Semi-automated Model Synchronisation in SOM

Christian Flender and Thomas Hettel

c.flender@qut.edu.au, t.hettel@qut.edu.au

Faculty of Information Technology,
Queensland University of Technology,
Brisbane, Australia.

Abstract. Model-driven engineering is at the forefront among recent attempts to information systems development. Models are gradually refined from domain specific descriptions to more concrete models closer to implementation. This is particularly relevant to the model transformation of collaborating business partners down to collaborating (web) services as they share a common interactional perspective. However, as requirements constantly evolve model layers change and so they have to be kept in sync. Model synchronisation keeps track of those changes and propagates them to other layers. This poster gives a brief introduction to model synchronisation as devised for the Semantic Object Model (SOM), a promising approach to model-driven service engineering. SOM allows for the gradual refinement of model layers through decomposition of business objects respective their interactional relationships.

1 Introduction

Model-driven development of information systems bares the indispensability of architectural frameworks. Architectures divide a complex model into several model layers and perspectives so as to reduce the amount of aspects to be considered at a point in time. Transformations do not only allow for the gradual refinement of a given layer but also hold together the whole system as a coherent architecture. Furthermore, it is the availability of Service-oriented Architecture (SOA) that increases the attractiveness of model-driven engineering as collaborative aspects gain relevance for both domain modelling (e.g. business process design) and implementation focus (e.g. compositions of web services). However, constantly changing requirements enforce changes of model layers. For instance, changes in the provision of business services (e.g. order processing) must be mirrored in terms of changes in the provision of technological services (e.g. automatic order entry). Models should be kept in sync without violating the consistency of the overall architecture. In a semi-automatic fashion, users are guided through all layers to apply dependent changes step-by-step.

The Semantic Object Model (SOM) [1] is an approach to model-driven service engineering. SOM allows for the deduction of executable process models from high-level networks of interacting business partners [3]. This poster presents the enhancement of SOM via change propagations for model synchronisation.

2 The Semantic Object Model (SOM)

The backbone of the Semantic Object Model (SOM) is an enterprise architecture as shown in Figure 1. The architecture is divided into three main layers. The enterprise plan defines the business system from an outside view in terms of its goals, objectives and strategies embedded in a broader socio-cultural context. From an inside view, the business process model implements the enterprise plan. It is specified as a system of interacting business objects which coordinate behaviour in purposefully providing and consuming services via transactions. Once a network of collaborating actors is decomposed down to a sufficient level of detail, resource assignments embody the system in terms of human actors and web-enabled software components (implementation support).

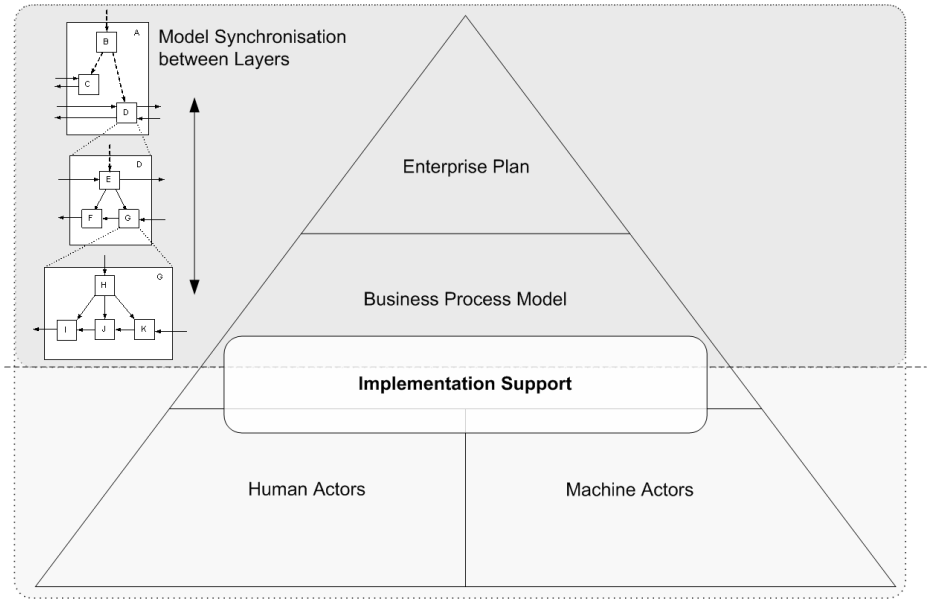

Fig. 1. The Semantic Object Model (SOM).

It is the business process model that can be further refined by decomposing interactions and objects. Interactions between objects are typed according to the coordination principles *negotiation* and *feedback-control*. Negotiation defines relationships between objects as either initiating (I), e.g. make offer, contracting (C), e.g. accept order, or enforcing (E), e.g. deliver service. Feedback-control relates objects through control transactions (R), e.g. give advice, and feedback transactions (F), e.g. state report. Model layers emerge in gradually refining networks of interacting objects. This is done by applying patterns of object-relations such as ICE, CE, RF or more complex combinations. For instance, consider the

decomposition of an object Supplier in two objects Sales and Retailer. In replacing Supplier relations between Sales and Retailer require the former to give sales advices (R) and the latter to confirm sold products (F). Following coordination principles in this manner new model layers emerge being in relationship with each other. Hence, transformations between layers constitute the trace of decomposition steps applied for their creation. However, once an architecture is modelled, requirements may change. For instance, industrial development of new products, services or markets may enforce new structures and relationships between managerial and operational actors. To avoid developing architectures anew from scratch each time requirements change, model synchronisation keeps track of changes.

3 Model Synchronisation in SOM

Consider the example of a Buyer- Supplier network. In the initial model layer as shown on the top-left side in Figure 2, both objects negotiate according to their needs. In the second model layer, Supplier was decomposed in two objects Sales

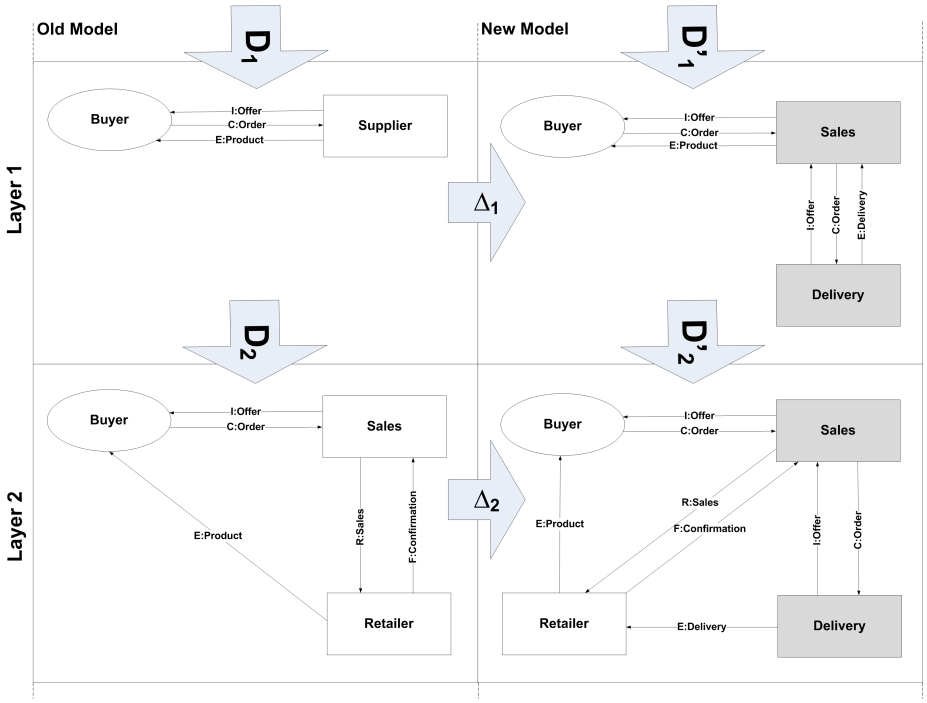

Fig. 2. Model Synchronisation in SOM.

and Retailer (see Figure 2 on the bottom-left side). The latter sells the product

to Buyer from stock according to advice from a salesman. However, in order to increase the range of products, the management decides to purchase selected items from external sources. These products must be delivered in order to resell them to buyers. On the top-right side in Figure 2 a new object Delivery is inserted according to the changed requirements. This enforces change propagations so as to keep Layer 1 and Layer 2 in sync. If one compares Layer 2 of the old model and Layer 1 of the new model, propagations result from both the decompositions which led to Layer 2 (D_2) and the the insertion of Delivery in Layer 1 (Δ_1).

We devised a complete set of change propagation algorithms keeping track of insertions, deletions and updates [2].

4 Conclusion and Future Work

The gradual refinement of model layers decreases complexity as systems are divided into manageable parts without loosing track of the whole architecture. Model-driven service engineering can unfold its full potential when architectures evolute toward sustainability. Therefore, mechanisms are needed assuring the propagation of changes on a particular layer to other layers so as to maintain consistency. SOM's transformation rules are derived from composable patterns of coordinations between objects. Having enhanced SOM's enterprise architecture with change propagations, model synchronisation introduces flexibility and accounts for evolving requirements. We are working on a SOM tool which is partly available for presentation.

References

1. O. K. Ferstl and E. J. Sinz. *Handbook on Architectures of Information Systems.*, chapter Modeling of Business Systems Using the Semantic Object Model (SOM) - A Methodological Framework. International Handbook on Information Systems. 1997.
2. C. Flender and T. Hettel. Model-driven Service Engineering with SOM. Technical report, FIT-TR-2008-02, Queensland University of Technology, Brisbane, 2008.
3. C. Flender and M. Rosemann. Service-oriented Design of an Enterprise Architecture in Home Telecare. In *18th Australasian Conference on Information Systems*, 2007.