Scenarios and Techniques for Choreography Design

Gero Decker[1]*, Michael von Riegen[2]**

[1] Hasso-Plattner-Institute, University of Potsdam, Germany
gero.decker@hpi.uni-potsdam.de
[2] University of Hamburg, Germany
riegen@informatik.uni-hamburg.de

Abstract. Choreography description languages have been put forward for capturing sets of interactions and their control and data dependencies, seen from a global perspective. Choreographies serve as starting point for generating interface processes for the different participants which in turn are used for implementing new services or adapting existing ones. However, such top-down approaches are not sufficient for scenarios where given implementations cannot be changed and are to be used as a starting point for choreography design. This paper identifies and classifies three categories of choreography design: *choreography identification*, *choreography context expansion* and *collaboration unification*. Each category is motivated through an example from the eGovernment domain. Existing techniques needed for the individual design categories are discussed and missing techniques are highlighted.

1 Introduction

Services are more and more used to support long-running business processes. This trend runs alongside with a shift from merely considering simple interaction behavior of services, like request-response interaction patterns manifested in standards like SOAP and WSDL, towards conversational services that engage in long-running conversations with other services.

In order to cope with the complexity of these conversations, a new viewpoint on interacting services was introduced. It describes interactions from a global point of view, i.e. from the perspective of an ideal observer who is able to see all interactions and their flow and data dependencies. The resulting global interaction models are called service choreographies. Standards such as the Web Service Choreography Description Language (WS-CDL [8]) were put forward for describing choreographies of web services. The main motivation for introducing such an abstraction layer was to enable a model-driven approach for service design and implementation. These top-down approaches, like e.g. presented in [7], propose choreographies as a starting point for generating interface processes for

* Research conducted while with SAP Research Center Brisbane
** Research is sponsored under the EU IST-2004-026650 project "R4eGov"

each service which are then the skeletons for implementing new services or for adapting existing services.

As the idea of global interaction models matures, more and more application domains adapt choreography languages and methodologies. However, it turns out that many application scenarios cannot be covered using top-down approaches. Existing services and processes are the starting point for identifying already existing choreographies or for creating to-be choreographies. In the course of this paper we will present several use cases from the EU funded project "Research for eGovernment" (R4eGov[3]) to display the need of different choreography design approaches. The project centers around interoperability and security in inter-organizational and even cross-border processes. Its main challenges are coping with heterogeneous systems and preserving local system and process ownership ([1]). Within this project, heterogeneity issues are tackled by resorting to service-oriented architecture concepts and the usage of web services. Thus, service choreography methods and techniques can be used for managing the peer-to-peer like collaborative processes.

The next section sets the scene for the paper by introducing different viewpoints for modeling collaborative processes. Section 3 presents *choreography identification*, section 4 presents *choreography context expansion* and section 5 *collaboration unification.* All three choreography modeling categories are illustrated using a use case and the corresponding techniques are discussed. Finally, a conclusion is drawn and future work is sketched. Related work will be mentioned in the course of each section.

2 Classification Framework

Figure 1 presents different perspectives on inter-organizational collaboration. *Process implementations* are the intra-organizational process definitions that are executed within one participant. In service-oriented architectures these definitions are called service orchestrations. While they also include internal actions that are not to be shown to other participants, the *provided interface processes* only describe the publicly visible behavior of a participant. I.e. only those actions are included in an interface process that directly relate to message exchanges with the outside world. Like it was the case for the process implementations, interface processes describe collaboration from the perspective of one single participant (endpoint-centric view). A set of interconnected interface processes describes all interaction taking place in a collaboration. As means to describe such collaboration from a truly global point of view (i.e. within one process definition), choreographies were introduced.

Furthermore, there are two different kinds of choreographies that we want to distinguish: An *observable choreography* describes the actually observable interaction behavior of a set of collaborating partners. One could imagine an ideal observer tracing all interactions belonging to a collaboration. In contrast to this

[3] See http://www.r4egov.info/

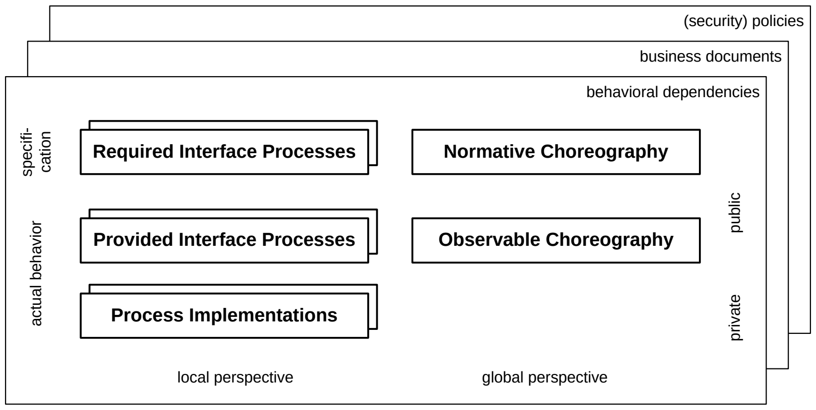


Fig. 1. Perspectives for describing collaborative processes

we find *normative choreographies* prescribing behavior for the participants in the collaboration. These choreographies are minimal sets of constraints in the collaborations and serve as contractual basis. Conformance between the actual interaction behavior of participants (the observable choreography) and the corresponding normative choreography can be checked. Most normative choreographies equal to a set of *required interface processes* constraining the interaction behavior of an individual participant. Complying to a required interface process is a prerequisite for a participant taking part in the collaboration. The required interface process defines what behavior other participants can expect from the implementing participant. If a participant violates a constraint in his process implementation, the other participant will be faced with unexpected behavior. However, participants still have different possibilities to implement a required interface process. This explains why there can be different provided interface processes that are all conforming to the same required interface process.

In addition to these different viewpoints we find three dimensions within each viewpoint.*Behavioral dependencies* between interactions cover the control flow in choreographies. The *business document* dimension takes care of the content and structure of the messages being exchanged and the data flow dependencies between different interactions. Content of business documents also influences branching decisions in the control flow dimension. Finally, *(security) policies* describe non-functional configurations in the collaboration.

Dijkman and Dumas have introduced some of the perspectives in [7]. However, they only focused on the control flow perspective and did not make a distinction between observable and normative choreographies.

Within the next sections we will display three approaches to do choreography design using this framework for describing the design procedure.

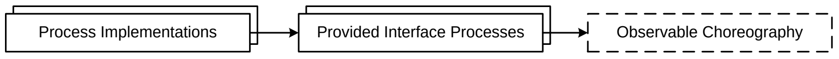


Fig. 2. Choreography identification

3 Choreography identification

In the case of *choreography identification*, different participants have working process implementations and already use them to collaborate with each other. However, every participant only knows about the interactions he is directly involved in (i.e. his own provided interface process). Therefore, the goal is to identify the observable choreography the participants already engage in so that everybody has a global view of the collaboration (see figure 2). The main motivation behind choreography identification is an optimization of the overall collaboration. The overall costs for the collaboration have to be reduced and/or the performance of the collaboration has to be ameliorated. Only having the global picture at hand, partners see what interactions and dependencies exist globally and which of these might be removed or organized differently. Finally, the changes in the choreography are pushed down again into the process implementations. The next section motivates this procedure with a use case.

3.1 Use Case: Eurojust / Europol Collaboration

Eurojust (European Judicial Cooperation Unit) and Europol (European Police Office) have been set up to help the EU member states to cooperate in the fight against cross-border organized crime. An objective is to establish a secure connection between Eurojust and Europol to insure collaboration and effective information exchange between both parties [12]. Both organizations already manage their information with computer systems but these are completely independent. On the one hand, Europol has three information systems: The Europol Information System that supports all intelligence activities within the Europol framework, the Europol Analysis System that is only available to the analysts employed within each analysis work file, and the Information Exchange System that enables bilateral exchange of data between Member States without necessarily involving Europol. On the other hand, Eurojust has only one system which can be used within collaboration: The Case Management System that enables to deal with case-related activities.

It should be noted that there are already different kinds of collaboration between Eurojust and Europol existing. Examples are the request for mutual legal assistance for witness protection during criminal proceedings, the execution of European arrest warrants (EAW) or an EAW with a rogatory letter. From now on, these existing channels should be supported by electronic ways of communication which means that paper-based communication should be converted to a secure electronic conversation. After the identification of existing interaction, the processes need to be built around the existing infrastructure. Furthermore, it is also the case that a participant might need to know all dependencies between others to be able to react to all possible scenarios or exceptions. To do this, he needs to identify the overall choreography. A choreography can be one part of the contractual grounding of electronic conversation between the parties.

3.2 Techniques

In the case of *choreography identification* the observable choreography has to be extracted from the existing collaborating processes. This can be done either

1. based on the process implementations and provided interface processes or
2. based on the runtime behavior of the processes.

Option 1 involves the extraction of provided interface processes if only the process implementations are given. Once the provided interface processes are known they can be interconnected for retrieving the overall collaboration process. E.g. in [11] different workflow modules are interconnected to a workflow net which can then be used for reasoning on the overall process. Other formalisms where proposed for capturing choreographies and interface processes. In [4] a formalism for choreographies and another for interface processes are presented together with a bisimulation-like conformance relation. However, it is not mentioned for this formalism how to retrieve the choreography from a set of given interface processes. The same holds true for other interaction modeling languages such as WS-CDL ([8]) and *Let's Dance* ([17]).

Option 2 can be done by looking at the traces produced in the actual collaboration. If a sufficient number of conversations have been traced, process mining techniques (cf. [15]) can be used to retrieve the observable choreography model.

4 Choreography Context Expansion

Normative choreographies are normally limited to a certain business context. Assumptions are present among the participants which are (only) valid for the specific context. In order to broaden the reach of the choreography, i.e. make the choreography applicable to a broader context, these assumptions have to be removed from the choreography and therefore also from the required interface processes. This might result in a situation where individual participants cannot conform to the choreography for all covered contexts any longer. To broaden a choreography, we have to consider the provided interface processes and the normative choreography in order to get an expanded normative choreography like depicted in figure 3. The next section motivates context expansion with a use case.

4.1 Use Case: Electronic Procurement Schema for European Public Administrations

This scenario describes the challenges in cross-border exchanges in public procurement [3]. Normally, each country has its own public procurement system

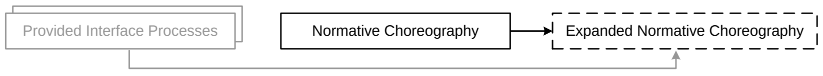


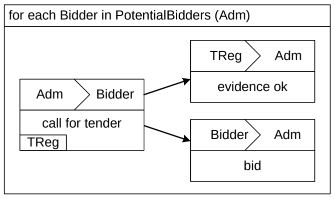
Fig. 3. Choreography context expansion

where local companies can bid. If a company wants to bid in another EU member state, it has to substantiate its legal existence and the buyer must be able to trust the received digital data. Due to the national differences in legal obligations and basis, this problem is getting really complex. There are at least two main issues that have to be solved within this context to have companies participating in cross-border public procurement: The electronic certification of companies willing to bid in another country and the diffusion of legal information; whenever a company makes changes in its status, it has to report it to the trade register.

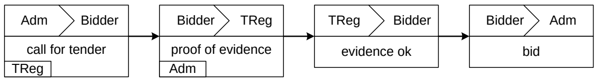
The European Commission released a legislative package consisting of two directives, 2004/18/CE and 2004/17/CE. These directives introduce a framework for open e-procurement standards and necessary conditions. The main problem is that an adoption of these directives will differ from country to country because of different technical standards, different legal requirements, different tax obligations, different laws on labor, different electronic certification processes, etc.

A collaboration model based on choreography can help to model the exchange of information between different countries without having to modify their national procedures. Thus, existing collaborations have to be opened for a wider audience. This is a scenario where we have to identify the minimal requirements that a new participant has to abide in order to join the cross-border collaboration. The process to accept cross-border participants has to be built around the existing collaboration. In contrast to this, a classic top-down redesign might change the local processes and interfaces and this would tamper the existing collaboration.

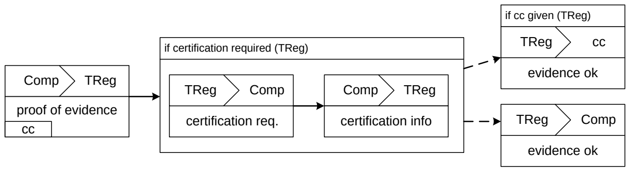
The example in figures 4(a) to 4(e) illustrates a part of the procurement use case using the choreography modeling language *Let's Dance* (cf. [17]). The French administrations have a provided interface process where a set of potential bidders are notified about a call for tender. Then, they expect a notification from the trade register that the bidder's proof of evidence is ok as well as the actual bid from the bidder (see figure 4(a)). The French bidders receive the call for tender, then they provide proof of evidence as well as the reference of the administration to the trade register. After being notified about a correct evidence, the bidder places the bid (see figure 4(b)). Finally, the trade register has a generic interface process for handling evidence cases. After a request containing the proof of evidence comes in, the trade register checks if the company is already registered or if further certification is required. If this is the case, interaction regarding the certification takes place with the requesting company. Finally, an "evidence ok" notification is sent back to the company as well as to another institution if given as "cc" (see figure 4(c)). In the case where only French participants are involved, the additional certification is not required since all French companies



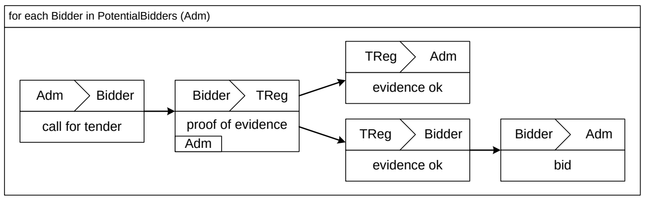
(a) Provided interface of the French administrations



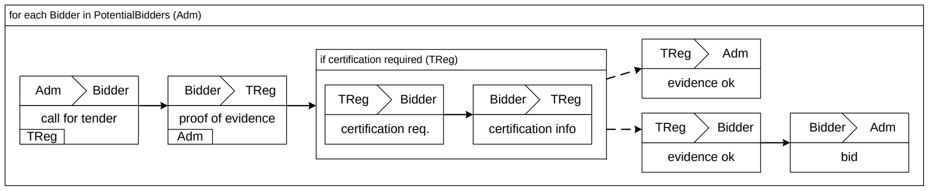
(b) Provided interface of the French bidders



(c) Provided interface of the trade register



(d) Normative choreography for the context "Bidding in France"



(e) Normative choreography for the context "Bidding in the European Union"

Fig. 4. eProcurement use case

are registered. The normative choreography for a French-only context is illustrated in figure 4(d). When broadening the context to Europe, the normative choreography includes the potential additional certification interactions. In this context, the French bidders cannot participate any longer unless they extend their provided interface process (see figure 4(e)).

4.2 Techniques

In the example we find that the French bidders can only operate in the context "bidding in France". They either do not know about the decision the trade register has to make or they make an assumption about how the trade register decides. The trade register on the other side is not limited to this context and can deal with the broader context "bidding in Europe" by considering the case that a company is not yet registered.

The situation that different partners can already deal with different contexts is typical for this choreography modeling category. The strategy is now to identify what parts of interface processes apply to which context and to then include corresponding interactions into the expanded normative choreography. We therefore need process model synthesis techniques. Techniques can be found in the space of object-oriented computing, where different scenario descriptions have to be merged to a single state machine describing the complete behavior of an object (cf. [9]). In the field of message sequence charts we find different scenarios of interacting system components that are to be merged into one global interaction model (cf. [14]).

The notion of business context is a central concept in ebXML's Core Component Technical Specification ([5]). It defines how to describe business document specifications and introduces "core components" as opposed to "business information entities". Business information entities are based on core components but are limited to a specific business context. There has not been any work on business contexts in choreography models. However, results from the field of process family engineering (e.g. [13]) could be used as a starting point.

Furthermore, conformance techniques are essential for checking if an existing provided interface process that conformed to the original normative choreography with the limited context still conforms to the expanded normative choreography. Many existing conformance checking techniques like protocol and projection inheritance in [2] and conformance in [4] are based on bi-simulation. [6] highlights that conformance relations are tightly linked to compatibility relations and shows that bi-simulation is too restrictive for common compatibility notions. It provides a means to check whether a conformance relation is optimal but it does not define a concrete relation fulfilling the criteria.

5 Collaboration Unification

In the case of *collaboration unification*, different observable choreographies exist for the same domain. A typical reason for the evolution of such "islands of collaboration" is that there are disjoint groups of collaborators each of which has

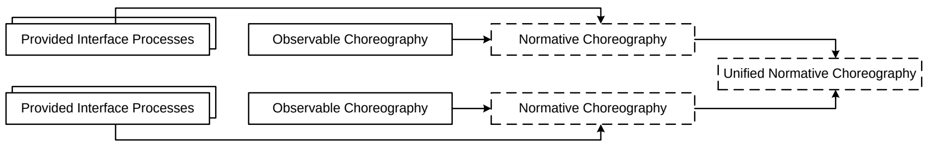


Fig. 5. Collaboration Unification

its own history for the interaction protocol. Now the goal is to enable the collaboration between participants from different islands through a unified normative choreography for all participants (See figure 5). A motivating use case will be described within the next section.

5.1 Use Case: eErasmus eHigher Education (eEH)

eErasmus is an international exchange program of higher education institutes among EU countries [10]. Students joining this program can take courses at foreign universities and might have the selected courses acknowledged by the home university. For a student, it is difficult to get the acknowledgment of foreign courses and examinations by the home university. The other way round, it is also difficult for a student to get an approval for courses without having any documents from the foreign host university. eErasmus should help a student by proposing the right courses or the right documents to get approved by the host university and should help the universities during the exchange process of grades and examination results.

This simple example depicts two high-level use cases. On the one hand, we have a *preparation process* where a student has to prepare the residence at a host university. He has to get approved by the host university in order to have courses. On the other hand, we have an *acknowledgment process* where the host university passes exams and grade results back to the home university. These have to be acknowledged by the home university if the student wants to continue studying at his home university.

The main problem within eErasmus is the setup of collaboration between different universities which are using different administration systems and different grading schemata for students. For this, some universities already have a working collaboration, others do not. This scenario displays the need of collaboration between the home and host university and is an example for collaborations that already exist on some islands within the same domain. These collaboration processes have to be merged or unified and can be adopted by other universities that do not have a collaboration running.

5.2 Techniques

As depicted in figure 5 collaboration unification consists of two steps:

1. The minimal interaction constraints in each island of collaboration have to be identified, leading to a normative choreography for each island.

2. The different normative choreographies have to be merged.

The first step mentioned in point 1 is to do identify normative choreographies. As already mentioned in section 2, normative choreographies allow more interaction scenarios than what observable choreographies describe. If a participant could participate in more interaction scenarios than what is captured in the observable choreography, this might be a hint where to extend an observable choreography into the direction of a normative choreography. Figure 6 illustrates an example from the preparation process within the eErasmus case study. A home university X and host university Y form an island of collaboration. The provided interface process of X consists of sending student details, sending the learning agreement (a document containing the list of classes the student wants to enroll in at the host university) and receiving an acknowledgment. In contrast to this, the provided interface process of Y includes the possibility to receive updates of the learning agreement during a time period of 2 weeks until the acknowledgment is sent. The observable choreography of the collaboration between X and Y is equal to the provided interface process of X. No update will ever be sent/received. In contrast to this, Y could also interact with other home universities that send agreement updates within two weeks. Therefore, the provided interface process of Y could be used as the normative choreography while X would still conform to it (see section 4.2 for a short discussion on conformance). Having broadened the possible interaction scenarios by extending the current observable choreography to the normative choreography, more home universities can join in.

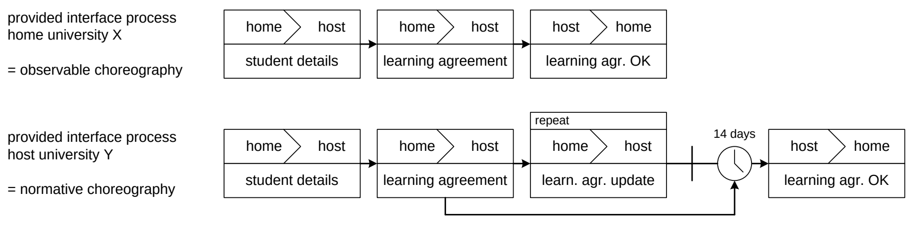


Fig. 6. Extracting a normative choreography

We see that identifying a normative choreography from given process implementations is the direct opposite of implementing conforming processes from a given normative choreography. During implementation certain decisions can be made (within the possibilities of conformance). E.g. assuming the normative choreography existed in the example, we could interpret that X has chosen not to send updates although it had the chance to do so. As a result, we have to remove these implementation decisions when identifying a normative choreography. Once we know what kind of decisions can be made during implementation while still preserving conformance, we can introduce techniques for identifying possible results of decisions and reverting the decision in given process implementations.

The second step mentioned in point 2 is to do a merge. Merging different normative choreographies requires for techniques that identify common structures in different process models and help to handle those parts of the process that are different. In the area of version control software these two functions are called *diff* and *merge*. The notion of process inheritance was introduced to reason on whether a changed process definition inherits properties from the original definition. Inheritance-preserving transformation rules are proposed in [16].

6 Conclusion

This paper has motivated the need for techniques supporting different categories of choreography design that were derived from eGovernment scenarios. As part of that the distinction between *observable choreographies* and *normative choreographies* was made. We have discussed existing techniques. However, there are still techniques missing for each of the three categories, namely for choreography identification, choreography context expansion and choreography unification. In the case of *choreography identification*, techniques for generating choreographies out of interconnected interface processes are missing for several languages. In the case of *choreography context expansion* extensions for choreography languages are missing that introduce the notion of explicit variability points, where different variants of the choreography can be defined for different business contexts. Process model synthesis techniques have to be conceived for integrating potential interactions into choreographies. The topic of conformance has to be revisited since bi-simulation-based techniques are too restrictive. Finally, in the case of *collaboration unification*, we have seen that techniques for extracting minimal constraints in a choreography and for merging conflicting process model structures into a unified normative choreography are missing. Introducing new techniques for filling the identified gaps and validating them using the presented use cases will be subject to future work.

References

1. R4eGov - Towards e-Administration in the large: Project Description, March 2006.
2. T. Basten and W. M. P. van der Aalst. Inheritance of behavior. *JLAP*, 47(2):47–145, 2001.
3. M.-C. Berneron, E. Deserevill, T. V. Cangh, O. Aydogmus, and A. Boujraf. SIXTH FRAMEWORK PROGRAMME, Information Society Technologies, R4eGov, Deliverable WP3-D5 - Case Study: Electronic procurement schema for European public administrations, July 2006.
4. N. Busi, R. Gorrieri, C. Guidi, R. Lucchi, and G. Zavattaro. Choreography and Orchestration: A Synergic Approach for System Design. In *B. Benatallah, F. Casati, and P. Traverso (Eds.): ICSOC 2005, LNCS 3826*, pages 228–240. Springer Verlag, 2005.

5. M. Crawford et al. ebXML Core Components Technical Specification 2.01. Technical report, UN/CEFACT, November 2003.
6. G. Decker and M. Weske. Behavioral Consistency for B2B Process Integration. In *Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE)*, LNCS, Trondheim, Norway, June 2007. Springer Verlag.
7. R. Dijkman and M. Dumas. Service-oriented Design: A Multi-viewpoint Approach. *International Journal of Cooperative Information Systems*, 13(4):337–368, 2004.
8. N. Kavantzas, D. Burdett, G. Ritzinger, and Y. Lafon. Web Services Choreography Description Language Version 1.0, W3C Candidate Recommendation. Technical report, November 2005. http://www.w3.org/TR/ws-cdl-10.
9. K. Koskimies and E. Makinen. Automatic synthesis of state machines from trace diagrams. *Software - Practice and Experience*, 24(7):643–658, 1994.
10. J. Lodge and R. Vermer. SIXTH FRAMEWORK PROGRAMME, Information Society Technologies, R4eGov, Deliverable WP3 D1-D4, Case Study e Erasmus eHigher Education (eEH), July 2006.
11. A. Martens. Analyzing Web Service based Business Processes. In M. Cerioli, editor, *Proceedings of Intl. Conference on Fundamental Approaches to Software Engineering (FASE'05), Part of the 2005 European Joint Conferences on Theory and Practice of Software (ETAPS'05)*, volume 3442 of *Lecture Notes in Computer Science*, Edinburgh, Scotland, April 2005. Springer-Verlag.
12. P.-E. Schmitz, T. V. Cangh, and A. Boujraf. SIXTH FRAMEWORK PROGRAMME, Information Society Technologies, R4eGov, Deliverable WP3-D2 - Eurojust / Europol collaboration, July 2006.
13. A. Schnieders and F. Puhlmann. Variability Mechanisms in E-Business Process Families. In *W. Abramowicz, H. Mayr (Eds.): 9th International Conference on Business Information Systems (BIS 2006)*, volume P-85 of *LNI*, pages 583–601, Bonn, Germany, 2006.
14. S. Uchitel, J. Kramer, and J. Magee. Synthesis of behavioral models from scenarios. *IEEE Trans. Softw. Eng.*, 29(2):99–115, 2003.
15. W. van der Aalst and A. Weijters. *Process-Aware Information Systems: Bridging People and Software through Process Technology*, chapter Process Mining, pages 235–255. Wiley & Sons, 2005.
16. W. M. P. van der Aalst and T. Basten. Inheritance of workflows: an approach to tackling problems related to change. *Theor. Comput. Sci.*, 270(1-2):125–203, uary.
17. J. M. Zaha, A. Barros, M. Dumas, and A. ter Hofstede. A Language for Service Behavior Modeling. In *Proceedings 14th International Conference on Cooperative Information Systems (CoopIS 2006)*, Montpellier, France, Nov 2006. Springer Verlag.