# A Three-Layer Meta-Design Model for Addressing Domain-Specific Customizations

**Carmelo Ardito, Maria Francesca Costabile, Giuseppe Desolda and Maristella Matera**

**Abstract** Meta-design has been proposed as a model to design systems able to support End-User Development (EUD). Meta-design means "design for designers." Differently than in traditional design, professional developers do not directly create a final application, but they build software environments thorough which non-technical end users, acting as co-designers, are enabled to shape up the application while they are using it. Allowing end users to participate to the creation of their applications, by modifying or even creating from scratch software artifacts, is very challenging. To make this possible, end users have to be provided with software environments customized to their specific domain, which they can easily understand and use. In order to cope with domain specificity, this chapter presents a new meta-design model that specifically addresses the customization to a domain of interest. Customization, performed by domain experts possibly in collaboration with professional developers, becomes the key activity to provide non-technical end users with software environments that are adequate to their knowledge and needs, thus allowing them to actually become co-designers of their applications. The model is illustrated by describing its successful application to the design of a mashup platform that allows end users to create new applications by integrating data and functionality taken from different resources. The customization of the platform to different domains, such as Cultural Heritage and Technology Enhanced Learning, is discussed.

**Keywords** Meta-design · End-User Development · Mashup platform

C. Ardito (✉) · M.F. Costabile · G. Desolda
Dipartimento di Informatica, Università degli Studi di Bari Aldo Moro, Via Orabona 4, Bari 70125, Italy
e-mail: carmelo.ardito@uniba.it

M.F. Costabile
e-mail: maria.costabile@uniba.it

G. Desolda
e-mail: giuseppe.desolda@uniba.it

M. Matera
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci 32, Milano 20134, Italy
e-mail: maristella.matera@polimi.it

F. Paternò, V. Wulf (eds.), *New Perspectives in End-User Development*,
DOI 10.1007/978-3-319-60291-2_5

## 1 Introduction

Meta-design is a model often applied to designing systems supporting End-User Development (EUD) (Costabile, Fogli, Mussio, & Piccinno, 2007; Fischer & Giaccardi, 2006; Fischer, Giaccardi, Ye, Sutcliffe, & Mehandjiev, 2004). It promotes the active involvement of software engineers and end users in a continuous cycle of development, use and evolution of systems. As defined by Fischer et al.: "Meta-design extends the traditional notion of system development to include users in an ongoing process as co-designers, not only at design time but throughout the entire existence of the system" (Fischer et al., 2004). The meta-design model encompasses different activities: *meta-design* activities consist of designing software environments; this leads to the next activities of *design and use*, where end users complete the design of the final application and use it. Meta-design is in line with the so-called culture of participation (Díez, Mørch, Piccinno, & Valtolina, 2013; Fischer, 2011; Jenkins, 2009), which has received a lot of attention as it promotes a shift from consumer cultures, where produced artifacts are passively consumed, to participatory approaches that greatly exploit computational media to support collaboration and communication. The aim behind this design model is to provide end users with the means to become co-creators of new ideas, knowledge and products that can effectively satisfy their specific needs (Porter, 2008).

Following this line of action, in this chapter we show how the original meta-design model is refined by explicitly modeling all those activities that enable domain experts, possibly in collaboration with professional developers, to customize general tools to the domain of interest. Customization is indeed instrumental to provide non-technical end users with software environments that are adequate to their knowledge and needs and that actually allow them to perform EUD activities. In the new model we therefore devise three different types of activities that conceptually can be organized in three different layers.

The chapter also discusses the application of the new model to the customization of a mashup platform. In the last years we have indeed worked extensively on *fostering the adoption, in real contexts and by non-technical end users, of mashup platforms enabling EUD*. Such class of tools accommodate very well EUD, as they allow end users to create new applications by integrating functions and content exposed by remote services and Web APIs. By means of two case studies in Cultural Heritage and Technology Enhanced Learning, this chapter illustrates how the three-layer meta-design model allowed us to customize a general mashup platform for its use in the two domains.

The adoption of mashup platforms in real contexts is largely debated (see for example (Casati, 2011)). So far, the research on mashup highlighted several advantages that can favor EUD. For example, the possibility to start from ready-to-use components certainly mitigates the complexity of creating a new application from scratch and can be also faced, under given assumptions, by non-technical end users who do not know how to program and do not want to be forced to do it. However, several disadvantages also emerged, for example in relation to the difficulties for end users in understanding and using the notations to compose resources, to the inadequacy of available components with respect to the end-user needs, and to the difficulty of adding new components into the composition platforms (Namoun, Nestler, & De Angeli, 2010). Our position, which also derives from observing people adopting our tools during field studies, is that these disadvantages occur because the proposed platforms are too "general," claiming that one single design might satisfy the requirements of many domains. We therefore propose domain customization as a solution to make meta-design still more effective in creating platforms that really fit the end-user needs. This position is also in line with the guidelines proposed in (Fischer, Fogli, & Piccinno, 2017).

This chapter is organized as follows. Sect. 2 illustrates the background of this research by discussing related work. Sect. 3 presents the three-layer meta-design model and illustrates how it has driven the development of the mashup platform according to an open architecture that specifically favors customization activities. Sect. 4 reports two case studies that show how the platform was used in two application domains, after a proper customization to each one of such domains. Sect. 5 concludes the paper.

## 2 Background and Related Work

In this section, we discuss the background of this article along two main dimensions, namely meta-design as a design model to support EUD and mashup platforms as tools for fostering user-driven innovation. The goal is to push end users to evolve from passive consumers of software to active producers of new knowledge and products.

### *2.1 Meta-Design to Foster EUD*

Traditionally, the life cycle of interactive systems distinguishes between design time and use time. At design time, system developers create a system that should satisfy the requirements they collected about end users' needs and objectives. At use time, end users exploit the system to accomplish their tasks. Design frameworks are based on the assumption that major design activities end at a certain point; then use time begins and people use the system. *Participatory design* was introduced to take into account the participation of end users in the design process (Schuler, 1993). It was based on the rationale that users are experts of the application domain, thus a system can be effective only if these experts are allowed to participate in its design, highlighting their needs and expectations. In participatory design, end users are members of the design team, but no tools are provided to let them create or modify software. EUD started the trend toward a more active involvement of end users in the overall software design, development, and evolution processes, to allow them becoming co-designers of the tools and products they will use. This does not imply transferring the responsibility of good system design to them. It actually makes the work of professional developers even more difficult, since: (a) it is still their responsibility to ensure the quality of the software artifacts created by end users (Ko et al., 2011), and (b) they have to create proper tools that support end users in these new roles of designers and developers.

The design of systems that enable EUD activities thus requires a different design paradigm, called *meta-design*, which literally means "design for designers" (Costabile et al., 2007; Fischer et al., 2004). It consists of two types of activities that might also alternate: *meta-design* activities are performed by professional developers, who create the design environments that allow the diverse stakeholders to participate in the creation of the final applications; *design* activities consist of designing the final applications and are performed by end users, and possibly other stakeholders, by using the design environments devoted to them. The two activities are not clearly distinct and are executed several times in an interleaved way because the design environments evolve, both as a consequence of the progressive insights the different stakeholders gain into the design process, and as a consequence of the feedbacks provided by end users working with the system in the field.

Since several years, Costabile et al. have been working on the creation of software infrastructures that support EUD activities (Costabile, Fogli, Fresta, Mussio, & Piccinno, 2003; Costabile, Fogli, Mussio, & Piccinno, 2006; Costabile et al., 2007; Costabile, Mussio, Parasiliti Provenza, & Piccinno, 2009). They defined a design approach that allows a team of stakeholders to cooperate in the design, development, use and evolution of interactive systems. The approach is based on a meta-design model, because it prescribes that, instead of developing the final interactive system as in traditional design approaches, professional developers design software environments for the different communities of stakeholders involved in the creation of the system. Such stakeholders will use such environments to carry out specific tasks at use time, and as a side effect they will also contribute to the design and evolution of the interactive system (Costabile et al., 2009). These software environments are called *Software Shaping Workshops* (SSWs or briefly workshops). The term *workshop* comes from the analogy with an artisan's workshop (e.g., the joiner's or the smith's workshop), i.e., the workroom where the artisan finds all and only those tools necessary to carry out her/his activities. According to the metaphor, the different software environments provide all and only the tools necessary to their users to perform their specific activities, as well as interaction languages tailored to their users' culture, defined by formalizing the traditional user notations and system of signs (Iverson, 1980). In the original definition, and in particular in (Costabile et al., 2006, 2007), the SSW model distinguished three levels of activities: (1) design by software engineers; (2) design by different communities of experts of the application domain or of experts of human factors; (3) use by different communities of end users. The design by software engineers is actually meta-design according to the definition provided in

(Fischer & Giaccardi, 2006; Fischer et al., 2004). Moreover, since the focus is on EUD, it is implicit that some communities operate at both level 2 and level 3, i.e., they perform both design and use of an application, at use time.

By applying the SSW model to real cases, it was soon realized that domain experts often need to perform meta-design. Several case studies are reported in (Ardito, Buono, Costabile, Lanzilotti, & Piccinno, 2012), which show that meta-design is not only performed by software engineers, but some domain experts have often to shape software artifacts that are used by other communities of experts and/or end users to design other artifacts. Specifically, most of such meta-design activities perform customization to a specific domain, in order to tailor generic tools to the needs of non-technical end users. In (Cabitza, Fogli, & Piccinno, 2014a, 2014b), Cabitza et al. introduce the "domain developer", i.e., a domain expert actively involved in the creation of artifacts more suitable for end users and the tasks in the work domain at hand. The three-layer meta-design model presented in this chapter makes explicit the customization activities, which are crucial for making EUD possible.

Fisher proposed the model called SER (Seeding, Evolutionary and Reseeding) (Fischer, 1998). Instead of building a complete system at design time, system design starts from seeds, which are developed by meta-designers in a participatory team involving end users. A subsequent evolutionary growth follows, and then a reseeding phase occurs. The seeding phase concerns the definition of the initial prototype, which will be used by end users to perform their activities. The reseeding is performed by meta-designers to modify the initial state of a software artifact, on the basis of the evolutions determined by end users. The evolving system continually alternates between periods of unplanned evolutions by end users and periods of deliberate restructuring and enhancement. Customization to a specific domain is not explicitly addressed.

Other authors present meta-design as an approach supporting end users to tailor the tools they use. Maceli and Atwood discuss that end users often adapt systems by a trial-and-error strategy (Maceli & Atwood, 2011). Koehne et al. show that meta-design is instrumental to provide useful tools for involving end users in the design of virtual worlds, such as online role-playing games like "Lord of the Rings Online," and open-ended virtual world like "Second Life" (Koehne, Redmiles, & Fischer, 2011). Sutcliffe and Papamargaritis suggest that customization is successful for "seeding" the adoption of EUD tools and propose the use of a configuration environment based on generic conceptual models of problem domains (Sutcliffe & Papamargaritis, 2014).

## 2.2 *User-Driven Innovation by Web Mashup*

Meta-design can be fruitfully exploited for the design of *Web mashup* (simply called *mashup*) platforms. As also remarked in (Fischer et al., 2017), given their component-based nature, mashups intrinsically favor EUD and meta-design. Mashups are "composite" applications constructed by integrating ready-to-use functions and content exposed by public or private services and Web APIs (Daniel & Matera, 2014). Mashups were initially exploited in the context of the consumer Web to rapidly create applications reusing programmable APIs and content scraped out from Web pages. Soon, the potential of such lightweight integration practice emerged in various domains. Several mashup platforms have been proposed in the last years to allow end users to visually compose data and services taken from different sources, so that they can satisfy their information needs (e.g., see Aghaee & Pautasso, 2014; Danado & Paternò, 2014; Daniel & Matera, 2014; Ghiani, Paternò, Spano, & Pintori, 2016; Mehandjiev & de Angeli, 2014). Very often these platforms are general, i.e., they do not show any specificity with respect to given domains. As observed in (Casati, 2011), the lack of specificity is a problem when platforms have to be adopted by users without expertise in computer programming. Methodologies are therefore needed to create platforms that, although designed to be generic, can be then effectively specialized when adopted in specific application domains.

Mashup development resembles service composition, a development practice traditionally covered by powerful standards and technologies that, however, can only be mastered by IT experts (Ro, Xia, Paik, & Chon, 2008). What makes mashup development different from plain Web service integration is the possibility, deriving from recent Web technologies, to merge ready-to-use resources at the client-side, thus with reduced efforts and without the need of complex integration platforms. Mashup development also emphasizes novel issues, such as the composition at different layers of the application stack of heterogeneous resources that make use of different technologies. In particular, the integration at the presentation layer is the most innovative aspect enabling the creation of full-fledged Web applications whose user interface (UI) can be easily obtained by synchronizing the UIs of different ready-to-use components. If supported by adequate tools, mashup development can be an alternative to service composition that goes towards the dream of a "programmable Web" (Maximilien, Wilkinson, Desai, & Tai, 2007) even by end users without any knowledge in programming.

Because of its intrinsic value as development practice to let end users produce new value, mashup composition is in line with the so-called "culture of participation" (Fischer, 2010); users are enabled to evolve from passive consumers of applications to active co-creators of new ideas, knowledge, and products. There is indeed a specific driver at the heart of the user participation to the mashup phenomenon: *user-driven innovation*, that is, the desire and capability of users to develop their own things, to realize their own ideas, and to express their own creativity (Von Hippel, 2005). According to recent works published in literature (Ardito, Costabile, Desolda, Latzina, & Matera, 2015; Latzina & Beringer, 2012), there is also an increasing need to replace fixed applications with *elastic environments* that can be shaped up flexibly, to accommodate different situational needs. New design principles are emerging to promote paradigms where end users can access contents and functions through different devices and flexibly use and compose such resources in several situations and across several applications. If the composition activity turns out to add significant new value, the advantage for the users is that they co-create effective applications matching exactly their needs. Additionally, an interesting side effect is that the providers of the original resources can integrate the user innovation back into their core products (Iyer & Davenport, 2008) and improve their services, in order to fulfill users' requirements without the need of carrying out the iterative experimentation generally required to identify requirements and develop and test a new product. In this new process, the end users are entirely in charge of these aspects because they are enabled to create solutions that closely meet their needs.

Such innovation potential requires adequate approaches and tools for enabling mashup by non-technical end users (Daniel & Matera, 2014). However, the research on mashups has been focusing especially on enabling technologies and standards, with little attention on easing the mashup development process. Research teams and industrial players tried to define simplified composition paradigms, mostly based on visual notations and lightweight design and execution platforms running on the Web. A number of tools have been proposed that offer composition paradigms based on graphical notations, which abstract relevant mashup development aspects and operations. The user defines diagrams to express the internal logic of a mashup, without writing code. However, many of such tools failed because they resulted non adequate for end users (Casati et al., 2012; Namoun et al., 2010). One of the main reasons is that they lack intuitive abstractions (Burnett, Cook, & Rothermel, 2004; Liu, Huang, & Mei, 2007). To support the user-driven innovation potential, the challenge is indeed to let users concentrate on the conception of new ideas, rather than on the technicalities beyond service composition. In other words, users should be enabled to easily access resources responding to personal needs, integrate them to compose new applications, and simply run such applications without worrying about what happens behind the scenes.

To achieve this goal, one direction is to restrict mashup platforms to a well-defined domain the user is comfortable with. General-purpose platforms are not adequate to the needs of specific application domains and specific end users. Some studies on composition approaches indeed showed that too general platforms are not used with satisfaction by end users (Casati, 2011; Namoun et al., 2010). This represents an obstacle to a wider adoption of such platforms by non-technical people, who need to interact with tools and notations they are familiar with (e.g., see Costabile et al., 2006, 2007). In order to develop generic platforms that can be valid in different domains, it is fundamental to design platform architectures able to support the easy customization of the platform. This is what our extension to the meta-design model supports.

## 3 A Three-Layer Meta-Design Model for a Mashup Platform

Since 2012, we have been developing a mashup platform where end users, at use time and according to their needs, can select and integrate content into Interactive Workspaces (IWs). The platform may be accessed through different devices, such as a desktop computer, a mobile device or a large multi-touch display; it shows content retrieved by dynamically querying Web data sources registered into the platform, and allows the users to select pertinent content items to fill-in *visual templates*, i.e., visualization skeletons through which users easily organize and instantiate with data their IWs. In other words, the visual templates are the “containers” in which raw data (i.e., content) retrieved from Web sources are shown in the visual interface (Ardito et al., 2015). Examples of visual templates are a map showing geo-referenced data, a list of items, a chart of values. A live programming paradigm let the users see immediately the effect on any composition action, having the possibility to assess directly the progressive definition of the final application. Users can therefore easily explore any feature offered by the platform and easily go back when they are not satisfied with their choice.

The result of the visual composition is an XML-based representation of the IW, which the user can store on the platform server and download anytime and anywhere for its execution on different devices. The schema specifies the selected services, the way they are queried in order to create the desired mashup, and how the mashup results are displayed through rendering elements of the visual template.

## 3.1 The Three-Layer Meta-Design Model

The mashup platform we developed is not tied to any specific domain. Indeed, a key feature of the platform is that it provides mechanisms for customization to specific usage domains. The only way to offer a composition paradigm and resources adequate to end users of a specific application domain is to capitalize on their domain knowledge. Thus, the general tools and interaction elements have to be customized to the domain of interest. To make this possible, the platform adopts a stratification into different design (and meta-design) layers where different stakeholders contribute to the creation of different artifacts (Fischer et al., 2004). The involvement of domain experts is instrumental for a successful customization.

As represented in Fig. 1, the top layer refers to a meta-design activity performed by *professional developers* (likely a multi-disciplinary participatory team), who design and develop the design environments for the other stakeholders. The team also develops *visual templates*, by using Web technologies (for example HTML and JavaScript) or specific languages for other devices (e.g., Java for Android). Visual templates are important ingredients for the successive customization, since customized visualizations can reflect the knowledge domain.

The middle layer refers to another meta-design activity, *Domain customization*. Domain experts, possibly collaborating with professional developers (not necessarily the same that act at the first level), customize the general-purpose tool resulting from the activities at the top layer. Domain experts are familiar with the types of information end users would retrieve, the manipulations they would perform and the most suitable visualizations. Thus, they exploit a platform tool, called

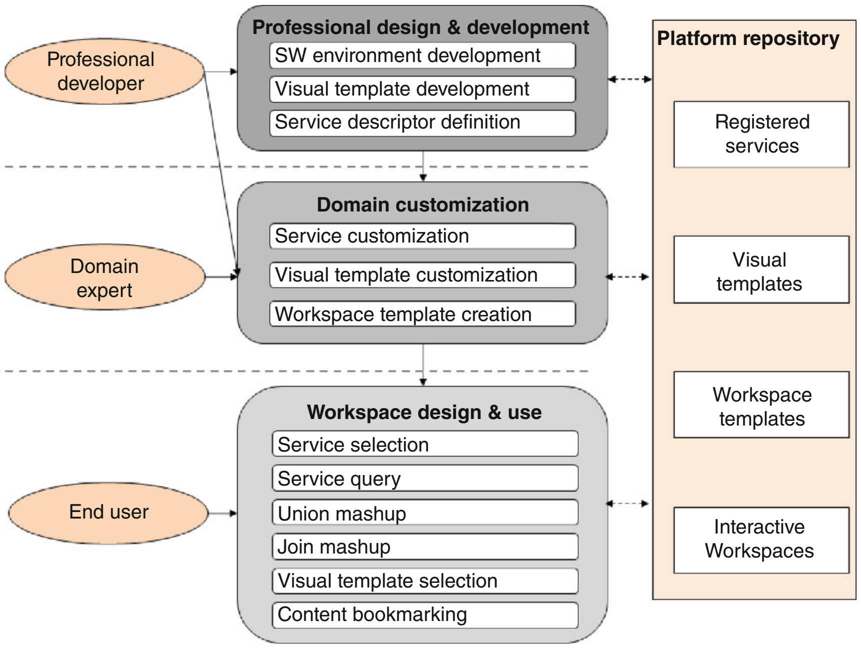

**Fig. 1** The three-layer meta-design model; the middle layer is devoted to the customization to the domain of interest

*Workspace composition environment*, to register services, compose registered services by exploiting data-composition operations (like join and union), select how to materialize service results by means of visual templates (e.g. map, list, graph) (Matera, Picozzi, Pini, & Tonazzo, 2013). Domain experts perform these technical activities in a way that is suitable for their skills. However, they do not have the skills and expertise to perform more complex customization activities like the registration of more sophisticated services (e.g. the ones requiring complex authentication mechanisms, proprietary technologies), advanced service compositions, the development of new and domain-specific visual templates, as well as the development of workspaces skeletons (i.e., pre-defined, typical aggregations of services). This is the reason why another environment, more devoted to advanced activities, is available for professional developers that integrate the domain experts' requests in the general-purpose tool.

At the bottom layer, end users finally design, use and update their IWs. This means that they start by a customized version of the mashup tool, which provides a selection of services composed and visualized according to the customization activity. In addition, in order to satisfy personal and situational needs, end users can manipulate content extracted from the registered services, for example, by using the union and join of different result sets. They can also associate different visualizations to the composed content and bookmark content in order to save it.

The possibility for end users to select pertinent services, query them and aggregate the retrieved content, and especially the opportunity to define and customize visual templates makes the entire approach *elastic*. So far, software systems have been conceived as pre-packaged sets of data, functionality, and visualizations that somebody else (the software developer) builds for us. *Elastic systems* diverge from such idea and try to promote paradigms where contents, functionality and access devices are totally decoupled from specific contexts of use and can be determined at use time. Elasticity is, in other words, an opportunity to accommodate multiple and variable contextual needs, moving the responsibility to end users of creating their own applications (Latzina & Beringer, 2012).

The customization is performed before using the platform in a new application domain; it can be later re-executed to satisfy specific needs emerging later, e.g., to register or to combine further services, as it emerges by the platform usage in the field. In Sect. 4, we illustrate customization activities by means of examples of the usage of a real platform in two different application domains.

## 3.2 Architecture for Mashup Platforms Implementing the Meta-Design Model

Adequate software architectures are needed, in order to make concrete the meta-design model illustrated above. We here report the architecture of EFESTO (Desolda, Ardito, & Matera, 2016), the mashup platform that we have designed with the specific purpose of supporting a meta-design methodology. The platform architecture complies with a separation of concerns so that the layers managing the different aspects of mashup creation and execution (presentation, logics, data) are decoupled. This means that each aspect, if needed, can be easily adapted to the application domain.

Separation of concerns is facilitated especially by the compositional nature of the platform. Being a mashup platform, EFESTO is indeed conceived for the integration of heterogeneous services. This openness facilitates the customization of the platform with respect to the characteristics and needs of specific communities of end users. Customization, for example, occurs by selecting and registering into the platform services and data sources (public or private) that, for any different domain, can provide content able to fulfill specific users' information needs. Service registration is kept as simple as possible, so that even non-technical users can possibly add new services if needed. Indeed. Except for particular cases, service registration requires the user to input, by means of visual forms, the service URI and the value of some search keys for executing basic service queries. Then the XML specification, i.e. the *Service Descriptor*, is automatically generated by the system and stored in the *Repository Server* (Desolda, 2015). A further customization activity performed by domain experts consists of reducing the initial data set of a registered service, so that only the attributes of interest for a specific domain are available to end users.

In EFESTO different *Visual templates*, which play the role of visualization containers (Cappiello, Matera, & Picozzi, 2015), can be easily introduced to represent metaphors and interaction paradigms that best suit the background and the needs of the addressed end users. Through visual templates, domain experts define how the content dynamically retrieved by querying a service will be visualized in proper visualization containers to be then adopted by end users to create their IWs. Visual templates, available in the *Repository Server*, provide end users with a schematic representation of how data extracted from services will be organized, i.e., aggregated and visualized. They also provide data integration schemas, as they determine how the involved data sources are queried and the resulting data integrated.

This schematic representation can be easily modified to reflect domain specificity. Providing a new visual template implies defining a new HTML template or a new View for execution on an Android smart phone. At composition time, by visually associating selected service attributes to visual template fields, the end user defines a projection of the only attributes of interest. In addition, if the attributes associated to a single visual template element are selected from multiple services, then the structure of the visual template determines a global integration schema mapping the attributes of single services into an integrated data set. In few words, to operate on data, end users actually manipulate visual representations that can be easily modified to accommodate the end-user mental model.

The overall organization of the platform is represented in Fig. 2. Thanks to the adoption of a live programming paradigm, end users create their IW through the

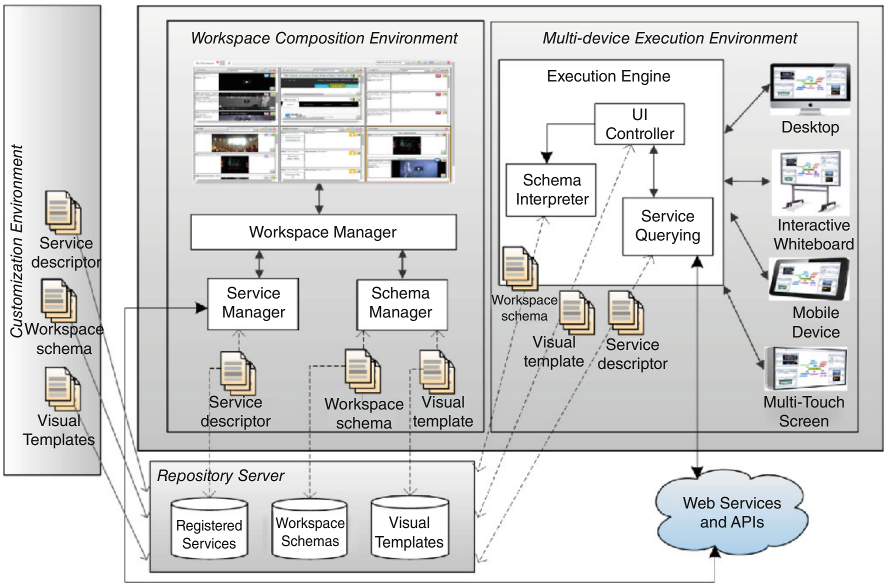

**Fig. 2** The architecture of the EFESTO mashup platform

*Workspace Composition Environment*, an HTML/JavaScript Web application that allows them to execute composition actions and immediately see the result, i.e., a running application. A *Workspace Manager* intercepts the visual mapping and synchronization actions performed by an end user. Through its *Schema Manager* module, such actions are automatically translated into elements of a *Workspace schema*, expressed in an XML-based domain specific language (Cappiello et al., 2015), which describes the service queries, the association of the query results with specific visual templates, and possible synchronizations among different visual templates.

The execution of an IW can occur on the device where it was created, as well as through an *Execution Environment* running on a different device (right side of Fig. 2). An *Execution Engine*, possibly implemented through any Web or device-native technology, interprets the Workspace schema (*Schema Interpreter*) and instantiates the adopted visual templates (*UI controller*), by rendering the corresponding user interface and filling the visual elements with data requested to the involved services (*Service Querying*). It is worth noting that the Model-Driven Architecture paradigm on which our approach is based allows the user to generate *one* platform independent model, representing the structure, in terms of integrated data sources and data visualizations, of the composed application, and to pervasively execute it on *different* devices and in *different* contexts of use.

## 4 Customization to Specific Application Domains

In order to verify the usefulness and validity of the extended meta-design model implemented in our mashup platform, we performed two field studies in different application domains. One study was carried out in the context of visits to archaeological parks. Two professional guides composed a mashup application for retrieving content relative to an archaeological park using a desktop application, accessible through a PC placed in his/her office. Later, during a guided visit of the archaeological park, two guides use the mashup application to show the content to visitors by using a large interactive display when introducing the visit and a tablet device during the tour in the park.

Another field study, performed in a context of Technology-Enhanced Learning (TEL), allowed us to analyze the use of the platform in a situation where students learn about a topic presented in class by their teacher, complementing the teacher's lecture by searching information on the Web. The retrieved information can also be communicated and shared with the teacher and the other students using interactive whiteboards, desktop PCs and personal devices (e.g., laptop, tablet and smartphone). These two studies are reported in details in (Ardito et al., 2014). The description in the next two subsections emphasizes the customization activities performed before the actual studies.

## *4.1 Customization in a CH Context*

In order to customize the mashup platform to the Cultural Heritage context, in particular to provide support to the activities of professional guides, we worked in a team that included two professional developers with HCI expertise and two guides with a long experience of conducting visit in archaeological parks. They met twice to perform various activities.

During the first meeting, the guides explained the way they usually organize a visit. The briefing phase performed before the actual tour through the ruins is fundamental. It aims at both introducing visitors to the history of the archaeological park and providing some preliminary information. It is usually carried out in front of a large panel showing the map or an aerial photography of the park. This phase would greatly benefit from making the panel interactive and able to show multimedia content related to the topics described by the guide. The team agreed that the debriefing should be supported by an interactive workspace displayed on a large display. Multimedia content (Web pages, images and videos) retrieved beforehand by the guide from the Web could be displayed as icons on a map.

After the meeting, the professional developers performed a first step of customization of their Interactive Workspaces (IW) by registering in the platform services like Google Search, Wikipedia, Google Images, FlickR and YouTube. In addition, developers integrated the map visual template by including the Google Maps service that, beyond the map, also provides some business logic; for example, it displays further details of a place by clicking on the corresponding pin on the map.

During the second meeting, the two guides had the possibility to directly perform a second step of customization using a desktop application, accessible through a PC placed in their office. First, they decided which services should be synchronized with the map, in order to show service data as pin on the map when a search was performed. Second, they saved favorites contents relative to the archaeological park of Egnathia (in Southern Italy) in a specific container with lists of items. Lastly, both the guides and the developers decided that the same interactive workspace should be made available on a tablet carried out by the guide, so that it could be accessed during the tour (Fig. 3).

Once the platform was customized, few days later the guides experimented the mashup platform with a large interactive display (46-inch) and a tablet device (7-inch) during two guided visits of the archaeological park, involving 28 visitors. To introduce the visit, the professional guides interacted with the IW they created, in order to “enhance” their presentation of the history of the park. The IW was then executed on a large interactive display available at the entrance of the park museum (Fig. 4a). Media contents, such as photos, videos, and wiki pages associated with park locations to be visited during the guided tour were represented by an icon and a title placed on a map centered on the park. By tapping on an icon, a pop-up window visualizes the corresponding media. During the park tour, the

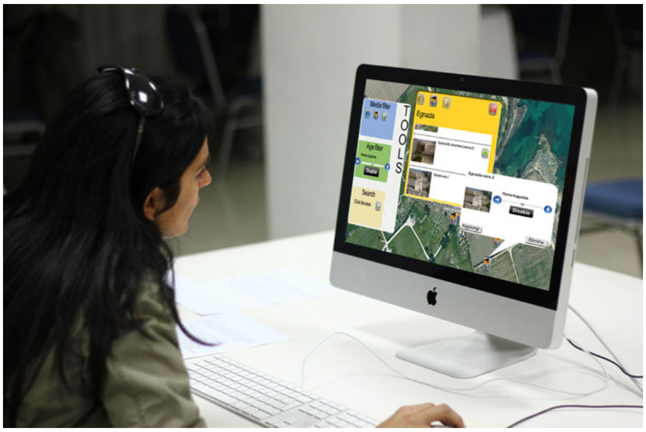

**Fig. 3** A guide performing the customization of the platform

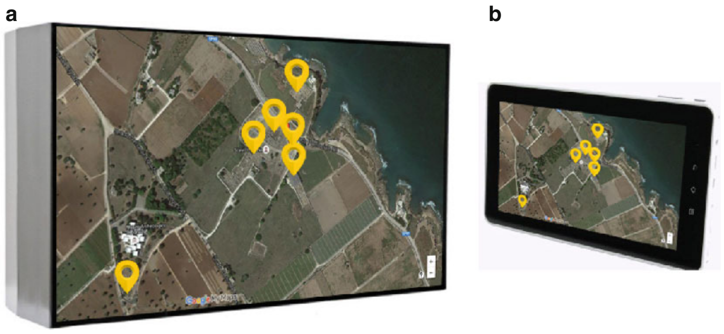

**Fig. 4** IW for the archaeological park of Egnathia visualized on a large interactive display (a) and on tablet (b)

guides accessed their IW on the tablet (Fig. 4b), in order to show photos, videos and other information when appropriate.

The study showed a general appreciation of the use of IW in the context of the visit and interesting insights emerged. The guides acknowledged the support of the mashup platform in composing the application and organizing the material for the visit. However, they complained about the scarce material they were able to find when searching the services available in the platform. This is a problem common to all service-based applications, which have to rely on content made available by third-parties. To limit this problem, more sensible services should be added into the platform; they can be further third-parties' services, if any responding to the user needs exists, but they can also be local and ad-hoc created collections of contents, maintained by domain experts and even fed by end users themselves by adding self-produced material. Also, since the services used for the study at the Egnathia park are Web 2.0 resources, the guides could publish online their own material (e.g., videos, pictures, Wikipedia pages) that can thus be easily accessed through the mashup platform. This of course requires a more intensive use of the system by the guides, since they have to realize which content is missing and to enrich consequently their public online collections.

It also emerged that guides would like to have the possibility of switching among different visualizations, according to the specific task they are performing. For example, it happens quite often that they want to refer to buildings or venues located in a different park. In this case, they are forced to navigate in the map for localizing the other park, which could be very far, and then show the content. Thus, they want the possibility to organize these contents, which cannot be positioned on the park they are currently visiting, in a different visual template, even a simple folder tree like the one used by Windows™ operating system they are familiar with.

## *4.2 Customization in a TEL Context*

The platform was also validated in a Technology Enhanced Learning (TEL) context. Nowadays, schools are provided with different computing devices, not only desktops but also different types of tablets and interactive whiteboard. Teachers and students are increasingly using such devices in their daily activities. The experience on TEL of some of the authors of this paper showed that, if used with proper techniques and tools, technology may be a valid support to learning and can even encourage people to become more active in their learning activities (Ardito, Costabile, De Angeli, & Lanzilotti, 2012). The proposed platform has a great potential to be one of such supporting tools.

The customization of the platform to the TEL domain was performed by a team of two professional developers with HCI expertise and two high school teachers. They met four times to perform various activities. Other activities were performed in between two consecutive meetings.

In the first two meetings, important information to identify new services to be registered was collected. Teachers illustrated their current use of technology in their school. Teachers and students regularly use Google Drive tools to support the activity of sharing and integrating information they find on the Web using students' laptops or tablets. The teacher organizes a Google Drive folder in subfolders, each related to a class topic. Web pages, images, videos, presentations, or part of them that the teacher has selected for her/his class are pasted into a document and saved in a folder, which is shared with students. In addition, each student has a folder on Google Drive (named with his/her name), containing his/her documents, some of which are shared with other students and with the teacher, others are only in view modality for other people. In class, the teacher uses the interactive white board, in order to show and discuss the contents available in the folder of that specific topic. A blank document is opened, in which s/he writes the titles of the topics that students will further deepen. Students individually perform their searches in the laboratory or at home and create documents that contain links to content on the Web and/or portions of Web documents that are copied and pasted in a new document. Each student saves these documents in the personal folder in the class folder. Back to classroom, the teacher, through the interactive whiteboard, examines and discusses with students the documents produced by them. During the discussions, the two teachers realized that, while Google Drive only permits manual operations to copy and paste into a new document text, images and links to Web pages, videos, etc., the mashup platform should be valuable in performing, in particular, the following activities: (1) creating more sophisticated search tools by composing data coming from different services; (2) updating the content returned by the components by simply re-running their queries; (3) organizing contents in appropriate visual templates. At the end of the second meeting, the team agreed that the teacher's class should be supported by multimedia content (Web pages, images, videos, presentations) retrieved beforehand by the teacher from the Web. Thus, the services Google Search, Wikipedia, SlideShare, Google Images, YouTube and Vimeo were registered in the platform.

In the third meeting, teachers customized the platform by registering new services (e.g. Wikipedia, SlideShare). They were able to manipulate content, performing join and union of services, primarily using various types of lists to visualize the results. They asked for having the possibility to save the current results of the composed services somewhere, replicating the classical operation they were used to do: they copy and paste the results of their searches in a document in Google Drive. This opened a discussion within the team. Teachers understood the different behavior of a widget in the workspace. Indeed, once a user, acting on that widget, performs a query, the original sources are accessed, but the results may be different than those obtained with a previous query on the same widget. This has many advantages, but the teachers explained that sometimes, when they find an interesting result, they want to keep it to show later to their students. In order to satisfy this requirement, the final decision was to implement in the platform a very primitive container, a kind of folder, in which they can save the results of a specific query. This "Favourite" folder was indeed implemented in the platform.

In the fourth and last meeting, teachers finalized the customization of the platform by refining the services and saving some contents in the favorite container. However, some concerns aroused about the appropriateness of the content visualization allowed by the "Favourite" container for supporting class activities. At the end, the teachers insisted on having a different visual template, such as a *concept map*, which permits to organize and structure the retrieved contents according to learning concepts and their relationship. Therefore, the design team specified the requirements of this new visual template so that a first prototype could be available.

The use of the customized platform was carried out at a high school in Southern Italy. It was organized over three days and involved a class of 16 students

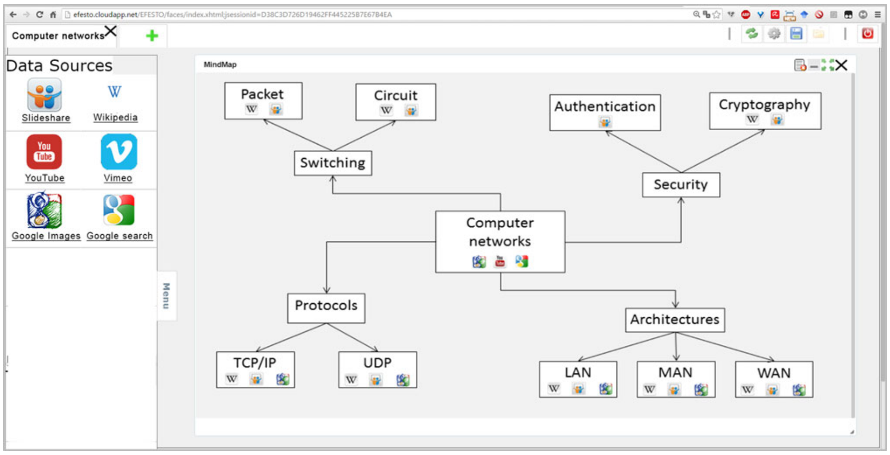

**Fig. 5** The Interactive Workspace on *Computer Networks* created by the teacher using a desktop PC

(9 females, 19-year-old on average) and a teacher. The first day, using a PC in his office, the teacher composed an IW relative to a specific topic, i.e., *Computer Networks*, searching and including content about *Protocols*, *Architectures*, *Switching* and *Security* retrieved from the registered services. The retrieved contents were saved and organized in the concept map container (see Fig. 5).

Two days later, the teacher taught a class supported by the IW visualized on an interactive whiteboard. The teacher very effectively presented the different contents; he visualized them using the concept map container, with which he was familiar. At the end, he divided the students in groups of 2–3; each group was assigned the task of creating an IW about a specific *Computer Networks* sub-topic, e.g., *Protocols*, *Packet Switching*, *Latency Period*. After a brief individual training session, all the groups accessed the laboratory to carry out their assignments.

Next day students presented in class their group work using a interactive whiteboard (see Fig. 6). The list container, used by students to organize the content they retrieved, proved very poor for presentation purposes. In particular, the list visualization makes difficult the identification of a specific content to be illustrated.

Students agreed that the concept map would be a better visual template, even if they did not realize this during the workspace composition. A group of students proposed a visual template in which contents could be organized in different folders; it is evident that this derived from the influence of Google Drive on their mental model. In general, they agreed on the value of more flexibility in organizing the interactive workspace. The provided visual templates should also be empowered with functionality that permits ordering, filtering and searching actions.

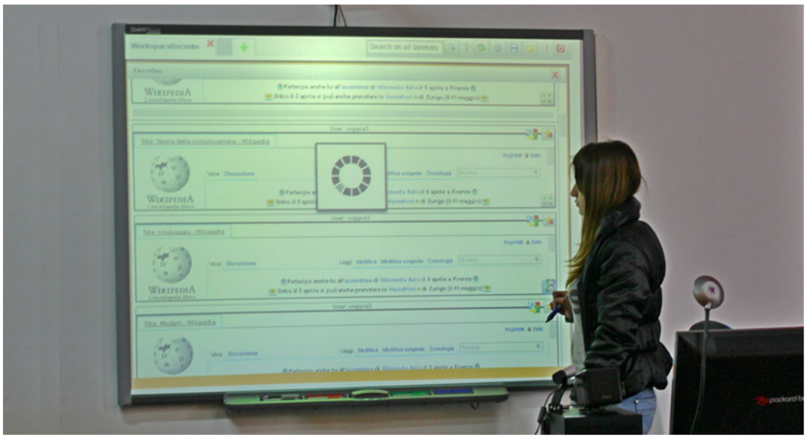

**Fig. 6** A student presenting the Interactive Workspace of her group organized as a list of contents and visualized on an Interactive White Board

## 4.3 Discussion

The studies conducted in the Cultural Heritage and in the Technology Enhanced Learning domains (see Ardito et al., 2014 for more details) demonstrated how the customization activities allowed domain experts to take advantage of their domain knowledge to adapt the general-purpose mashup platform to the specific end-user requirements in those domains. Customization was functional to foster the adoption of mashup platforms in real contexts, also favoring the "seeding" of such EUD tools in a specific domain.

Besides allowing us to assess the advantages of the customization activities introduced in the meta-design model, the studies demonstrated that the platform is sufficiently easy to use and users felt quite supported in accomplishing their tasks. Most participants appreciated the value of the platform in enabling easy and effective integration of content retrieved on the fly from online APIs. Low response time of the platform was indicated as a negative aspect, but this was due to the very poor technology infrastructure available both at the archaeological park and at the school lab. In other studies that we performed to evaluate the platform, none ever complained about this problem.

Participants highlighted the lack of collaboration tools, such as chats or forums. Other remarks also concerned distributed collaborative creation of components and functions to annotate them services, widgets and information items. In earlier versions of the platform (Matera et al., 2013) we already included these functions, thus their provision would be possible. They were not made available during the study as our main focus was on the adequateness of the composition paradigm.

The studies also revealed new requirements that mashup platforms should feature to foster their adoption in real contexts. First, the users expressed the need to "manipulate" data extracted from services. They highlighted that through the platform they could not perform much more than visualizing data, modifying visualizations, and inspecting data details. They would instead appreciate functions to make the displayed information *actionable*, i.e., suitable for being manipulated according to their task goals. For example, in the content retrieval task, beyond composing services and choosing how to visualize retrieved content, participants also wanted to perform ordering, filtering, or selecting a specific part of a content item, possibly annotating the selected parts with comments. A suggestion came out about the adoption of the *mind map* as further visual template, because teachers are familiar with it and it is indeed appropriate in the learning domain where concept relationships are very significant.

Second, they needed to satisfy complex information needs by gathering data from the entire Web - not only from pre-packaged components. Inspired by these requirements, the most recent version of EFESTO offers: (1) a set of tools to organize, visualize and manipulate extracted data according to specific functions (Ardito et al., 2015); (2) a new "polymorphic" data source that exploits the Linked Open Data cloud (Desolda, 2015); (3) visual mechanisms to integrate data retrieved from different data sources (Ardito et al., 2014). Further studies have been planned to assess the benefit of these new features.

## 5 Conclusion

This chapter presented a three-layer meta-design model to build systems that enable EUD and that leverage domain specificity to provide end users with tools that really make sense in real contexts of use. The peculiarity is the introduction of additional methodological activities, which address the customization of systems to specific domains. This customization is performed by domain experts, possibly in collaboration with professional developers.

Although the studies were conducted in two specific domains, we are confident that the proposed methodology can be effectively applied to the customization of any domain. Our current work is devoted to further refining the customization activities. For this purpose, following a bottom-up approach, we are applying the methodology for customizing the mashup platform to other domains, in particular home automation to support the elderly. This domain poses some more challenges: even the composition paradigm needs to be revised, as also smart objects needs to be composed and synchronized with Web services. Some preliminary results however already confirmed the effectiveness of the three-layer design model and the adequateness of the architecture organization of the EFESTO platform.

**Acknowledgments** This work is partially supported by the Italian Ministry of University and Research (MIUR) under grants PON03PE_00136_1 - "DSE - Digital Services Ecosystem" and CTN01_00128_111357 - "SHELL - Cluster Smart Living Technologies".

## References

Aghaee, S., & Pautasso, C. (2014). End-user development of mashups with naturalmash. *Journal of Visual Languages & Computing*, *25*(4), 414–432.

Ardito, C., Bottoni, P., Costabile, M. F., Desolda, G., Matera, M., Picozzi, M. (2014). Creation and use of service-based distributed interactive workspaces. *Journal of Visual Languages & Computing*, *25*(6), 717–726.

Ardito, C., Buono, P., Costabile, M. F., Lanzilotti, R., Piccinno, A. (2012). End users as co-designers of their own tools and products. *Journal of Visual Languages & Computing*, *23* (2), 78–90.

Ardito, C., Costabile, M. F., De Angeli, A., Lanzilotti, R. (2012). Enriching exploration of archaeological parks with mobile technology. *ACM Transactions on Computer-Human Interaction*, *19*(4), 1–30. Article 29.

Ardito, C., Costabile, M. F., Desolda, G., Lanzilotti, R., Matera, M., Picozzi, M. (2014). Visual composition of data sources by end users. In *Advanced visual interfaces (AVI '14), Como, Italy* (pp. 257–260). New York: ACM.

Ardito, C., Costabile, M. F., Desolda, G., Latzina, M., Matera, M. (2015). Making mashups actionable through elastic design principles. In P. Díaz, V. Pipek, C. Ardito, C. Jensen, I. Aedo, A. Boden (eds.). *End-user development - IS-EUD 2015* vol. LNCS 9083, (pp. 236–241). Berlin Heidelberg: Springer.

Burnett, M., Cook, C., Rothermel, G. (2004). End-user software engineering. *Communications of the ACM*, *47*(9), 53–58.

Cabitza, F., Fogli, D., Piccinno, A. (2014a). "Each to his own": distinguishing activities, roles and artifacts in EUD practices. In L. Caporarello, B. Di Martino, M. Martinez (eds.). *Smart organizations and smart artifacts: fostering interaction between people, technologies and processes* vol. 7, (193–205). Cham: Springer International Publishing.

Cabitza, F., Fogli, D., Piccinno, A. (2014b). Fostering participation and co-evolution in sentient multimedia systems. *Journal of Visual Languages & Computing*, *25*(6), 684–694.

Cappiello, C., Matera, M., Picozzi, M. (2015). A UI-centric approach for the end-user development of multidevice mashups. *ACM Transactions on the Web*, *9*(3), 1–40.

Casati, F. (2011). How end-user development will save composition technologies from their continuing failures. In M. Costabile, Y. Dittrich, G. Fischer, A. Piccinno (eds.). *End-user development - IS-EUD 2011* LNCS vol. 6654, (4–6). Berlin Heidelberg: Springer.

Casati, F., Daniel, F., Angeli, A. D., Imran, M., Soi, S., Wilkinson, C. R., et al. (2012). Developing mashup tools for end-users: on the importance of the application domain. *International Journal of Next-Generation Computing*, *3*(2), 144–172.

Costabile, M. F., Fogli, D., Fresta, G., Mussio, P., Piccinno, A. (2003). Building environments for end-user development and tailoring. In *IEEE symposium on human centric computing languages and environments (HCC'03)* (pp. 31–38). Auckland, New Zealand: IEEE Computer Society.

Costabile, M. F., Fogli, D., Mussio, P., Piccinno, A. (2006). End-user development: the software shaping workshop approach. In H. Lieberman, F. Paternò, V. Wulf (eds.). *End user development* (pp. 183–205). Dordrecht, The Netherlands: Springer.

Costabile, M. F., Fogli, D., Mussio, P., 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*(6), 1029–1046.

Costabile, M. F., Mussio, P., Parasiliti Provenza, L., Piccinno, A. (2009). Supporting end users to be co-designers of their tools. In V. Pipek, M. Rosson, B. de Ruyter, V. Wulf (eds.). *End-user development - IS-EUD 2009* vol. LNCS 5435, (70–85). Berlin Heidelberg: Springer.

Danado, J., & Paternò, F. (2014). Puzzle: a mobile application development environment using a jigsaw metaphor. *Journal of Visual Languages & Computing*, *25*(4), 297–315.

Daniel, F., & Matera, M. (2014). *Mashups - concepts, models and architectures*. Berlin Heidelberg: Springer-Verlag.

Desolda, G. (2015). Enhancing workspace composition by exploiting linked open data as a polymorphic data source. In E. Damiani, R. J. Howlett, L. C. Jain, L. Gallo & G. De Pietro (Eds.), *Intelligent interactive multimedia systems and services - IIMSS '15* 40, (97–108). Cham: Springer.

Desolda, G., Ardito, C., Matera, M. (2016). EFESTO: a platform for the end-user development of interactive workspaces for data exploration. In: F. Daniel, C. Pautasso (Eds.), *Rapid mashup development tools.* Communications in Computer and Information Science. Vol 591 (pp. 63–81). Cham: Springer.

Díez, D., Mørch, A., Piccinno, A., Valtolina, S. (2013). Cultures of participation in the digital age: empowering end users to improve their quality of life. In Y. Dittrich, M. Burnett, A. Mørch, D. Redmiles (eds.). *End-user development - IS-EUD 2013* vol. LNCS 7897, (pp. 304–309). Berlin Heidelberg: Springer.

Fischer, G. (1998). Seeding, evolutionary growth and reseeding: constructing, capturing and evolving knowledge in domain-oriented design environments. *Automated Software Engineering*, *5*(4), 447–464.

Fischer, G. (2010). End user development and meta-design: foundations for cultures of participation. *Journal of Organizational and End User Computing*, *22*(1), 52–82.

Fischer, G. (2011). Understanding, fostering, and supporting cultures of participation. *Interactions*, *18*(3), 42–53.

Fischer, G., Fogli, D., Piccinno, A. (2017). Revisiting and broadening the meta-design framework for end-user development. In F. Paternò & V. Wulf (eds.), *New perspectives in end-user development* (pp. 61–98). Cham: Springer.

Fischer, G., & Giaccardi, E. (2006). Meta-design: a framework for the future of end-user development. In H. Lieberman, F. Paternò & V. Wulf (eds.), *End user development* (pp. 427–457). Dordrecht, The Netherlands: Springer.

Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A., Mehandjiev, N. (2004). Meta-design: a manifesto for end-user development. *Communications of the ACM*, *47*(9), 33–37.

Ghiani, G., Paternò, F., Spano, L. D., Pintori, G. (2016). An environment for end-user development of web mashups. *International Journal of Human-Computer Studies*, *87*(C), 38–64.

Iverson, K. E. (1980). Notation as a tool of thought. *Communications of the ACM*, *23*(8), 444–465.

Iyer, B., & Davenport, T.H. (2008). Reverse engineering: Google's innovation machine. *Harvard Business Review*, *86*(4).

Jenkins, H. (2009). *Confronting the challenges of participatory culture: media education for the 21st century*. Cambridge, MA: MIT Press.

Ko, A. J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwig, M., et al. (2011). The state of the art in end-user software engineering. *ACM Computing Surveys*, *43*(3), 1–44.

Koehne, B., Redmiles, D., Fischer, G. (2011). Extending the meta-design theory: engaging participants as active contributors in virtual worlds. In M. F. Costabile, Y. Dittrich, G. Fischer, A. Piccinno (eds.). *End-user development - IS-EUD 2011* vol. 6654, (pp. 264–269). Berlin Heidelberg: Springer.

Latzina, M., & Beringer, J. (2012). Transformative user experience: beyond packaged design. *Interactions*, *19*(2), 30–33.

Liu, X., Huang, G., Mei, H. (2007). Towards end user service composition. In: *Computer software and applications conference (COMPSAC '07)*, Beijing, China, 24–27 July (pp. 676–678). IEEE.

Maceli, M., & Atwood, M. E. (2011). From human crafters to human factors to human actors and back again: bridging the design time – Use time divide. In M. F. Costabile, Y. Dittrich, G. Fischer, A. Piccinno (eds.). *End-user development - IS-EUD 2011* vol. 6654, (pp. 76–91). Berlin Heidelberg: Springer.

Matera, M., Picozzi, M., Pini, M., Tonazzo, M. (2013). PEUDOM: a mashup platform for the end user development of common information spaces. In F. Daniel, P. Dolog, Q. Li (eds.). *Web engineering - ICWE '13* vol. LNCS 7977, (pp. 494–497). Berlin Heidelberg: Springer.

Maximilien, E. M., Wilkinson, H., Desai, N., Tai, S. (2007). A domain-specific language for web APIs and services mashups. In B. Krämer, K.-J. Lin, P. Narasimhan (eds.). *Service-oriented computing – ICSOC 2007* vol. LNCS 4749, (pp. 13–26). Berlin Heidelberg: Springer.

Mehandjiev, N., & de Angeli, A. (2014). Guest editors introduction: representations and environments for user-driven development of service applications. *Journal of Visual Languages & Computing*, *25*(4), 251–252.

Namoun, A., Nestler, T., Angeli, A. (2010). Conceptual and usability issues in the composable web of software services. In F. Daniel, F. M. Facca (eds.). *International conference on web engineering - ICWE 2010 workshops - revised selected papers* vol. LNCS 6385, (pp. 396–407). Berlin Heidelberg: Springer.

Namoun, A., Nestler, T., De Angeli, A. (2010). Service composition for non-programmers: prospects, problems, and design recommendations. In: *IEEE European conference on web services (ECOWS '10), Ayia Napa, Cyprus* (pp. 123–130). Washington, DC: IEEE Computer Society.

Porter, J. (2008). *Designing for the Social Web*. Thousand Oaks, CA: New Riders Press.

Ro, A., Xia, L.-Y., Paik, H.-Y., Chon, C. (2008). Bill organiser portal: a case study on end-user composition. In S. Hartmann, X. Zhou, M. Kirchberg (eds.). *Web information systems engineering – WISE 2008 workshops* vol. LNCS 5176, (pp. 152–161). Berlin Heidelberg: Springer.

Schuler, D. (1993). *Participatory design: principles and practices*. Hillsdale, NJ: L. Erlbaum Associates.

Sutcliffe, A., & Papamargaritis, G. (2014). End-user development by application-domain configuration. *Journal of Systems and Software*, *91*, 85–99.

Von Hippel, E. (2005). *Democratizing innovation*. Cambridge, MA: MIT Press.