the staring node is the rule event while the final node is the action. G3 and G5 proposed two different solutions to specify event and action details; G5 suggested a button in each node (see Figure 3), while G3 a wizard procedure automatically activated when the arrow (i.e., the relationship) between nodes is drawn.

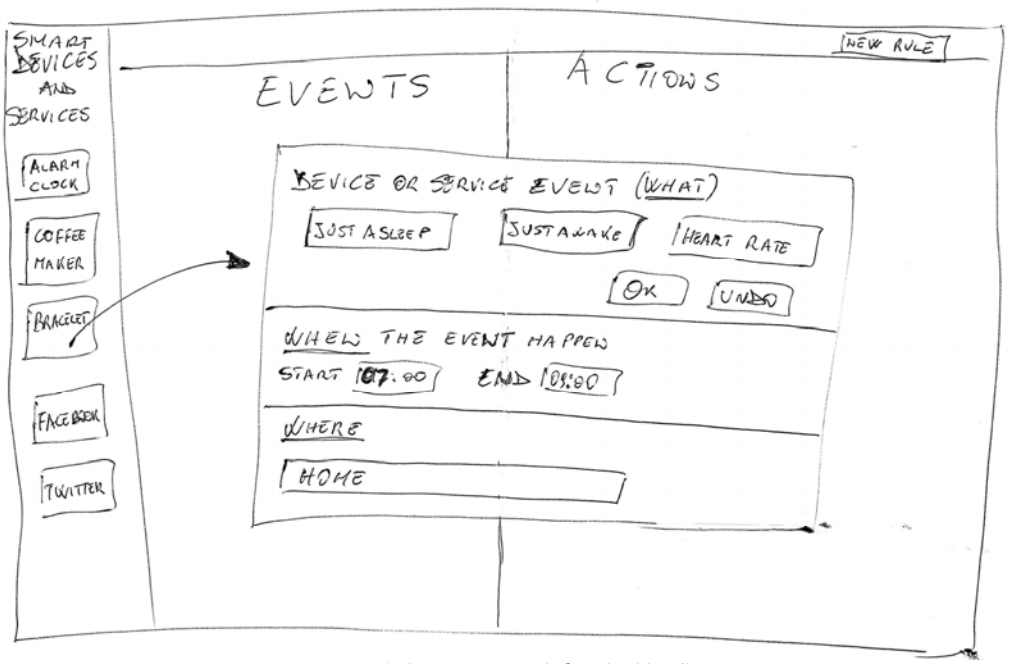


Figure 2. A design proposal sketched by G3.

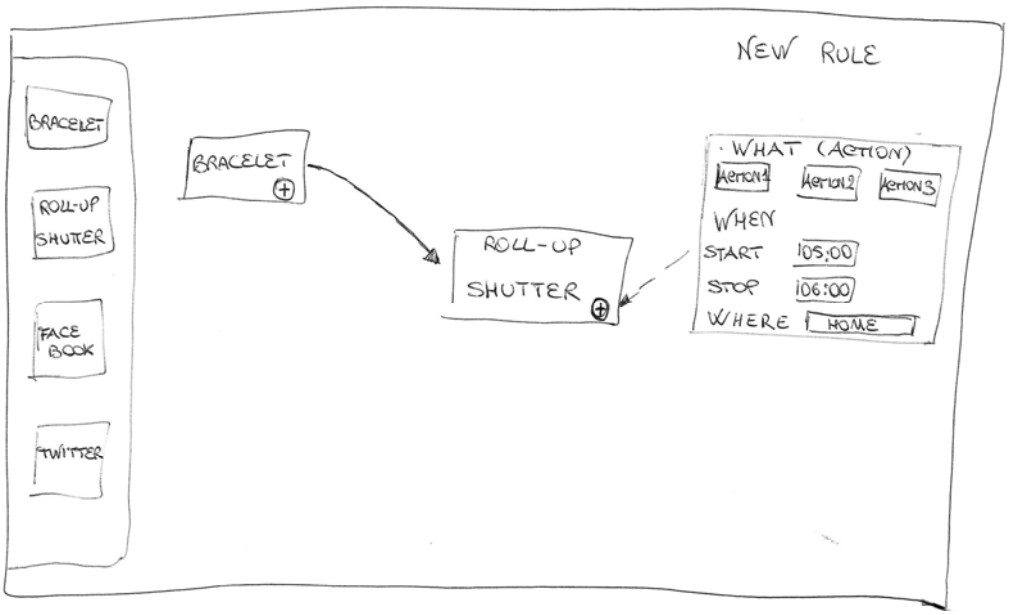


Figure 3. A design proposal sketched by G5.

4.5 Results from the online questionnaire data

The analysis of the online questionnaires referring to demographic data, experience in using IT technology and in programming provided the participants' characterization already reported in Section 4.1. The usefulness of composing services and smart objects was rated 6.09 out of 7. Regarding the advantages expressed in the open questions, participants reported that: *i)* task automations performed through their solutions save time and reduce errors; *ii)* their solutions can be exploited by non-technical users; *iii)* it is useful to add constraints based on time and spatial conditions; *iv)* it is interesting to create rules for performing powerful task automations. Concerning the disadvantages, participants commented on the limited possibilities of configuring useful smart environments because of the currently poor spreading of smart objects and on the lack of device interoperability due to protocol heterogeneity.

5. IMPLEMENTED PROTOTYPES

Based on the three main designs emerged during the elicitation study, we developed three interactive prototypes: EFESTO-Free, EFESTO-Wizard and EFESTO-Wired, abbreviated to *E-Free*, *E-Wizard* and *E-Wired*, respectively. EFESTO is the name of the mashup platform proposed by the authors in their previous works (e.g., [Ardito et al. 2014b; Ardito et al. 2014c; Desolda et al. 2016]. EFESTO was used as the basis for the implementation of the three prototypes as the EUD paradigms for smart object composition can exploit the EFESTO layers for service invocation.

The three prototypes share some design choices:

- In the user interface, the terms *events* and *actions* are used instead of *causes* and *effects*, respectively, because they are the most adopted terms in IoT and task-automation domains.
- In principle, rule events can be chained by means of *AND* and *OR* logical operators; however, in order to avoid complex logical expressions, all the events in a rule are in either AND or OR, depending on the operator used for connecting the first two events.
- Actions can be chained only by the *AND* logical operator, otherwise it would not be possible for the system to disambiguate which action has to be performed. In other words, if the event conditions are fulfilled, all the specified actions are executed.
- Both events and actions are specified by answering the Rule_5W questions, Which, What, When, Where and Why. In particular, the answer to the Why question is meant to provide a title briefly describing the rule (e.g. "Synchronize my social posts" in Figure 1), which can be useful when rules are shared with other users.
- Spatial constraints can be defined only if the smart device provides its GPS position; temporal constraints can be always defined because the prototypes compare the user defined temporal conditions with the user account timestamp instead of the device timestamp, which could not be always provided by the device.

The prototypes were developed as Web applications using the Java Spring framework[4]. Their UIs were programmed by using Thymeleaf[5], a Java HTML5 template engine, and the Bootstrap[6] front-end framework. The use of Bootstrap

[4] https://spring.io/
[5] http://www.thymeleaf.org/
[6] http://getbootstrap.com/

allowed us to build responsive UIs, which adapt their layout to the device on which they are run (e.g. PCs, smartphone, tablet). To create nodes and edges in the E-Wired prototype, a specific JavaScript plugin[7] was used. All the prototypes have been deployed on a virtual machine created in the Windows Azure cloud platform (4 core, 8Gb RAM, Windows Server 2012).

We devoted particular attention to the smart objects and services made available in our prototypes. We indeed wanted to propose realistic scenarios for rule creation, comparable to the ones supported by the existing and most popular tools. A set of services available in IFTTT was therefore selected by analyzing the 20 most popular IFTTT recipes. The services and smart objects used in these rules were then registered in our prototypes, together with their events and actions. The three prototypes are described in the rest of this section.

5.1 E-Free

The first prototype is called E-Free because of the few limitations the users encounter in the rule definition process. As an example of interaction with E-Free, in the following a user, who we suppose is 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. In order to create this rule, the user clicks the "New Rule" button in the navigation bar (Figure 4, 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.

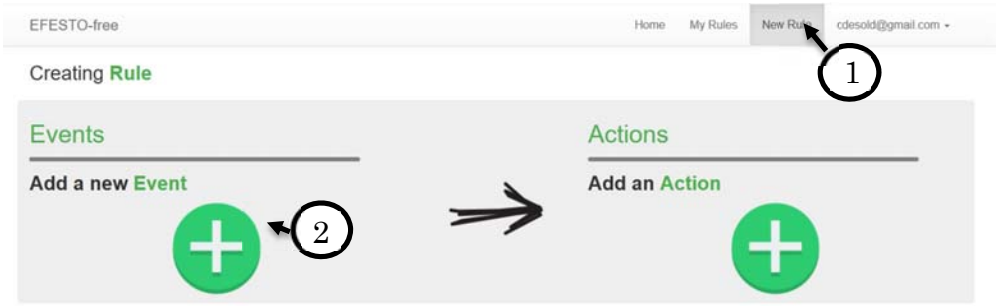
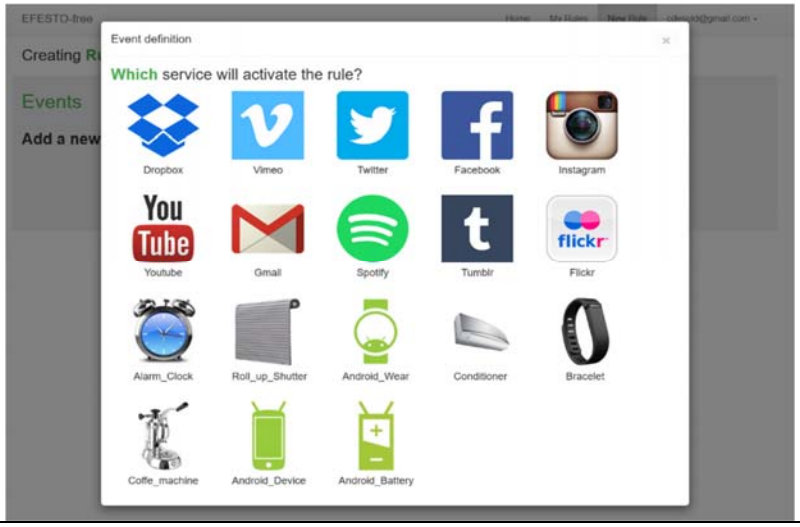


Figure 4. E-Free: the interface for rule creation.

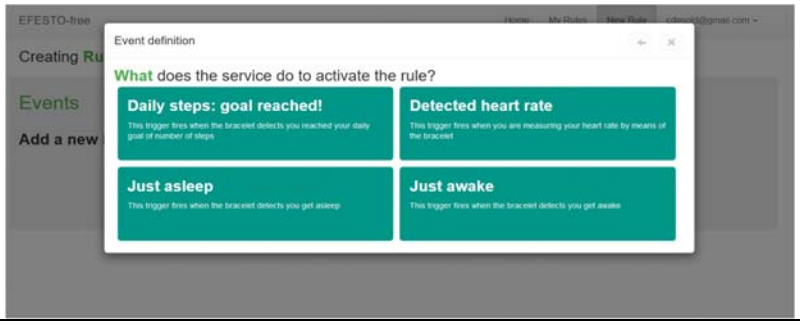
A wizard procedure, activated by the green "+" button highlighted by circle 2 in Figure 4, guides the users in defining the events. The wizard sequentially shows some pop-up windows in which the service, the events and the conditions are specified. The first wizard step is to define an event in terms of *Which* is the service to be monitored for detecting the triggering event (Figure 5a). The second step asks for *What* service event has to be monitored (Figure 5b). The last step allows the definition of *When* and *Where* the event has to be triggered (Figure 5c). 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 5d, circle 1). In the example of Figure 5d, 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 by circle 3 in Figure 5d. The button activates a wizard procedure that guides the user in the definition of 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.

[7] https://github.com/Rodsevich/JointTooledViewPlugin

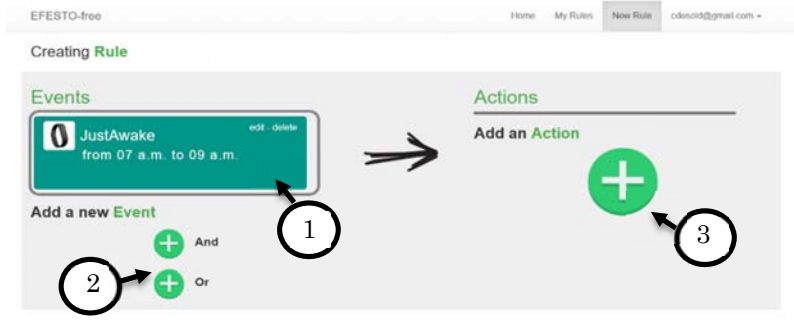


a



b

c



d

Figure 5.E-Free 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.

In E-Free, users may either define first all the events and then the actions, or define first a basic rule including one event and one action and then include new events and new actions. Events and actions can be added or removed at any time. Further events can be added by clicking one of the two green "+" buttons labelled And / Or (Figure 5d, 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 (the *Why* in the Rule_5W model). Figure 6 shows the example rule created by the user: as soon as the smart *Bracelet* detects that the user has woken up *OR* when the *Alarm Clock* rings, the *Coffee_Machine* is turned on and the *Roll-up Shutter* is opened.

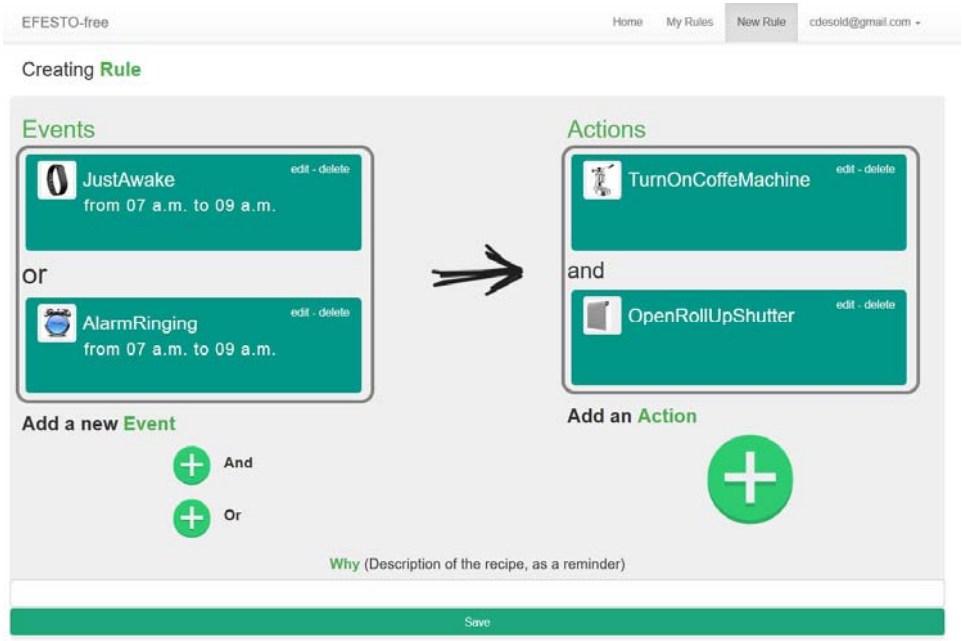


Figure 6. E-Free: example of rule including two events and two actions.

5.2 E-Wizard

With respect to E-Free, E-Wizard implements the same user interface but it also offers a wizard procedure that guides users in event and action specification. As first step, once the "New Rule" button is clicked, users are compelled to follow a wizard procedure to create a "basic rule" composed of one event and one action. Then, they can add further events and actions. This should facilitate the rule creation, because users can incrementally extend and adapt the rule to their needs. Even if the two paradigms appear very similar, in E-Free the user can freely compose the rule by adding events and actions at any moment and in no particular order. In E-Wizard, rule creation is more under the control of the system. This paradigm is also similar to the one proposed by IFTTT. However, in IFTTT the combination of multiple events and actions, as wells as temporal and spatial conditions are not allowed. We found it worth investigating the differences in the two paradigms (free vs. incremental), thus we implemented both E-Free and E-Wizard.

In E-Wizard, the first step is clicking the "Create Rule" button in the navigation bar. A wizard procedure starts by informing users what to do to create the basic rule. Then, it continues as in E-Free to guide the definition of an event in terms of *Which*,

What, *When* and *Where*. Having defined the Event, the wizard procedure, without returning to the main screen, continues with the definition of the Action. At the end, the interface in Figure 7 (identical to the E-Free interface) shows the rule that has just been created. From now on, the user might continue by adding further events or actions as she would do with E-Free.

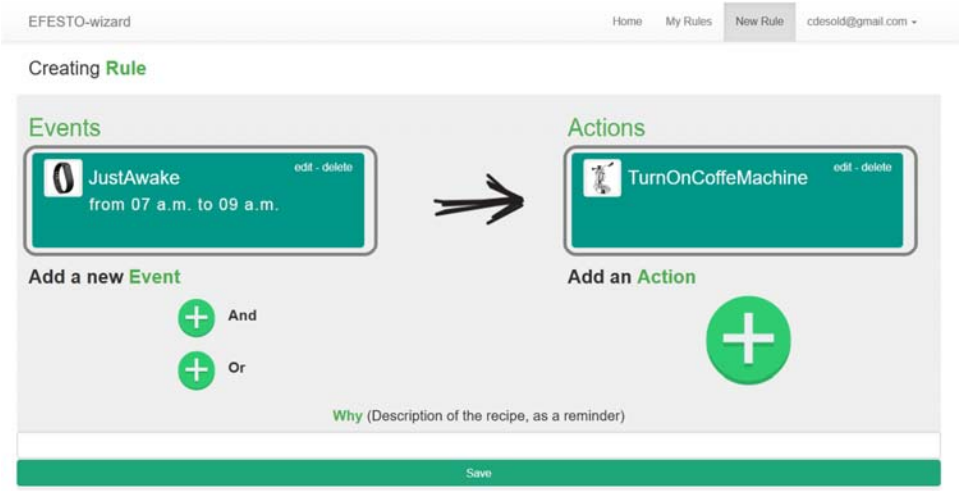


Figure 7. E-Wizard: example of 'basic rule' including one event and one action.

5.3 E-Wired

The E-Wired interaction paradigm is based on the graph metaphor: nodes represent services involved in a rule, while directed edges, i.e. arrows, represent cause-effect relationships between services. As reported in Figure 8, the E-Wired UI has two main areas. The sidebar on the left provides the list of all the available services: Web services are light-yellow, while smart objects are light-green. In the workspace area, the user builds the rule. She first selects one of the services in the left sidebar, which is added to the workspace and represented as a box augmented with two small circles, light-blue and purple, which represent the connection points for the arrows representing cause-effect relationships.

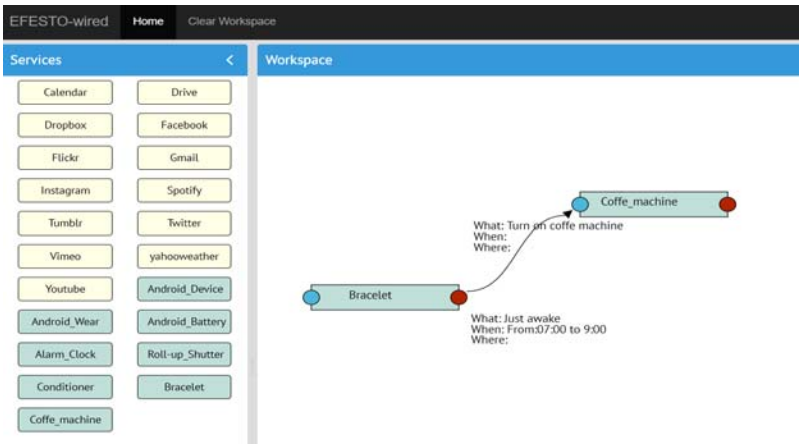


Figure 8. E-Wired: example of rule including one event and one action.

In the example illustrated in Figure 8, a Bracelet and a Coffee_machine have been added into the workspace to provide the answer to the *Which*. The two objects have been connected by drawing an arrow from the purple circle of the Bracelet, whose events have to be monitored, to the light-blue circle of the Coffee_machine that will

execute the actions. As soon as the arrow is drawn, two pop-up windows in sequence allow the user to specify the parameters of the Event (Figure 9a) and of the Action (Figure 9b) in terms of *What, When,* and *Where*. The "Create Rule" button of the second pop-up window permits to save the rule, also specifying *Why*, i.e., a title shortly describing the rule.

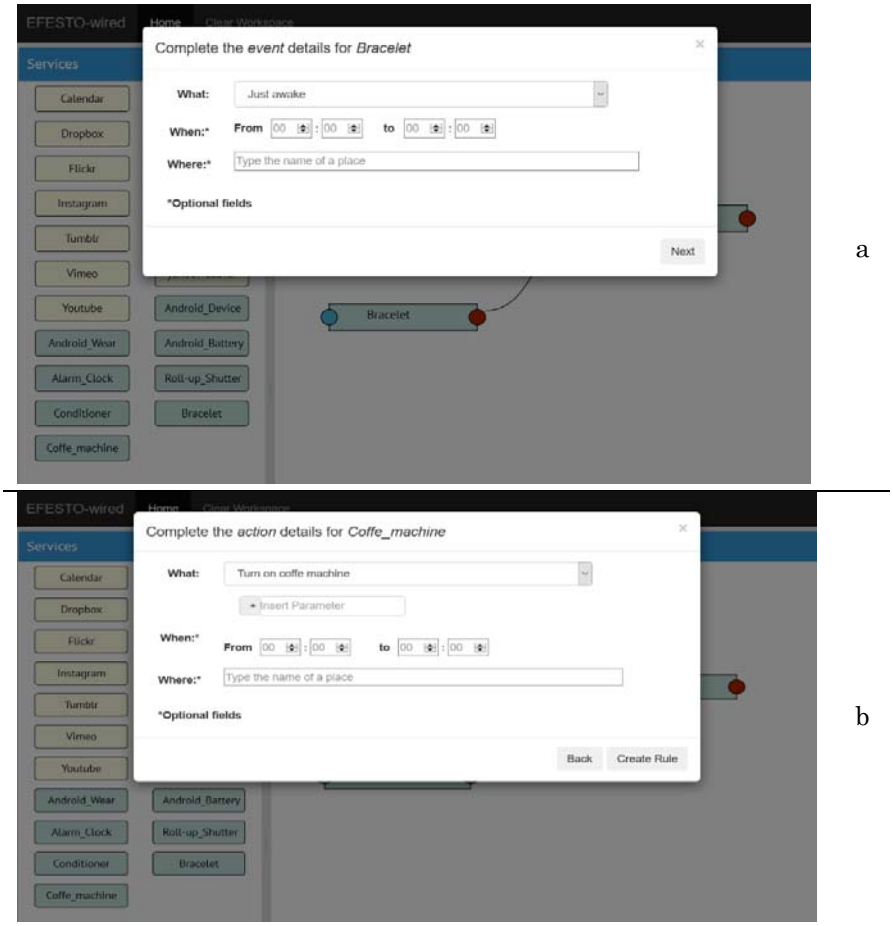


Figure 9. E-Wired: definition of a) event parameters and b) action parameters.

6. COMPARING THE PROTOTYPES

We carried out an experimental study, in order to understand if and how different composition paradigms might affect the definition of task-automation rules that include elements of the Rule_5W model. We compared our prototypes with IFTTT, which was chosen as baseline for three main reasons. First, it is widely acknowledged that IFTTT is one of the most representative tools for the class of task- automation systems [Coronado and Iglesias 2016]. IFTTT is also considered one of the most popular tools for non-programmers, and indeed it was used as baseline in other comparative studies focusing on EUD paradigms for task automation (see for example [Cabitza et al. 2015; Lucci and Paternò 2015]). Second, even though more powerful TA tools offer visual composition mechanisms, sometimes they also require programming skills to accomplish specific tasks. For example, Node-RED offers a wired visual paradigm; however, temporal and spatial constraints within rules have to be defined by writing JavaScript code. Since our target users are non-programmers, we exploited our E-Wired prototype as it can be considered a representative of the class of wire-based TA tools (a class that we wanted to include

in the comparison) and at the same time does not require writing code for rule specification. Third, we had to found a reasonable balance among number of tools, tasks, and session duration. Therefore, we could not consider as baseline more than one tool.

Further analyses were carried out by comparing only the three EFESTO systems. The reason for this second comparison is that we wanted to assess performances and satisfaction of users with the definition of more expressive rules, which is allowed by our prototype but not by IFTTT or by other TA tools without requiring some coding.

6.1 Participants and Study Design

Since the target users of the experimented applications are mainly non-technical users, i.e., persons without technical skills in computer programming or Computer Science, we recruited both technical and non-technical users. In particular, 40 participants (27 males, 13 females) were recruited among the students of the second and third year of Computer Science (n = 27), Business (n = 10) and Physiotherapy (n = 3) Bachelor degree courses of the University of Bari. The mean age was 23.38 years (SD = 2.75, min = 18, max = 30). We considered the 27 Computer Science students as technical users because they had experience with computer programming. The other 13 students were considered non-technical users because they did have neither a Computer Science background nor any experience with computer programming. This mixed sampling has been necessary also because we wanted to verify which of the experimented tools provide a composition paradigm fitting the mental model of both technical and non-technical users. The recruitment started 2 weeks before the study execution and it was performed via email. Participants were rewarded with an 8Gb USB memory stick.

Two research questions guided the study:

- RQ1) What is the difference between the considered systems in terms of *user performance* in creating rules?
- RQ2) What is the difference between the considered systems in terms of *user satisfaction*?

To answer these research questions, we performed a controlled experiment adopting a *within-subject design,* with *system* as an independent variable and four within-subject factors: E-Free, E-Wizard, E-Wired and IFTTT.

6.2 Tasks

The tasks executed during the comparative study required participants to create rules for the composition of Web services and smart objects. As reported in Table 2, four rule schemas (RS) were considered to guide the definition of tasks with different complexity. RS1 refers to the simplest rule, characterized by one event and one action, without temporal or spatial constraints. The other rule schemas introduce different operators and constraints.

Table 2. Rule schemas used to design the tasks.

Rule Schema	Rule structure	Constraints
RS1	event ⇒ action	
RS2	event1 OR event2 ⇒ action	
RS3	event ⇒ action	Temporal and spatial constraints
RS4	event1 OR event2 ⇒ action1 AND action2	Temporal and spatial constraints

RS2 represents rules with multiple events combined with the OR operator. Tasks based on the RS2 schema can be performed with IFTTT. In fact, the RS2 schema (*event1 OR event2 ⇒ action*) is logically equivalent to two rules (*event1 ⇒ action)* and (*event2 ⇒ action).* Thus, to accomplish RS2 tasks with IFTTT participants had to create two rules.

Tasks based on RS3 and RS4 schemas were not performed with IFTTT, because logical operators on actions and temporal and spatial constraints are not allowed.

For each rule schema, two tasks were defined: one required the composition of Web services, the other required the composition of smart objects. Thus, every participant performed eight tasks with each system, except for IFTTT, for which they performed only the 4 tasks allowed by this tool. Each participant performed 28 tasks, for a total of 40 users x 28 tasks = 1120 trials. The defined tasks, with their short identification (Task ID), are reported in Table 3. To improve the external validity of the study, all the tasks required the inclusion of Web services and smart objects used in the most popular rules created by the IFTTT community. Four out of the eight proposed tasks required the parameterization of events and/or actions.

Table 3. The 8 experimental tasks used in the comparative study.
Tasks labeled with * cannot be performed with IFTTT.

Rule Schema	Task ID	Task statement
RS1	T1	If I post a picture on Instagram, then post the same picture on Twitter
	T2	If my Android device battery drops below 15%, then send a notification on my Android Wear
RS2	T3	If I post a picture on Instagram OR Flickr, then post the same picture on Twitter
	T4	If my Android Device battery drops below 15% OR my Android device connects to any Wi-Fi network, then send a notification to my Android Wear
RS3	T5*	If I post a text on Twitter geo-localized in Milan between 8-12 a.m., then create a Google Drive document having as its name the tweet id and as its content the tweet text.
	T6*	If my Android Wear changes position to the address of my home between 7-11 p.m., then switch-on the home air conditioners setting 25 °C as temperature.
RS4	T7*	If I post on Instagram or Flickr a picture geo-localized in Rome between 8-12 a.m., then post the same picture on Twitter AND Tumblr with the same geo-localization data.
	T8*	If between 7-8 a.m. my Android Wear changes to the address of my home OR my smart alarm clock rings, then switch-on the coffee machine and open the roll-up shutter of my bedroom.

6.3 Procedure

The study took place in a quiet university room where the study apparatus was installed. Two HCI experts were involved: one acted as observer, the other as facilitator. A laptop with a 15-inch display provided with an external mouse was available. The observation of the user interaction with the systems was facilitated by an external monitor that duplicated the laptop screen (see Figure 10). The comparative study lasted 10 days. Four participants were individually observed each day. Every participant followed the same procedure. First, they were introduced to the study purpose and what they had to do. Nobody refused to participate in the study. Participants were asked to sign a consent form.

The participants were provided with a booklet composed of four pages. Each page reported three training and eight experimental tasks to be performed with one of the compared systems. Only four experimental tasks were reported on the IFTTT page, namely those supported by this system. To avoid carry-over effect, the booklet pages, as well as the experimental tasks on each page were ordered to have the system test order counterbalanced across the participants, and the task set order counterbalanced across the experimental conditions, both according to a Latin Square design.

The facilitator introduced the first system, i.e., the one reported on the first page of the participant booklet, and demonstrated the creation of a rule based on the RS4 schema, being this the most complete one. The demonstration did not involve services used in the experimental tasks. Then, the participants were invited to perform the three training tasks, possibly asking the facilitator for help.

After the training, the participants had to execute the experimental tasks alone. They had to read aloud the task text and then start the rule creation. At the end of each task execution, participants had to tick a checkbox associated to the task statement, in order to indicate if, in their opinion, the created rule fulfilled the task requests or not. At the end of all the experimental tasks, they filled in an online questionnaire about the system they had used. Before repeating the same procedure with the next system, the participants were invited to relax for five minutes.

A paper questionnaire was administered at the end of the participant's session. It asked to rank the four systems on the basis of their usefulness, completeness and ease of use and to vote for the best system.

This procedure was preliminarily assessed by a pilot study involving three participants.

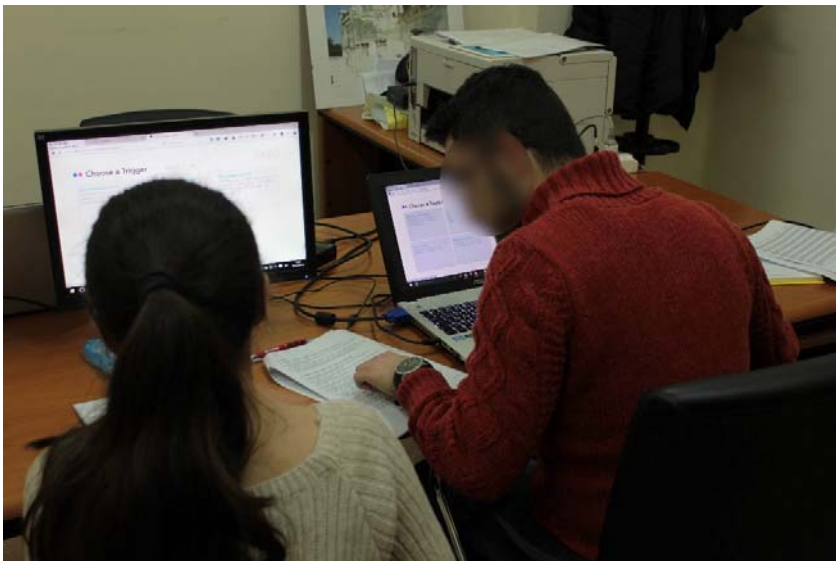


Figure 10. Comparative study setting.

6.4 Data Collection

Different types of data, both quantitative and qualitative, were collected to evaluate respectively user performances and user preferences.

Regarding user performances, all the interactions were audio-video recorded by using a screen-capture tool. Notes were taken by the observers on significant behavior or externalized comments. The two researchers transcribed their notes and performed an audio-video analysis of the screen-capture records. As result, they built an excel file reporting for each task performed by each user the following data: user ID (from 1 to 40), user experience (technical – non-technical), system name, task ID (T1-T8), task schema (RS1, RS2, RS3, RS4), type of service (Web service – smart-object), time (in seconds), clicks, number of errors (from 0 to n), type of error, user perception about rule correctness (right – wrong). Then, they independently double-checked such data. The initial reliability value was 94%, thus the researchers discussed the differences and reached a full agreement.

Regarding user preferences, online and paper questionnaires were administered during the study. The online questionnaire addressed two main dimensions:

satisfaction with system and *satisfaction with created rules*. The former was assessed through a semantic-differential scale that required participants to judge the system on 12 pairs of adjectives describing satisfaction in using the system. Participants could modulate their evaluation on 7 points (after re-coding of reversed items 1 = very negative, 7 = positive). Such tool for measuring user satisfaction with system was already used in some previous similar studies of ours (see, for example, [De Angeli et al. 2003]) and was designed with the involvement of an experimental psychologist working in the HCI field. The latter was assessed directly by a Likert-type item asking participants to express their gratification on a 7 point-scale (from "not at all" to "very much") and indirectly by a percentage estimation of the number of tasks accomplished correctly. The paper questionnaire asked participants to rank experimented systems along three dimensions, i.e., completeness, easiness and usefulness.

6.5 Results on User Performance

In order to answer to the first research question, namely if there is a difference in terms of user performance in creating rules between the interaction paradigms, we structured our analyses along five dimensions. The first dimension, *Rule Complexity Impact*, compared the systems along different rule schemas implying different levels of complexity. The second dimension, *Type of Service Impact*, aimed to understand, for each system, if and how the type of service (Web service vs. smart object) affects user performance. The third dimension, *User Expertise Impact*, refers to the effect of the user expertise (technical vs. non-technical) on the user performance. The fourth dimension, *User Gender Impact*, refers to the effect of gender (male vs. female) on the user performance. The fifth dimension, *Expressiveness of rule representation*, aims to evaluate if the way in which rules under creation are represented helps users understand and perceive correctly the effect of their composition actions, and if there are differences among the four systems. This analysis actually addressed both performance and satisfaction: the wrong perception of the created rule not only impacts users' satisfaction but can also generate errors.

The variables *time*, *clicks* and *error severity* were used in the first four analysis dimensions. *Time* is the number of seconds the participant took to execute a task; we considered as task starting point the moment when the participant finished reading aloud the task statement. The ending point was the moment when the participant saved the rule. The variable *Clicks* refers to the number of clicks. It was calculated by analyzing the system logs. *Error severity* is an index that indicates the seriousness of the task errors. A rate was assigned to each error depending on its severity. In particular, triangulating the notes and video analysis, three different types of error rates were identified. The first type is about wrong events or actions in a rule. The second error type is related to wrong parameters in the specification of events and actions. The last error type is about using a wrong logical operator (AND instead of OR) to connect the events. A different score was assigned to each type of errors, respectively 3, 2 and 1 (3 is the most serious). The error rate of a specific task execution was thus calculated as sum of the scores of all the errors observed in a screen snapshot showing the final rule. The final error severity index was calculated as ratio between the sum of the error rates and the number of errors (0 was assigned in case of no errors).

One-way repeated measures ANOVAs (all Greenhouse-Geisser corrected) with post-hoc pairwise comparisons (Bonferroni corrected) were adopted in the first analysis that addressed the role of different rule schemas. In Sections 6.5.1 we report in detail the analyses of data related to this dimension. By means of tables, for each system we report mean ($\bar{x}$) and standard deviation (SD) of the dependent variables, the results of the ANOVA tests (p-values below .05 indicate a significant difference),

and the results of the *Post-hoc* tests. The latter highlight the couples of systems for which a statistically significant difference exists (e.g., *E-Free – E-Wired (p=.019)* indicates that the first system performed better than the second one). Log transformations were needed to achieve a normal data distribution (assessed with Shapiro-Wilk test). However, the tables report the original data, without log-transformation, to better show the real system performances.

For the second, third and fourth analyses, paired-sample t-tests were performed: in the related tables, reported in Section 6.5.2, 6.5.3 and 6.5.4, respectively, the column *T-test* reports test results (a significant difference exists when p-value < .05).

Lastly, chi-square tests were applied to data in the fifth analysis, as shown in Section 6.5.5.

6.5.1 Rule complexity impact

A detailed comparison among the systems was performed to analyze if and how the rule schema (RS1, RS2, RS3 and RS4) affects the dependent variables.

Table 4 shows that there was a significant difference among the systems in the time to create an RS1 rule, i.e., a rule not requiring either logical operators not even temporal and spatial constraints. In particular, IFTTT required more time than E-Free and E-Wizard, while there are no significant differences between E-Wired and IFTTT. Regarding clicks, significant differences emerged: E-Wired required less clicks than E-Free and E-Wizard. IFTTT required less clicks than E-Free. No differences emerged for the error severity.

Table 4. Performances in task execution time, clicks and error severity of all the systems along rule schemas RS1.

	System	Time		Clicks		Error Severity	
		$\bar{x}$	SD	$\bar{x}$	SD	$\bar{x}$	SD
Rule Schema 1	**E-Free**	33.82	15.84	13.27	4.63	.00	.00
	E-Wizard	35.20	17.78	12.23	4.22	.05	.35
	E-Wired	38.06	21.27	10.61	3.36	.06	.24
	IFTTT	46.22	24.91	11.14	2.77	.00	.00
	ANOVA Test	F(2.771, 218.879) =7.725 p < .000 partial η2 = .098		F(2.798, 221.023) = 7.275 p < .000 partial η2 = .084		F(1.702, 134.442) = 1.953 p = .152 partial η2 = .024	
	Post-hoc	E-Free – IFTTT (p=.001) E-Wizard – IFTTT (p=.002)		E-Wired – E-Free (p<.000) IFTTT – E-Free (p=.010) E-Wired – E-Wizard (p=.044)		---	

A possible explanation of the two EFESTO systems advantage over IFTTT for the time variable can be ascribed to the different organization of the wizard procedure for selecting events, actions and their parameters. In order to define a rule, IFTTT requires passing through every single step, even when some elements do not need to be specified for a rule under definition. For example, even if actions do not need the specification of parameters, as it happens in one of the two RS1 rules, the users have to pass anyway through the page dedicated to parameter definition. More than a critical problem, this could be considered a hint for designers of similar tool.

Table 5. Performances in task execution time, clicks and error severity of all the systems along rule schemas RS2.

	System	Time		Clicks		Error Severity	
		$\bar{x}$	SD	$\bar{x}$	SD	$\bar{x}$	SD
Rule Schema 2	**E-Free**	61.39	27.64	19.62	7.04	.18	.56
	E-Wizard	66.86	29.21	18.23	6.21	.24	.73
	E-Wired	78.63	42.78	23.61	12.82	.05	.21
	IFTTT	100.69	40.42	24.42	4.87	.05	.35
	ANOVA Test	F(2.532, 200.065) =32.508 p < .000 partial η^2 = .292		F(2.714, 211.689) =21.365 p < .000 partial η^2 = .215		F(2.350, 185.658) =2.842 p = .052 partial η^2 = .035	
	Post-hoc	E-Free – E-Wired (p<.000) E-Free – IFTTT (p<.000) E-Wizard – IFTTT (p<.000) E-Wired – IFTTT (p<.000)		E-Free – E-Wired (p=.007) E-Free – IFTTT (p<.000) E-Wizard – E-Wired (p<.000) E-Wizard – IFTTT (p<.000)		---	

The time gap between IFTTT and the EFESTO systems, already evident in RS1 tasks, is even more accentuated in RS2 tasks that required the use of the OR logical operator between two events (Table 5). It is not uncommon that a user needs to activate the same action as reaction to different events; in this situation, the paradigms of the EFESTO systems are more efficient than IFTTT that, as it does not support logical operators, requires the definition of two rules to perform RS2 tasks.

E-Free, E-Wizard and E-Wired were then compared along the RS3 and RS4 tasks to determine which paradigm is most suitable for managing rules that also include logical operators and temporal and spatial constraints (see Table 6 and Table 7, respectively). IFTTT was not considered because it does not support these features.

There were significant differences in the time to create a rule. In particular, post-hoc analysis revealed that participants were faster with E-Free and E-Wizard than with E-Wired. With respect to clicks, it emerged that E-Wired requires a number of clicks significantly higher than E-Wizard. In terms of error severity, the only significant difference emerged in RS3 tasks: participants performed less severe errors using E-Wizard than using E-Wired.

The video analysis of the user interaction highlighted that participants spent more time with E-Wired because of the rule graph representation. Drag&drop of the nodes representing services and definition of the relationships by arrow drawing was time-consuming.

Table 6. Performances in task execution time, clicks and error severity of the EFESTO systems along rule schema RS3.

	System	Time		Clicks		Error Severity	
		$\bar{x}$	SD	$\bar{x}$	SD	$\bar{x}$	SD
Rule Schema 3	**E-Free**	62.95	31.22	21.42	6.11	.16	.46
	E-Wizard	69.34	34.78	20.41	5.81	.08	.26
	E-Wired	98.14	55.13	30.89	19.16	.23	.42
	ANOVA Test	F(1.808, 142.794) =14.771 p < .000 partial η^2 = .158		F(1.581, 124.885) =10.624 p < .000 partial η^2 = .119		F(1.761, 139.112) =3.200 p = .050 partial η^2 = .039	
	Post-hoc	E-Free – E-Wired (p<.000) E-Wizard – E-Wired (p<.000)		E-Free – E-Wired (p=.003) E-Wizard – E-Wired (p=.001)		E-Wizard – E-Wired (p=.020)	

Table 7. Performances in task execution time, clicks and error severity of the EFESTO systems along rule schema RS4.

	System	Time		Clicks		Error Severity	
		$\bar{x}$	SD	$\bar{x}$	SD	$\bar{x}$	SD
Rule Schema 4	**E-Free**	122.32	47.31	41.31	13.40	.41	.80
	E-Wizard	147.36	76.79	39.57	13.54	.51	.90
	E-Wired	198.93	85.78	57.61	28.89	.59	.73
	ANOVA Test	F(1.760, 139.017) =33.726 p < .000 partial η^2 = .299		F(1.854, 146.494) =20.671 p < .000 partial η^2 = .207		F(1.893, 149.566) =1.132 p = .323 partial η^2 = .014	
	Post-hoc	E-Free – E-Wired (p<.000) E-Wizard – E-Wired (p<.000)		E-Free – E-Wired (p<.000) E-Wizard – E-Wired (p<.000)		---	

6.5.2 Type of service impact

For each system, we investigated if there are statistical differences in performing tasks with different types of service, i.e., *Web service* and *smart object* (see Table 8).

Time and *clicks* values were significantly different in favor of smart objects for all systems. A possible reason can be the number of events and actions exposed by smart objects, which in general is lower than the number of events and actions exposed by Web services. As also observed in [Lucci and Paternò 2015], an excessive number of service elements to choose from can make it difficult to identify the right one. Furthermore, the parameterization of events is typically more complex in Web services. About the error severity, there was a statistical difference in favor of smart objects only in E-Free. Even in this case, the reason could be related to the greater complexity of parameterization in Web services.

Table 8. Performances in task execution time, clicks and error severity of the four systems along the type of service. Tests labeled with * are statistically significant.

System	Variable	Web Services		Smart objects		T-Test
		$\bar{x}$	*SD*	$\bar{x}$	*SD*	
E-Free	*Time*	87.50	49.81	52.74	33.32	t(159) = 8.296, p < .000*****
	Clicks	30.12	14.54	17.69	8.69	t(159) = 10.747, p < .000*****
	Error Severity	.24	.63	.13	.47	t(159) = 1.795, p = .075
E-Wizard	*Time*	96.46	67.97	62.92	48.59	t(159) = 6.928, p < .000*****
	Clicks	27.64	13.07	17.58	11.19	t(159) = 9.096, p < .000*****
	Error Severity	.24	.66	.20	.62	t(159) = .564, p = .574
E-Wired	*Time*	119.23	86.95	87.64	72.80	t(159) = 5.237, p < .000*****
	Clicks	33.39	24.36	27.97	25.86	t(159) = 3.716, p < .000*****
	Error Severity	.18	.47	.00	.00	t(159) = 2.023, p = .045*****
IFTTT	*Time*	88.44	46.69	58.48	33.49	t(79) = 5.622, p < .000*****
	Clicks	19.40	8.28	16.16	6.85	t(79) = 2.431, p = .017*****
	Error Severity	.00	.00	.05	.35	t(79) = –1.270, p = .208

6.5.3 User expertise impact

We compared, for each system, the performance of *technical users* vs. *non-technical users*. As shown in Table 9, the only significant difference regards the error severity of E-Wizard that is in favor of non-technical users. However, since the error severity value is very low in both cases, we can safely assume that E-Wizard does not specifically lead the users to perform important errors.

Table 9. Performances in task execution time, clicks and error severity of the four systems along the user expertise. Tests labeled with * are statistically significant.

System	Variable	Technical users		Non-technical users		T-Test
		$\bar{x}$	*SD*	$\bar{x}$	*SD*	
E-Free	*Time*	69.73	46.66	70.94	44.02	t(103) = .457, p = .649
	Clicks	24.19	13.85	23.33	12.73	t(103) = .719, p = .473
	Error Severity	.21	.63	.17	.44	t(103) = .649, p = .518
E-Wizard	*Time*	81.09	62.11	76.79	59.88	t(103) = 1.319, p = .190
	Clicks	23.22	14.09	21.35	10.91	t(103) = .634, p = .527
	Error Severity	.35	.81	.14	.44	t(103) = 2.221, p = .029*****
E-Wired	*Time*	99.45	75.06	111.72	93.61	t(103) = –.305, p = .761
	Clicks	30.15	24.26	31.79	27.22	t(103) = –.948, p = .345
	Error Severity	.30	.52	.23	.47	t(103) = –.961, p = .339
IFTTT	*Time*	71.88	41.57	76.73	46.66	t(51) = .017, p = .987
	Clicks	17.57	7.73	18.21	7.83	t(51) = –.524, p = .603
	Error Severity	.00	.00	.06	.42	t(51) = –1.000, p = .322

No differences emerged in terms of time and clicks. Thus, the EFESTO system paradigms, as well as the one implemented in IFTTT, seem to fit the mental model of non-technical users since they do not affect their performances. This result confirms

that the elicitation study (see Section 4) helped us identify paradigms that can be exploited also by non-technical users, despite the use of logical operators and temporal and spatial constrains that represent the step forward compared to the existing TA systems analyzed in Section 2.2.

6.5.4 User gender impact

Gender is considered an issue relevant in EUD research, especially in the smart environment [Blackwell et al. 2009]. In order to assess the gender effect on the evaluated systems, we also compared the performance of *male* vs. *female* participants. As shown in Table 10, no significant differences emerged from this analysis.

Table 10. Performances in task execution time, clicks and error severity of the four systems along the user gender. Tests labeled with * are statistically significant.

System	Variable	Male		Female		T-Test
		$\bar{x}$	*SD*	$\bar{x}$	*SD*	
E-Free	*Time*	71.08	47.80	67.89	40.69	t(95) = 1.120, p = .266
	Clicks	24.08	13.85	23.52	12.62	t(95) = .371, p = .712
	Error Severity	.29	.767	.22	.527	t(95) = .758, p = .451
E-Wizard	*Time*	79.73	62.13	79.60	59.74	t(95) = 1.047, p = .298
	Clicks	22.75	13.56	22.27	12.21	t(95) = .350, p = .727
	Error Severity	.41	.878	.21	.640	t(95) = 1.807, p = .074
E-Wired	*Time*	101.18	76.98	108.71	91.73	t(95) = .133, p = .894
	Clicks	30.33	25.81	31.51	23.91	t(95) = –.759, p = .450
	Error Severity	.32	.514	.27	.594	t(95) = .644, p = .521
IFTTT	*Time*	75.43	44.89	68.85	39.03	t(47) = 1.197, p = .237
	Clicks	17.85	7.76	17.63	7.78	t(47) = .032, p = .795
	Error Severity	.02	.144	.06	.433	t(47) = –.628, p = .533

6.5.5 Expressiveness of rule representation

An important feature of task-automation systems is the capability of communicating to the user if the created rule, statically visualized on the computer display, will generate the expected service behavior when executed on actual objects and services. In fact, one of the limitations of current TA systems is the limited debugging possibilities [Coutaz and Crowley 2015].

This aspect was investigated by means of two dichotomous variables: *perceived rule correctness* (right – wrong) and *actual rule correctness* (right – wrong). The value of the former was explicitly stated by participants, who, for each task, ticked a checkbox indicating if the created rule fulfilled or not the task requests. The actual rule correctness value was indicated by the two HCI researchers, who examined the created rule through the screen snapshots.

For each system, chi-square tests were applied on the two dichotomous variables. There was a statistically significant association between *perceived rule correctness* and *actual rule correctness* in E-Free, E-Wizard and E-Wired ($\chi^2_{\text{E-Free}}(1) = 46.506$, $p < .000$; $\chi^2_{\text{E-Wizard}}(1) = 39.811$, $p < .000$; $\chi^2_{\text{E-Wired}}(1) = 20.080$, $p < .000$), with a moderately strong association in all cases ($\varphi_{\text{E-Free}} = 0.381$, $p < .000$; $\varphi_{\text{E-Wizard}} = 0.250$, $p < .000$; $\varphi_{\text{E-Wired}} = 0.353$, $p < .000$). No statistical difference emerged in IFTTT ($\chi^2_{\text{IFTTT}}(1) = .121$, $p = .728$).

Table 11 summarizes and offers a different perspective on the results of this analysis. The *Correctly perceived* column reports, for each system, the number of

rules participants correctly declared to fulfill (*As right* column) or not (*As wrong*) the task requests. The *Total* column, calculated as sum of the two previous columns, indicates the number of rules participants correctly perceived. Percentage, calculated on a total of 320 rules for the three EFESTO versions and 160 for IFTTT, is reported in the *%* column. Similarly, the *Wrongly perceived* column reports the same information related to the rules wrongly perceived by participants.

Triangulating the chi-square test results with the data reported in Table 11, we can assert that the EFESTO systems offer more adequate rule representation. The chi-square test revealed that no association exists between the two variables in the case of IFTTT, thus indicating that its rule representation generates a wrong user perception, even if it has the lowest percentage of wrongly perceived rules (6.9%). This contrasting result is explained by observing that participants were never able to identify wrong rules: the 2 wrong rules were never recognized and the 9 rules perceived as wrong were right (100% of failures). This is the main cause of the weak association between the two dichotomous variables. To identify the source of this problem, we considered E-Wizard that, although very similar to IFTTT, does not suffer this problem. In fact, 12 out of 30 wrong rules (40% of failures) were perceived correctly. IFTTT proposes a similar wizard as in E-Wizard with the same service names, events and actions. The very difference between these two systems is in the rule representation. E-Wizard shows the rule in a horizontal panel, with events on the left and actions on the right (see Figure 7); this representation is used both at creation time and when the rule has been saved, according to a "What You See Is What You Get" (WYSIWYG) approach. IFTTT shows, at creation time, the rule elements in a scrolling page, where they appear in sequence as soon they are defined; when the rule is saved, it is shown as a text that summarizes its features in natural language.

Table 11. Participants' perception of the created rule correctness.

System	Correctly perceived				Wrongly perceived			
	As right (N)	As wrong (N)	Total (N)	%	As right (N)	As wrong (N)	Total (N)	%
E-Free	268	14	282	88.1	28	10	38	11.9
E-Wizard	259	12	271	84.7	31	18	49	15.3
E- Wired	241	22	263	82.2	41	16	57	17.8
IFTTT	149	0	149	93.1	2	9	11	6.9

We were also interested in investigating if the participants' background affected their ability to evaluate rule correctness. Thus, the analysis has been executed considering the two different participant samples, i.e., *technical* vs. *non-technical*.

In the case of technical users, there was a statistically significant association between *perceived rule correctness* and *actual rule correctness* in E-Free, E-Wizard and E-Wired ($\chi^2_{\text{E-Free}}(1) = 33.252$, $p < .000$; $\chi^2_{\text{E-Wizard}}(1) = 9.896$, $p = .002$; $\chi^2_{\text{E-Wired}}(1) = 48.756$, $p < .000$) with a moderately strong association ($\varphi_{\text{E-Free}} = 0.392$, $p < .000$, $\varphi_{\text{E-Wizard}} = 0.214$, $p = .002$, $\varphi_{\text{E-Wired}} = 0.475$, $p < .000$). No statistical difference emerged in IFTTT ($\chi^2_{\text{IFTTT}}(1) = .049$, $p = .825$).

In the case of non-technical users, there was a statistically significant association between *perceived rule correctness* and *actual rule correctness* in E-Free and E-Wizard ($\chi^2_{\text{E-Free}}(1) = 13.505$, $p < .000$; $\chi^2_{\text{E-Wizard}}(1) = 12.486$, $p < .000$) with a moderately strong association between the variables ($\varphi_{\text{E-Free}} = 0.360$, $p < .000$, $\varphi_{\text{E-Wizard}} = 0.346$, $p < .000$, $\varphi_{\text{E-Wired}} = 0.475$, $p < .000$). No statistical difference emerged in E-Wired and IFTTT ($\chi^2_{\text{E-Wired}}(1) = 1.878$, $p = .171$, $\chi^2_{\text{IFTTT}}(1) = .085$, $p = .771$).

Table 12 reports participants' performance in determining rule correctness, with a distinction between technical and non-technical users. The main evidence is that the E-Wired rule representation, based on a graph metaphor, is not correctly interpreted by non-technical users, who are not acquainted with a language typical of the Computer Science domain.

Table 12. Influence of participant background (technical vs. non-technical) on the perception of the created rule correctness.

System	Technical Users								Non-technical Users							
	Correctly perceived				Wrongly perceived				Correctly perceived				Wrongly perceived			
	As right (N)	As wrong (N)	Total (N)	%	As right (N)	As wrong (N)	Total (N)	%	As right (N)	As wrong (N)	Total (N)	%	As right (N)	As wrong (N)	Total (N)	%
E-Free	183	9	192	88.9	18	6	24	11.1	85	5	90	86.5	10	4	14	13.5
E-Wizard	177	6	183	84.7	25	8	33	15.3	82	6	88	84.6	6	10	16	15.4
E-Wired	168	17	185	85.6	25	6	31	14.4	73	5	78	75.0	16	10	26	25.0
IFTTT	102	0	102	94.4	1	5	6	5.6	47	0	47	90.4	1	4	5	9.6

6.6 Results on User Satisfaction

User satisfaction was assessed by means of two types of questionnaires. The first one was an online questionnaire filled in by participants after the use of each system. The second one was a paper questionnaire administered at the end of the participant's session; it asked to rank the four systems on the basis of their usefulness, completeness and easiness, and to vote for the best system.

6.6.1 User satisfaction with system and with created rules

The online questionnaire addressed two main dimensions: *satisfaction with system* and *satisfaction with created rules*. One-way repeated measures ANOVAs (all Greenhouse-Geisser corrected) with post-hoc pairwise comparisons (Bonferroni corrected) were adopted to determine whether there were statistically significant differences in these two dimensions.

User *satisfaction with system* was addressed through a semantic-differential scale that required participants to judge the system on 12 pairs of adjectives describing satisfaction in using the system. The questionnaire had a high level of internal consistency, as determined by a Cronbach's alpha of .940. The index of user satisfaction with system was computed averaging the scores for the 12 items.

User *satisfaction with created rules* was assessed directly by a Likert-type item asking participants to express their gratification on a 7 point-scale (from "not at all" to "very much") and indirectly by a percentage estimation of the number of tasks accomplished correctly. The two variables are highly correlated ($r = .498$, $p < .000$), thus indicating that the more tasks participants think they accomplished correctly, the more satisfied they are with system performance. Consequently, the final index of user satisfaction with created rules was computed multiplying the two scores.

Table 13 shows that E-Free, E-Wizard and IFTTT were the systems most preferred by participants. About satisfaction with created rules a significant difference emerged only between IFTTT and E-Wired. Therefore, E-Wired in general was perceived as the worst one.

Table 13. User satisfaction results (the highest $\bar{x}$ value is the best).

System	Satisfaction with system		Satisfaction with created rules	
	$\bar{x}$	SD	$\bar{x}$	SD
E-Free	5.81	.89	488.50	167.095
E-Wizard	5.70	.86	485.75	485.69
E-Wired	4.55	1.14	425.75	184.97
IFTTT	5.73	.98	526.00	155.13
ANOVA Test	$F(2.806, 109.423) = 19.897$ $p < .000$ partial $\eta^2 = .339$		$F(2.868, 2374835) = 3.387$ $p = .022$ partial $\eta^2 = .080$	
Post-hoc	E-Free – E-Wired (p<.000) E-Wizard – E-Wired (p<.000) IFTTT – E-Wired (p<.000)		IFTTT – E-Wired (p=.031)	

We also investigated if the participants' expertise and gender affected their satisfaction with system and with created rules. Paired-sample t-tests were executed considering *technical* vs. *non-technical* and *male* vs. *female*.

Table 14. User satisfaction results: technical vs. non-technical (the highest $\bar{x}$ value is the best).

System	Variable	Technical Users		Non-Tech. Users		T-Test
		$\bar{x}$	*SD*	$\bar{x}$	*SD*	
E-Free	*Satisf. with system*	5.68	.88	6.07	.87	t(12) = .861, p = .406
	Satisf.with created rules	460.00	169.27	547.69	151.78	t(12) = .928, p = .372
E-Wizard	*Satisf. with system*	5.56	.91	5.97	.67	t(12) = 3.108, p = .009*****
	Satisf. with created rules	458.52	189.18	542.31	171.42	t(12) = 1.653, p = .124
E-Wired	*Satisf. with system*	4.41	.98	4.84	1.41	t(12) = 1.268, p = .229
	Satisf. with created rules	415.96	181.90	446.15	197.08	t(12) = .584, p = .570
IFTTT	*Satisf. with system*	5.80	.83	5.55	1.26	t(12) = –.068, p = .947
	Satisf. with created rules	517.78	161.54	543.08	145.62	t(12) = .265, p = .796

Table 15. User satisfaction results: male vs. female (the highest $\bar{x}$ value is the best).

System	Variable	Male		Female		T-Test
		$\bar{x}$	*SD*	$\bar{x}$	*SD*	
E-Free	*Satisf. with system*	5.91	.52	5.62	1.13	t(11) = .739, p = .476
	Satisf.with created rules	487.00	158.98	505.00	154.06	t(11) = -.241, p = .814
E-Wizard	*Satisf. with system*	5.22	.86	5.63	.92	t(11) =-1.037, p = .322
	Satisf. with created rules	430.83	185.88	477.50	201.54	t(11) = -.565, p = .584
E-Wired	*Satisf. with system*	4.24	1.28	4.65	1.27	t(11) = -1.099, p = .295
	Satisf. with created rules	391.67	212.80	436.66	176.85	t(11) = .-576, p = .576
IFTTT	*Satisf. with system*	5.56	.92	5.32	1.30	t(11) = .513, p = .618
	Satisf. with created rules	511.66	139.79	508.33	150.26	t(11) = .055, p = .957

In case of participants' expertise, with respect to the *Satisfaction with system*, the average values reported in Table 14 show that there is a positive attitude of technical users towards using IFTTT, while non-technical users prefer E-Free. There is only one significant difference between technical and non-technical users: the latter are more satisfied with the E-Wizard system. The same trend emerged in the *Satisfaction with created rules*, even if no significant differences were highlighted. In case of participants' gender, no differences emerged in any case (Table 15).

6.6.2 User ranking of systems along completeness, easiness and usefulness

Participants were asked to rank the four systems along three dimensions, i.e., completeness, easiness and usefulness. As shown in Table 16, rankings were significant in all the cases (see the Kendall's W coefficient reported in the Test row of the table). Wilcoxon signed-rank tests were used as post-hoc tests to determine which scores were significant. E-Free resulted as the favorite system in all the three dimensions, followed by E-Wizard, IFTTT and lastly E-Wired.

Table 16. User preference in terms of completeness, easiness and usefulness (the lowest $\bar{x}$ value is the best).

System	**Completeness**		**Easiness**		**Usefulness**	
	$\bar{x}$	SD	$\bar{x}$	SD	$\bar{x}$	SD
E-Free	1.85	.95	1.80	.91	1.62	.93
E-Wizard	2.50	.96	2.45	.86	2.10	.90
E-Wired	2.88	1.16	3.25	1.01	3.20	.97
IFTTT	2.78	1.14	2.50	1.20	3.10	.81
Kendall's Test	Kendall's W= .128 $\chi^2(3) = 15.33$ p = .002		Kendall's W= .211 $\chi^2(3) = 25.32$ p < .000		Kendall's W = .357 $\chi^2(3) = 42.895$ p < .000	
Post-hoc	E-Free – E-Wired (p=.001) E-Free – E-Wizard (p=.023) E-Free – IFTTT (p=.001)		E-Free – E-Wired (p<.000) E-Free – E-Wizard (p=.011) E-Free – IFTTT (p=.019) E-Wizard – E-Wired (p=.003) IFTTT-E-Wired (p=.022)		E-Free – E-Wired (p<.000) E-Free – E-Wizard (p=.042) E-Free – IFTTT (p<.000) E-Wizard – E-Wired (p<.000) Wizard – IFTTT (p<.000)	

We also investigated if the participants' background and gender affected the rank along the three dimensions. For each system, Wilcoxon signed-rank tests were executed with respect to each dimension, comparing technical vs. non-technical as well as male vs. female participants. As confirmed by data and tests reported in Table 17 and Table 18, neither participants' background nor gender do affect their ranking of the experimented systems.

About voting the best system, the result is: E-Free = 27, E-Wizard = 8, IFTTT = 3 and E-Wired = 2. Voting results are in line with the user preferences expressed in the three rankings.

Table 17. User ranking along completeness, easiness and usefulness: technical vs. non-technical (the lowest $\bar{x}$ value is the best).

System	Variable	Technical Users		Non-Technical Users		Wilcoxon signed-rank test
		$\bar{x}$	*SD*	$\bar{x}$	*SD*	
E-Free	*Completeness*	1.89	.93	1.76	1.01	z = –.092, p = .927
	Easiness	1.92	.87	1.54	.97	z = –.784, p = .433
	Usefulness	1.63	.93	1.61	.96	z = –.333, p = .739
E-Wizard	*Completeness*	2.55	.89	2.38	1.12	z = –.628, p = .530
	Easiness	2.44	.89	2.46	.88	z = –.973, p = .331
	Usefulness	2.19	.88	1.92	.95	z = –1.200, p = .230
E-Wired	*Completeness*	2.93	1.24	2.79	1.01	z = –.575, p = .565
	Easiness	3.37	1.04	3.00	.91	z = –.730, p = .465
	Usefulness	3.22	1.02	3.15	.80	z = –.263, p = .793
IFTTT	*Completeness*	2.63	1.81	3.08	1.04	z = –.000, p = 1.000
	Easiness	2.26	1.16	3.00	1.15	z = –1.856, p = .063
	Usefulness	3.00	.83	3.30	.75	z = –1.645, p = .100

Table 18. User ranking along completeness, easiness and usefulness: male vs. female (the lowest $\bar{x}$ value is the best).

System	Variable	Male		Female		Wilcoxon signed-rank test
		$\bar{x}$	*SD*	$\bar{x}$	*SD*	
E-Free	*Completeness*	1.93	.94	1.67	.96	z = –.632, p = .527
	Easiness	1.93	.94	1.50	.79	z = –.997, p = .319
	Usefulness	1.68	.91	1.50	1.00	z = –.319, p = .750
E-Wizard	*Completeness*	2.57	.99	2.33	.88	z = –.586, p = .558
	Easiness	2.50	.92	2.33	.78	z = –1.221, p = .222
	Usefulness	2.18	.98	1.92	.67	z = –1.040, p = .298
E-Wired	*Completeness*	2.89	1.19	2.83	1.12	z = –.812, p = .417
	Easiness	3.32	.945	3.08	1.17	z = –.355, p = .722
	Usefulness	3.11	1.03	2.18	.98	z = –.303, p = .762
IFTTT	*Completeness*	2.61	1.17	3.17	1.93	z = –.359, p =.719
	Easiness	2.25	1.21	3.08	.99	z = –1.606, p = .108
	Usefulness	3.07	.86	3.17	.72	z = –.811, p = .417

6.7 Threats to Validity

We now analyze some issues that may have threatened the validity of the comparative study, also to highlight under which conditions the study design offers benefits that can be exploited in other contexts, and under which circumstances it might fail.

6.7.1 Internal validity

Internal validity can be threatened by some hidden factors compromising the achieved conclusions:

—*Learning effect*. In our experiment, this factor was minimized by counterbalancing the systems and the experimental tasks order across the systems, both according to a Latin Square design.

—*Subject experience*. It was alleviated by the fact that none of the subjects had any experience with the experimented tools, as well as with TA tools in general.

—*Subject-expectancy Effects*. Students are not the best participants for a user study due to the subject-expectancy effect they can produce, i.e., a form of reactivity that occurs when a research subject expects a given result and therefore unconsciously affects the outcome. We mitigated this effect by masking details that can produce bias. In particular, we presented the experiment to the participants in a way that suggests that we had no stake in the outcome. For example, we introduced all the experimental tools as already available web sites that we wanted to observe during the creation of task-automation rules by users; furthermore, in order to foster the credibility of this aspect, we developed our tools with a professional look-and-feel and we deployed them on a remote Web server, so that the participants had to connect to a remote URL to them, similarly to IFTTT.

—*Method authorship*. We eliminated the biases that different facilitators running the experiment could introduce, as we had the same instructor for every session of the study. In this way, we avoided any variability in the initial training as well as in the way users had been observed.

—*Information exchange*. Since the study took place over 10 days, it is difficult to be certain whether the involved subjects did not exchange any information. However, participants were recruited from different classes and during exams period thus, for many of them, it was difficult to know each other and to communicate. The participants were asked to return all the material (e.g., the booklet) at the end of each session. We asked participants coming from the same classes and that typically study and travel together to perform the test in the same session.

—*Understandability of the material*. A pilot study with further 3 participants was carried out to assess the understandability of experiment procedures and materials.

6.7.2 External validity

External validity refers to the possible approximation of truth of conclusions in the attempt to generalize the results of the study in different contexts. With this respect, the main threats of our study are:

—*Users age and domain experience*. Since the study participants were young students not experienced with IoT and TA tools, we have to take into account two potential limitations of the study results. The first one is the participants' age that limits the prediction of the tools benefits to older people. Thus, we can safely accept the experiment results for *digital natives* [Prensky 2001] but further studies have to be carried out including older people. The second potential limitation is related to the participants' domain experience: in fact, they had not experience with IoT technology, as well as with TA tools. We intentionally recruited inexperienced young people because we aimed to experiment the composition paradigm with users that would potentially adopt such tools in the next years, without affecting results on composition paradigms usability evaluation due to prior users' knowledge of other tools and IoT technologies. However, this criterion can limit the generalizability of our results to end users of smart environments that can have different behaviors and needs.

This is the reason why, as described in Section 7, we also performed a validation of EFESTO-Free by involving end users expert in the configuration of home automation systems.

—*Tasks Complexity*. The tasks used for the study took inspiration from the most popular rules created by the IFTTT community. We also extended some tasks in order to accommodate more complex and real user's needs, by including temporal and spatial constraint, as well as logical operators. The possibility to consider tasks that are more complex was limited by the experimented tools. Thus, the results obtained and the design indications proposed are valid for a particular class of tasks, i.e., simple ECA rules that include logical operators and temporal and spatial constraints.

—*Comparison with other tools*. In the comparative experiment, we considered only IFTTT as baseline because the main goal of our comparison was to observe the limits and advantages of the conceived EFESTO paradigms with respect to a tool that is mature and popular among non-technical users. However, as reported in Section 2.2, different TA tools are available other than IFTTT and we have no evidence about the advantages of E-Free with respect to these tools. This is the reason why we are going to perform new controlled experiments including other relevant tools.

6.7.3 Construct validity

Construct validity might have been influenced by the measures that we applied in the quantitative analysis and by the reliability of the questionnaire. We alleviated the first threat by adopting measures, such as efficiency (e.g., time to complete a task), that are commonly employed in user studies [Dix et al. 2003]. The reliability of the questionnaire was tested by applying the Cronbach test to each set of closed questions intended to measure subjective variables. As reported in the previous section, the value obtained (.940) was higher than the acceptable minimum threshold (>0.70) [Maxwell 2002].

6.7.4 Conclusion validity

Conclusion validity refers to the validity of the statistical tests applied. In our study, this was alleviated by applying the most common tests that are employed in the empirical software engineering field [Juristo and Moreno 2010].

7. STUDY WITH HOME-AUTOMATION EXPERTS

After the comparison of the three prototypes, we performed a further study to investigate how to improve task-automation tools in the home automation domain by taking into account the perspective of domain experts. We also evaluated if E-Free and the underlying Rule_5W could represent a valid proposal in this direction. To this aim, we validated the E-Free prototype with 15 experts of both IoT and home automation. E-Free was chosen because in the comparative study it outperformed the other prototypes in terms of participants' performances and preferences.

The new study consisted of two phases. The first one was a utilization study to evaluate user performances and satisfaction with E-Free. This first phase then fostered a discussion that during the second phase was held in a focus group session: utterances, comments and hints were gathered from the participants about usability and functionalities of a TA tool in general, and about the adoption of E-Free in the home-automation domain.

7.1 Participants and Design

We were able to recruit a total of 15 participants (9 female), aged between 25 and 34 ($\bar{x}$ = 28.8, SD = 2.42). More specifically, participants were 14 construction engineers

and 1 biomedical engineer. They were concluding at the "Politecnico di Milano" University an advanced (Post-Master) training course on smart technologies in smart environments. The course, titled "Home Automation and Technology for Living Environments", was organized over 15 months (2.400 hours) and included a phase of coaching and project work in a company of the home-automation sector. The course is part of the SHELL Project - Cluster Smart Living Technologies[8]. Participants were rewarded with a 8GB USB memory stick.

In the first utilization study, each participant was asked to complete two training tasks and three experimental tasks. In order to propose tasks that would be significant, engaging and able to stimulate home-automation end users, we involved a domain expert. First, we asked the expert to use E-Free to perform some tasks (some of them were those of the comparative study). Then, he was required to select the tasks, among those just performed, that could be actually useful in the home automation domain. He was also invited to design further significant domain tasks. At the end, the three following tasks were produced:

1. If my Android Wear changes position to the address of my home between 8-10 p.m., then switch-on the home air conditioning and set 25 °C as temperature;
2. If my smart bracelet detects that I'm waking-up between 7-8 a.m. or my smart alarm clock rings, then open the roll-up shutters and switch-on the coffee maker;
3. If my car changes position to the address of my home and I push the button of my Android Wear, then open the garage door and switch-on the boiler.

The first two tasks were taken from the comparative study because they were very similar to some tasks proposed by the expert; the third one was completely new.

To evaluate user satisfaction, the same questionnaire of the comparative experiment, integrated with the SUS statements [Brooke 1996], was administered after the tasks execution. We introduced the SUS statements because it is highly reliable [Bangor et al. 2008], technology agnostic and effective also for evaluating usability of modern technology [Brooke 2013].

The utilization study was followed up by two focus groups (7 participants in the first group). Participants were stimulated to discuss the following topics:

1) *Which scenarios in home-automation environments can benefit from the use of E-Free*. The goal was to identify specific and real situations so that future research on TA tools applied to home automation can be more focused and driven by realistic scenarios;
2) *Which aspects of user interface and interaction can be improved*. The goal was to identify interface and interaction aspects that can impact on system usability;
3) *Which functionalities should be included/removed to make E-Free more compliant to real contexts*. The goal was to identify functionalities, possibly independent from a specific composition paradigm, that foster the adoption of TA tools in real contexts.

7.2 Procedure

The entire study took place in a quiet and isolated room at the Politecnico di Milano campus, where we installed the study apparatus (a laptop and a web camera) 30 minutes before the start. Two HCI researchers were involved in the study. In particular, during the utilization study, one (facilitator) was in charge of introducing users to the study and following them during the tasks accomplishment; the second

[8] http://shell.smartlivingtech.it/

one (observer) took notes. During the focus groups, instead, one (facilitator) was in charge of stimulating the discussion, the second one (observer) took notes. The entire study lasted 1 day (about 8 hours).

During the utilization study, each participant interacted for about 10 minutes for a total of 4 hours (breaks included). They all followed the same procedure. First, each participant was asked to sign a consent form. Then, the facilitator showed a quick introduction about the use of E-Free. Then, the participant was provided with a list of two training tasks during which they could ask for help, and three main tasks to be performed alone. At the end, participants filled in the online questionnaire.

For the focus group session, in order not to have a too large group, we split participants in two separate groups. Both the HCI researchers participated in the focus groups. Participants sat around a table and were provided with pencil and sheets in order to sketch their ideas. Each focus group lasted about 1 hour and was video recorded.

7.3 Data Collection & Analysis

During the utilization study, we collected quantitative and qualitative data. Quantitative data regarded user performances measured through time and number of clicks to perform tasks. Qualitative data regarded user satisfaction measured through a questionnaire that included all questions already used during the comparative study and the 10 statements of the SUS questionnaire.

During the focus groups, the observer took notes about the discussion; video and audio of the discussion were also recorded. The set of collected notes was extended by video and audio analysis, performed by two researchers that transcribed videos and audios and independently double-checked some 85% of the material. The initial reliability value was 80%, thus the researchers discussed the differences and reached a full agreement. The transcripts were analyzed by thematic analysis following a semantic approach. Themes were identified within the explicit or surface meaning of the data [Braun and Clarke 2006b].

7.4 Results of the utilization study

Even if the actual goal of the utilization study was to introduce participants to the use of a TA tool to foster the discussion in the following focus group, during this phase we also collected data about user performances and satisfaction.

Regarding user performances, the average time participants spent for each task was 115 seconds (SD = 53.25), while the average number of clicks was 25 (SD = 11.58). Because the tasks were similar but not identical to those assigned in the comparative study, it is not possible to derive further information through the comparison of user performance in the two studies.

Regarding qualitative data, the questionnaire results allowed us to measure two particular aspects, i.e., *User satisfaction with system* ($\bar{x}$ = 5.58, SD = 0.72) and *User satisfaction with created rules* ($\bar{x}$ = 566.67, SD = 93.63). Being user satisfaction less dependent by tasks than user performances, a t-test has been used to compare the satisfaction results between the utilization and the comparative study. The test demonstrated that no statistical differences emerged for the *User satisfaction with system* ($t(53) = .8954$, $p = .3746$) and the *User satisfaction with created rules* ($t(53) = 1.7085$, $p = .0934$). These results are encouraging as they highlight that also expert users, who are typically more demanding with respect to the adopted technology and also like to have full control on it, were satisfied in using E-Free for the accomplishments of tasks typical of their domain.

The user satisfaction was also evaluated by means of the 10 SUS questions, which gave us a more general indication about the perceived system usability and learnability. The SUS global score was 73.1/100 (SD = 13.8), which is higher than the

average SUS scores (69.5) of one thousand studies reported in [Bangor et al. 2009]. In addition, according to [Lewis and Sauro 2009], we split the overall SUS score into two factors, i.e., System Learnability (considering statements #4 and #10) and System Usability (all the other statements). The *System Learnability* score was 68.1 (SD = 22.6), while the *System Usability* score was 74.3 (SD = 13.9). According to the SUS adjective rating scales [Bangor et al. 2009], both the scores can be considered a very good result. Besides providing an objective indication about the usability and learnability of the tested system, SUS results can be used as benchmarks in the comparison of further TA tools similar to E-Free.

7.5 Results of the focus groups

Through the discussion on *scenarios* we aimed to identify realistic home-automation tasks that could benefit from the use of our TA tool. Such scenarios would be fundamental for customizing the general methodology and the prototypes to the home-automation domain. During the thematic analysis we identified the following categories of new emerged elements (examples of possible rules suggested by the participants are provided):

- **Security**. "IF my position is far from my house THEN activate the system alarm"; "IF a thief enters the house THEN switch on TV and lights"; "IF the time is between 09 p.m. and 06 p.m. THEN keep close the garage door and all the house doors".
- **Home Assistance**. "IF my smart bracelet detects an anomalous heartbeat THEN send a message to my cardiologist"; "IF a proper device detects that an elderly falls down THEN send a message to his sons".
- **Education for children**. "IF the time is between 07 p.m. and 07 a.m. THEN disable the TV in the children room".
- **Energy Optimization.** This was one of the most intriguing categories. In fact, today the sources of green energy installed at home cannot send back to the energy factory the surplus of produced energy, which is wasted; this because the energy networks are built to have only a main direction, i.e., from the factory to the houses. A system like E-Free can allow a smarter setting of energy production and consumption, for example a rule like "IF the energy production is more than the energy used in the home at a certain moment THEN switch on the washing machine". In addition, this energy optimization can be set at different granularity levels. For example, in a building with different apartments, the administrator can create rules to transfer/sell the surplus energy of some houses to other houses. The same logic could be adopted by the energy factory on a large scale and systems like E-free make easier its management.

The second part of the focus groups was about *usability problems* as we also wanted to improve the usability of our tools taking into account the perspective of real users. No critical problems emerged, but the participants highlighted different aspects that could be improved. For example, users should be helped in the definition of temporal constraints by an auto-complete function (as in Google Maps); it should be possible to specify more details in the definition of temporal constraints, e.g., as in Google Calendar, a greater variety in defining recurring events/actions (all days, certain days, weekly, etc.) should be allowed; icons should be used also for events and actions to make more clear their meaning.

Finally, through the focus group we wanted to identify new functional requirements that, independent of a specific composition paradigm, could facilitate the adoption of TA platforms in the home-automation domain. Interesting *missing*

functionalities were highlighted. In the following, we list those that we consider important for TA tools and also generalizable to further domains – not only home automation:

- **Meta-rules** would allow users to define rules that control rules. For example, a user can have different states that determine different rules activation, e.g. at home, out of home, in travel, at work. Each status can be associated with the activation/deactivation of a set of rules. A meta-rule can be "IF my status is IN TRAVEL (e.g. from a date to another date of my calendar), activate rules X, Y and deactivate rules Z, W".
- **Warning mechanisms** would alert users about possible dangerous rules. For example, if a rule set the opening of the house doors, users should be alerted for the behavior that can cause the involuntary opening of doors.
- **Rule debug** would help users to simulate and foresee rule behavior under different conditions.
- **Rules conflict identification** would support users of a smart environment (e.g. wife, husband and children) to identify the rules that affect the same smart objects and that can create potential conflicts (e.g., two rules created by the wife and the husband to switch on/off the washing machine but with different conflicting conditions).
- **Service recommendations** would be important to guide the rule composition, helping users be aware of useful services that they could not know.
- **Different complexity levels** would accommodate different user skills and attitudes. For example, an advanced modality should be available for skilled users to allow a more powerful rule customization (e.g., more detailed conditions or more expressive logic connections).
- **Multiple object management** would allow users to manage multiple instances of the same objects in each account. E-Free, as well as most TA tools, typically allow managing only one smart object per account.
- **Access management policies** would allow defining restrictions on the object access by the home users. For example, children should not access the configuration of smart cooking, garage doors, and in general those smart objects dangerous for them or for the home security.

Summing up, home-automation experts found E-Free, and the Rule_5W model fully implemented in the prototype, satisfactory for accomplishing their configuration activities. In particular, questionnaire results and focus group discussions confirmed the usefulness of temporal and spatial constraints, which are the characterizing feature introduced by the Rule_5W model and that we wanted to validate with expert users too. Furthermore, experts provided valuable hints for identifying interesting directions for our research.

8. SUMMARY OF RESULTS AND DESIGN IMPLICATIONS

In this section, we summarize the main findings of the studies and discuss some design implications. According to our research questions, the comparative study was conducted to detect possible differences, in terms of user performance and satisfaction, among the four systems that were considered. This study allowed us to identify some characteristics that composition paradigms in TA tools should feature to be adequate with respect to end users who are not expert in programming. It also allowed us to assess the adequateness of the rule elements introduced by the Rule_5W model, even with respect to the skills of non-technical users.

The successive validation study confirmed the usefulness and adequateness of the E-Free composition paradigm and the Rule_5W model also with respect to the skills of expert users. It also allowed us to identify further requirements that might encounter the needs and preferences of users that are expert in a specific domain (i.e., home automation).

Expressive power does not downgrade user performance and satisfaction. In relation to the user performance assessed in the comparative study, E-Free clearly emerged as the most promising composition paradigm. In particular, when using E-Free for creating rules, the users always took significantly less time than when using the other systems. This was true for simple rules (RS1, RS2) as well as for complex rules (RS3, RS4) exploiting the extra-elements introduced by our Rule_5W model. It is exactly rule complexity that helped us identify the lacks of the other composition paradigms. For example, IFTTT does not support the occurrence in a rule of multiple events. IFTTT was indeed conceived for creating very simple rules, not allowing the inclusion of logical operators for concatenating multiple events and actions and of conditions to constraint rule activation. In IFTTT, similar composite behavior can be achieved through the creation of multiple rules, each one focusing on one single event and one single action. However, this modus operandi negatively impacts on the user performances. We are aware that the simplicity of IFTTT rules is a fundamental factor for the success of this platform. However, we believe that more expressive composition paradigms, as the one that can be defined on top of our Rule_5W model, are fundamental to empower end users to create meaningful services.

The comparative study also allowed us to understand that end users can master more complex rules if adequate interaction paradigms are provided. In particular, the study highlighted that E-Free is the system that the users preferred most along all the satisfaction dimensions (results achieved, completeness, ease of use and usefulness). Moreover, the comparison between technical and non-technical users along performance and satisfaction in general highlighted that the interaction with all the four systems is not influenced by the user expertise. The lack of significant differences between the two groups enforces our belief that more expressive paradigms still result adequate for non-technical users, even if the specification of rules is more complex.

Wired paradigms downgrade user performance and satisfaction - even for technical users. The comparative study revealed that wired paradigms, as the one implemented in the E-Wired prototype, downgrade both user performance and satisfaction. This result is in line with some findings already reported in literature. Wired paradigms have been largely used in the field of Web services and smart object composition [Blackstock and Lea 2012; Guinard et al. 2011; JS_Foundation 2016]. Graphs are good candidates to represent the flow of parameters (the edges) that are generated by events to trigger the activation of different services and objects (the nodes). This notation fits very well the mental model of expert programmers, who are used to adopt graphs for representing the semantics of programs. In addition, wired concepts are fruitfully used to support the tailoring of component-based applications at run-time [Wulf et al. 2008]. However, this notation introduces severe problems when non-expert users need to program web-service behaviors [Namoun et al. 2010a]. Additionally, in our study we also found that the performance with E-Wired was not influenced by the user expertise as there are no significant differences between technical and non-technical users. We can say therefore that even technical users, who should be acquainted with wired notations, perform better and are more satisfied with other interaction paradigms.

Rules under creation need adequate representations highlighting the composing elements. Another relevant result regarding the user performance is that the rule representation in IFTTT induces the users to judge as wrong rules that

instead are correct. In other words, users are not in control of the rule definition task. The study reported in [Cabitza et al. 2015] already highlighted the need for clear descriptions, to let users understand the effect of their created rules without being forced to activate them on real objects. As already discussed in Section 6.5.4, to identify possible reasons of the low performance of IFTTT, we compared step by step the E-Wizard and the IFTTT paradigm. We purposely kept E-Wizard similar to IFTTT, as we wanted to analyze the impact of the new operators. In both the systems the rule elements are incrementally visualized as soon as they are selected or defined. The only difference between the two paradigms is a "synthesis" of the created rule, expressed in natural language, that IFTTT displays at the end of the process. We already planned further studies to assess whether this is the element that actually downgrades the expressiveness of rule representation. However, our hypothesis is that *i)* the synthesis forces the users to jump to a rule representation that is different from the one adopted during rule creation, and *ii)* the new representation does not highlight adequately the details of events and actions that the users specify during rule creation.

Not imposing any specific order on the composition steps improves user's performance and satisfaction. The improved performance and the higher satisfaction of E-Free emerged in the comparative study are due to the very difference of this paradigm with respect to the others: the freedom that it leaves to the users, who can define events and actions without being forced to follow a specific order. This finding is coherent with the results of the study discussed in [Lucci and Paternò 2015]: by comparing some TA tools, the study highlighted that the composition paradigm should not impose any temporal constraint regarding what to specify first. In addition to this result, in our study we observed that the absence of constraints on the order of element specification allows the user to explore "freely" the available elements, and this in turn improves their understanding of what elements can be composed and how, and their performance. In other words, even if it can appear as a paradox, the absence of constraints helps users be in control.

It is worth noticing that in the comparative study a slightly higher preference for IFTTT emerged for non-technical users. Observing participants during the study made evident that non-technical users are more acquainted with wizard procedures, which are for example used for installing software or configuring services. Therefore, they did not get disturbed by the extra steps needed to configure a rule with IFTTT, being them aware that wizards sometimes propose steps that are useless in a specific situation and that can be simply skipped without influencing the correctness of the final result.

Offering assistance mechanisms can reduce the occurrence of errors. In order to facilitate rule definition, assistance mechanisms can be used to guide users in discovering elements made available by the platform and help them define sound and correct rules. As emerged from the validation study involving home automation experts, users would appreciate having a set of pre-defined rules, and also *meta-rules* corresponding to typical scenarios of use. In other words, especially when users need to repeatedly define similar rules, they could benefit from the availability of *pre-defined system configurations*, which can then be customized according to the actual situational needs. Also, users would benefit from recommendations helping them discover the services that can be exploited for rule definition. These findings are in line with some choices at the basis of the IFTTT paradigm, which offers pre-configured recipes. As already discussed in the previous sections, these predefined recipes especially determined the IFTTT success. The findings are also in line with the results of previous studies on mashup composition paradigms, which showed that users find helpful any kind of assistance that the system is able to provide during the composition process [Cappiello et al. 2011; Namoun et al. 2010b].

Expert users, probably due to their awareness of the consequences of wrong rule definition, also highlighted the need to be assisted by debugging mechanisms to identify errors and possible rules conflicts. This aspect was also recently discussed in other independent studies (see for example [Fogli et al. 2016b]).

Meta-design can help adopting proper abstractions hiding technical details. When comparing the user performances along the type of services (smart object vs. Web service), we found that the number of parameters to be set up for Web services negatively influences the user performance and satisfaction. This is in line with some findings reported in [Lucci and Paternò 2015], where the authors state that an excessive number of service properties to choose from makes it difficult to identify the right one. With this respect, we believe that a meta-design approach can help pre-configuring the resources to be composed by the end users, so that to avoid unnecessary complexity. Meta-design indeed prescribes involving domain and/or technology experts to customize the system for its initial use by the end users [Ardito et al. 2014b; Fischer et al. 2004]. Therefore, in a smart-object composition scenario, domain experts can exploit their domain knowledge to select those properties of objects and Web services that are really useful for the end users. Expert programmers can then define techniques for accessing services, based on adequate "adapters" and service descriptors that provide the logic to mediate between the whole set of properties natively provided by the resources and the set of properties to be exposed to the end users.

Supporting different complexity levels could accommodate the attitudes and preferences of different classes of users. Another ingredient emerged in the validation study relates to accommodating different users' skills and attitudes and also varying composition contexts. Different "composition styles" can be offered to reflect different users' needs and skills. For example, as discussed above, users should be provided with totally pre-defined rule configurations, or at the other extreme should be enabled to define by themselves, even by writing code, articulated conditions and logic connections between different services. This is in line with previous, well-known findings reported in literature, which say that users should be provided with different abstraction levels [Green and Petre 1996] to ensure a "gentle slope of difficulty" [Lieberman et al. 2006b].

Supporting domain specificity can help accommodate variable user needs. The results of the comparative study indicate that E-Free is the most suitable composition paradigm. However, since notations are adequate if they really meet the characteristics and the background of the end users, we believe that it is important to rely on flexible platform architectures that can be easily adapted to address varying needs. In other words, it is important to foster *domain-specificity*, a quality that is fundamental in EUD platforms [Casati 2011]. In order to allow end users to understand the possibilities offered by the platform and to make sense of the services and objects that are available for composition, it is indeed important to restrict the platform to a well-defined domain, represented through adequate notations the users are comfortable with. As we will illustrate in Section 9, to address this requirement, TA platforms should privilege separation of concerns. The interaction layer, which manages the visual composition of rules by end users, should be kept independent of the other platform components. The adoption of different interaction paradigms is therefore possible to accommodate the needs and the background of specific users' communities.

9. PLATFORM ARCHITECTURE

In this section, we illustrate the organization of the platform that we designed for the definition and execution of event-action rules[9] and that facilitated the generation of the prototypes exploited during the user studies. By presenting the platform architecture we aim to facilitate the replicability of our studies and to show how an adequate organization of TA platforms can enable customizability with respect to varying users and usage domains.

In our research, building prototypes offering different composition paradigms was facilitated by the modularity of our platform, and especially by the decoupling between the interaction layer and the other platform modules. Software design patterns, first of all the MVC (Model-View-Controller), already address this separation of concerns. In our work, however, the emphasis is not on programming practices to facilitate the development and maintenance of an interactive system; rather we want to stress the possibility to adapt easily the composition paradigm offered by the platform, to comply with domain-specific requirements. It is indeed important to restrict the platform to a well-defined domain the user is comfortable with. That is, it is important to develop a general platform that can be, however, easily customized as far as the provided composition metaphor is concerned [Ardito et al. 2014b].

9.1 Platform organization

The platform inherits some modules for service invocation and management already developed in the EFESTO mashup framework [Desolda et al. 2016]. The focus of the new implementation, which we will call *E-5W* in the sequel, is however on the *Rule Engine*. As reported in Figure 11, the E-5W platform is organized in three layers, each one managing a separate aspect.

The *Interaction Layer* is the system client that manages the User Interface (UI) through which the users can create task-automation rules. In addition, it implements two modules, the *Service Builder* and the *Rule Generator*. The first one is in charge of materializing in the UI the list of attributes of registered services, as resulting from the *Service Descriptor* repository. Thus, it is invoked each time users need to add an event or an action to the rule. The UI layer is in principle agnostic to the registered services; to build the visualization of available services, the Service Builder requests to the *Service Engine* the JSON file containing the list of available services, each of them described by attributes like name, events, actions and thumbnail URL.

The second module is the *Rule Generator:* it is an interpreter that translates the user visual actions for rule creation into a JSON specification that describes the rule in terms of events, actions, logical operators and spatial and temporal constraints (see Figure 12). It is worth remarking that the *when* is codified according to the syntax provided by *Quartz Job Scheduler*[10], a job scheduling JAVA library implemented in our platform to manage the rule scheduling.

[9] A demo video, showing the prototype at work for the definition of task automation rules, is available at https://www.dropbox.com/s/nb4v1v6ompe9vey/EFESTO-Free.mp4?dl=0

[10] https://www.quartz-scheduler.org/

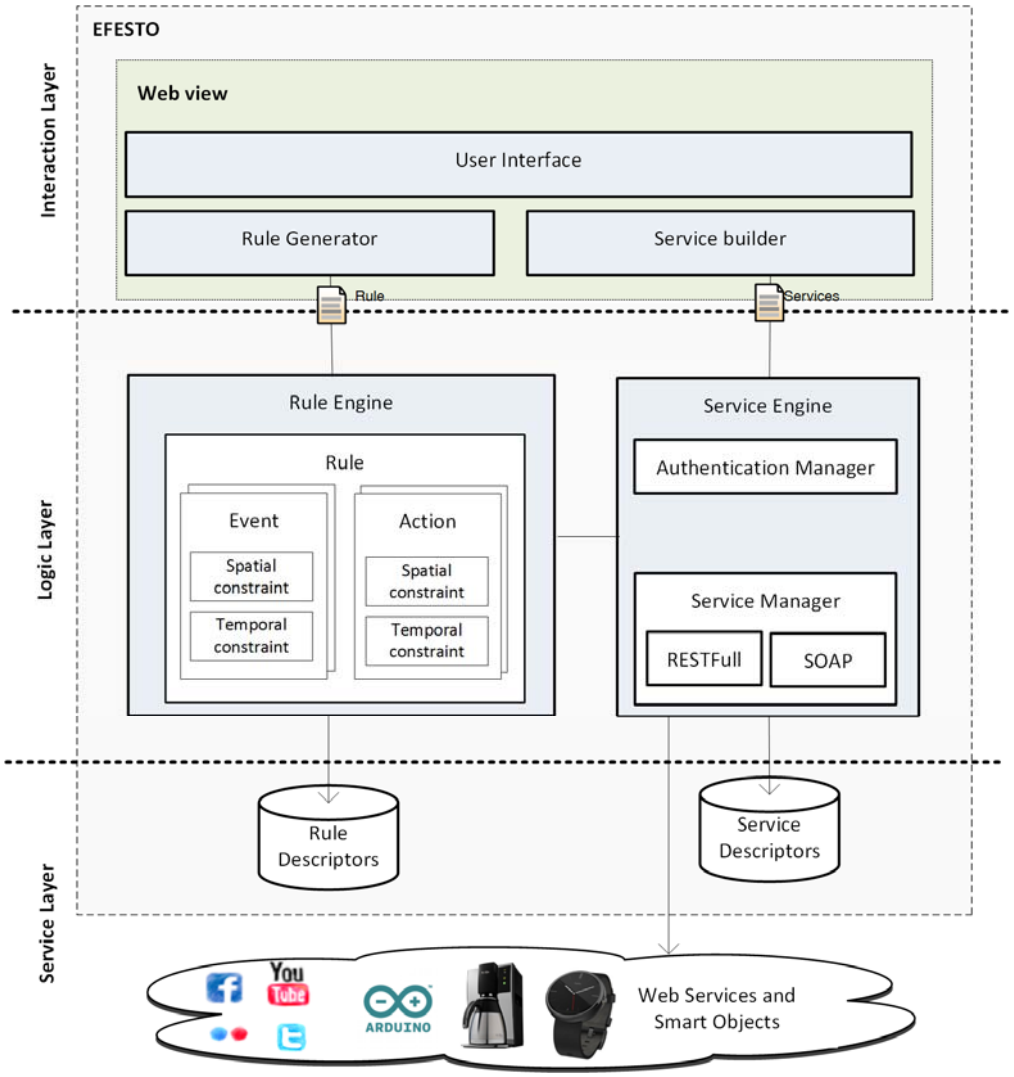
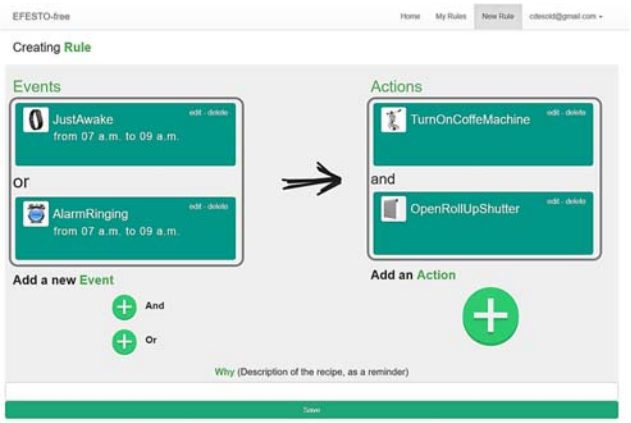


Figure 11. Overall organization of the platform architecture and structure of the rule engine.

```
{
 "userId" : 2,
 "ruleId" : "21.json",
 "why" : "Beautify my wakeup",
 "triggers_logical_opearator": "OR",
 "actions_logical_opearator": "AND"
 "events" : [ {
     "who" : "Bracelet",
     "what" : "JustAwake",
     "when" : "0 0/1 9 ? * ?",
     "where" : "City, Address, number"
   },
   {
     "who" : "SmartAlarm",
     "what" : "Ringing",
     "when" : "0 0/1 9 ? * ?",
     "where" : ""
   }
 ],
 "actions" : [ {
     "who" : "Roll-upShutter",
     "what" : "Open",
     "when" : "",
     "where" : ""
   },{
     "who" : "CoffeMachine",
     "what" : "TurnOn",
     "when" : "",
     "where" : ""
   }
 ]
}
```

a b

Figure 12. a) JSON descriptor of a rule with 2 causes and 2 actions and b) its graphical representation in the E-Free system.

The *Logic Layer* at server side manages rules and services by means of respectively the *Rule Engine* and the *Service Engine* modules. The first one receives the rule JSON file from the client (from the *Rule Generator* module) and instantiates the rule object based on a publish-subscribe event-action model [Cappiello et al. 2015; Cappiello et al. 2011] (see Figure 13). This model is natively managed and handled by a Java Spring class [11]. Each rule object is characterized by a set of *Publisher* services, each of them associated with an event that can be complemented with temporal and spatial constraints, and by a set of *Subscriber* services, each of them associated to an action that can be complemented with temporal and spatial constraints. Moreover, details about the logical operators used among events or actions are stored in the rule object.

The Rule Engine acts as an event bus that mediates the communication between the different components. Components are decoupled: they do not need to be explicitly aware of each other or be blocked waiting for events from other

[11] ThreadPoolTaskScheduler (http://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/scheduling/concurrent/ThreadPoolTaskScheduler.html)

components. The Rule Engine checks every N minutes (3 minutes of *sample rate* in our systems) if the publisher events are triggered (all of them or just one of them depending on the logical operator chosen, respectively AND or OR). This check is performed by a *listener* associated to the rule. If the events are triggered, the Rule Engine controls if there are temporal and spatial constraints on the events and, in case, if they are satisfied. If the events meet all the conditions, the Rule Engine runs all the subscribed actions associated with the rule or schedules the action execution according to the when constraint (see Figure 13).

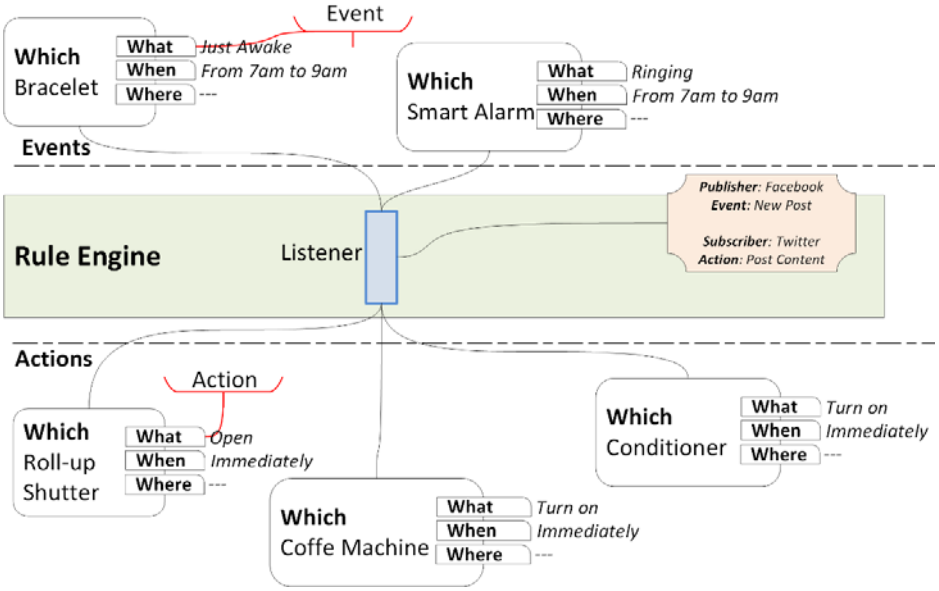


Figure 13. Event-driven paradigm for service coupling definition and rule execution.

It is worth noting that the current platform prototype provides a default sample rate that can be also customized by the users when they configure their profile. We however adopt a specific policy to manage the execution of rules that require checking events in specific time intervals. In this case, the sample rate is not the one defined by default (or by the user), but it is defined automatically taking into account the time interval specified in the rule. In this way, we also optimize the exploitation of resources, since rule checks are performed only in a pertinent time interval, avoiding useless checks out of that interval. Similar strategies are adopted for other sensible cases, e.g., precise-time constraints or multiple events with multiple time intervals. In all these cases, the rule engine optimizes rule checking, restricting it only to pertinent time intervals.

The *Service Layer* is located at the server side and stores service and rule descriptors by using JSON files. A *service descriptor* contains all the information useful to query an API and contributes to decouple the registered services from the rest of the platform. It is created when a new object is added into the platform. For the implementation of the current prototype, we decided to consider only the RESTful technology since it is widely adopted by most of the Web API and smart object providers. However, different technology can be easily accommodated as the EFESTO service layer [Desolda et al. 2016] is structured so that different types of adapters can be plugged in to manage the access to different API technologies. Alternatively, without developing further adapters, it is possible to adopt a dedicated middleware, as for example Azure IoT Suite[12], to mediate the access to additional

[12] https://www.microsoft.com/en/server-cloud/internet-of-things/azure-iot-suite.aspx

service technologies [Li et al. 2015]. The platform is indeed open and each layer can be also implemented by external services.

Figure 14. Service descriptor of the bracelet smart object

An example of service descriptor is provided in Figure 14. It is divided into two main sections: header and body. The header has the attributes *name* and *url* that specify respectively the service name and the API documentation URL. The body section is characterized by a set of attributes (*appID*, *appSecret*, *restUri*, *redirectUri*, *tokenExpiredCode*, *authentication*) that the Service Engine uses to establish the connection with the API. Moreover, the *functions* JSON array contains a list of *events* and *actions*, each of them characterized by the attributes *type*, *name*, *path*, *method* and *response*, which are respectively the type of function (event or action), the event/action name displayed to the users in the UI, the event/action path chained to the *restUri* URL to invoke the event/action, the type of API call (e.g. GET, POST) and the provider response format (e.g. JSON, XML).

9.2 Layer decoupling for domain-specificity

The three layers illustrated above are strongly separated from each other, thus each element of the visual paradigm, the policy for rule management and the service technology as well can change without impacting on the others. In particular, the separation of the UI layer from the other two layers allowed us to develop the three different E-5W composition paradigms by acting exclusively at the Interaction level. All the systems, however, exploit the same Logic and Service Layers. This separation

of concerns also enables the definition of multiple front-ends addressing different execution platforms, i.e., different devices. The interaction layer, indeed, mainly acts as an interpreter of models that specify the rules. As already discussed in [Ardito et al. 2014b; Cappiello et al. 2015], such logic can be replicated also in form of apps running on mobile devices. This feature is in line with the findings reported in [Cabitza et al. 2015], which highlight the need of multi-platform (i.e., Web-based, Android, iOS) tools supporting EUD for IoT.

10. CONCLUSIONS

In this article, we presented our perspective on the EUD for the Internet of Things, by showing how the definition of rich rules for smart object composition can be mastered by end users. This perspective mainly derived from user studies in which we analyzed the performance and satisfaction of a sample of users interacting with three prototypes of task-automation systems that we developed, and with IFTTT, one of the most popular platforms freely available online. Our prototypes are grounded on a new model for rule specification that includes a rich set of operators for coupling multiple events and including temporal and spatial constraints on rule activation. In addition, such prototypes take advantage of the experience that we gained in the last years in the development of EUD platforms for the mashup of heterogeneous Web resources. Smart object composition, indeed, has several commonalities with Web API mashups, being smart objects very often controlled and configured through remote Web services.

The findings of the performed studies highlighted some features of composition paradigms that have to suit the expertise of end users. We also gained useful insights on how to organize a supporting composition platform so that to privilege the adaptability of the adopted interaction paradigm to accommodate domain-specific requirements. Although the study allowed us to identify a candidate paradigm among those analyzed, we are aware that this paradigm cannot be considered the only possible solution, and that it would require adaptations when adopted in specific usage domains. With this respect, our current work is devoted to analyze the advantages and the limits of the best-evaluated composition paradigm in Home Automation and Ambient Assisted Living (AAL) scenarios. As reported in this paper, we already conducted some preliminary studies with home-automation experts. In particular, inspired by the results of the focus group results performed with home-automation experts, we are planning longitudinal and field studies with other users' categories, for example non-frail elderly who need to be supported in an independent lifestyle and health preservation.

Of course, we are aware that many aspects need to be considered to achieve full-fledged platforms for the EUD of IoT systems. Besides introducing more sophisticated strategies for rule execution (e.g., managing rule execution at "relative time" with respect to the occurrence of past events), we want more in general to focus on the interoperability of connected devices. This is a problem that now limits the diffusion of IoT systems, and that is the object of many research projects. We are now investigating the adoption of ontologies to define a common ground for the communication among heterogeneous devices. We are also investigating different approaches for the detection and processing of events from sensor data, an aspect that is fundamental for the robustness of the platform. With this respect, we are considering pros and cons of Complex Event Processing engines versus the adoption of specific data stream query languages.

As far as the composition paradigm is concerned, we plan to further develop the E-Free prototype by also taking into account the suggestions coming from the conducted user studies, especially the utilization study involving home-automation experts. We also want to improve the user control on rule execution. We will

therefore focus on the visualization of the data generated and consumed by the different objects. The aim is to let the users understand, thus control, the system behavior (i.e., "what happens if I define a given rule"), to in turn improve the overall comprehension of the rule definition process.

To extend the capability of the platform to support EUD, we are also planning to introduce collaboration features, to let different stakeholder to cooperate for the definition of a smart space. These features are very interesting in domains that require the involvement of different stakeholders, such as the AAL for non-frail elderly we are currently focusing on. In our past research we already defined a collaborative composition paradigm and some architectural extensions for the EFESTO platform [Ardito et al. 2014a; Matera et al. 2013]. We will revise the achieved results to define customizations that are sensible to the specific domain.

Our future work will also consider the addition and the initial configuration of new objects into smart environments by non-technical users. This aspect has not been addressed in this article. Actually, our current prototype requires the intervention of expert programmers to define JSON-based object descriptors. We would like to understand whether EUD practices would (at least partially) enable non-technical users to perform this activity. This implies the identification of a "component model", i.e., a set of conceptual elements abstracting the underlying technology, which can mediate between the technical features to be addressed to program smart objects (the components) and the interaction layer supporting the customization by end users of objects by means of high-level programming constructs.

Finally, it would be very interesting exploring alternative paradigms to allow users to configure smart object by means of their physicality, thus designing mechanisms based on object proximity as well as on hand and body gestures.

ACKNOWLEDGMENTS

This research is partially funded by the Italian project SHELL (Shared Interoperable Home Ecosystems for a Green, ComfortabLe and Safe Living - CTN01 00128 111357). We are grateful to Prof. Maria Francesca Costabile for her valuable and constant support. We also thank Prof. Rosa Lanzilotti for providing useful suggestions about the experimental study. Finally, we deeply thank the students of the SHELL post-master program for the hints they were able to provide for improving the composition paradigm.

REFERENCES

Apiant, Inc. 2016. We Wired Web. Retrieved from https://wewiredweb.com/

Ardito, C., Bottoni, P., Costabile, M.F., Desolda, G., Matera, M. and Picozzi, M. 2014a. Creation and Use of Service-based Distributed Interactive Workspaces. *Journal of Visual Languages & Computing 25*, 717-726.

Ardito, C., Buono, P., Costabile, M.F., Lanzilotti, R. and Piccinno, A. 2012a. End users as co-designers of their own tools and products. *Journal of Visual Languages & Computing 23*, 78-90.

Ardito, C., Costabile, M.F., Desolda, G., Lanzilotti, R., Matera, M., Piccinno, A. and Picozzi, M. 2014b. User-Driven Visual Composition of Service-Based Interactive Spaces. *Journal of Visual Languages & Computing 25*, 278-296.

Ardito, C., Costabile, M.F., Desolda, G., Lanzilotti, R., Matera, M. and Picozzi, M. 2014c. Visual Composition of Data Sources by End Users. In *Proceedings of the Advanced Visual Interfaces (AVI '14)*. ACM, New York, NY, USA, 257-260.

Ardito, C., Costabile, M.F., Desolda, G., Matera, M., Piccinno, A. and Picozzi, M. 2012b. Composition of situational interactive spaces by end users: a case for cultural heritage. In *Proceedings of the Nordic Conference on Human-Computer Interaction: Making Sense Through Design (NordiCHI '12)*. ACM, New York, NY, USA, 79-88.

Atooma. 2016. Atooma mobile App. Retrieved from https://www.atooma.com/

Atzori, L., Iera, A. and Morabito, G. 2010. The Internet of Things: A survey. *The International Journal of Computer and Computer Networks 54*, 2787-2805.

Bangor, A., Kortum, P. and Miller, J. 2008. The system usability scale (SUS): An empirical evaluation. *International Journal of Human-Computer Interaction 24*, 574-594.

Bangor, A., Kortum, P. and Miller, J. 2009. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of Usability Studies 4*, 114-123.

Barricelli, B.R. and Valtolina, S. 2015. Designing for End-User Development in the Internet of Things. In *International Symposium on End-User Development, IS-EUD 2015*, P. Díaz, V. Pipek, C. Ardito, C.

Jensen, I. Aedo and A. Boden (Eds.) Lecture Notes in Computer Science, Vol. 9083 Springer International Publishing, Cham, 9-24.

Beckmann, C. and Dey, A. 2003. Siteview: Tangibly programming active environments with predictive visualization. In *adjunct Proceedings of UbiComp 2003*, 167-168.

Bellucci, A., Aedo, I. and Díaz, P. 2014a. ECCE toolkit: prototyping UbiComp device ecologies. In *Proceedings of the International Conference on Advanced Visual Interfaces (AVI '14)*. ACM, New York, NY, USA, 339-340.

Bellucci, A., Díaz, P., Aedo, I. and Malizia, A. 2014b. Prototyping device ecologies: physical to digital and viceversa. In *Proceedings of the International Conference on Tangible, Embedded and Embodied Interaction (TEI '14)*. ACM, New York, NY, USA, 373-376.

Blackstock, M. and Lea, R. 2012. IoT mashups with the WoTKit. In *Proceedings of the Conference on the Internet of Things (IOT '12)*. IEEE Xplore, 159-166.

Blackwell, A.F., Rode, J.A. and Toye, E.F. 2009. How do we program the home? Gender, attention investment, and the psychology of programming at home. *Int. J. Hum.-Comput. Stud. 67*, 324-341.

Braun, V. and Clarke, V. 2006a. Using thematic analysis in psychology. *Qualitative Research in Psychology 3*, 77-101.

Braun, V. and Clarke, V. 2006b. Using Thematic Analysis in Psychology Qualitative Research in Psychology, 3, 77-101. *Bristol: University of the West of England*.

Brooke, J. 1996. SUS-A quick and dirty usability scale. *Usability evaluation in industry 189*, 4-7.

Brooke, J. 2013. SUS: a retrospective. *Journal of Usability Studies 8*, 29-40.

Burnett, M. and Kulesza, T. 2015. End-User Development in Internet of Things: We the People. In *Proceedings of Workshop on End User Development in the Internet of Things Era (EUDITE '15) - CHI '15*

Cabitza, F., Fogli, D., Lanzilotti, R. and Piccinno, A. 2015. End-user development in ambient intelligence: a user study. In *Proceedings of the 11th Biannual Conference on Italian SIGCHI Chapter (CHItaly '15)* ACM, Rome, Italy, 146-153.

Cappiello, C., Matera, M. and Picozzi, M. 2015. A UI-Centric Approach for the End-User Development of Multidevice Mashups. *ACM Transaction Web 9*, 1-40.

Cappiello, C., Matera, M., Picozzi, M., Sprega, G., Barbagallo, D. and Francalanci, C. 2011. DashMash: A Mashup Environment for End User Development. In *Web Engineering - ICWE 2011*, S. Auer, O. Díaz and G. Papadopoulos (Eds.) Lecture Notes in Computer Science, Vol. 6757 Springer Berlin Heidelberg, 152-166.

Casati, F. 2011. How End-User Development Will Save Composition Technologies from Their Continuing Failures. In *International Symposium on End-User Development - Is-EUD 2011*, M.F. Costabile, Y. Dittrich, G. Fischer and A. Piccinno (Eds.) Lecture Notes in Computer Science, Vol. 6654 Springer Berlin Heidelberg, 4-6.

Casati, F., Castano, S. and Fugini, M. 2001. Managing Workflow Authorization Constraints through Active Database Technology. *Information Systems Frontiers 3*, 319-338.

Ceri, S., Daniel, F., Matera, M. and Facca, F.M. 2007. Model-driven development of context-aware Web applications. *ACM Trans. Internet Technol. 7*, 2.

Coronado, M. and Iglesias, C.A. 2016. Task Automation Services: Automation for the Masses. *IEEE Internet Computing 20*, 52-58.

Costabile, M.F., Fogli, D., Mussio, P. and Piccinno, A. 2007. Visual Interactive Systems for End-User Development: A Model-Based Design Methodology. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 37*, 1029-1046.

Coutaz, J. and Crowley, J.L. 2015. Learning about End-User Development for Smart Homes by "Eating Our Own Dog Food". In *Proceedings of Workshop on End User Development in the Internet of Things Era (EUDITE '15) - CHI '15*.

Daniel, F. and Matera, M. 2014. *Mashups - Concepts, Models and Architectures*. Springer.

Daniel, F., Matera, M. and Pozzi, G. 2008. Managing runtime adaptivity through active rules: the Bellerofonte framework. *J. Web Eng. 7*, 179-199.

Daniel, F., Matera, M. and Weiss, M. 2011. Next in mashup development: User-created apps on the web. *IT Professional Magazine 13*, 22.

Daniel, F., Yu, J., Benatallah, B., Casati, F., Matera, M. and Saint-Paul, R. 2007. Understanding UI Integration: A Survey of Problems, Technologies, and Opportunities. *IEEE Internet Computing 11*, 59-66.

De Angeli, A., Matera, M., Costabile, M.F., Garzotto, F. and Paolini, P. 2003. On the Advantages of a Systematic Inspection for Evaluating Hypermedia Usability. *International Journal of Human–Computer Interaction 15*, 315-335.

Desolda, G. 2015. Enhancing Workspace Composition by Exploiting Linked Open Data as a Polymorphic Data Source. In *Intelligent Interactive Multimedia Systems and Services (KES-IIMSS '15)*, E. Damiani, J.R. Howlett, C.L. Jain, L. Gallo and G. De Pietro (Eds.) series Smart Innovation, Systems and Technologies, Vol. 40 Springer International Publishing, Cham, 97-108.

Desolda, G., Ardito, C. and Matera, M. 2016. EFESTO: A Platform for the End-User Development of Interactive Workspaces for Data Exploration. In *Rapid Mashup Development Tools - ICWE '15*, F. Daniel and C. Pautasso (Eds.) series Communications in Computer and Information Science, Vol. 591 Springer International Publishing, 63-81.

Dix, A., Finlay, J.E., Abowd, G.D. and Beale, R. 2003. *Human-Computer Interaction (3rd Edition)*. Prentice-Hall, Inc.

Eisenhauer, G., Wolf, M., Abbasi, H. and Schwan, K. 2009. Event-based systems: opportunities and challenges at exascale. In *Proceedings of the ACM International Conference on Distributed Event-Based Systems (DEBS '09)*. ACM, New York, NY, USA, 1-10.

Crafty Apps EU. 2016. Tasker. Retrieved from http://tasker.dinglisch.net/index.html

Fischer, G. 2009. End-User Development and Meta-design: Foundations for Cultures of Participation. In *International Symposium on End-User Development - Is-EUD 2009*, V. Pipek, M.B. Rosson, B. De Ruyter and V. Wulf (Eds.) Lecture Notes in Computer Science, Vol. 5435 Springer Berlin Heidelberg, Berlin, Heidelberg, 3-14.

Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A. and Mehandjiev, N. 2004. Meta-design: a manifesto for end-user development. *Communications of the ACM 47*, 33-37.

Fogli, D., Lanzilotti, R. and Piccinno, A. 2016a. End-User Development Tools for the Smart Home: A Systematic Literature Review. In *Distributed, Ambient and Pervasive Interactions, in DAPI 2016*, N. Streitz and P. Markopoulos (Eds.) Lecture Notes in Computer Science, Vol. 9749 Springer International Publishing, Cham, 69-79.

Fogli, D., Lanzilotti, R., Piccinno, A. and Tosi, P. 2016b. AmI@Home: A Game-Based Collaborative System for Smart Home Configuration. In *Prooceedings of the International Working Conference on Advanced Visual Interfaces (AVI '16)* ACM, Bari, Italy, 308-309.

Ghiani, G., Manca, M. and Paternò, F. 2015. Authoring context-dependent cross-device user interfaces based on trigger/action rules. In *Proceedings of the International Conference on Mobile and Ubiquitous Multimedia (MUM '15)*. ACM, New York, NY, USA, 313-322.

elastic.io GMBH. 2016. elastic.io. Retrieved from http://www.elastic.io/

Green, T.R.G. and Petre, M. 1996. Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework. *Journal of Visual Languages & Computing 7*, 131-174.

LAB at Rockwell Group. 2016. Spacebrew. Retrieved from http://docs.spacebrew.cc/

Guinard, D., Trifa, V., Mattern, F. and Wilde, E. 2011. From the Internet of Things to the Web of Things: Resource-oriented Architecture and Best Practices. In *Architecting the Internet of Things*, D. Uckelmann, M. Harrison and F. Michahelles (Eds.) Springer Berlin Heidelberg, 97-129.

IBM. 2016. WebSphereJRules. Retrieved from https://www-01.ibm.com/software/integration/business-rule-management/jrules-family/

IFTTT. 2016. IFTTT. Retrieved from https://ifttt.com/

WigWag Inc. 2016. WigWag Smart Home. Retrieved from http://www.wigwag.com/

Zapier Inc. 2016. Zapier. Retrieved from https://zapier.com/

Jennifer, W. and Stefano, C. 1996. *Active Database Systems: Triggers and Rules for Advanced Database Processing*. Morgan Kaufmann Publishers Inc.

JS_Foundation. 2016. Node-RED. Retrieved from http://nodered.org/

Juristo, N. and Moreno, A.M. 2010. *Basics of Software Engineering Experimentation*. Springer Publishing Company, Incorporated.

Kubitza, T. and Schmidt, A. 2015. Towards a Toolkit for the Rapid Creation of Smart Environments. In *International Symposium on End-User Development - IS-EUD 2015*, P. Díaz, V. Pipek, C. Ardito, C. Jensen, I. Aedo and A. Boden (Eds.) Lecture Notes in Computer Science, Vol. 9083 Springer International Publishing, Cham, 230-235.

Lewis, J. and Sauro, J. 2009. The Factor Structure of the System Usability Scale. In *Human Centered Design - HCD 2009*, M. Kurosu Ed. Lecture Notes in Computer Science, Vol. 5619 Springer Berlin Heidelberg, 94-103.

Li, S., Xu, L. and Zhao, S. 2015. The internet of things: a survey. *Information Systems Frontiers 17*, 243-259.

Lieberman, H., Paternò, F., Klann, M. and Wulf, V. 2006a. End-User Development: An Emerging Paradigm. In *End User Development*, H. Lieberman, F. Paternò and V. Wulf (Eds.) Human-Computer Interaction Series, Vol. 9 Springer Netherlands, 1-8.

Lieberman, H., Paternò, F. and Wulf, V. 2006b. End User Development Springer.

Itrios LLC. 2016. itDuzzit. Retrieved from http://cloud.itduzzit.com/

SmarterApps Ltd. 2016. AutomateIt - Smart Automation. Retrieved from http://automateitapp.com/

Lucci, G. and Paternò, F. 2015. Analysing How Users Prefer to Model Contextual Event-Action Behaviours in Their Smartphones. In *International Symposium on End-User Development - IS-EUD 2015*, P. Díaz, V. Pipek, C. Ardito, C. Jensen, I. Aedo and A. Boden (Eds.) Lecture Notes in Computer Science, Vol. 9083 Springer International Publishing, Cham, 186-191.

Marquardt, N., Hinckley, K. and Greenberg, S. 2012. Cross-device interaction via micro-mobility and f-formations. In *Proceedings of the ACM symposium on User interface software and technology (UIST '12)*. ACM, New York, NY, USA, 13-22.

Matera, M., Picozzi, M., Pini, M. and Tonazzo, M. 2013. PEUDOM: A mashup platform for the end user development of common information spaces. In *Web Engineering - ICWE 2013* Lecture Notes in Computer Science, Vol. 7977 Springer, 494-497.

Maxwell, K. 2002. *Applied Statistics for Software Managers*. Prentice Hall.

Morris, M.R., Danielescu, A., Drucker, S., Fisher, D., Lee, B., Schraefel, M.C. and Wobbrock, J.O. 2014. Reducing legacy bias in gesture elicitation studies. *interactions 21*, 40-45.

Namoun, A., Nestler, T. and Angeli, A. 2010a. Conceptual and Usability Issues in the Composable Web of Software Services. In *International Conference on Web Engineering - ICWE 2010 Workshops - Revised Selected Papers*, F. Daniel and F.M. Facca (Eds.) Lecture Notes in Computer Science, Vol. 6385 Springer, Berlin Heidelberg, 396-407.

Namoun, A., Nestler, T. and De Angeli, A. 2010b. Service Composition for Non-programmers: Prospects, Problems, and Design Recommendations. In *Proceedings of the IEEE European Conference on Web Services (ECOWS '10)*. IEEE Computer Society, Washington, DC, USA, 123-130.

OpenRules, Inc. 2016. OpenRules Business Rules - Time to Excel Retrieved from http://openrules.com/ruleengine.htm

Pane, J.F., Ratanamahatana, C.A. and Myers, B.A. 2001. Studying the language and structure in non-programmers' solutions to programming problems. *International Journal of Human-Computer Studies 54*, 237-264.

Prensky, M. 2001. Digital natives, digital immigrants part 1. *On the horizon 9*, 1-6.

Red Hat, Inc. 2016. Drools. Retrieved from http://www.drools.org/

Rode, J.A., Toye, E.F. and Blackwell, A.F. 2004. The fuzzy felt ethnography—understanding the programming patterns of domestic appliances. *Personal Ubiquitous Comput. 8*, 161-176.

Rogers, Y., Sharp, H. and Preece, J. 2015. *Interaction Design: Beyond Human - Computer Interaction*. Wiley.

Rosson, M.B. and Carroll, J.M. 2003. Scenario-based design. In *The human-computer interaction handbook*, A.J. Julie and S. Andrew (Eds.) L. Erlbaum Associates Inc., 1032-1050.

Tetteroo, D., Markopoulos, P., Valtolina, S., Paternò, F., Pipek, V. and Burnett, M. 2015. End-User Development in the Internet of Things Era. In *Proceedings of the CHI '15 Extended Abstracts on Human Factors in Computing Systems (EUDITE '15)*. ACM, New York, NY, USA, 2405-2408.

Tetteroo, D., Soute, I. and Markopoulos, P. 2013. Five key challenges in end-user development for tangible and embodied interaction. In *Proceedings of the ACM International conference on multimodal interaction (ICMI '13)*. ACM, New York, NY, USA, 247-254.

Ur, B., Mcmanus, E., Ho, M.P.Y. and Littman, M.L. 2014. Practical trigger-action programming in the smart home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 803-812.

Voida, S., Podlaseck, M., Kjeldsen, R. and Pinhanez, C. 2005. A study on the manipulation of 2D objects in a projector/camera-based augmented reality environment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05)*. ACM, New York, NY, USA, 611-620.

Wajid, U., Namoun, A. and Mehandjiev, N. 2011. Alternative Representations for End User Composition of Service-Based Systems. In *End-User Development - Is-EUD 2011*, M.F. Costabile, Y. Dittrich, G. Fischer and A. Piccinno (Eds.) Lecture Notes in Computer Science, Vol. 6654 Springer Berlin Heidelberg, 53-66.

wot.io. 2016. Bip.io. Retrieved from https://bip.io/

Wulf, V., Pipek, V. and Won, M. 2008. Component-based tailorability: Enabling highly flexible software applications. *International Journal of Human-Computer Studies 66*, 1-22.

Zancanaro, M., Not, E., Petrelli, D., Marshall, M., Van Dijk, T., Risseeuw, M., Van Dijk, D., Venturini, A., Cavada, D. and Kubitza, T. 2015. Recipes for tangible and embodied visit experiences. In *Proceedings of the Museums and the Web conference (MW '15)*.

Zang, N. and Rosson, M.B. 2008. What's in a mashup? And why? Studying the perceptions of web-active end users. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VLHCC '08)*. IEEE Computer Society, Washington, DC, USA, 31-38.

Zipato. 2016. Zipato. Retrieved from https://www.zipato.com/