Adaptive Process Management with ADEPT2

Manfred Reichert
University of Twente, IS Group
7500 AE Enschede, The Netherlands
m.u.reichert@cs.utwente.nl

Stefanie Rinderle, Ulrich Kreher, Peter Dadam
University of Ulm, DBIS Group
89069 Ulm, Germany
{rinderle,kreher, dadam}@informatik.uni-ulm.de

1. Introduction

In the ADEPT project we have been working on the design and implementation of a next generation process management software for several years [1, 2]. Based on a conceptual framework for dynamic process changes, on novel process support functions, and on advanced implementation concepts, the developed system enables the realization of adaptive, process-aware information systems (PAIS).

Basically, process changes can take place at the type as well as the instance level: Changes of *single process instances* may have to be carried out in an ad-hoc manner (e.g., to deal with an exceptional situation) and must not affect system robustness and consistency. *Process type changes*, in turn, must be quickly accomplished in order to adapt the PAIS to business process changes. This may require the concomitant migration of thousands of instances to the new process schema (if desired). Important requirements are to perform respective migrations on-the-fly, to preserve correctness, and to avoid performance penalties.

2. Change Features of ADEPT2

ADEPT2 offers powerful concepts for modeling, analyzing, and verifying process schemes. Particularly, it ensures schema correctness, like the absence of deadlock-causing cycles or erroneous data flows. This, in turn, constitutes an important prerequisite for dynamic process changes as well. In detail, ADEPT2 supports both ad-hoc changes of single process instances and the progagation of process type changes to running instances:

Ad-hoc changes of single instances: We support different kinds of ad-hoc deviations from the pre-modeled process template (e.g., to insert, delete, or shift activities). Such ad-hoc changes do not lead to an unstable system behavior, i.e., none of the guarantees achieved by formal checks at buildtime are violated due to the dynamic change. ADEPT2 offers a complete set of operations for defining changes at a high semantic level and ensures correctness by introducing pre-/post-conditions for these operations. All complexity associated with the adaptation of instance states, the remapping of activity parameters, or the problem of missing data (e.g., due to activity deletions) is hidden from users.

Process type changes and change propagation: In order to deal with business process changes we enable quick and efficient schema adaptations at the process type level (*schema evolution*). In particular, it is possible to *propagate* type changes to running instances (of this type) as well. We provide a comprehensive correctness criterion for deciding on the *compliance* of process instances with a modified type schema. This criterion is independent of the used process meta model and is based on a relaxed notion of trace equivalence [2]. It considers control as well as data flow changes, and it works correctly in connection with loop backs. In order to enable efficient compliance checks, for each change operation we provide precise and easy to implement *compliance conditions* (cf. Fig. 1). Finally, efficient procedures exist for adapting the states of instance when migrating them to the new schema (cf. Instance I_1 in Fig. 1).

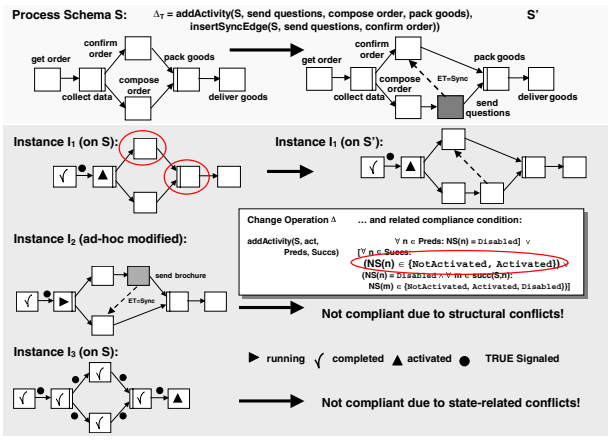

Figure 1. *Migration Process (Example)*

The correct interplay between concurrent type and instance changes is indispensable to provide real benefit for

Proceedings of the 21st International Conference on Data Engineering (ICDE 2005)
1084-4627/05 $20.00 © 2005 **IEEE**

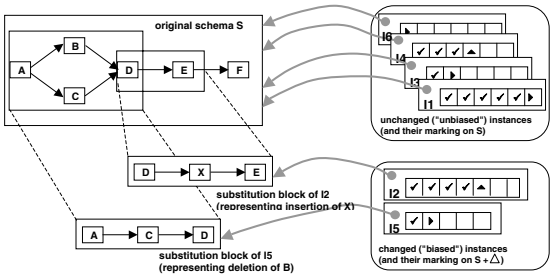

Figure 2. *Managing Schema and Instance Data*

practical applications. Therefore, we have also dealt with the question how to propagate type changes to running instances that may be in different states and may have undergone preceding ad-hoc modifications. For such "biased" instances, the current execution schema differs from the original one. We apply a comprehensive correctness principle in this context, which excludes state-related, structural, and semantical conflicts. Fig. 1 shows an example: Instance I_2 has been individually modified such that type change Δ_T cannot be applied to it; otherwise the resulting instance schema would contain a deadlock-causing cycle.

ADEPT2 comprises a number of buildtime and runtime components. They support the correct modeling of process schemes, enable (distributed) process control, prove the feasibility of dynamic process changes (also in case of distributed process control), indicate which interfaces are required, and show how the different concepts work in conjunction with each other.

The implementation of ADEPT2 has raised many challenges, e.g., with respect to storage representation of schema and instance data: Unchanged instances are stored in a redundant-free manner by referencing their original schema and by capturing instance-specific data (e.g., activity states). As an example, consider instances I_1, I_3, I_4, and I_6 from Fig. 2. For changed ("biased") instances, this approach is not applicable. One alternative would be to maintain a complete schema for each biased instance, another to materialize instance-specific schemes on the fly. We follow a hybrid approach: For each biased instance we maintain a minimal substitution block that captures all changes applied to it so far. This block is then used to overlay parts of the original schema when accessing the instance (I_2 and I_5 in Fig. 2).

3. Demo Description

In our prototype the effects of ad-hoc instance modifications can be visualized by a special monitoring component. The same applies for process type changes (cf. Fig. 3). Users define a new process type by introducing a respective process template. Based on such a template new instances can be created and new schema versions be derived; Fig. 3 shows version $V2$ of an *online ordering process*. After committing the change, the system automatically checks compliance conditions and reports migration results to the user (cf. Fig. 3). This report summarizes which instances are compliant with the new schema version. For non-compliant instances the report indicates state-related (I_3 in Fig. 3) or structural conflicts (I_2 in Fig. 3).

Fig. 3 also illustrates the interplay between type and instance changes: I_2 has been individually modified. Due to a structural conflict, it cannot be migrated to the new schema version and therefore remains running on the old one.

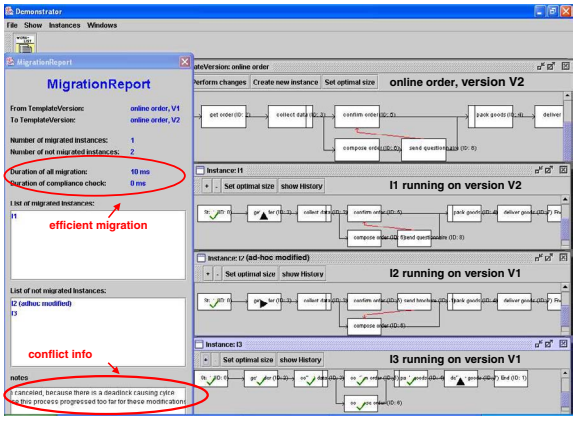

Figure 3. *ADEPT Prototype: Migration Process*

ADEPT2 is one of the few available research prototypes for adaptive, high-performance process management. In order to gain usability experience we have deployed this system to different research groups [3, 4]. They have used it as platform for realizing advanced PAIS in domains like e-health and e-business. The experiences made have helped us to refine our conceptual framework and to develop new system components with advanced programming interfaces.

References

[1] M. Reichert, P. Dadam: *ADEPT$_{flex}$ – Supporting Dynamic Changes of Workflows Without Losing Control.* JIIS 10(2): 93–120 (1998).

[2] S. Rinderle, M. Reichert, P. Dadam: *Flexible Support Of Team Processes By Adaptive Workflow Systems.* Distributed and Parallel Databases, 16(1):91–116 (2004).

[3] S. Bassil, R. Keller, P. Kropf: *A Workflow-oriented System Architecture for the Management of Container Transportation.* Proc. BPM'04, Potsdam, June 2004, pp. 116–131.

[4] R. Müller, U. Greiner, E. Rahm: *AgentWork: A Workflow System Supporting Rule-based Workflow Adaptation.* Data & Knowledge Engineering, 51(2): 223–256.