Improving Business Process Models with Reference Models in Business-Driven Development

Jochen M. Küster, Jana Koehler, and Ksenia Ryndina

IBM Zurich Research Laboratory
8803 Rüschlikon, Switzerland

Abstract. Reference models capture best-practice solutions for a specific industry such as retail, banking, or insurance. The models usually cover the whole range of solution components such as product models, business rules, data models, and service models. Over the past years, *business process reference models* have gained increasing attention. *Process merging* is a technique that brings together several process models to create a new process model. In this paper, we introduce process merging for a scenario which focuses on the improvement of an existing AS-IS business process by using a reference process model. We describe an approach that enables a business architect to establish *correspondences* between two process models in a systematic way and show how these correspondences define concrete *refactoring operations* that serve to improve the AS-IS model.

Keywords: Reference models, process merging.
Category: Industry paper.

1 Introduction

Over the past years, the role of business processes has been continously growing. The need to sense, analyze and respond more effectively to continuously changing market conditions and risks is the main driver behind this development. As a consequence, greater flexibility is required from business models and the supporting IT architecture. In order to meet this requirement, companies begin to eliminate "line of business" silos and move towards networked models. At the business level, composable business processes are key, while at the IT level, Web services [18] and the adoption of a Service-Oriented Architecture [5] are at the core of the new technologies.

The field of business process modeling has a long standing tradition. Recently, new requirements and opportunities have been identified, which result from the need to directly derive the IT solution from a business process model. Business-driven development (BDD) [11] is a methodology for developing IT solutions that directly satisfy business requirements and needs. BDD begins with the business strategy and requirements and takes them through an execution framework that is standardized, well understood, and that can be executed repeatedly and successfully [11]. Business process models are an essential means in BDD to create a link between the business needs and the IT implementations.

Modeling the processes of an enterprise is a time-consuming and methodologically challenging task. It is therefore not surprising that *reference models* have been developed that capture processes and data at an abstract level. Reference models describe the best practices of an industry and are also often aligned with emerging industry-specific

J. Eder, S. Dustdar et al. (Eds.): BPM 2006 Workshops, LNCS 4103, pp. 35–44, 2006.
© Springer-Verlag Berlin Heidelberg 2006

and cross-industry standards. Many examples of reference models exist, see [7] for an overview. An example of an industry-specific reference model is IBM's Insurance Application Architecture [9], which has been developed with the assistance of more than 40 leading international insurance companies, while [2] provides a process classification framework for cross-industry relevant business processes.

Using reference models in BDD has several advantages. First, they significantly speed up the design of business process models by providing reusable and high quality content. Secondly, reference models lead to better and optimized process designs as they have been developed over a longer period and usually capture the business insight of more than one industry player. Thirdly, the reference model content usually bridges the business and the IT domain. For example, business process models can be linked with predefined interface definition models and Web service models.

There are two main different scenarios, which make use of reference models. On the one hand, there is the approach of *reference model customization*, which starts from a process model that is equipped with configuration options and configures this model to the needs of an enterprise, see for example the work described in [3, 4, 14, 13, 16]. In this scenario, a reference model provides the starting point of the configuration process. This reference model is adapted to the needs of the customer e.g., by refining system roles, by adding and removing business activities in the process, by setting the values of process attributes, or by applying configuration patterns.

On the other hand, there is the increasingly important scenario of *process merging*. In this scenario, two or more process models have to be brought together in order to create an improved business process. One scenario for process merging is the improvement of an existing process model (AS-IS model) by a reference model where some parts of the existing model should be preserved and others should be replaced. Process merging is also required when companies become subject to acquisitions and mergers. In such situations, processes have to be aligned at the business and IT level, however, differences also have to be identified and preserved if appropriate.

Process merging differs from process configuration in the way that it usually requires to preserve certain parts of the AS-IS process model and the underlying IT systems. Therefore, a pure configuration of the reference model is not possible as this would not consider the existing business processes sufficiently enough. Process merging has similarities to model composition [8] but needs to be tailored to the characteristics of process models. A key technique for process merging is the ability to establish *correspondences* between the elements contained in two business processes. These correspondences allow to clearly identify related parts of process models and provide the basis for systematic process merging.

The paper is organized as follows: Section 2 briefly reviews *business-driven development* and discusses why *process merging* is becoming increasingly important. It also introduces two process models as an example for process merging. In Section 3, we introduce the two main steps of our method: the comparison of process models to establish *correspondences* and the derivation of a TO-BE model using *refactoring operations* driven by the correspondences. Correspondences are introduced in detail in Section 4, while the derivation of refactoring operations is covered in Section 5. We conclude with an outlook on next steps in Section 6.

2 Process Merging and the Role of Reference Models

Business-driven development provides a model-driven approach to business-IT alignment. We distinguish between *analysis* and *design models* of business processes [10]—a distinction which is also made in object-oriented modeling. An analysis model describes what the process is doing. It shows the initial partitioning of the process into subprocesses and activities with the main flow of control and, optionally, of data. It completely abstracts from IT-related aspects, but can be used for simulation and discussion with business analysts. A design model contains a refined partitioning of the process that reflects existing application systems and shows an IT-based flow of data and control and it describes how the process is realized using hardware, software, and people.

In business-driven development, we distinguish between *vertical* and *horizontal* scenarios of process merging. Horizontal scenarios involve the merging of process models at the same level of abstraction and usually in the same modeling notation whereas vertical scenarios involve models at different levels of abstraction, e.g., merging an analysis and a design model. Another vertical example scenario is the merging of a business process model and its underlying implementation, e.g., given in the Business Process Execution Language, BPEL [6], when either the process model or the BPEL have been modified and changes need to be identified.

A particular scenario that we are going to investigate in this paper is the merging of an AS-IS process model with a reference business process model with the goal to improve the AS-IS process and to capture this improvement in a TO-BE process model. This process improvement scenario is presented as a purely horizontal, analysis-level scenario. However, we would like to point out that this is only capturing the initial process improvement phase. In realistic BDD, business requirements flow downwards from the analysis model to the design model, while IT requirements flow upwards from the design model to the analysis model. This means, while improving an AS-IS process with a reference model, it can happen that certain improvement steps are inhibited by the underlying IT infrastructure. Furthermore, design-level decisions also have to be reflected in the improvement scenario when a *migration plan* has to be derived showing how the AS-IS process is migrated to the TO-BE process at the IT level. Due to space restrictions, we cannot discuss these vertical merging elements.

We present our example analysis models in the notation of IBM's WebSphere Business Modeler [1], which is based on UML 2.0 activity diagrams [12]. In these models, we distinguish *task* and *subprocess* elements. While tasks capture the atomic, not further dividable activities in the business process models, subprocesses can be further refined into more subprocesses and tasks. Control and data flow edges can connect tasks and subprocesses. The control and data flow can be split or merged using control nodes such as *decision*, *fork*, *merge*, and *join*. Process start and end points are depicted by *start* and *end nodes*.

Figure 1 introduces a simplified AS-IS process to handle an insurance claim submitted by a customer. The process model shows five subprocesses and two decisions. In this analysis model, we completely abstract from data flow and show the basic process structure and control flow only. In the *Check Requirements* subprocess, the claim requirements are first checked against the customer's insurance policy. If insufficient

information was provided by the customer, this information needs to be obtained first and thus the process cycles back to the initial state. If sufficient information is available, the subprocess *Make Accept-Reject Decision* makes an accept-reject decision about the claim. Depending on the outcome of this decision, a payment is made by the insurance or the claim is rejected.

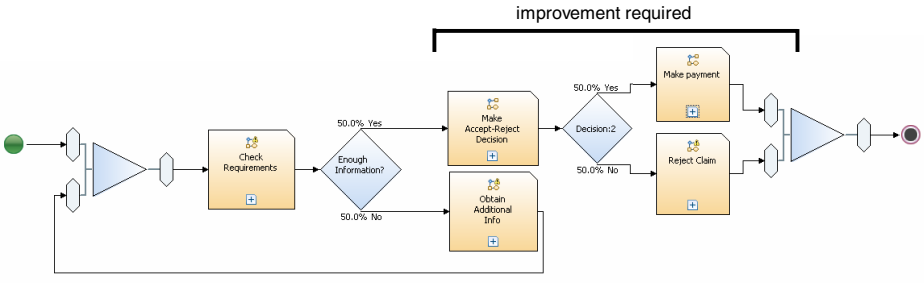

Fig. 1. AS-IS model of the claim handling process

Figure 2 shows a simplified reference model for the claim-handling process. We assume that this reference process has been selected, because of its better separation of the recording of the claim from its validation and the subsequent settlement process, which is much better worked out than the currently used payment process. This reference model therefore fits our anticipated improvement goal, which aims at improving the compliance of the current AS-IS process with new legal requirements. In order to meet these requirements, a more fine-grained process has to be developed, showing the detailed steps of making a payment to a customer.

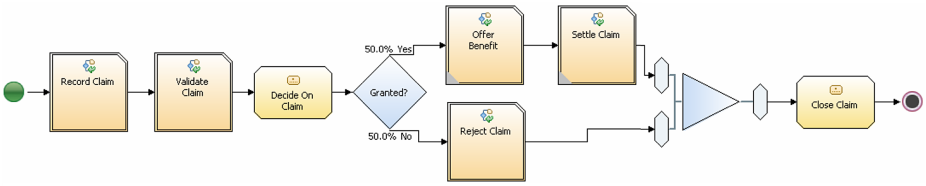

Fig. 2. Reference model for the claim handling process

3 Overview of Improvement Approach

An existing business process model can be improved under a diverse set of criteria, including *functional* aspects as well as *non-functional* aspects such as the cost it imposes while being run in an enterprise. The functional aspect of a business process model is determined by its tasks as well as their overall organization. For example, a business process can support *Claim Validation* functionality by including tasks that perform claim validation. Our improvement approach concentrates on these functional aspects, leaving out data flow and non-functional aspects.

Improvement can either be performed in a *revolutionary* or *conservative* approach. In the revolutionary approach, the reference model is taken as the initial TO-BE model. This TO-BE model is iteratively customized by integrating parts of the AS-IS model. As the IT legacy is disconnected of the initial TO-BE model with this approach, monitoring and evaluating intermediate results on the IT level is usually not possible.

In the conservative approach, the AS-IS model is taken as the initial TO-BE model. This initial TO-BE model is then adapted by considering tasks and subprocesses in the reference model. Changes introduced in the TO-BE model can be propagated to the IT-level immediately and an incremental approach can be adopted with regards to IT-level changes. This leads to the ability to monitor intermediate results by implementing intermediate process models and thereby reduces the risk of process migration. We favor the conservative approach and assume for the following discussion that the AS-IS model is taken as an initial TO-BE model. Figure 3 illustrates the main steps of our approach:

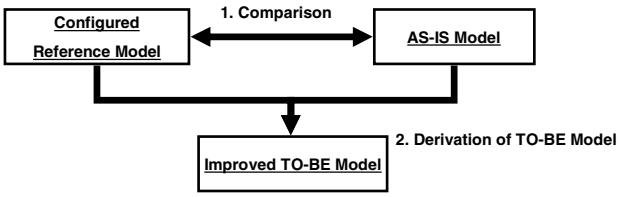

Fig. 3. Main steps of the approach

- *Comparison of AS-IS and Reference Model*: The first step is a detailed comparison of the AS-IS and reference model to detect similarities and differences, with regards to tasks, subprocesses and their organization. The comparison can be focused on improvement areas or extended to the complete models in order to perform a detailed analysis. In the first case, business goals lead to the identification of improvement areas. In the latter, a form of delta analysis [17] is used to identify those parts (hot spots) of the AS-IS model that should be improved by integrating parts of the reference model. A migration plan can be elaborated that clearly determines milestones for the envisioned improvement.
- *Derivation of TO-BE Model*: The AS-IS model is taken as the initial TO-BE model. This model is iteratively improved by refactoring operations: task contents are adapted, additional tasks and subprocesses are introduced or removed, the hierarchical structure is reorganized and the control flow is adjusted.

Optionally, the first step can be preceded by a process configuration step (e. g. [3, 4]). This involves configuring a configurable process model to the needs of the current domain.

The output of the approach is a TO-BE model that has been improved by integrating tasks and subprocesses of the reference model. The relationships of each AS-IS task or AS-IS subprocess with regards to the reference model are clearly defined.

In the next sections, we will discuss our solution to the problem of comparing process models and methods for derivation of an improved TO-BE model from the initial TO-BE model.

4 Comparison of AS-IS Model and Reference Model

This section introduces a systematic approach for a comparison between the AS-IS model and the reference model. In principle, two process models can be compared along different criteria, for example, tasks and subprocesses used, ordering of tasks and subprocesses by control flow as well as data used within processes.

In this section, we focus on comparison of tasks and subprocesses used in the processes without taking into account detailed control flow. As tasks represent the atomic unit of behavior, a relation on the task and subprocess level is a prerequisite for control flow or data flow comparison which is beyond the scope of this paper.

In order to visualize task and subprocess relations, we use a *tree structural view* which is a tree constructed of modeling elements of the business process model: The tree contains as root the process name node and each level i of the tree contains all activities of the process model with depth i. Tree edges represent hierarchical relationships of the process model, e.g., , if a subprocess contains a task, then the tree contains an edge from the subprocess to the task.

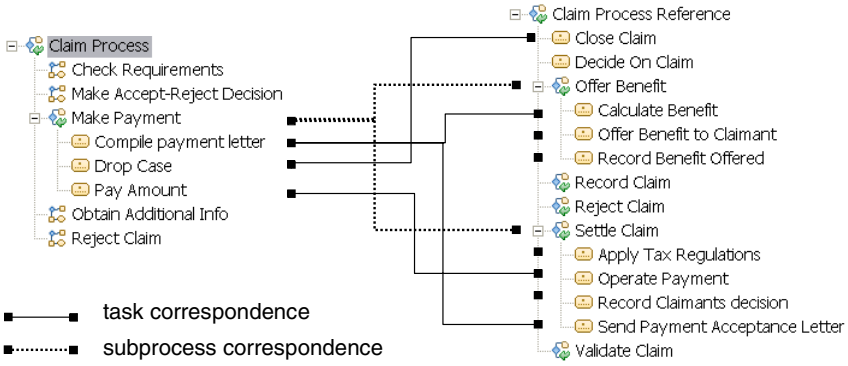

Fig. 4. Correspondences for the claim-handling scenario

In Figure 4, two tree structural views are shown. On the left, the structural view of the AS-IS claim handling process is displayed, showing the details of the *Make Payment* subprocess. On the right, the reference claim process is displayed.

When comparing business processes, one can find tasks that model the same business activity. The goal of a detailed comparison is to find all these *correspondences* between tasks of AS-IS and reference models and also to identify those tasks that do not have corresponding tasks. The idea is that a task has a correspondence to another task if they represent the same functionality. Typically, establishing correspondences requires a reference model expert and the business architect who designed the AS-IS model. Together, they have to review every task in the AS-IS and reference models, discuss the task content and establish correspondences manually. Optionally, correspondences can be established automatically if similarity of task content can be identified automatically e.g. by using automated semantic matching (see e.g. [15]). We identify the following types of task correspondences:

- *1-to-1*: One AS-IS task has a direct correspondence with a reference task.
- *1-to-0*: The AS-IS element does not have a corresponding reference task.
- *1-to-many*: The AS-IS task has several corresponding tasks in the reference model.
- *many-to-1*: Several AS-IS tasks have one corresponding task in the reference model.
- *0-to-1*: The reference task has no corresponding AS-IS task.

In Figure 4, we show task correspondences for our example, focusing on the tasks of the *Make Payment* subprocess:

- *Drop Case* has a 1-to-1 correspondence to *Close Claim* in the reference process,
- *Pay Amount* has a 1-to-1 correspondence to *Operate Payment*,
- *Compile Payment letter* has a 1-to-many correspondence to *Calculate Benefit* and *Send Payment Acceptance letter*,
- reference tasks *Offer Benefit to Claimant*, *Record Benefit Offered*, *Apply Tax Regulations* and *Record Claimants decision* have no corresponding AS-IS tasks.

Tasks that are in a correspondence relation can be compared with respect to their content which represents an even finer degree of comparison. Given two tasks, the *content* can either be *equal*, *inclusive* or *overlapping*. Note that the content of tasks has to be compared with respect to their semantics. A pure naming comparison is not enough.

After task correspondences have been established, subprocess correspondences can be automatically derived: The basic idea is to relate those subprocesses to each other that contain related tasks. Subprocess correspondences can be used to identify those subprocesses that have identical or overlapping behavior. In our example, the AS-IS *Make Payment* subprocess has a 1-to-many correspondence to the reference *Offer Benefit* and *Settle Claim* subprocesses. Details of the algorithm to derive subprocess correspondences are beyond the scope of this paper.

If two process models are captured on different abstraction levels, it can happen that a subprocess in the AS-IS model only has a single task as its counterpart in the reference model and vice versa. We assume that subprocess correspondences also include these types of correspondences between an AS-IS subprocess and one or more reference tasks.

In general, if there are many 0-to-1 correspondences, this means that the reference model contains functionality that is currently not included in the AS-IS model. On the contrary, 1-to-0 correspondences either show that the AS-IS model is very specific or that the reference model may not be suitable. A large number of many-to-1 and 1-to-many correspondences hints at a mismatch with regards to the abstraction levels at which the two models are captured.

Correspondences allow the definition of various quantifiable degrees of similarity between processes. For example, one can calculate a functionality gap between the two processes by dividing the reference tasks having 0-to-1 correspondence by the overall number of reference tasks. The example shows a 57% gap in the area of *Make Payment* because 4 of the total 7 tasks of the reference model have 0-to-1 correspondences.

The correspondences provide the basis for the derivation of the improved TO-BE model, which is discussed in the following section.

5 Derivation of Improved TO-BE Model

The objective of this step is to create a TO-BE model that takes into account the results of the mapping between the AS-IS model and the reference model. This TO-BE model represents an improved version of the AS-IS model by incorporating parts of the reference model. The more parts are incorporated into the initial TO-BE model, the closer will the resulting TO-BE model be to the best practice. The fewer changes are made, the closer will the resulting TO-BE model be to the existing AS-IS model. The ideal amount of changes is different for each customer situation and is found by also considering IT legacy constraints.

Based on the task and subprocess correspondences, we can identify the following refactoring operations that can be applied to the initial TO-BE model:

- 0-to-1 correspondence: addition of reference task or reference subprocess,
- 1-to-0 correspondence: removal of AS-IS task or subprocess,
- 1-to-many correspondence: splitting of AS-IS task or subprocess,
- many-to-1 correspondence: merging of AS-IS task or subprocess,
- correspondence between task and subprocess: conversion of tasks to subprocesses and vice versa,
- correspondence between elements with different depth *i*: changes in the hierarchy such as moving a task from one subprocess to another one.

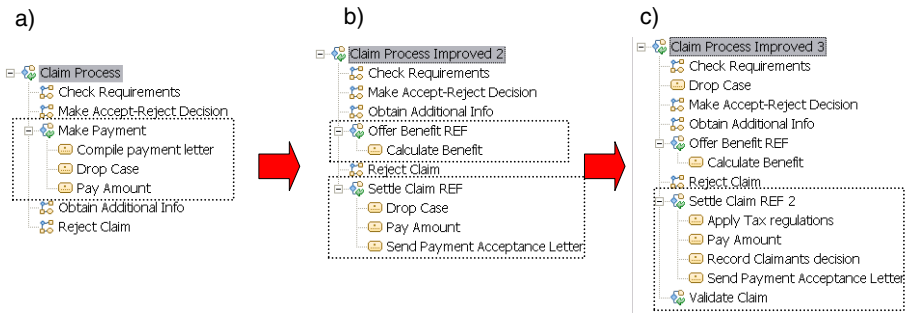

Fig. 5. Evolution of the TO-BE model

Figure 5 shows the application of these refactoring operations to the example. In the first step, it is decided to split the *Make Payment* subprocess into two subprocesses,*Offer Benefit REF* and *Settle Claim REF*, because the *Make Payment* subprocess is the area where improvement of the process is required. The contained tasks are split according to their correspondences, e.g., *Pay Amount* is placed into the *Settle Claim REF* subprocess. The *Compile Payment Letter* task is split (note the 1-to-many correspondence in Figure 4) and replaced by the *Calculate Benefit* and *Send Payment Acceptance Letter* tasks, leading to the result shown in Figure 5 b).

In the second step, the *Apply Tax Regulations* and the *Record Claimants decision* tasks are added to the *Settle Claim REF* subprocess (note the 0-to-1 correspondences in

Figure 4). Further, the *Drop Case* task is moved into its parent process, because it has a 1-to-1 correspondence to the *Close Claim* task in the reference model. It is decided to integrate the *Validate Claim* subprocess into the TO-BE model because *Validate Claim* represents a functionality that is not at all represented in the AS-IS model. This leads to an improved TO-BE model shown in Figure 5 c).

In a last step, all control flows between new and existing tasks and subprocesses are manually adjusted. Each subprocess or task that has been modified needs to be examined and the control flow needs to be reconnected because subprocesses or tasks cannot automatically be integrated into an existing control flow. Figure 6 a) shows the *Settle Claim* subprocess of the TO-BE model after the previously described refactoring operations, with unconnected control flow. The right order of the tasks has to be determined by adding control edges between the tasks, leading to the result shown in Figure 6 b). The resulting TO-BE model models the payment on a fine-grained level and therefore fulfills the original improvement goal.

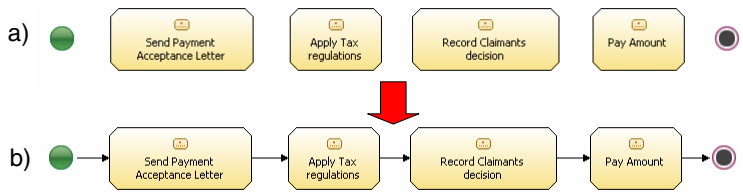

Fig. 6. Establishing the control flow in the *SettleClaim* subprocess of the TO-BE model

6 Conclusion

Within business-driven development, process merging allows the systematic integration of different processes to create a new business process. One scenario is the improvement of an existing business process by integrating parts of reference models.

In this paper, we propose a *process merging* approach that allows the detailed comparison of two process models and systematic derivation of an improved TO-BE model. Our approach has a number of benefits: The correspondences between tasks and subprocesses can be used to clearly capture the degree of alignment and to identify the hot spots of the AS-IS model that need to be improved. Concrete refactoring operations are determined by the correspondences and can be executed semi-automatically. As the refactoring operations can be recorded, process model improvement can be made reproducible, which can be a major cost reduction effort if new versions of the reference model have to be customized for the same company or if improvement proceeds over a longer time period.

One important area of future work is tool support for refactoring process models with the capability of change tracking. Further future work includes the elaboration of the concept of migration plans and the integration with a bottom-up approach that takes into account existing legacy applications.

References

1. IBM WebSphere Business Modeler. http:///www-306.ibm.com/software/integration/wbimodeler/.
2. Process Classification Framework. http://www.apqc.org/portal/apqc/ksn/PCF_Complete_May_5_2004.pdf, 2004.
3. J. Becker, P. Delfmann, A. Dreiling, R. Knackstedt, and D. Kuropka. Configurative Process Modeling - Outlining an Approach to Increased Business Process Model Usability. In *Proceedings of the 2004 Information Resources Management Association Conference. New Orleans.*, pages 615–619, 2004.
4. A. Dreiling, M. Rosemann, W. van der Aalst, W. Sadiq, and S. Khan. Model-Driven Process Configuration of Enterprise Systems. In S. Eckert O.K. Ferstl, E.J. Sinz and T. Isselhorst, editors, *Wirtschaftsinformatik 2005*, pages 687–706. Physica-Verlag, 2005.
5. T. Erl. *Service-Oriented Architecture: Concept, Technology, and Design*. Prentice Hall, 2005.
6. T. Andrews et al. Business Process Execution Language for Web Services. http://www.ibm.com/developerworks/webservices/library/ws-bpel, 2002.
7. P. Fettke, P. Loos, and J. Zwicker. Business Process Reference Models: Survey and Classification. In Christoph Bussler and Armin Haller, editors, *Business Process Management Workshops, BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005, Revised Selected Papers*, volume 3812, pages 469–483, 2006.
8. M. Gervais, K. Engel, D. Kolovos, D. Touzet, Y. Shaham-Gafni, R. Paige, and J. Aagedal. MODELWARE Delivery 1.5 Model Composition: Definition of Model Composition Properties. http://www.modelware-ist.org/.
9. IBM Insurance Application Architecture. http://www.ibm.com/industries/financialservices/iaa.
10. J. Koehler, R. Hauser, J. Küster, K. Ryndina, J. Vanhatalo, and M. Wahler. The Role of Visual Modeleling and Model Transformations in Business-Driven Development. In *Proceedings of the 5th International Workshop on Graph Transformations and Visual Modeling Techniques*, pages 1–12, 2006.
11. T. Mitra. Business-driven development. IBM developerWorks article, http://www.ibm.com/developerworks/webservices/library/ws-bdd, IBM, 2005.
12. Object Management Group (OMG). *UML 2.0 Superstructure Final Adopted Specification. OMG document pts/03-08-02*, August 2003.
13. J. Recker, M. Rosemann, W. M. P. van der Aalst, and J. Mendling. On the Syntax of Reference Model Configuration - Transforming the C-EPC into Lawful EPC Models. In *Business Process Management Workshops, BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005, Revised Selected Papers*, pages 497–511, 2005.
14. M. Rosemann and W. van der Aalst. A Configuarable Reference Modeling Language. *Information Systems*, 2006. To appear.
15. T. Syeda-Mahmood, G. Shah, R. Akkiraju, A. Ivan, and R. Goodwin. Searching Service Repositories by Combining Semantic and Ontological Matching. In *2005 IEEE International Conference on Web Services (ICWS 2005), 11-15 July 2005, Orlando, FL, USA*, pages 13–20. IEEE Computer Society, 2005.
16. O. Thomas, O. Adam, and P. Loos. Using Reference Models for Business Process Improvement: A Fuzzy Paradigm Approach. In *9th International Conference on Business Information Systems (BIS 2006), Klagenfurt, Austria, May 31-June 2, 2006.* To appear.
17. W. van der Aalst. Business Alignment: Using Process Mining as a Tool for Delta Analysis and Conformance Testing. *Requirements Engineering Journal*, 2006. to appear.
18. O. Zimmermann, M. Tomlinson, and S. Peuser. *Perspectives on Web Services - Applying SOAP, WSDL and UDDI to real-world projects*. Springer, 2003.