Journal of Visual Languages and Computing (xxxx) xxxx–xxxx

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

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

A visual language and interactive system for end-user development of internet of things ecosystems

Barbara Rita Barricelli*, Stefano Valtolina

Department of Computer Science, Università degli Studi di Milano, Via Comelico 39/41, 20135 Milan, Italy

ARTICLE INFO

Keywords:
Internet of things
Event detection
Unwitting developers
Sociotechnical design
End-user development
eWellness
Visual language
Visual interactive system
Rule Editor

ABSTRACT

This paper presents the definition of a visual language and its implementation with the design of a visual interactive system for the collaborative management of Internet of Things (IoT) sensors (e.g., wearable fitness trackers, ambient sensors, fitness apps, nutrition apps, sleep trackers) for improving people's quality of life and promoting wellness awareness. The system, called SmartFit Rule Editor, is designed to be used by coaches and trainers of non-professional teams of athletes for monitoring and analyze fitness and wellness data streams and to support them in detecting relevant events and specifying rules for actions taking. Our research is framed under the scope of computer semiotics and semiotic engineering theories. This allows us to study how to support coaches and trainers as a community of domain experts – but not IT and IoT experts – to use elements of a visual language to indirectly manage physical devices and their data streams without the need to know technical specification of the devices, the apps, and the data. We apply a socio-technical approach to design being able to study the social and the technological aspects of the use of the Internet of Things ecosystem, considering them as closely interconnected and dependent. Such an approach underpins user-centered design and development methodologies in order to design the most suitable User eXperience according to users' culture, needs, context of use, and activity.

1. Introduction

Community of domain experts but not IT and Internet of Things (IoT) experts need an effective, easy-to-use and easy-to-learn strategy for managing physical devices and their data streams, without the need to know technical specification of the devices, the apps, and the data. The relationship between domain experts and IT experts has often been fraught with misunderstandings and even frustration. Part of this disconnection between the Communities of Practice [1] is a natural disparity between their work cycles. For a domain expert, the IT system development cycle appears to be a long, drawn-out process. Moreover, once the requirements are explained, it is not easy for domain experts to follow them through the implementation process once they are translated into technical forms. In the developers' defense, what they receive as requirements is often not sufficiently detailed and precise and many requirements are “refined” in the development process. More importantly, once the translation is accomplished, the process for changing a requirement requires going through the same complex consultation and translation process for the change. In several cases, the owners of the requirements (i.e. the domain experts) lose control over their expression and evolution when it is handed off to be embedded in a software solution, and never regain control. To this aim, business rule management systems (BRMSs) [2] enable a great leap forward in bridging the gap between domain and IT experts. Using BRMSs, the domain experts define their policies and rules and provide the clear communication between them and the developers implementing the application system solution. Thus, domain experts have control on exactly how their rules are being executed, and, perhaps more importantly, how the system is designed to facilitate change when business rules change. In traditional information systems, the rules and policies get hard-coded into the application. When changes need to be made, the entire software development life cycle must restart and all its phases need to be repeated: requirements analysis, system design, making sure it does not adversely affect anything else, implementing the change, testing it, and deploying it. With a BRMS, the business logic (again, the part most likely to require change) is separated out and can be changed without impacting the remainder of the application. An assessment of the impact of the change on these rules must still be done, but this is much simpler than a full system impact assessment and it can be done in a more understandable way by domain experts themselves. For this reason, rules should first be specified at the conceptual level, using concepts and languages easily understood by

* Corresponding author.
E-mail address: barricelli@di.unimi.it (B.R. Barricelli).

http://dx.doi.org/10.1016/j.jvlc.2017.01.004
Received 11 February 2016; Received in revised form 26 July 2016; Accepted 8 January 2017

the domain experts who are best qualified to validate the rules. This is particularly true for the quantified-self movement, the result of technology advances in the field of lifelogging, where thanks to the sensors used as small and wearable devices that are affordable and easily interconnectible, it has become a wide spreading phenomenon that sees people to keep track, by using rules, of their habits, health conditions, physiological data, and behavior, and to monitor conditions and quality of the environments in which they work and live. Typical BRMSs such as Drools,[1] OpenRules[2] or IBM Operational Decision Manager[3] offer different solutions for editing, managing and executing rules and in some cases they also provide functions for modeling, in a graphical way, the execution data-flows by applying a set of rules. Moreover, as described in Section 2, the literature reports several solutions for modeling business rules by using UML-Based Rule Modeling Language [3] or standard ontology modeling notation such as RDF or OWL, or other strategies based on Extended Tabular Trees or Decision Tables approach. Anyway, all these approaches require significant knowledge of standard modeling notation, and the programming is still the central metaphor. However, what is necessary to support nontechnical people in keeping track of their daily activities is a new manner for expressing rules that can meet their expectations and decision making attitude. This requirement pushes towards the design of new manners for visually specifying rules at conceptual level by adopting a graphical interaction strategy for expressing rules and related constraints and actions in an easier and more natural way. The emphasis of the interaction needs to be the communication process that takes place among users and the system in order to meet the users' expectations as thus defined in their mental model and tacit knowledge. Our approach, specifically implemented in the context of eWellness domain, aims at helping domain experts, in our case coaches and trainers, in expressing rules based on their working settings by using a set of conditions that if met, have to trigger suitable actions, therefore allowing the establishment of a correct interpretation and semiosis process among users and the system. The IoT panorama that characterizes current eWellness domain faces the problem to monitor a huge quantity of data collected by sensors and services that need to be exchanged together with their users' needs and/or preferences, in order to keep track and influence behaviors and critical situations. In this context one of the main problems is the need to express conditions and spatial-temporal and thematic relations that typically affect the sensors' data-stream management. In general, sensors besides spatial and temporal information provide thematic information in order to discover and analyze data. Thematic domain contains metadata that describe a real-world state from sensor observations, such as a sensor that is used for gather data about calories burned, heartrate, or duration of a physical activity.

Section 2 describes current solutions based on the use of decision tables, decision trees or mashup techniques that are still highly complex and negatively affect the effectiveness of the communication process between users and the system. On the other hand, a visual approach to temporal conditions definition could help the user in successfully expressing them without having to learn specific languages and their syntax. Defining suitable visual representations for temporal structures for rules needs the integration of theoretical and methodological work both from traditional areas devoted to temporal representation (logic, reasoning, and databases) and from the information visualization research field.

Stemming from these considerations, this paper presents the Rule Language we defined and its implementation in a visual interactive system that allows domain experts to unwittingly develop an IoT ecosystem [4–7]. We use the term unwitting developers to identify those people who are motivated in using specific software environments to reach their objectives and thanks to their motivation they are more keen to overcome any eventual difficulty. As reported in [4,5], programming is not these people's goal, instead constructing and deconstructing software objects are. The architecture of the SmartFit Framework is designed on Software Shaping Workshop (SSW) methodology [8], one of the most established methodologies for End-User Development (EUD) [9].

Specifically, this work results in the design of a Rule Editor as part of the SmartFit Framework whose aim is to gather, compute, and diffuse data originated and streamed by physical and social IoT devices, sensors, and applications. The graphical Rule Editor uses a notation able to facilitate team sport members in taking under control their athletes' physical activities and their nutrition and sleep behaviors.

The essential idea of using a semiotics-driven design is to define a new perspective in designing and developing interactive systems to support collaboration in multidisciplinary projects, especially adopting participatory design techniques. The key concept is to involve domain experts in activities for mapping and translating their professional knowledge into proper vocabularies, notations, and suitable visual structures of navigation among interactive systems interface elements. Such an approach to design is aimed at supporting communication among different communities of experts (e.g., IT, domain, HCI, and interaction design) and enabling their collaboration in designing together effective interactive systems for specific application domains.

The paper is organized as follows. Section 2 describes how EUD methods and techniques can be used to empower the end user in the IoT domain and illustrates the state-of-the-art solutions provided by Workflow Management Systems (WfMSs) and BRMSs in supporting nontechnical people in keeping track of their activities and behaviors. In Section 3, the methodology at the base of the design of the SmartFit Framework is illustrated and discussed in detail. Section 4 presents the architecture of the Framework and its systems. The Rule Language is presented in Section 5, while its implementation in the SmartFit Rule Editor is described in Section 6. The evaluation process and its results are reported in Section 7, while Section 8 concludes the paper offering an overview of future works and developments.

2. Related work

One of the key objectives in EUD is to provide people with the capability to create and modify software. In particular, the goal of EUD is to extend that capability to a wider range of people, beyond professional programmers, in a way that will help these people in achieving successful results in their daily activities. EUD represents the ideal approach for empowering end users and make them becoming unwitting developers in their own IoT environment [4–7]. As widely reported in the literature, EUD can be enabled by applying methods and techniques and by offering specific tools that support end users in the development of solutions with limited programming skills and knowledge about programming languages. Specifically, the solutions offered by EUD include tools for the customization of applications by parameters setting, control of a complex device (like a home-based heating system), and even scripting of interactive Web sites [10]. In the case of EUD, we are particularly interested in those who are not professional programmers, but might still engage in some kinds of programming activity. Significant researches have highlighted the differences between end-user programming from professional programming; with an intent-based differentiation: End-user programming produces program for personal use or for their Community of Interest [11]; Professional programming produces program to be used by larger and more generic groups of users [12]. In the IoT context, end-user programming allows users to configure, adapt, and evolve their software by themselves [13] and such tailoring activities, together with personalization, extension, and customization, aim at programming not (only) the user interface and the behavior of an interactive

[1] http://www.drools.org/
[2] http://openrules.com/
[3] http://www-03.ibm.com/software/products/it/odm