On the Controlled Evolution of Process Choreographies

Stefanie Rinderle
University of Ulm
Datbases and Inf. Systems Group
rinderle@informatik.uni-ulm.de

Andreas Wombacher, Manfred Reichert
University of Twente
Information Systems Group
a.wombacher@utwente.nl,m.u.reichert@utwente.nl

Abstract

Process–aware information systems have to be frequently adapted due to business process changes. One important challenge not adequately addressed so far concerns the evolution of process choreographies. If respective modifications are applied in an uncontrolled manner, inconsistencies or errors might occur in the sequel. In particular, modifications of private processes performed by a single party may affect the implementation of the private processes of partners as well. In this paper we sketch a framework that allows process engineers to detect how changes of private processes may affect related public views and - if so - how they can be propagated to the public and private processes of partners. Our approach exploits the semantics of the applied changes in order to automatically determine the adaptations necessary for the partner processes.

1. Motivation

The economic success of an enterprise more and more depends on its ability to flexibly and quickly react on changes at the market, the development, or the manufacturing side. For this reason companies are developing a growing interest in improving the efficiency and quality of their internal business processes (BP) and in optimizing their interactions with business partners and customers. Recently, we have seen an increasing adoption of BP automation technologies by enterprises as well as emerging standards for BP orchestration and BP choreography in order to meet these goals. Altogether they enable the definition, execution, and monitoring of the operational processes of an enterprise. In connection with Web service technology, in addition, the benefits of BP automation from within a single enterprise can be transferred to cross-organizational BP (*process choreographies*) as well. The next step in this evolution will be the emergence of the agile enterprise being able to rapidly set up new processes and to quickly adapt existing ones to changes in its environment.

One important challenge not adequately dealt with so far concerns the evolution of process choreographies, i.e., the controlled change of the interactions between partner processes in a cross-organizational setting. If one party changes its process in an uncontrolled manner, inconsistencies or errors regarding these interactions might occur in the sequel. Generally, the partners involved in a process choreography exchange messages via their public processes, which can be considered as special views on their private processes (i.e., the process orchestrations). If one of these partners has to change the implemenation of his private process (e.g., to adapt it to new laws or optimized processes) the challenging question arises whether this change also affects the interactions with partner processes and their implementation. Obviously, as long as a modified business process is not part of a process choreography, change effects can be kept local. The same applies if changes of a private process have no impact on related public views.

In general, however, we cannot assume this. The modification of a private process may not only influence the corresponding public process, but also the public and private processes of its partners. We therefore need methods for (automatically) *propagating* changes of a private process to the partner processes (if required). This issue has not been considered so far. As a consequence adaptations of process choreographies have turned out to be both costly and error-prone. Note that the handling of changes is not trivial since we must be able to precisely state which effects on partner processes result when adapting a (private) process.

In this paper we sketch an approach that addresses these challenges and allows for the controlled evolution of process choreographies (for a detailed report see [7]). We discuss how changes of a private process may affect related public views and - if so - how they can be propagated to the public and the private processes of partners. In order to be able to precisely state whether change propagations to partner processes become necessary or not we have introduced a formal framework based on annotated Finite State Automata (see [7]). In this framework we exploit the semantics of the applied change operations in order to derive

necessary adaptations automatically. Due to the autonomy of partners, however, private partner processes cannot be adapted automatically to changes of a process choreography. However, our approach allows for the comprehensive assistance of users in accomplishing this task in a correct and effective manner.

Sect. 2 gives an overview of our framework for process choreography evolution. Sect. 3 discusses related work and Sect. 4 gives a summary and an outlook on future work.

2. Process Choreography Evolution

Consider the scenario depicted in Fig. 1a. The accounting department approves an order (*order* message) sent by a buyer and forwards the order to the logistics department (*deliver* message) to deliver the requested goods. The logistics department confirms the receipt (*deliver_conf* message) and forwards it to the buyer extended by the expected deliver date and the parcel tracking number using the *delivery* message. Further, the buyer may perform parcel tracking (*get_status* and *status* messages) of the shipped goods, which is forwarded by the accounting department to the logistics department. – The sketched scenario represents a *process choreography*, i.e., a conversation between partner processes. More precisely, the partners exchange messages via their *public* processes, which constitute special views on the associated *private* processes. In our approach we describe private processes based on BPEL4WS. Public processes are represented using annotated Finite State Automata (aFSA) in order to be able to reason about the correctness of choreography definitions and changes [7].

As motivated process-oriented information systems have to be continuously adapted. As long as the modified processes are not part of a process choreography, change effects can be kept local. The same applies if changes of a private process have no impact on related public processes. In general, however, we cannot assume this. Regarding process choreographies the modification of a private process may not only influence related public processes, but also the public and private processes of partners. As an example take an activity inserted into a private process and invoking an external operation of a partner process (by sending a corresponding message to it). If the partner process is not adapted accordingly (e.g., by inserting a receive activity to the respective BPEL flow processing the message sent) execution of the modified process choreography could fail. Thus it is crucial to provide adequate methods to *propagate* changes of a private process to partner processes.

Fig. 1b depicts our overall approach for the controlled evolution of process choreographies. Assume that Private Process 1 (left side of the figure) is modified. Then, at first, the public view on this process is recreated in order to reflect changes that might affect the interactions with partner

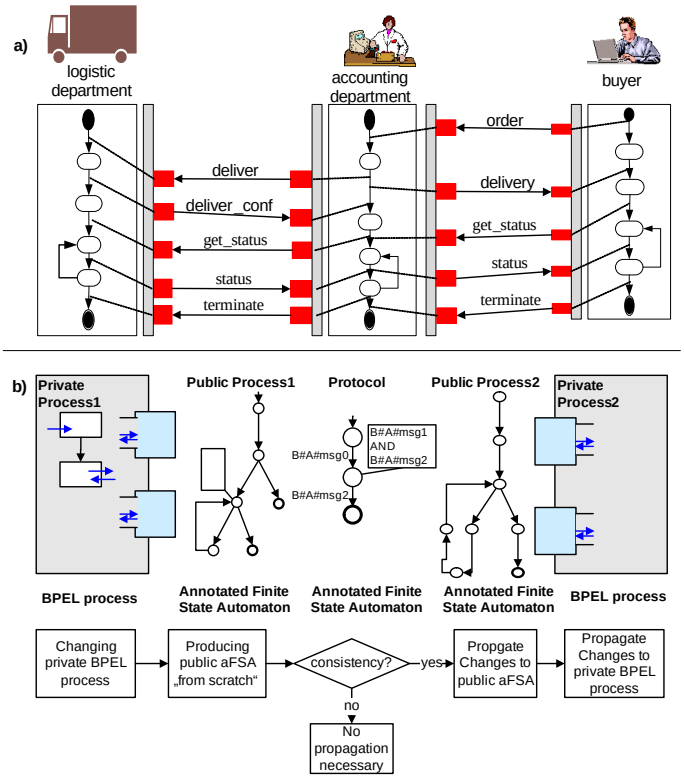

Figure 1. Example Overview

processes, i.e., the corresponding aFSA is automatically rebuilt. If this results in a modification of Public Process 1 (and only then) we further check whether adaptations of Public Process 2 (right side of the figure) become necessary as well. This is accomplished by calculating the consistency of the two public processes, i.e., the guarantee of a deadlock free execution of the interaction (see [7] for details). In case of inconsistency the change of Public Process 1 has to be propagated to Public Process 2; otherwise the execution of the process choreography will fail.

The used consistency criterion is based on non-empty intersection of the public partner processes represented as aFSAs. If the intersection between the aFSAs becomes empty the successful partner interaction is no longer guaranteed. Therefore Public Partner Process 2 has to be adapted. We call such changes *variant* whereas changes not affecting the partner interaction are called *invariant* changes [7]. We exploit the semantics of the applied changes to automatically adapt Public Process 2 in such a case. So far we distinguish between additive and subtractive changes with respect to the number of messages sent or received by the particular partner. After having performed modifications the adaptation of Private Process 2 becomes necessary as well. However, due to the autonomy of the partners and due to the privacy of the

mission critical business decisions (represented in the private process), an automatic adaptation of private processes is generally not desired. Nevertheless the system should adequately assist process engineers in accomplishing this task by suggesting respective adaptations of Private Process 2.

3. Related Work

Related work exists for consistency checking and dynamic changes in workflows. With regard to consistency checking, there are several approaches based on different models using centralized decision making. However, a centralized solution is not appropriate in the addressed scenario, rather local decisions based on bilateral knowledge are required. There exist alternative approaches to the presented one based on aFSAs (e.g., [1, 3, 4]). However, they require centralized decision making and are also not constructive; i.e., they only specify criteria for various notions of consistency but do not provide an approach to adapt public processes in a way making the overall cross-organizational process consistent. In addition, these approaches neither address synchronous communication nor allow for decentralized consistency checking.

Issues related to the dynamic change of workflows have been investigated in great detail in literature (e.g., [2, 5]). Respective approaches address ad-hoc changes of single process instances as well as process schema evolution (i.e., the controlled change of process types and the propagation of these modifications to already running process instances [2, 5]). However, these approaches focus on the adaptation of process orchestrations, i.e., process instances controlled by a single endpoint. By contrast, issues related to changes of process choreographies have been neglected so far. What can be learned from approaches dealing with dynamic changes of process orchestrations is the idea of controlled change propagation. These approaches aim at propagating process type changes to running process instances without loosing control, i.e., without causing inconsistencies or errors in the sequel. Similarly, we have provided an approach for the controlled propagation of the changes of private processes within a choreography to the choreography itself and the respective partner processes.

4. Conclusion and Future Work

The controlled evolution of private processes, the correct adaptation of related public views, and the effective propagation of these changes to partner processes will be key ingredients of future service-oriented infrastructures, ultimately resulting in highly adaptive process choreographies. Together with our previous work on process choreographies [8, 9] and process evolution [6, 5] the sketched framework will enable a powerful approach for realizing adaptive, cross-organizational business processes.

In this paper we have sketched an approach on structural process changes. In particular, we sketeche our framework for changing private processes, for recalculating related public views automatically, and for propagating resulting modifications to partner processes if required. The very important aspects of our work are its practical relevance and its formal foundation. In [7] we have provided a formal model and precise criteria allowing us to automatically decide which adaptations become necessary due to changes of private partner processes. The treatment of different change scenarios adds to the completeness of our approach. Finally, we have implemented the basic mechanisms presented in this paper in a proof-of-concept prototype.

In future work we will extend the described concepts by the treatment of running process instances (participating in a choreography) when changing private and public process models. This work will be based on concepts developed in the ADEPT project [5, 6], where we realized advanced concepts for the controlled evolution of process schemes and the dynamic migration of related process instances to new schema versions.

References

[1] W. Aalst. Interorganizational workflows: An approach based on message sequence charts and petri nets. *Systems Analysis - Modelling - Simulation*, 34(3):335–367, 1999.

[2] F. Casati, S. Ceri, B. Pernici, and G. Pozzi. Workflow evolution. *DKE*, 24(3):211–238, 1998.

[3] X. Fu, T. Bultan, and J. Su. Realizability of conversation protocols with message contents. In *Proc. IEEE Intl. Conf. on Web Services (ICWS)*, pages 96–103, 2004.

[4] E. Kindler, A. Martens, and W. Reisig. Inter-operability of workflow applications: Local criteria for global soundness. In *Business Process Management, Models, Techniques, and Empirical Studies*, pages 235–253. Springer-Verlag, 2000.

[5] S. Rinderle, M. Reichert, and P. Dadam. Correctness criteria for dynamic changes in workflow systems – a survey. *DKE*, 50(1):9–34, 2004.

[6] S. Rinderle, M. Reichert, and P. Dadam. Flexible support of team processes by adaptive workflow systems. *Distributed and Parallel Databases*, 16(1):91–116, 2004.

[7] S. Rinderle, A. Wombacher, and M. Reichert. On the controlled schema evolution of process choreographies. Technical Report TR-CTIT-05-47, University of Twente, 2005.

[8] A. Wombacher, P. Fankhauser, B. Mahleko, and E. Neuhold:. Matchmaking for business processes based on choreographies. *Intl. Journal of Web Services*, 1(4):14–32, 2004.

[9] A. Wombacher, P. Fankhauser, and E. Neuhold. Transforming BPEL into annotated deterministic finite state automata enabling process annotated service discovery. In *Proc. of Intl. Conf. on Web Services (ICWS)*, pages 316–323, 2004.