Declarative Specification of Web Applications exploiting Web Services and Workflows

Marco Brambilla, Stefano Ceri, Sara Comai, Marco Dario, Piero Fraternali, Ioana Manolescu*
DEI, Politecnico di Milano, Italy / * INRIA Futurs, France
mbrambil | ceri | comai | fraterna@ elet.polimi.it, ioana.manolescu@inria.fr

ABSTRACT

This demo presents an extension of a declarative language for specifying data-intensive Web applications. We demonstrate a scenario extracted from a real-life application, the Web portal of a computer manufacturer, including interactions with third-party service providers and enabling distributors to participate in well-defined business processes. The crucial advantage of our framework is the high-level modeling of a complex Web application, extended with Web service and workflow capabilities. The application is automatically verified for correctness and the code is automatically generated and deployed.

Keywords

Web, Workflow, Web services, application design.

1. INTRODUCTION

This demo shows the capabilities of high-level modeling languages in describing complex Web applications. For this purpose, we present a set of new primitives and metadata, which extend a traditional Web modeling language called WebML [3, www.webml.org]. A WebML site definition consists of a data schema structured in Entities and Relationships, and a hypertext describing the content of the site in terms of the underlying data model. In the hypertext model, Web pages are defined as sets of interconnected units. A unit is an elementary piece of content corresponding to a parametric query over the underlying E-R data model. Units, pages and their correlations are specified using WebML's library of graphical primitives.

The extensions to WebML proposed in this paper describe: (i) business processes and activity flows, described by generic workflow specifications (Section 2); (ii) complex interactions between HTTP applications and remote services (Section 3); (iii) publishing of more complex Web services from existing ones and wrapping of applications as Web services (Section 3); (iv) wrapping of heterogeneous data sources through Web services (Section 4). These extensions are shown on a real industrial case: a business portal developed for Acer Europe, providing services to the distribution channel (resellers and distributors).

Main actors of the portal are: resellers, who browse among product descriptions, distributors' availability, and store locations (including maps and driving directions, provided by third party companies as Web services); distributors, who can submit sales reports and ask for stock refunding, following complex approval processes; and internal Acer employees, in charge of evaluating and verifying requests and submissions from distributors.

Therefore, the Acer portal is a very complex Web application, incorporating traditional navigation, workflow-based activities, remote services invocation and heterogeneous data sources access. In this demo we present these aspects, as well as a sketch of the compile-time and run-time architecture, concluding with results discussions.

2. WORKFLOW ENHANCEMENTS

While the Web consolidates as a ubiquitous application delivery platform, Web applications need to cover new requirements, like the capability of managing complex workflows spanning multiple users and organizations. Our proposal consists of a set of new conceptual primitives that leverage workflow capabilities for the Web context [1], starting from terminology defined by the Workflow Management Coalition [www.wfmc.org].

Starting from a typical representation of a process through a workflow schema, we model it in the WebML hypertext through some new WebML units. Moreover, a metadata structure is put in place at the purpose of describing processes and tracking their enactment.

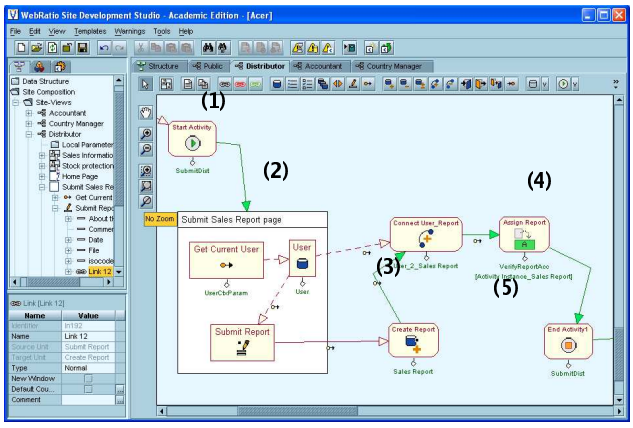

Figure 2.1: Hypertext diagram for Report Submission activity

The resulting hypertext is described by usual WebML schemas, enriched by the new workflow-oriented units. Figure 1 shows the hypertext fragment that specifies the navigation of the user for the submission of a sales report; by navigating a link, he starts the submission activity by means of a *Start activity operation* (1) and he is brought to a page containing a form for the submission (2); when he submit his input, the new report is created and connected to the current user (3), then the report is assigned by

the *Assign unit* to the next activity to be executed (4); at this point the activity is closed by the *End activity* unit (5).

3. WEB SERVICES INTERACTIONS

One of the main requirements of the Acer portal is the retrieval of information from third party remote services. To grant remote services interaction capabilities, we added some new WebML primitives for supporting Web services enactments, modeling both WSDL Web service operations and data conversion operations, required to grant correct data marshalling between the local and remote information format.

(9) (7) (10)

(1)

(6)

Figure 3.1: Hypertext model with Web service calls

Again, the Web application schemas, exemplified in Figure 3, include the new primitives together with hypertext pages and links. The example shows how to get a map of a geographical area, enriched with resellers positioning. The user has to specify his own country and location (1); when he submits the data, a chain of Web service calls is triggered: the first call retrieves all the addresses matching the request (2), the second one asks for the coordinates of each match (3); then an XSL transformation is applied (4), to convert the received message in a known format, and finally an *XML-in unit* (5) stores the received information in the local database. If any error occurs, the Error page (6) is shown, otherwise the user is lead to the Success page (7), where he can choose one of the found locations from an index. If he chooses a location, another *Request-Response unit* (8) asks for a map of the area, enriched with the positions of the Acer resellers (provided by the Partners unit (9)). The response is stored into the application database (10), and then the map is shown (11).

(11)

(2) (3) (4) (5)

Acer needs also to provide Web access to the repository of marketing materials, whose information are stored into a native XML repository. In the application, queries are modeled and implemented through calls to the Web service interface of an XML database, called Xyleme Zone Server.

4. SYSTEM ARCHITECTURE

The architecture of our solution is an extension of the WebRatio [www.webratio.com] architecture presented in [2], which supports the WebML model. At compile time (Figure 4, top), each graphical site specification is used to generate, for each unit, link and page included in the definition, an XML descriptor containing the information needed to instantiate it in the functional Web site, and a collection of JSP page templates. At runtime (lower part of Figure 4) the site is instantiated and "run". Clients issue requests via a browser to the Web server, which in turn calls the JSP engine; the generated pages include customized JSP tags that define the placement of the WebML units into the JSP pages. The content required by the user is assembled by the WebML runtime component, using the descriptors and the data sources in the Data Layer.

We have recently extended the runtime to support the exchange of Web service messages: we added a SOAP sender/listener, and a Conversation Manager for conducting long-running interactions with a service. Finally, we extended the site generation algorithm to produce descriptors for each of the new primitives, which will be used in the generated JSP pages. These extensions are depicted in bold lines and fonts in Figure 4.

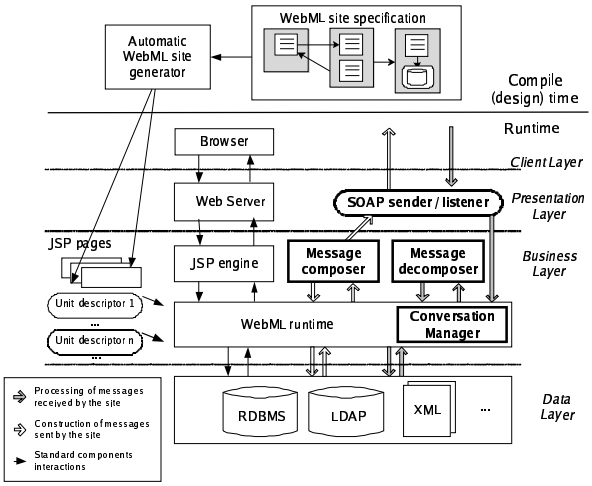

Figure 4.1: Architecture of WebML Web applications.

5. CONCLUSIONS

The proposed demonstration illustrates a complex Web application, including process management primitives, calls to Web services, and integration of heterogeneous data sources. Application developers can concentrate only on the requirements of the application and on its high-level design, because code is automatically generated by the CASE tool and correctness is automatically verified. An important benefit of our approach stands in the orthogonality of the solutions. In particular, Web services and Workflows can be mixed at will in the application design, describing workflow activities that include remote service calls, and vice-versa, new Web services that are composition of existing services and workflow activities.

ACKNOWLEDGMENT. This work is part of the WebSI project, supported by the EU. We thank G. Morbello and E. Tosetti for their co-operation in building the application.

6. REFERENCES

1. M. Brambilla, S. Ceri, S. Comai, P. Fraternali, I.Manolescu, Specification and design of workflow-driven hypertexts, *JWE (Journal of Web Engineering)*, 1(2) April, 2003.
2. S. Ceri, P.Fraternali, R. Acerbis, A. Bongio, et al., Architectural Issues and Solutions in the Development of Data-Intensive Web Applications, *CIDR 2003*, Asilomar.
3. S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera, *Designing Data-Intensive Web Applications*, Morgan-Kaufmann, December, 2002.