Towards the Integration of Data and Functionality in Web Applications. A Model Driven Approach [1]

Victoria Torres, Ricardo Quintero, Marta Ruiz, Vicente Pelechano

Department of Information Systems and Computation
Technical University of Valencia
Camino de Vera s/n, 46022
Valencia, Spain
{vtorres, iscrquinter, mruiz, pele}@dsic.upv.es

Abstract. Web Engineering methods have evolved during the last years to deal with new requirements that have arisen as a natural evolution of Web Applications. However, a new challenge that has been little handled by some of these methods is the data and functionality integration with third-party systems. Relying on the basic principles of the OMG Model Driven Architecture, we present a solution to achieve this integration at the conceptual modelling level. Therefore, solving the integration problem at this level of abstraction let us (1) give a general solution that is not tied to any particular implementation technology and (2) to handle external components at the conceptual modelling phase. To achieve this goal we present a new model that characterizes external functionality at a high level of abstraction.

1 Introduction

In the last years it has been done a very hard work in the Web Engineering field. As a result, a set of proposals have been developed to tackle with features such as navigation. The most outstanding proposals developed are WebML [3], OOWS [2], OOHDM [6], UWE [7], WSDM [5] or OO-H [8] among others. These proposals were conceived at a first stage to provide a methodological guide for the development of Web Applications. Therefore, the main functionality provided by the systems developed under any of these proposals was the retrieval and the maintenance of data stored in their persistent layer. In the following years, these approaches have been extended to tackle with new requirements that have arisen as a natural evolution of the Web. However, these extensions have always been addressed for the construction of isolated artifacts where little methodological solution has been provided to achieve data and functionality integration from different sources.

Web Engineering Methods, such as WebML, have addressed directly the integration of Web Services [4]. To achieve this integration they have included new

1 The work reported in this paper has been funded by the MEC under grant TIN2004-03534 and cofinanced by FEDER

Proceedings of the CAiSE'05 Forum - O. Belo, J. Eder, J. Falcão e Cunha, O. Pastor (Eds.)

WebML constructs that characterize each kind of Web Service operations. However, we think that this solution is too dependent to the implementation technology, what entails to provide a different integration solution for each technology. Moreover, it mixes abstraction units from different levels.

The amount of data and functionality available in the Web make us to think in the benefits that we can bring to Web Applications. Now we can provide Web users not only with the data and functionality maintained by our own system. Therefore, we are in conditions to create value added Web Applications. However, the functionality available in the Web is given in terms of an underlying technology, and this fact makes difficult the integration at the modelling level.

The main contribution of this work is to present a MDA [1] based solution to integrate third-party data and functionality in a technology independent way. To achieve this goal we introduce a new model (the Services Model) to the set of conceptual models that are commonly used by Web Engineering Methods. The aim of this new model is to characterize in the most general way the external functionality that is going to be integrated in the Web Application. The work also describes how to integrate this new model with the rest of models specified during the development process.

The rest of the paper is structured as follows. Section 2 presents the Services Model. In section 3 we introduce the mechanisms used to integrate the Services Model previously introduced with the existing Web Engineering models. Section 4 illustrates through the use of an example how to achieve data and functional integration with third-party systems. Finally, Section 5 draws some conclusions and outlines some further work.

2 The Services Model

For the design of the Services Model we have followed the MDA principles aimed to provide portability, interoperability and reusability through architectural separation of concerns. This practice allows us to define integration independently from technology. The Services Model is made up of a set of operations with their input and output parameters. As a result, we have defined the Services Model at two different levels, at the PIM level and at the PSM level. The Services Model defined at the PIM level let us to characterize in a *general way* the functionality imported from third-party systems. However, the Services Model at the PSM level is intended to characterize the peculiarities of each specific technology. As a final goal what we want to achieve is the automatic code generation. So, we have to keep specific information from each technology. For this reason, we have to define as many Services Model at the PSM level as technologies we want to integrate with our system. This separation of concerns let us to integrate coherently external functionality (defined in general terms in the Services Model at the PIM level) with the native elements from the rest of the PIM models (structural, behavioural and navigational models). Due to space constraints we provide here just a general idea of the proposal, avoiding the inclusion of specific details. Fig. 1 shows that the integration with

external functionality is done through the Service Model defined at the PIM level, instead of dealing directly with the PSM models. Moreover, it shows that we need a new Service Model for each technology that we want to support integration.

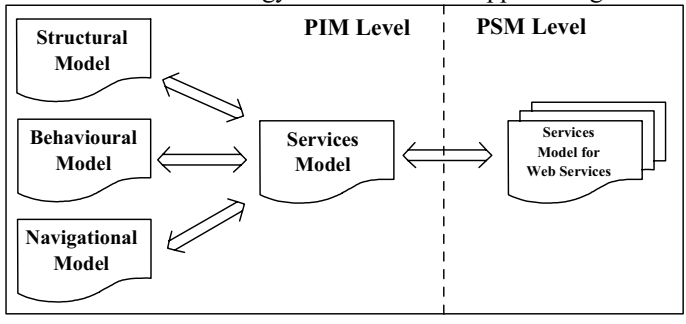

Fig. 1 Integration through PIM models

Once we have external functionality defined in our system we have to integrate it with our native modelling elements. In next section we identify the mechanisms to achieve integration between models.

3 Mechanisms for the Integration of the Services Model

Depending on the use that we want to do of the imported functionality, we distinguish three different ways to achieve the *Services Model* integration through:

- the *structural model*,
- the *behavioural model* and
- the *navigational model*.

The first mechanism is to associate external functionality to our structural model. This association let us to enrich the definition of a class (defined in our structural model) adding new attributes or operations not modelled initially in our system. When we detect that the results provided by some external functionality can complement the data contained in our structural model we create a new "*derived"* attribute in the class to be enriched. With the word "*derived"* we mean an attribute which value is not kept in the persistent layer. Moreover, we have to provide the "*formula"* that feeds this new attribute. This formula will be based on the definition of the external functionality. However, we also can find external services which provide extra functional value to our classes. In these cases, we should wrap the external functionality into a new operation created in the enriched class.

The second mechanism to associate external functionality is by means of the behavioural model. This association is made when we want to include some external functionality in the definition of a complex operation (an operation defined as a sequence of steps/sub-operations). We can now compose operations based on internal as well as external operations.

Finally, the third mechanism is to associate external functionality directly to the Navigational Model. This kind of association is made when we want to include new

data or functionality that it is not "directly" related to any of the elements modelled originally in our system.

Following this approach now the use of external functionality is almost transparent for the web designer. Note however, that we have to keep the information that allows us to know how to invoke imported functionality (via SOAP or REST in the case of Web Services). At modelling level we do not need to know these details, but when generating the final code of the Web Application this information will be necessary.

In next section, by means of a Library Web Application, we illustrate each one of the integration mechanisms introduced in the previous paragraphs.

4 A Real Scenario: The Library Web Application

The Library Web Application provides users with information (title, author, publication year, etc.) and functionality (to make a booking or ask for a buying) related mainly to books. To complement the data and functionality modelled and handled by our library system we are going to integrate functionality published by external providers such as an e-shop and another on-line library system.

Based on the initial requirements of the Web Application, we have defined the class *Book* with the following attributes: *title*, *publication year*, *edition*, *language* and *issn*. Through the use of external services we could add to this information the *price* or the *list of e-shops* which sell a specific book. In case a book is not available in the library at this moment, we also could provide the facility to reserve this specific book in another on-line library system.

In the following subsections we provide details of the examples that illustrate how to accomplish this integration.

4.1 Integrating the SM with the Structural Model via Attributes

We are going to create the "derived" attribute *price* in the class *Book*. This attribute will be fed by the result provided by the imported service *itemSearch*. We must define the formula that will provide the value to the *price* attribute. Sometimes this result provides more information than we need. To navigate through the returned data structure we use the dot syntax.

```
price = (itemSearch(title)).price
```

The definition of this new attribute in the class diagram allow us to include it in the navigational model in an equivalent way as any other attribute of the class.

4.2 Integrating the SM with the Structural Model via “Simple” Functionality

To include external functionality to the class *Book* we are going to create a new operation. Once the external operation is wrapped we can use it equally as the rest of internal operations. Note that the interface of the wrapping operation must be coherent

with the wrapped functionality. We are going to include the operation *bookReservation* in the class *book* to wrap the *bookReservation* operation imported in the Services Model. Fig. 2 shows graphically this integration.

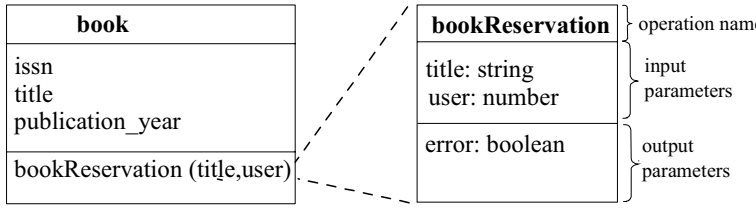

Fig. 2 (left) an excerpt of the class book definition (from the class diagram) and (right) bookReservation definition (from the Services Model at the PIM level)

4.3 Integrating the SM into the Behavioral Model via "Complex" Operations

In the definition of an operation such as *searchBook* we want to search not only in our catalog, moreover, we want to search in other external libraries. For this purpose, we specify the sequence of services to execute when this operation is called. This specification can be done through an activity diagram as depicted in fig. 3. If the specified book has not been found in our catalog we complement the search with an external service provided by another library (the UPV library).

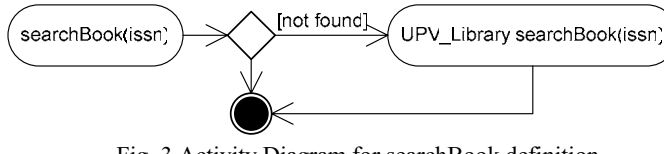

Fig. 3 Activity Diagram for searchBook definition

4.4 Integrating the SM with the Navigational Model

The Navigational model defines the navigational structure of a Web application. This structure is defined usually by means of a graph based representation technique, where graph nodes represent web pages and arcs among nodes represent links. Graph nodes are made up of a set of views over the structural model; these views define the information that is shown in each web page. However, following the presented approach, we allow the construction of these views also from third-party systems. For example, if we want to include in a web page information such as the *most purchased books* from Amazon we would include it by adding a view in the node from the corresponding Amazon service. Fig. 4 shows a web page that illustrates this case.

Fig. 4 An excerpt from the library web page including the best seller books imported from Amazon

5 Conclusions and Further Work

In this work we have presented a high level integration approach for Web Applications. This integration is achieved through a set of mechanisms that associate at the PIM level the Services Model with the rest of models.

As further work we have planned to take these ideas to practice by providing a prototype. Moreover, once this proposal had been completely defined we want to study how to achieve data integration through the use of Ontologies and Semantic Web representations.

References

1. OMG. *Model Driven Architecture. A Technical Perspective*. Object Management Group, January 2001. OMG document ab/2001-01-01
2. J. Fons, V. Pelechano, M. Albert and O. Pastor. “Development of Web Applications from Web Enhanced Conceptual Schemas”. Proc. Of the International Conference on Conceptual Modelling, 22nd Edition, ER'03, pp 232-245. Chicago, EE.UU, 13 - 16 October 2003
3. S. Ceri, P. Fraternali and A. Bongio, “Web Modeling Language (WebML): a Modeling Language for Designing Web Sites”. In WWW9, Vol. 33 (1-6), pp 137-157. Computer Networks, 2000
4. M. Brambilla, S. Ceri, S. Comai, P. Fraternali and I. Manolescu: "Model-driven Development of Web Services and Hypertext Applications", SCI2003, Orlando, Florida, July 2003
5. O. De Troyer and C. Leune. “WSDM: A user-centered design method for Web sites”. In Proc. of the 7th International World Wide Web Conference, 1998
6. D. Schwabe and G. Rossi, “An Object Oriented Approach to Web-Based Application Design”, Theory and Practice of Object System 4(4), 1998. Wiley and Sons, New York, ISSN 1074-3224)
7. N. Koch and M. Wirsing. “Software Engineering for Adaptive Hypermedia Applications”. In: 3rd Workshop on Adaptive Hypertext and Hypermedia, 2001
8. J. Gómez, C. Cachero and O. Pastor. “Extending a Conceptual Modeling Approach to Web Application Design”. Proc. Conference on Advanced Information Systems Engineering, CAiSE’00, Springer- Verlag, LNCS 1789, pp. 79-93, 2000