# A Wizard Based EUDWeb Development Process

**Loredana Caruccio, Vincenzo Deufemia and Giuseppe Polese**

**Abstract** The pervasiveness of technological media in the daily life of the people has given rise to the end-user development (EUD) research area, which aims to empower end-users to be developers of their applications. The moving towards Web technologies introduced further challenges for EUD researchers. With respect to the significant number of solutions that have been developed, including methodologies and tools, little effort has been produced to support end-users in a proper development process of their applications. In this paper, we propose a wizard-based development process guiding the users towards the construction of own Web applications (Webapps), letting them accomplish complex tasks, such as those related to security and access control.

**Keywords** End-user development · Wizard-based development process · Web application development · Access control

## 1 Introduction

In the last decades number of people accessing digital information has increased tremendously, mainly due to the pervasiveness of technological media, such as the new generation of mobile phones that allow users to browse the Web, share contents, and create customized pages on social networks. The familiarity of users with these new technologies yields the need to customize applications to their requirements. In other words, there is the necessity of creating tools capable of supporting

L. Caruccio (✉) · V. Deufemia · G. Polese
Università di Salerno, Via Giovanni Paolo II, 132, 84084 Fisciano, SA, Italy
e-mail: lcaruccio@unisa.it

V. Deufemia
e-mail: deufemia@unisa.it

G. Polese
e-mail: gpolese@unisa.it



T. Torre et al. (eds.), *Empowering Organizations*, Lecture Notes in Information Systems and Organisation 11, DOI 10.1007/978-3-319-23784-8_14

end-users in developing software applications for their needs. For this reason, end-users can be defined as domain developers: experts of a specific domain [6], in which their main goal is more oriented to the development of capabilities available in their setting, than just write software code [2, 3].

The end-user development (EUD) research area aims to empower end-users in developing their own applications [18]. With the growth of the WWW, including new user friendly devices to surf the Web, such as smartphones and tablets, the development of Web applications by end-users (EUDWeb) has received a considerable interest from researchers. However, web-based programming requires dealing with additional aspects with respect to stand-alone programming, like for example, the client/server paradigm, the access control [16], the stateless protocol, and distributed databases. Often, these aspects are known to be difficult even for expert programmers, hence at first glance they appear insurmountable to end-users [26].

A further challenge in EUDWeb is the support of the development process by hiding the software engineering activities that expert programmers usually employ [19]. However, although an end-user can hardly follow a development process like those used by expert programmers, a qualitatively good application cannot be built without a development process with specific steps and activities to be followed.

In this paper we propose an EUDWeb process guiding end-users through wizards in the development of web applications. In this way, end-users can follow a specific development process, which lets them focus on specific tasks with predefined steps, reducing the perception of dealing with a complex work. Furthermore, the proposed process relies on several visual metaphors, which simplify its enactment, by also removing technical aspects, and consequently, the need for programming skills. Finally, particular attention has been given to the access control management that is one of the most complex tasks for end-users, as previously stated.

The paper is organized as follows. Section 2 presents related work by discussing methodological solutions and tools in the EUD area, whereas Sect. 3 describes the proposed wizard-based process for EUDWeb development. Finally, conclusions and future works are included in Sect. 4.

## 2 Related Work

In the literature there exist many approaches for EUD. In particular, we can find both methodological solutions [10–15] and tools [1, 17, 27–29].

Most methodological solutions are based on the use of the Meta Design paradigm [11–14] in the EUD context. As an example, the Seeding-Evolutionary growth-Reseeding (SER) Model [10] is a collaborative development process where end-users become an active part of it. Another solution following the Meta Design paradigm is the CBEADS framework [15], which requires the development of a Meta-Model of the application and provides a set of tools through which end-users can build and evolve their applications.

Other approaches propose online application development processes that exploit the contents and services available on the Web, such as open API or reusable services [4]. In other words, the development of web applications consists in the combination of data and services provided by web resources into a new integrated service [31]. To this end several programming environments assisting end-users into web-mashup application development have been created [1, 17, 27–29].

Finally, some solutions are based on the WYSIWYG (What You See Is What You Get) paradigm, which allows end-users define all the application's features through a simple composition of user interface elements [5, 7, 22].

Regarding the tools, a first classification of them has been introduced by Rode et al. [25]. In general, many tools exist for developing specific applications as the web and commercial content management [20, 23, 30], or online database management [9, 21, 24].

## 3 The Wizard-Based Process for EUDWeb Development

In this section we present our wizard-based process for EUDWeb development. It aims to support end-users step-by-step during the construction of general purpose web applications. More specifically, as shown in Fig. 1, we have defined five general steps guiding the users towards the *configuration* and *generation* of web applications. Each step of the process allows users to focus on specific tasks, such as the services to be embedded in the application or the definition of access control policies. Following the development process shown in Fig. 1 the users can generate their own applications and iteratively refine them by moving forward and backwards across steps.

In the following sections, we detail each step of the development process by using a running example concerning the construction of a web application for the management of student information.

### *3.1 Step 1: What*

The first step is indicated with the word *What* and consists of the selection of the application domain. The proposed solution supports the definition of the domain by providing a set of pre-defined management applications.


**Fig. 1** The step-by-step development process

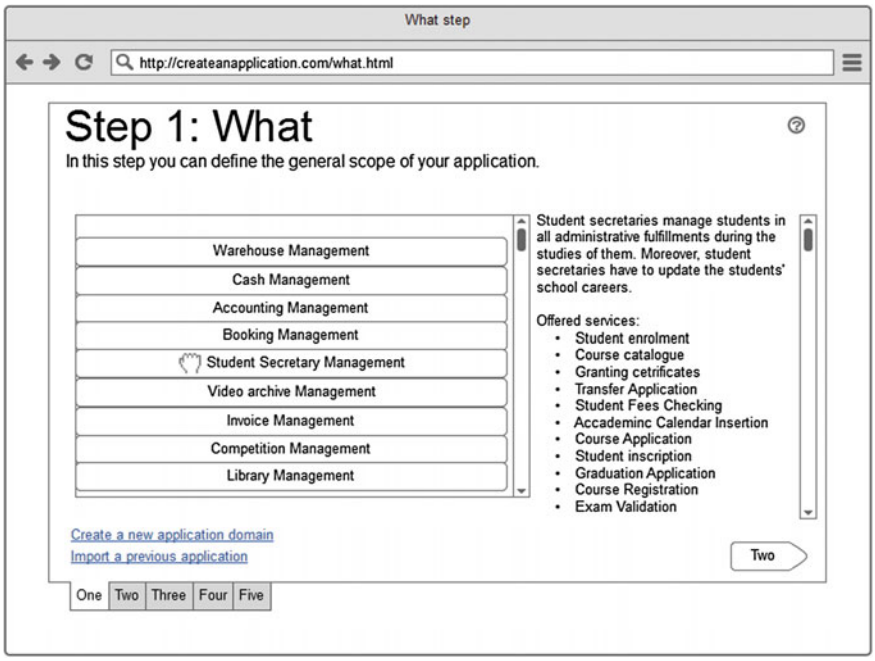


**Fig. 2** The user interface for selecting the application domain

An application domain includes the description of the main services offered for the specific application area. In this way, users can read each domain description and select the domain offering the services more suited to their needs. As an example, in Fig. 2 the selection of the *Student Secretary Management* a plication domain visualizes the main purposes of the university secretaries and the list of pre-defined operations. It is worth to note that if the available application domains do not match the users requirements, users have the possibility of defining new application domains or importing an application domain from previous applications.

## *3.2 Step 2: How*

Once the application domain has been specified, the step *How* requires users to define the business services and processes that the application should manage. The proposed solution simplifies the specification of this information through a visual language. In particular, the initial user interface of this step allows users define the services to be embedded in the application by selecting them from a list of pre-defined services or by visually creating them from scratch. As an example, Fig. 3 shows the list of available services for the Student Secretary application. In

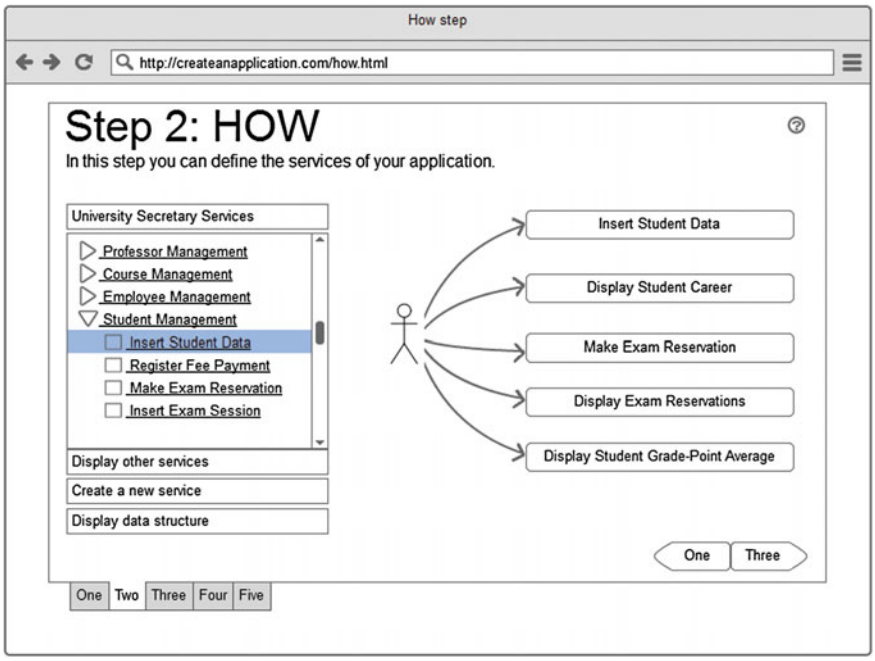


**Fig. 3** Selection of the predefined services

particular, the user interface highlights the services for the management of student information. The predefined services can be updated by end-users in order to customize the service-workflow and/or to integrate own business processes.

Figure 4 shows the user interface for visually specifying the services to be included in the service-workflow. As it can be noticed, the user interface is composed of a central workspace, where the user can integrate the application concepts in a linked flow fashion by applying the icon operators.

A *concept* represents a reference data unit that contains all the features of an application domain entity. For example, the Student concept in Fig. 4 has associated all the characteristic elements required by the application domain, e.g., `firstname`, `lastname`, `ID number`, `degree`, `faculty`. The *icon operators* enable the specification of how the application service must work (service-workflow) by linking concepts to operations. The available operators are:

- *Input* It enables the definition of input fields that users have to fill in order to trigger the associated service-workflow. This operator together with the input field names is also used to define the type of each field.
- *Store data* It enables the specification the data to be permanently stored and to define how the concepts coming from the input flow are related.
- *Request data* It enables the request of permanent data from the previous concepts in the flow. It also permits to define conditions on the request, similarly to the WHERE clause in SQL.

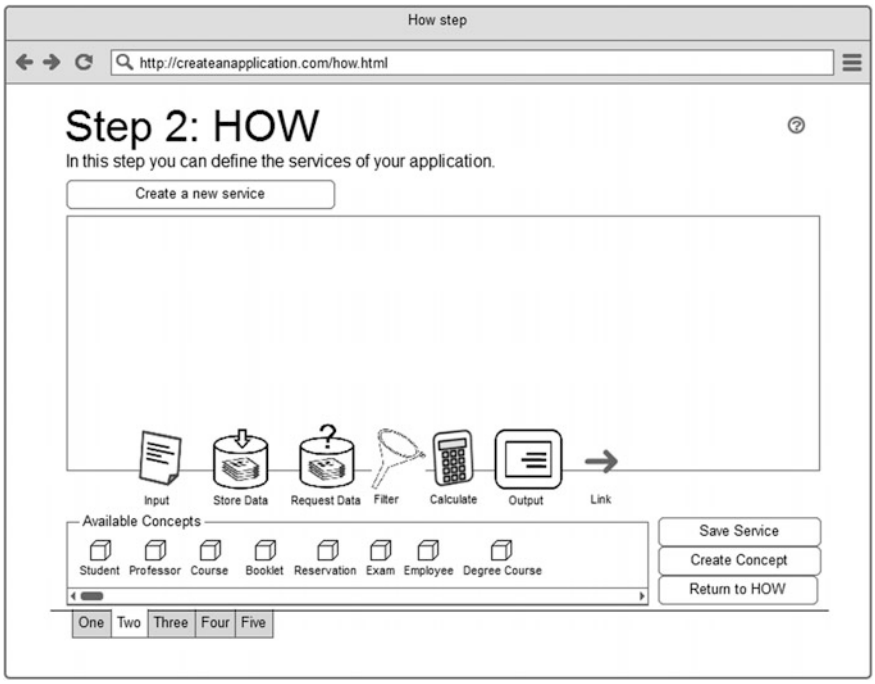


**Fig. 4** Editor for creating a new services

- *Filter* It enables the specification of input flow elements to be carried on into the composition process.
- *Calculate* It enables, where possible, the definition of operations on the input flow elements through operators or functions.
- *Output* It enables the specification of the fields to be displayed as output.
- *Link* It enables the definition of the flow that the application service has to follow.

The user interface allows users to apply the selected icon operators and concepts and to add specific details. The latter represent metadata useful for the automatic generation of the application. Figure 5 shows an example of service created for the Student Secretary application. The diagram defines the service-workflow for the exam reservation service. The input form receives information on the student and the exam sessions, and requests user to select the session to be reserved. Once these data are submitted, the student and exam session information are saved in a reservation concept, which is then permanently saved.

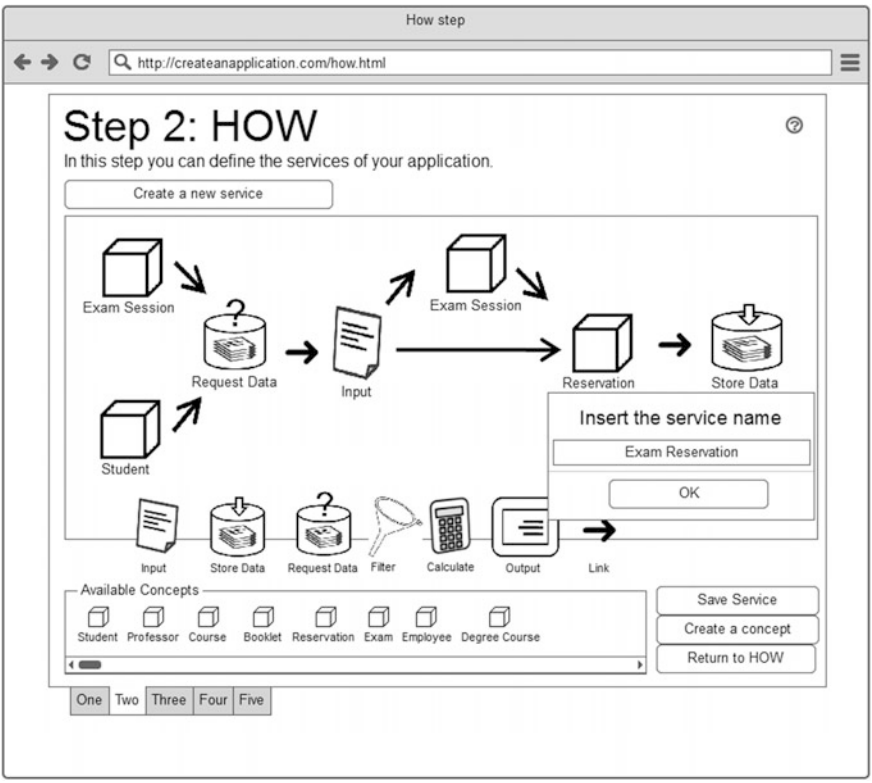


**Fig. 5** Visual specification of the "Exam reservation" service

## *3.3 Step 3: Who*

The third step of the proposed development process, named *Who*, allows the specification of application roles and of service restrictions, that is, the specification of access control mechanisms. This task is very relevant for applications running on the Web, since they can be attacked by malicious users willing to access the system and the underlying database.

Since the end-user is a domain developer, s/he strongly knows "who can make what", hence s/he only needs tools simplifying the management of this task. In the third step of the proposed process, end-users can implement access control mechanisms by using the role-based access control (RBAC) model [8]. Figure 6 shows the user interface supporting this activity. Here, end-users can define the roles involved in the application by associating a different color to each of them. Moreover, "padlock" and "key" colored icons are associated to the roles in order to represent role restrictions and access concepts, respectively. The latter are used to identify possible restrictions on the services. As shown in Fig. 7, the user is able

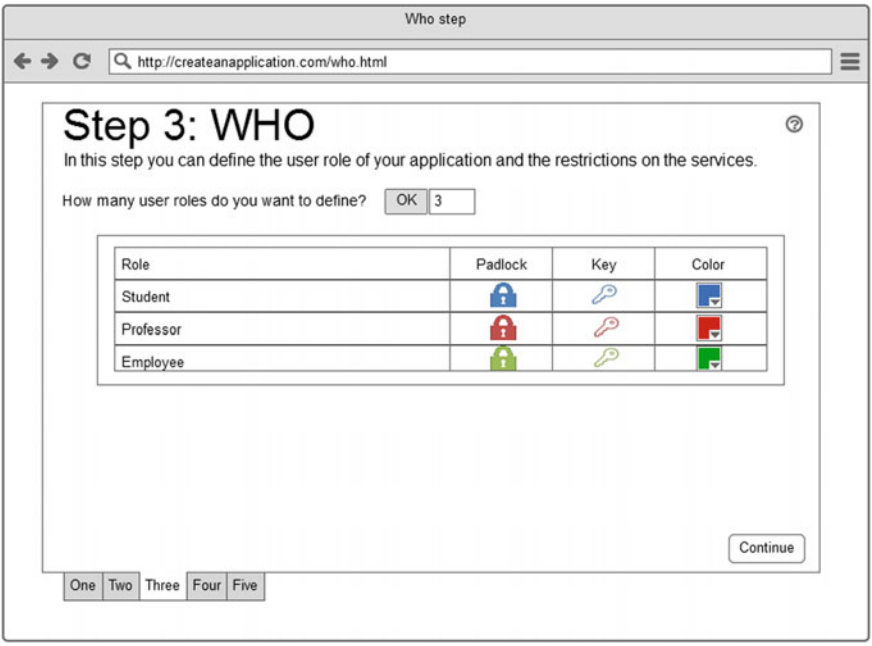


**Fig. 6** The definition of user roles


**Fig. 7** Specification of access policies for the available services

to specify the restrictions by dragging the colored padlocks on the services defined in the previous step.

### 3.4 Step 4: Where

The fourth step, named *Where*, enables the end-user to visually organize the application contents and services. The user interface shown in Fig. 8 contains a workspace where the user can define the page flow. A page can be built from scratch or it can be derived through the inclusion of a service. In the last case, the page content is initialized with the elements defined in the service-workflow. A click on a page opens a new interface for content management. As an example, Fig. 9 shows the user interface for specifying the content of web pages. An end-user can select the graphical widgets from the left bar and organize them on the right workspace.

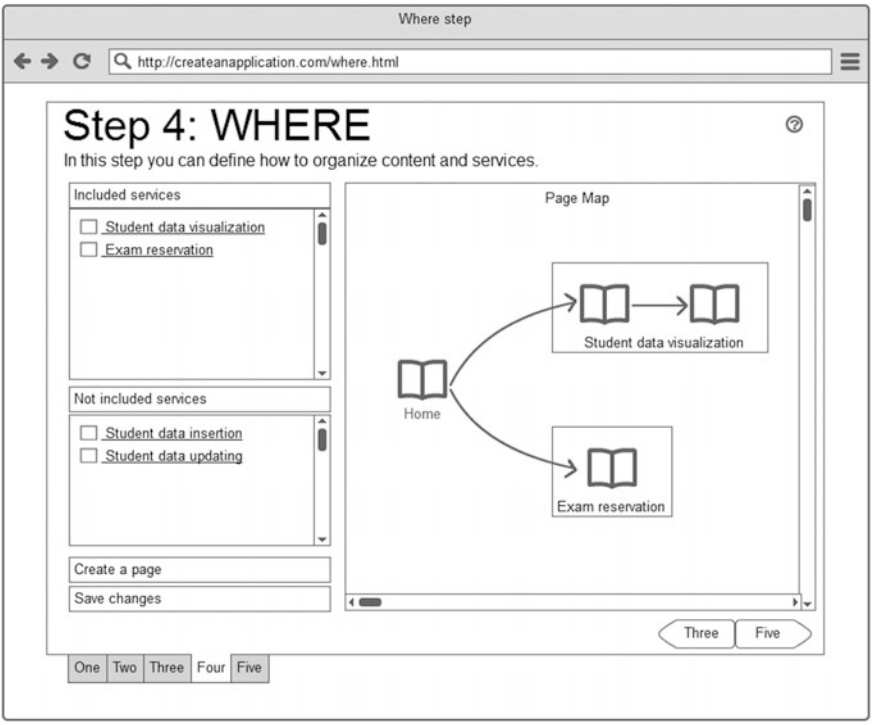


**Fig. 8** Creation of pages and their flow

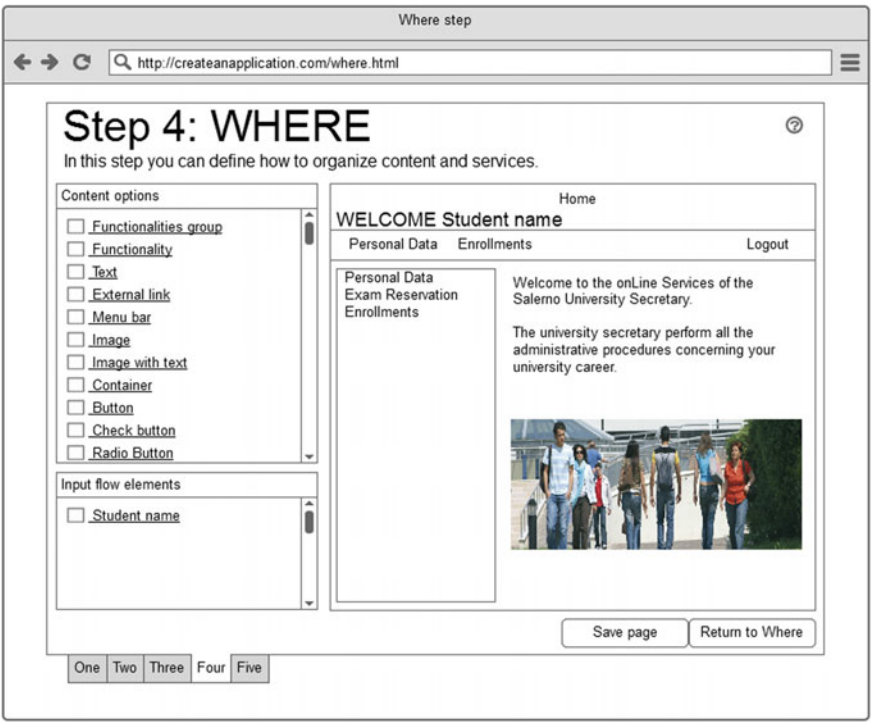


**Fig. 9** Content specification for a web page

### *3.5 Step 5: When You Want*

The last step, *When you want*, enables the end-user to perform final checks before generating the application (see Fig. 10). In particular, it is possible to:

- visualize the defined services;
- visualize the application data structure;
- visualize the role-based permissions;
- visualize the page map;
- visualize a demo of the application;
- generate the application.

## 4 Conclusion and Future Work

The EUD concept for Web application development has been somehow conceived as the possibility for a person of building do-it-yourself (DIY) objects. Expert application developers will continue perform low level programming tasks in order

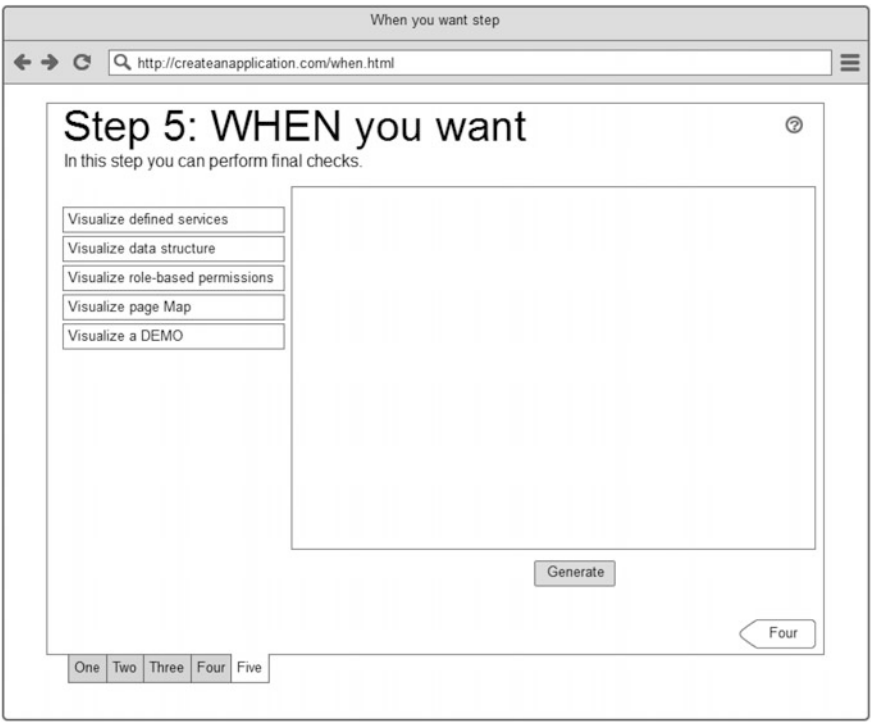


**Fig. 10** User interface for checking and generating the Web application

to satisfy customers willing to have qualitatively excellent objects, with proper attention to details. However, end-users often want to build their own objects, in order to satisfy their tastes, to bring their creativity, and to save costs.

Industry has engaged a task force to make the DIY become a reality; in fact, they have created both tools and semi-finished products to simplify the development work, and to let end-users build their applications through simple compositions. Analogous efforts have been produced by the research community.

In this paper, we propose a wizard-based EUDWeb development process, driving end-users step-by-step towards the construction of the application. In addition, the proposed process enables end-users to manage access control mechanisms, which are mandatory for Web applications, and are particularly complex to deal with.

We are currently implementing tools supporting the proposed process, in order to analyze and empirically evaluate if it is easy to use. Moreover, we would also like to define a visual language for the specification of the application domain, i.e., the *What* step.

## References


1. Albinola, M., Baresi, L., Carcano, M., Guinea, S.: Mashlight: a lightweight mashup framework for everyone. In: Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (2009)
2. Cabitza, F., Fogli, D., Piccinno, A.: Cultivating a culture of participation for the co-evolution of users and systems. In: International Workshop on Cultures of Participation in the Digital Age (2010)
3. Cabitza, F., Fogli, D., Piccinno, A.: "each to his own": distinguishing activities, roles and artifacts in EUD practices. In: Smart Organizations and Smart Artifacts, pp. 193–205. Springer (2014)
4. Cappiello, C., Daniel, F., Matera, M., Picozzi, M., Weiss, M.: Enabling end user development through mashups: requirements, abstractions and innovation toolkits. In: End-User Development, pp. 9–24. Springer (2011)
5. Caruccio, L., Deufemia, V., D'Souza, C., Ginige, A., Polese, G.: Supporting access control within a mockup-based EUDWeb. In: Proceedings of 7th International Symposium on Visual Information Communication and Interaction (VINCI), pp. 88–97 (2014)
6. Costabile, M.F., Fogli, D., Mussio, P., Piccinno, A.: End-user development: the software shaping workshop approach. In: End User Development, pp. 183–205. Springer (2006)
7. Deufemia, V., D'Souza, C., Ginige, A.: Visually modelling data intensive web applications to assist end-user development. In: Proceedings of 6th International Symposium on Visual Information Communication and Interaction (VINCI), pp. 17–26 (2013)
8. Ferraiolo, D.F., Kuhn, R.D., Chandramouli, R.: Role-Based Access Control. Artech House Inc., Norwood (2007)
9. FileMaker, I.: File maker pro 13. http://info.filemaker.com/GL_it-IT_FileMaker_Overview_information.html. Accessed 27 Feb 2015
10. Fischer, G.: Seeding, evolutionary growth and reseeding: constructing, capturing and evolving knowledge in domain-oriented design environments. Autom. Softw. Eng. **5**(4), 447–464 (1998)
11. Fischer, G.: End-user development and meta-design: foundations for cultures of participation. In: End-User Development, pp. 3–14. Springer (2009)
12. Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A.G., Mehandjiev, N.: Meta-design: a manifesto for end-user development. Commun. ACM **47**(9), 33–37 (2004)
13. Fischer, G., Nakakoji, K., Ye, Y.: Metadesign: guidelines for supporting domain experts in software development. IEEE Softw. **26**(5), 37–44 (2009)
14. Fischer, G., Scharff, E.: Meta-design: design for designers. In: Proceedings of the 3rd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques, pp. 396–405. ACM (2000)
15. Ginige, A., De Silva, B.: CBEADS©: a framework to support meta-design paradigm. In: Universal Access in Human Computer Interaction. Coping with Diversity, pp. 107–116. Springer (2007)
16. Giordano, M., Polese, G.: Visual computer-managed security: a framework for developing access control in enterprise applications. IEEE Softw. **30**(5), 62–69 (2013)
17. Google: Google maps Editor. http://www.google.it/mapmaker. Accessed 27 Feb 2015
18. Lieberman, H., Paternò, F., Klann, M.,Wulf, V.: End-User Development: An Emerging Paradigm. Springer (2006)
19. Lizcano, D., Alonso, F., Soriano, J., Lopez, G.: A web-centred approach to end-user software engineering. ACM Trans. Softw. Eng. Methodol. (TOSEM) **22**(4), 36 (2013)
20. Microsoft: Visual studio express 2013 for web. http://www.asp.net/vwd. Accessed 27 Feb 2015
21. Microsoft: Visual studio lightswitch. http://msdn.microsoft.com/it-it/vstudio/lightswitch.aspx. Accessed 27 Feb 2015

22. Nestler, T., Namoun, A., Schill, A.: End-user development of service-based interactive web applications at the presentation layer. In: Proceedings of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems, pp. 197–206. ACM (2011)
23. Open Source Matters, I.: Joomla. http://www.joomla.org/. Accessed 27 Feb 2015
24. Pesce, M.: Gabama. http://www.gabama.com/. Accessed 27 Feb 2015
25. Rode, J., Howarth, J., Pérez-Quinones, M.A., Rosson, M.B.: An end-user development perspective on state-of-the-art web development tools. Technical Report TR-05-03, Virginia Tech Computer Science (2005)
26. Rode, J., Rosson, M.B., Qui, Pérez-Quinones, M.A., et al.: End user development of web applications. In: End User Development, pp. 161–182. Springer (2006)
27. Wang, G., Yang, S., Han, Y.: Mashroom: End-user mashup programming using nested tables. In: Proceedings of the 18th International Conference on World Wide Web, pp. 861–870. ACM (2009)
28. Wong, J., Hong, J.I.: Making mashups with Marmite: towards end-user programming for the web. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1435–1444. ACM (2007)
29. Yahoo Inc., Y.: Yahoo's pipes. https://pipes.yahoo.com. Accessed 27 Feb 2015
30. Yes Software: Code charge studio 5. http://www.yessoftware.com/. Accessed 27 Feb 2015
31. Yue, K.B.: Experience on mashup development with end user programming environment. J. Inf. Syst. Educ. **21**(1), 111 (2010)