# A Framework for the Behavior Based Integration of Business Processes

Georg Grossmann, Michael Schrefl, and Markus Stumptner

University of South Australia, Advanced Computing Research Centre, Mawson Lakes, SA 5095, Adelaide, Australia. E-mail: {cisgg, cismis, mst}@cs.unisa.edu.au. [*]

**Abstract.** We propose a meta-meta framework architecture for supporting the behavior based integration of two business processes. The meta-meta level provides basic integration operators to the system administrator to create integration options for specific domains. Based on semantic relationships between nodes of two business processes these integration options are executed and transform parts of the business processes. The outcome of the model transformation is an integrated business process. The main advantage of this framework is the support of integration on multiple levels of varying domain independence as supported by the Model Driven Architecture.
*Keywords:* Behavior based integration, business process.

## 1 Introduction

Recent research initiatives such as a special issue of *Advanced Engineering Informatics* on Enterprise Modelling and System Support [2] have pointed out the potential of enterprise systems to support integration of various functions in an organisation as well as in value chains and business networks. The implementation, however, is not without problems and failures can be costly. A variety of models, tools and techniques have been developed to address this problem by supporting the analysis of business process structure and performance.

Our contribution to this topic is a framework for behavior based integration of business processes with an underlying metaclass architecture [6]. It represents a generic approach where high-level integration operators can adapt and produce individualized integration options, i.e., groups of operations, for the integration of processes from a particular domain.

The framework uses instantiation relationships in a metaclass architecture to implement context-dependent integration patterns. The application of the integration approach is shown in Figure 1 and consists of the meta model level and the model level. The meta model provides a set of basic integration tools which are instantiated on the model level. On the meta level, three steps are undertaken to provide the necessary context for model level integration.

1. **Definition of semantic relationships**: On the meta level, the system administrator defines semantic relationships between element types of a business process.
2. **Definition of integration options**: The second task of the system administrator is the definition of integration options. Several predefined basic integration operators are available for the development of integration options.
3. **Definition of integration choices**: The last step on the meta level covers the definition of integration choices. Each relationship defined in the first step must be associated with at least one integration option created before.

[*] This research was partially supported by the Australian Research Council under Discovery Grant DP0210654 and by the CIEAM CRC Project "Integrated Reliability Management".

On the model level, 4 steps must be performed to lead from two separated models given as input to an integrated business process:

1. **Modeling of business processes**: On the model level, the software developer creates or imports two business processes which are to be integrated into one model.
2. **Set semantic relationships**: The developer subsequently inserts the semantic relationships defined in step 1 on the meta level between nodes of the two business processes.
3. **Choose integration option**: Depending on the defined integration choices, there exist either one or more integration options for each semantic relationship. In the latter case, the developer has to decide which option is the proper one for the following model transformation. At the end, each semantic relationship refers to one integration option.
4. **Model transformation**: In the last step, the developer executes the model transformations. This implies the execution of all integration options that are related to the semantic relationships identified in the second step on the model level.

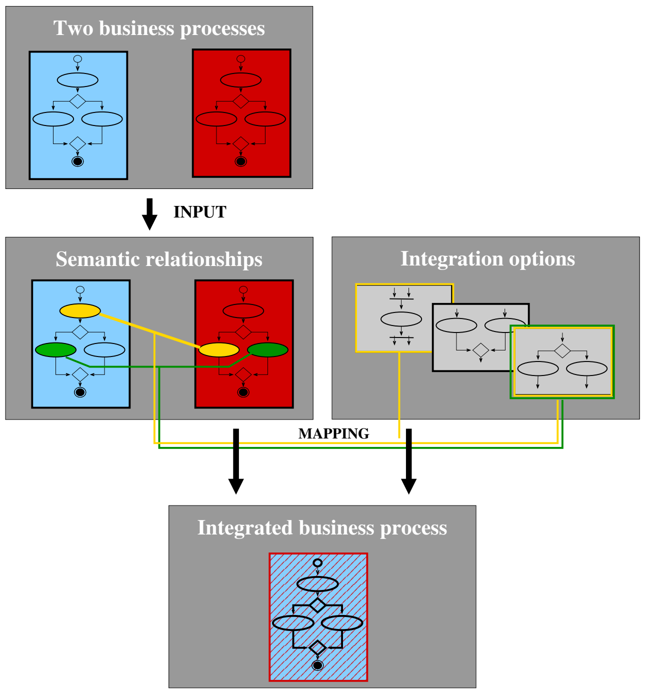

**Fig. 1.** The concept of the integration framework.

The main advantages of this framework is the support of integration being specified in multiple levels of varying domain independence, i.e., independent from system environment, and domain specific on the meta model level, as supported by Model Driven Architecture (MDA) techniques [4]. It assists the development of an integration application that can be used in a specific domain, e.g., supply chain management or e-commerce. The framework also supports the reuse of integration options which might be applied in different integration situations in a specific domain.

In the following section we give a short introduction to the diagram notation that we are using in Section 3. We assume that the reader is familiar with Petri net semantics and UML Activity Diagrams (UML-AD) notation.

## 2 Diagram notation

The proposed integration framework operates on the level of diagrammatic representations of the processes. Our diagram notation is derived from the UML 2.0 Activity Diagrams (UML-AD) [5]. However, we have adapted activity diagrams to our needs as a result of the experience with our earlier work in this area. We use a simplified subset of the nodes defined by the UML 2.0 AD standard which is sufficient to express business process semantics and preserves or even enhances the underlying Petri-net-resembling semantics of Activity Diagrams. On the other hand, we have made two significant extensions.

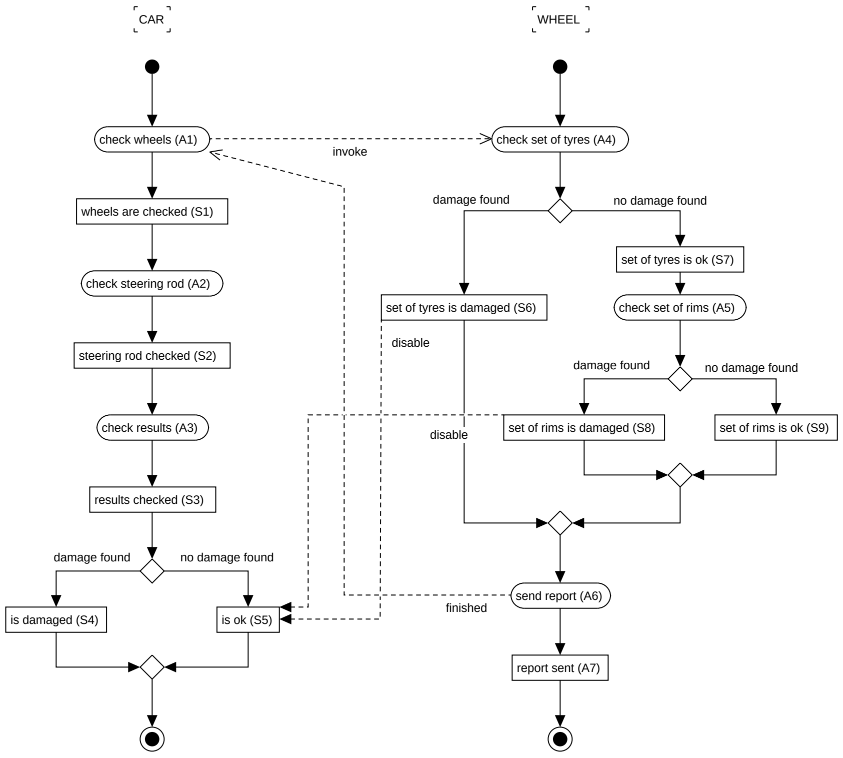

**Fig. 2.** An integrated model for $CAR$ and $WHEEL$.

First, we found that both "state-biased" or "activity-biased" diagrams impose limitations on the integration process, and explicitly dealing with states significantly facilitates the clear separation between the different integration options in a number of cases. Therefore we introduce an explicit representation of states and include the additional node type *state*.

Second, as described in [1] activity edges are not sufficient for modelling composition and association relationships between business processes, because they always imply passing of tokens, but situations exist where only checking for token existence should occur. We have introduced a new edge type, *links*, to handle this situation. We use 6 different link types which are sufficient for modeling of composite business processes. We refer to the diagrams resulting from these extensions as *Activity State Diagrams* (ASDs). For a more detailed discussion and examples of ASDs, see [1].

## 3 Example

We demonstrate the output of our integration framework on two simple business processes of a garage $CAR$ and a wheels store $WHEEL$. $CAR$ decided to outsource the wheels branch and to cooperate with $WHEEL$. In the example a customer brings his vehicle to $CAR$ because he feels vibrations on the steering wheel while driving. The search for the reason of the vibration consists of checking the wheels ($A_1$) and the steering rod ($A_2$), where $A_1$ should be performed by the business process $WHEEL$. This circumstance is defined by the semantic relationship *sub_act* between $A_1$ and $WHEEL$ [1]. One integration option for *sub_act* would be that $A_1$ invokes $A_4$ and the token in $A_1$ is blocked until the token in $WHEEL$ arrives at the activity "send report" ($A_6$), i.e., checking the tyres ($A_4$) and rims ($A_5$) is finished as shown in Figure 2. This integration option is called *blocking* [1]. An alternative option would be the *future synchronization*, where a token is not blocked in an activity, e.g., $A_1$, but can continue execution until another activity is reached, e.g., $A_3$. In this case the steering rod can be checked ($A_2$) while the checking of wheels is being executed. The software developer chooses the proper integration option which are created by the system administrator previously. Between the states $S_5$, $S_8$, and $S_9$ exists an *interprocess dependency*, meaning that a process cannot enter a specific state, e.g., $S_5$, if another process is or was in another specific state, e.g., $S_8$ or $S_9$. The dependency is realised through *disable* links.

## 4 Conclusion and Future Work

This paper has presented an integration framework for two business processes on different meta levels [1]. We are currently implementing the framework in the meta modelling tool DoME [3]. In future we are going to verify integration options by checking the consistency of the integrated model in respect to the input models similar to research in [7, 9, 8].

## References

1. Georg Grossmann, Yikai Ren, Michael Schrefl, and Markus Stumptner. Behavior Based Integration of Composite Business Processes. In *Proc. BPM 2005*, volume 3649 of *LNCS*, pages 186–204, September 2005.
2. Elsevier North Holland. Special Issue on Enterprise Modelling and System Support. *Advanced Engineering Informatics*, 18(4):191–253, October 2004.
3. Honeywell Inc. Domain Modeling Environment (DoME) . http://www.htc.honeywell.com/dome.
4. Anneke Kleppe, Jos Warmer, and Wim Bast. *MDA explained - The Model Driven Architecture: Practice and Promise*. Object Technology. Addison-Wesley, 2003.
5. James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual, 2nd edition*. Object Technology Series. Addison-Wesley, 2004.
6. Markus Stumptner, Michael Schrefl, and Georg Grossmann. On the road to behavior-based integration. In *Proc. of APCCM 2004*, pages 15–22. Australian Computer Society, 2004.
7. W. M. P. van der Aalst. Verification of workflow nets. In *Proc. of ICATPN 1997*, volume 1248 of *LNCS*, pages 407–426, London, UK, June 1997. Springer.
8. Kees van Hee, Lou Somers Natalia Sidorova, and Marc Voorhoeve. Consistency in model integration. *Data and Knowledge Engineering*, 56(1):4–22, January 2006.
9. S. J. Woodman, D. J. Palmer, S. K. Shrivastava, and S. M. Wheater. Notations for the Specification and Verification of Composite Web Services. In *Proc. of EDOC 2004*. IEEE Computer Society Press, 2004.