Top Down Versus Bottom Up in Service-Oriented Integration: An MDA-Based Solution for Minimizing Technology Coupling

Theo Dirk Meijler, Gert Kruithof, and Nick van Beest

Information Systems, Faculty of Management and Organization
{t.d.meijler, g.h.kruithof}@rug.nl,
n.r.t.p.van.beest@student.rug.nl

Abstract. Service-oriented integration typically combines top-down development with bottom-up reverse engineering. Top-down development starts from requirements and ends with implementation. Bottom-up reverse engineering starts from available components and data sources. Often, the integrating business processes are directly linked to the reverse-engineered web services, resulting in a high degree of technology coupling. This in turn leads to a low level of maintainability and low reusability. The Model-Driven Architecture (MDA) provides an approach aimed at achieving technology independency through full top-down development. However, this approach does not handle the bottom-up reverse-engineered components very well. In this paper, an approach is introduced that combines top-down with bottom-up realization, while minimizing the technology coupling. This is achieved by an explicit buffer between top-down and bottom-up. "High-level" web services are derived through top-down development, whereas "Low-level" web services are reverse-engineered, and a mapping is created between the two. The approach focuses on engineering web services reversely, while retaining the advantages of the MDA with respect to platform independency, maintainability and reusability.

Topics: Business Service Modeling, Service Assembly.

Scientific Area: Service-oriented software engineering.

1 Introduction

Various approaches have been proposed for developing enterprise information systems (EISs) in a top-down manner [1],[2]. An important characteristic of a pure top-down development approach is that it starts from the desired business situation (rather than from the current situation) and finally results in the implementation of a new system. In the approaches adopted by [1],[2] the Model-Driven Architecture (MDA) plays an important role. Models serve as a better form of communication with the domain expert than code does, and so the domain expert can be given a bigger role in the development process. A Service-Oriented Architecture (SOA) [3] can also play an important role in the top-down development of reconfigurable business processes. A bottom-up approach, in contrast, starts from a system's existing components

A. Dan and W. Lamersdorf (Eds.): ICSOC 2006, LNCS 4294, pp. 484 – 489, 2006.

(including data sources). The SOA enables publishing of legacy components into the standard form of web services. An effective combination of top-down and bottom-up development is clearly required because enterprises both need to reuse their legacy components and work toward a desired business situation.

According to the standard SOA integration approach top-down development results in process realizations. These processes are directly linked to the web services that result from the bottom-up reverse engineering of legacy components. Changes in the relatively autonomous legacy components have an impact on the business processes. Thus, due to this form of technology coupling, the process model no longer reflects the "real" business process and becomes less understandable to the domain expert, making it more difficult for him/her to maintain. So, linking top-down and bottom-up development in this way undermines flexibility.

This paper presents an alternative approach to the linking of top-down and bottom-up development in a SOA on the basis of which technology decoupling is achieved. This approach improves the integration of MDA and SOA, as already presented by other authors [4][5].

2 Example from the Financial Sector

Generally, banks already have several components and data sources at their disposal, and they may integrate these by adopting a SOA approach. This example represents a highly simplified process of loan provision. The main legacy component is a data source. The schema of the initial database contains two tables: client and loan (Fig. 1a). Fig. 1 also shows the bottom-up reverse-engineered web services that represent this data source.

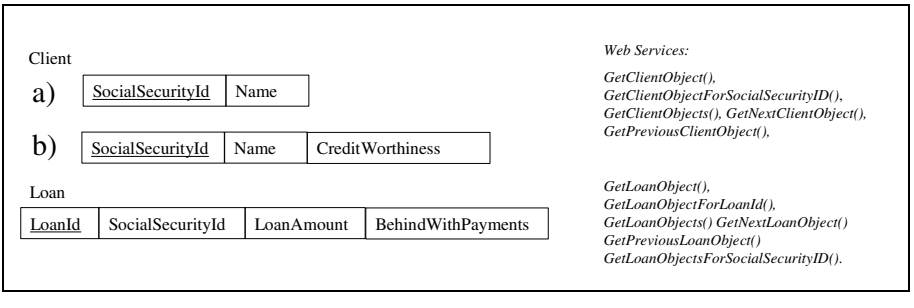

Fig. 1. Initial data structure and some web services

Fig. 2 shows a part of the business process model. The business process invokes the web service GetClientForSocialSecurityID() directly from an activity in the process. The SocialSecurityID of the Client has to be passed on to the LoanProvision subprocess, which can only continue if the client is creditworthy. The second activity in the process therefore involves the calculation of creditworthiness by iterating over all loans to assess whether in one of these loans the value of the "BehindWithPayments" property is set to true.

Now, the bank may decide to add a new "CreditWorthiness" property to the Client table (Fig. 1b). From the perspective of the business process, it is irrelevant whether creditworthiness is calculated in advance and stored as a Client attribute, or calculated at the time it is being requested in the process. However, the process will be strongly impacted by this change: the additional activity of calculating creditworthiness is no longer required.

This example shows that in a standard top-down and bottom-up integration, changes in the structure of the data sources of the bottom-up web services have an undesired impact on the structure of the business process.

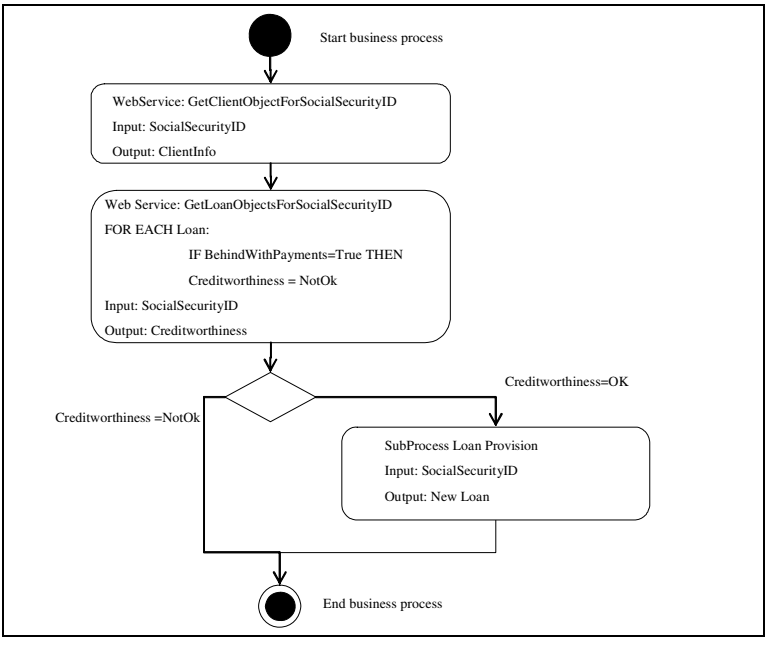

Fig. 2. Initial Business process of providing loans to clients

3 Approach

Figure 3 illustrates the solution presented in this paper. The solution itself is represented in the middle of the figure. Figure 3a, b and c are used to indicate the application of the solution to the example. The solution contains the following main components:

1. The solution applies the standard MDA subdivision between PIM (Platform Independent Model), PSM (Platform Specific Model) and Implementation. The lowest implementation part is represented by BPEL and WSDL files for the executable business process and web services that can be invoked. In a PSM a process is modeled in a SOA-specific way, indicating which activities call which web services, where these are located etc. (See [4] for a SOA specific UML

profile). A PIM may be a process model invoking interfaces that map to web services (See also [5]); Figure 3a depicts the PIM process model in our example, which abstracts from the implementation of creditworthiness.

2. New in comparison to other approaches to integrating MDA and SOA [4][5] is that the PIM not only contains a process model but also a data model. While the PIM process model serves to do top-down development of the BPEL process implementation, the PIM data model serves to do a top-down development of a set of so-called "high-level" (abbreviated: hl) web services (abbreviated: ws). Figure 3b shows the PIM data model and corresponding high-level web services. Thus, the addition of a data-model serves to enable a full-fledged *top-down* derivation of an implementation.

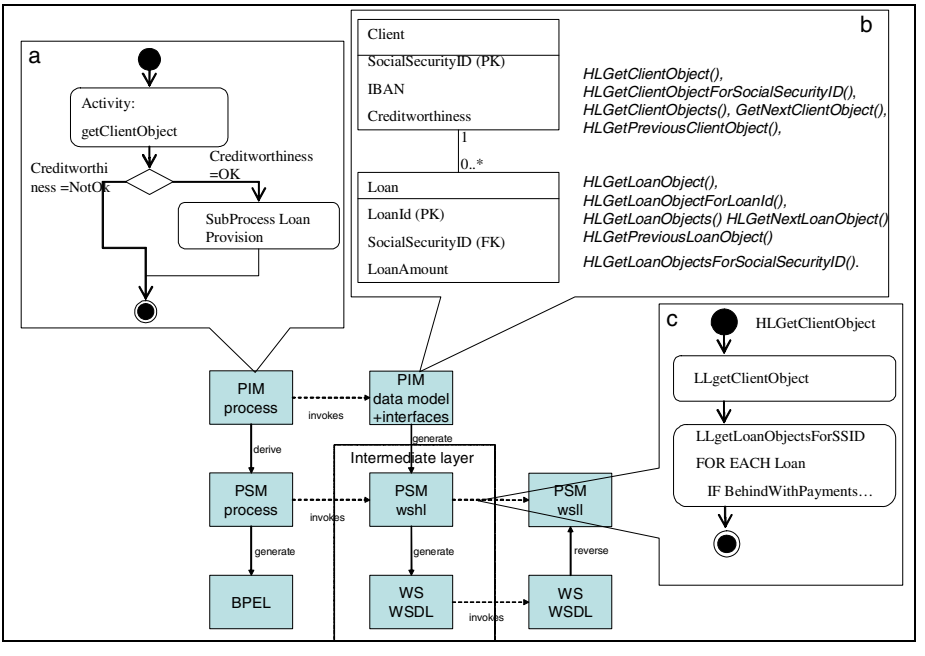

Fig. 3. Illustration of the approach

3. Another new element is the use of an intermediate decoupling layer between the top-down derived high-level web services and the web service that have been bottom-up reverse-engineered on the basis of legacy components. These web services will be called the "low-level" (abbreviated: ll) web services. The intermediate layer has been implemented by means of a simple design-time composition. In figure 3c the mapping is presented between high-level web services based on the data model of b, and the web services based on the database schema as presented in Fig. 1a. This mapping itself is presented and realized by means of a process model. The high-level low-level mapping can absorb unwanted changes from bottom-up to top-down. Thus, changing the database to the one presented in Fig. 1b does not affect the process. The mapping can, moreover, hide semantic heterogeneity aspects and integrate the results of different low-level web services.

4 Related Work and Future Research

Both technology decoupling as well as web services composition for the purpose of heightening the level of abstraction are well-known in SOA literature. Aspects such as automatic composition [6], matching the Quality of Services to what is requested and what is provided [7] and dynamic composition, e.g. on the basis of semantic information [8] have generated quite some interest.

Due to the focus on integrated EIS, the reuse of existing components and the creation of right mappings are more important than automatic composition and dynamism [8]. Furthermore, dynamic matching undermines performance, which should be avoided in this context.

Technological support for our approach is useful and has not been elaborated in this paper. This could be:

- Technological support for the development of a mapping to determine whether a used web service "fits" the requirements of the high-level web service [7].
- Simplifying the composition mechanism for these specific purposes, e.g., describing and realizing the mapping in a model-driven manner.

Moreover, methodological support is required to prevent "unconstrained" top-down development. Developers should be supported in case there is a large difference in functionality between top-down derived business models and the legacy components.

The approach presented is related to other (non-SOA) approaches to decoupling interfaces and languages. It is for example related to work on Database integration [9] where a federal database provides a newly modeled view on underlying databases that are integrated. It is also related to the so-called "data mapper" [10], and the Adapter and Bridge patterns [11], patterns for decoupling a client from an underlying implementation.

5 Conclusion

To develop an integrated Enterprise Information System both a top-down model driven development approach is needed as well as a bottom-up development. A standard top-down bottom-up linking however breaks the principle of top-down model driven development, as the process model is partly determined by the availability of components and data sources. Thus, a technology coupling is introduced that impairs the understandability and maintainability of the process model.

This paper introduces a decoupling between full-fledged top-down development following the Model-driven Architecture (MDA) and bottom-up development in the context of a SOA. Full-fledged top-down development consists of a platform independent process model as well as a data model as normal in the MDA. The data-model is used to generate so-called "high-level" web services. Thus a process model invokes these high-level web services and is no longer dependent of the availability of the bottom-up derived "low-level" web services of which the change can also not be controlled. The high-level web services are mapped using a simple design-time

composition to the bottom-up derived low-level web services. The mapping can buffer the unwanted impact of change. The simplicity of a design-time composition enables its direct application in industrial setting, this while more complex dynamic composition mechanisms are neither needed nor optimal for these purposes.

Acknowledgements

This work is framed the project "Software Mass Customization" and is paid by the Dutch ministry of Economic Affairs. We thank Cordys, one of the partners of this project for their time and giving us access to their platform. We furthermore thank Bart Orriëns and Douwe Postmus for valuable comments.

References

[1] R. Hubert Convergent Architecture, Wiley Computer Publishing, ISBN 0-471-105600 2002
[2] D.S. Frankel Model Driven Architecture Applying MDA to Enterprise Computing, OMG Press – Wiley Publishing, ISBN 0-471-31920-1 2003
[3] T. Erl. A Field Guide to Integrating XML and Web Services, Pearson Education, Publishing as Prentice Hall Technical Reference, ISBN 0-13-142898-5 2004
[4] D. Skogan, R. Groenmo, I. Solheim, Web service composition in UML Proceedings of the Eighth IEEE International Enterprise Distributed Object Computing Conference, (EDOC 2004) IEEE 2004
[5] B. Orriëns, J. Yang and M.P. Papazoglou, Model Driven Service Composition, Service-Oriented Computing - ICSOC 2003 Eds. M. E. Orlowska, S. Weerawarana, M. P. Papazoglou, J. Yang LNCS 2910 / 2003, ISBN 3-540-20681-7 2003
[6] M. Aiello, M. Papzoglou, J. Yang, M. Carman, M. Pistore, L. Serafini, P. Traverso, A request language for web-services based on planning and constraint satisfaction, Proceedings of the VLDB workshop on Technologies for E-Services, Hongkong, China, 2002
[7] J. Cardoso, A. Sheth, J. Miller, J. Arnold, K. Kochut, Quality of Service for Workflows and Web Service Processes, Journal of Web Semantics, Elsevier, Vol. 1, No. 3, pp. 281-308, Amsterdam, The Netherlands, 2004
[8] P. Rajasekaran, J. Miller, K. Verma, A. Sheth, Enhancing Web Services Description and Discovery to Facilitate Composition, International Workshop on Semantic Web Services and Web Process Composition, 2004 (Proceedings of SWSWPC 2004)
[9] H. Balsters, E.O. de Brock, An object-oriented framework for managing cooperating legacy databases; 9th International Conference Object-Oriented Information Systems; Lecture Notes in Computer Science LNCS 2817, Springer, September 2003
[10] M. Fowler, Patterns of Enterprise Application Architecture, Eddison Wesley Pearson Education, ISBN 0-321-12742-0, 2003
[11] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design patterns: elements of reusable object-oriented software, Addison-Wesley Longman Publishing Co., Inc., Boston, MA ISBN 0-201-63361-2 1995