Towards a Framework for the Agile Mining of Business Processes

Barbara Weber[1], Manfred Reichert[2], Stefanie Rinderle[3], and Werner Wild[4]

[1] Quality Engineering Research Group, University of Innsbruck, Austria
Barbara.Weber@uibk.ac.at
[2] Information Systems Group, University of Twente, The Netherlands
m.u.reichert@cs.utwente.nl
[3] Dept. Databases and Information Systems, University of Ulm, Germany
rinderle@informatik.uni-ulm.de
[4] Evolution Consulting, Innsbruck, Austria
werner.wild@evolution.at

Abstract. In order to support business processes effectively, their implementation by a process management systems (PMS) must be as close to the real world's processes as possible. Generally, it is not sufficient to analyze and model a business process only once, and then to handle respective business cases according to the defined model for a long period of time. Instead, process implementations must be quickly adaptable to changing needs. A PMS should enable process instance changes and provide facilities for analyzing these instance-specific changes in order to derive optimized process models. In this paper we introduce a framework for the agile mining of business processes which supports the whole process life cycle in an integrated way. Our framework is based on process mining techniques, adaptive process management, and conversational case-based reasoning. On the one hand, it allows annotating execution and change logs with semantical information to gather information about the reasons for ad-hoc deviations, which can then be analyzed by the process engineer (with support from the PMS). On the other hand, it enables the process engineer to adapt process models based on the outcome of these analyses and to migrate related process instances to the new model.

1 Introduction

Companies are developing a growing interest in aligning their information systems in a process-oriented way. Recently, process mining, in particular Delta analysis [1,2], has been proposed to improve business alignment [3]. If no process models are available yet, process mining techniques can be applied to identify repetitive process fragments. However, if the business processes are already captured in process models, Delta analysis can help to detect discrepancies between the modeled and the observed execution behavior of a process. Though process execution logs can be used to reveal malfunctions or bottlenecks, they do not provide any semantical information about the reasons for the observed

C. Bussler et al. (Eds.): BPM 2005 Workshops, LNCS 3812, pp. 191–202, 2006.

discrepancies. In respect to the optimization of the process models these logs therefore provide only limited information to process engineers. Furthermore, current PMS do not sufficiently support process engineers to incorporate the results of a Delta analysis into an improved process model and to smoothly migrate running process instance to the new model version.

Obviously, the practical benefit of process mining depends on the quality of the available log data. In PMS, for instance, respective execution logs can only reflect situations the PMS is able to handle. Particularly, if the PMS does not support process instance changes it has to be bypassed in exceptional situations (e.g., by executing unplanned process activities outside the scope of the PMS). Consequently, the PMS is unaware of the applied deviations and thus unable to log information about them. This missing traceability of process instance changes significantly limits the benefits of process mining and Delta analysis approaches.

Continuous process improvement requires adaptive PMS which enable authorized users to flexibly deviate from premodeled processes as needed. Since this results in more meaningful execution logs, which implicitly reflect the applied process instance changes, process mining and Delta analysis approaches become more useful. If such analyses result in an optimized version of a process model, adaptive PMS support the process engineer to quickly implement the model change and smoothly migrate running process instances to the new model.

In addition to process execution logs, adaptive PMS maintain information about applied instance modifications in change logs. Minimally, a change log should keep syntactical information about the kind and the context of the applied changes (e.g., the type and position of a dynamically inserted process activity). While this information is useful for process mining, it is not sufficient to effectively support process optimization efforts, process engineers also need semantical information about the reasons for the change. This is particularly important if the same or similar instance changes happen over and over again. Assume that in a patient treatment process an unplanned lab test is dynamically added for a considerable number of process instances. Then the respective change logs should also reflect information about the semantical context of the applied instance changes (e.g., that insertions have been mainly performed for patients older than 40 years and suffering from diabetes).

In this paper we introduce a framework for the agile mining of business processes which supports the whole process life cycle in an integrated way and fosters continuous and quick adaptation to change. Our framework is based on process mining [3], adaptive process management (PM) [4,5], and, to close the semantic gap, conversational case-based reasoning (CCBR) [6]. CCBR is an interactive extension of the case-based reasoning (CBR) paradigm [7]. In particular, we combine the advantages of these three approaches in an integrated prototype. We enhance execution and change logs with semantical information, which is then analyzed by the process engineer with support from the PMS to provide knowledge about the context of and the reasons for discrepancies between process models and related instances. This semantical information is also reused to support users when similar deviations become necessary. Finally, our framework

enables the process engineer to adapt process models based on the outcome of process mining efforts and to smoothly migrate related process instances to the new model version.

Section 2 describes building blocks for the agile mining of business processes. Section 3 gives an overview of our framework and Section 4 discusses a sample application. The paper closes with a summary and an outlook in Section 5.

2 Building Blocks for Agile Mining of Business Processes

This section summarizes main characteristics of our framework's building blocks: process mining, case-based reasoning, and adaptive PM.

2.1 Process Mining

Process mining denotes the extraction of process knowledge from log data related to past process and application executions (cf. Fig. 1). Respective logs are usually provided by workflow systems, but also by other process-oriented applications, like enterprise resource planning or supply chain management systems. Typically, all these systems log event-based data (e.g., related to the start or completion of task executions) together with additional context information (e.g., about actors).

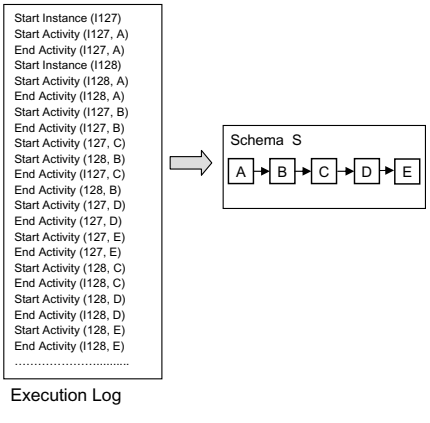

Fig. 1. Process Mining

The main objective of process mining is to effectively use automatically collected data in order to gain process knowledge from the logs. In particular, process mining extracts formal process models from execution logs [3,8,9,10,11]. So far, the focus has been put on issues related to control-flow mining. However, first approaches have been developed which use event-based data for mining organizational and performance aspects as well [12,13]. Process mining is generally

considered as an alternative to interviews or questionnaires to acquire process knowledge. Its results can be used for further analysis. For instance, Delta analysis [1,2] compares existing process models with the results of process mining in order to detect discrepancies between the modeled and the observed execution behavior. This information is then used to improve the process model.

2.2 Case-Based Reasoning

Case-based reasoning (CBR) is a contemporary approach to problem solving and learning [7]. New problems are dealt with by drawing on past experiences – described in cases – and by adapting their solutions to the new problem situation (cf. Fig. 2).

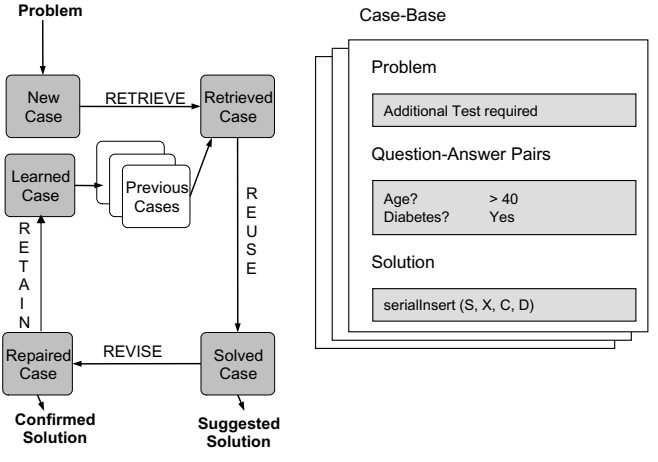

Fig. 2. Case-Based Reasoning

Reasoning based on past experiences is a powerful and frequently applied way to solve problems by humans [14]. A physician, for example, remembers previous cases to determine the disease of a newly admitted patient. A case is a contextualized piece of knowledge representing an experience [7], which typically consists of a problem description and the corresponding solution.

Our framework applies conversational CBR, an extension of the CBR paradigm, which actively involves users in the inference process [15]. CCBR systems are interactive systems that, via a mixed-initiative dialogue, guide users through a question-answering sequence in a case retrieval context. Unlike traditional CBR, CCBR neither requires users to provide a complete a priori problem specification for case retrieval nor to provide knowledge about the relevance of each feature for problem solving. Instead, the system assists users in finding relevant cases by presenting a set of questions to assess a given situation. Furthermore, it guides users who may supply already known information on their

initiative. Therefore, CCBR is especially suitable for handling exceptional or unanticipated situations which cannot be dealt with in a fully automated way.

CBR has been applied to PM for several purposes [16]. Our research prototype CBRFlow, for example, uses CCBR to perform ad-hoc changes of single process instances, to memorize these changes, and to support their reuse in similar future situations [16].

2.3 Adaptive Process Management

Adaptive PM technology increases the flexibility of process-oriented information systems by enabling (dynamic) changes of different process aspects (e.g., control and data flow). Such process changes can be performed at two levels – the *process type* and the *process instance* level [5,17] (cf. Fig. 3).

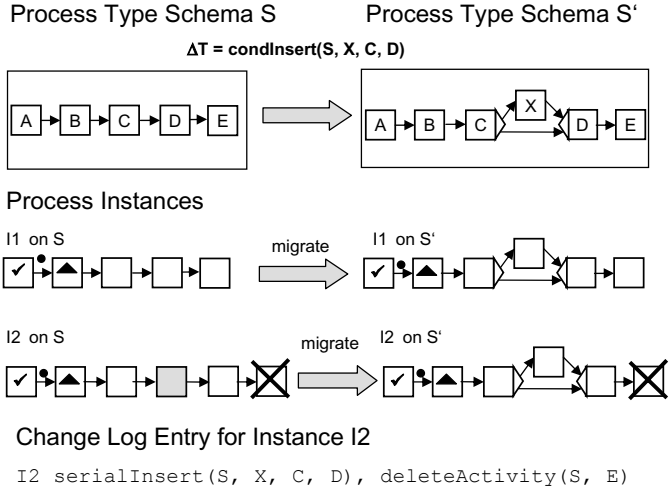

Fig. 3. Adaptive Process Management

When a process template or, more precisely, the *process type schema* representing this template is changed, the respective changes should be propagated to already running process instances [18]. This has to be done in a correct and consistent manner, and respective *migration procedures* must efficiently handle a high number of concurrently running process instances. In this context it must be possible to propagate process type changes to both unbiased and biased process instances. We denote process instances as *unbiased* if they are running according to the original process type schema they were derived from, whereas process instances are denoted as *biased* if they have been individually modified (e.g., due to an ad-hoc change) [17].

Instance–specific ad-hoc changes (e.g., switching the order of two activities or adding new ones) must be performed to deal with exceptional situations [4].

Usually, they are stored in change logs, which provide information about the type of the applied change and related context information (e.g., the position of a newly inserted activity). This log information contributes to the analysis of potential conflicts between process type and process instance changes as well as to the documentation of the instance history. The described functionality has been implemented in our ADEPT PMS [4,5,17,18].

3 Framework for Agile Mining of Business Processes

The implementation of a framework for the agile mining of business processes raises a number of challenges. This section outlines basic requirements (Section 3.1), gives an overview of the designed framework (Section 3.2), and describes the current state of its implementation (Section 3.3).

3.1 Requirements for an Agile Process Mining Framework

An agile process mining framework must provide comprehensive facilities for business process monitoring and must allow quick responses to observed discrepancies between the modeled and the executed processes. In particular, process engineers should be supported in learning from previous (ad-hoc) instance changes and in deriving optimized process models from log data.

When performing a process instance change information about the reasons for the change should be kept in a case–base. This can be done by either adding a new case (when no similar change has been applied before) or by reusing an existing one (representing a previously applied, similar ad-hoc change). A pure syntactical approach is not sufficient to stimulate the reuse of change information, semantical information about the changes must be maintained as well. Consider, for example, a patient treatment process and assume that an additional activity (i.e., a lab test) has been frequently inserted for patients older than 40 years and suffering from diabetes. If such context information is explicitly maintained, respective cases can be reused in similar situations and optimized process models can be derived from instance change logs.

To continuously optimize business processes, extended mining techniques must be provided. They should use information from execution and change logs as well as the semantical information (cases) associated with the changes. When similar ad-hoc changes occur frequently enough (i.e., their occurrence exceeds a certain threshold), process engineers should be notified and assisted in optimizing the process model. Semantical change information must be represented in such a way that useful suggestions can be made. For example, assume that our lab test activity has been inserted for a significant number of process instances in the context just mentioned. When moving this change to the process type level the simple insert operation (used at the instance level) cannot be applied directly as the additional lab test activity shall only be performed if the specific conditions (older than 40 years, diabetes) are met. Therefore, the PMS should be

able to translate instance changes and related semantical information to process type changes (i.e., to respective transformations of the process type schema) and to suggest them to the process engineer.

Process engineers should not only be supported in deriving new schema versions from log data, but also in migrating already running instances to a modified schema. For this, the PMS has to check whether these instances are compliant with this new schema version or not. Depending on whether an instance is biased or unbiased (cf. Section 2.3) and depending on the degree of overlap between process type and process instance change, different migration strategies must be supported by the PMS [19]. Furthermore, the system has to migrate the semantical information associated with the process schema as well (i.e., the cases representing ad-hoc changes on instances of this schema), as only information not yet covered by the new process schema must be migrated. Information on the ad-hoc change which triggered the process type evolution should be omitted, i.e., those cases should be dropped from the new case-base version.

3.2 Overview of the Framework

In order to meet these requirements and to reflect a company's business processes adequately, process models must be continuously monitored and adapted to changing needs. For this, both execution and change logs must be enriched with semantical information. As illustrated in Fig. 4, different information sources (i.e., execution logs, change logs, and case-bases) are relevant in this context. These sources must be continuously evaluated and changes to the process model should be triggered when discrepancies between it and its instances occur frequently. Based on the information maintained in the execution logs, in the change logs, and in the case-base, the process engineer can then adapt the process model and migrate related process instances by using adaptive PM technology (cf. Section 2.3).

Execution and change logs provide the syntactical information on what happened, i.e., which activities were executed and which deviations occurred at what time. CCBR (cf. Section 2.2) is used to provide semantical information on why changes happened. More precisely, experiences from previous changes are stored as cases in a case-base. A case represents a concrete ad-hoc modification of one or more process instances, it consists of a textual problem description briefly explaining the problem that made the deviation necessary, a set of question-answer pairs, and a solution part. The question-answer pairs describe the reasons and the context of the ad-hoc deviation and the solution part reflects the concrete change operations that have been applied to the respective instance(s) to perform the desired ad-hoc change by using services of the adaptive PMS (for a more detailed description see [20,21]).

In Fig. 4 the mining of the process execution log reveals that activities A, B, C, D and E are always executed in sequence. The change log further indicates that for some process instances running on schema S the additional activity X was dynamically inserted between activities C and D. However, the change log does not provide any semantical information on why activity X was inserted. By

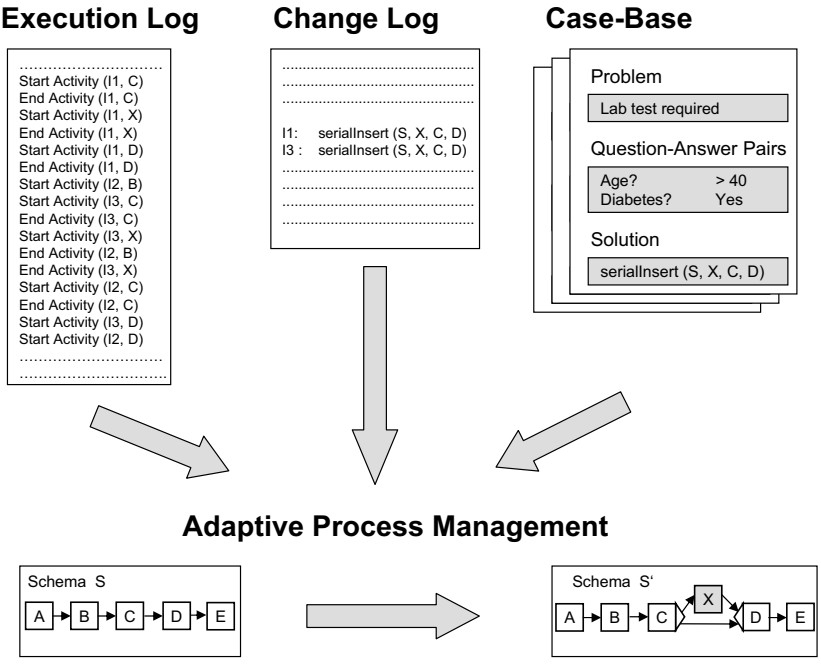

Fig. 4. Information Sources Triggering Process Evolution

analyzing the cases in the case-base, the process engineer learns that activity X was primarily added and executed for patients older than 40 years who suffers from diabetes. Based on this information he can derive a new process schema containing the additional activity X for patients matching these two conditions. Finally, after the new version of the process type schema is released, process instances can be migrated to the new schema version.

In the following we describe how the building blocks (cf. Section 2) and their respective information sources (i.e., execution log, change log and case-bases) work together to complete the process life cycle (cf. Fig. 5). An initial process schema can either be created by applying process mining techniques (i.e., by observing process and/or task executions) or by business process analysis (1). During run-time new process instances are created from such a predefined process schema (2). In general, process instances are then executed according to their process type schema and execution logs are written (3). However, when deviations from the predefined schema become necessary (e.g. due to exceptions or unanticipated situations) process actors must be able to deviate from it. They can either specify a new ad-hoc deviation and document the reasons for their changes in the CCBR subsystem, or reuse a previously specified ad-hoc modification from the case-base (4). In both situations an appropriate change log entry is written (5) and the execution continues. When a particular ad-hoc modification is frequently reused, the process engineer is notified to perform a process

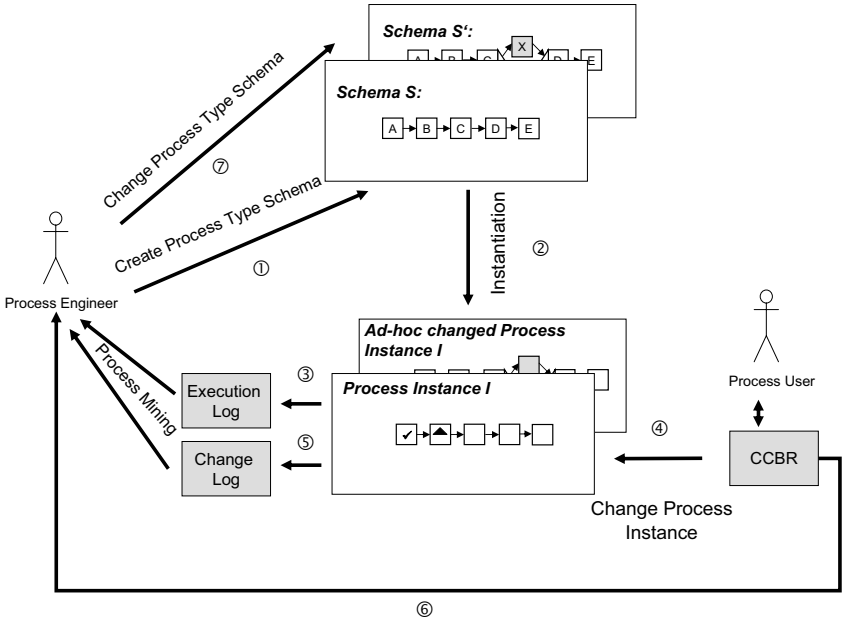

Fig. 5. Integrated Process Life Cycle Support

type change (6). Both execution and change logs are needed to know how often a particular schema has been instantiated and how frequently deviations occurred. The process engineer can then perform a schema evolution (7), as supported by the adaptive PMS, and, if possible, migrate running instances to the new schema version.

Thus, combining process mining and CCBR techniques with adaptive PM allows full process life cycle support and the seamless integration of the detected discrepancies into a new, optimized process schema.

3.3 Current State of the Framework

Our ongoing research focuses on the integration of adaptive PM technology and CCBR. We currently work on the integration of the methods, concepts, and prototypes developed by our groups in the ADEPT and the CBRFlow projects.

ADEPT [4] is a next generation PMS that offers full functionality in respect to the modeling, analysis, execution, and monitoring of business processes [4,5,18]. To our best knowledge it is the only PMS providing full support for adaptive processes at both the process instance and the process type level. When performing process instance changes, ADEPT ensures consistency and correctness. In the context of process type changes it additionally allows to migrate running process instances efficiently to the new schema version [5,17,18]. A powerful prototype demonstrating this functionality exists and is used by several research groups and industry partners.

CBRFlow [16] supports users in performing simple ad-hoc instance changes and allows them to express the semantics of these changes by applying CCBR techniques. It documents the reasons for a process instance change and enables the user to reuse information about previously performed ad-hoc deviations when creating new ones for other instances of the same process type. Like with ADEPT, a powerful prototype exists.

By integrating ADEPT and CBRFlow significant synergies can be exploited, fostering integrated process life cycle support [20,21]. On the one hand the combined system provides a powerful process engine, supporting all kinds of changes in one system. On the other hand it enables the intelligent reuse of process instance changes and fosters deriving process type changes from collected information. Currently we implement a prototype which integrates both systems; future work will include process mining tools as well [22]. We will apply and evaluate our prototype in different application settings, including healthcare processes and emergent workflows (e.g., in the automotive sector).

4 Sample Application of the Framework

Contemporary PMS rely on predefined process models requiring large upfront investments to analyze and model business processes before process-aware information systems can be deployed. It is especially difficult to create adequate process models for knowledge-intensive processes (e.g., engineering processes in the automotive sector) comprising both routinized and non-routinized work. In some cases process schemes are already obsolete when their specification is "completed". However, in today's dynamic business environment not all eventualities and deviations can be considered in advance as requirements change or evolve over time. Covering too many details in a process schema early on raises the risk of including rarely needed, not yet needed or never needed parts. For these reasons process modeling on demand and focusing on core functionality offer promising perspectives to foster shortened modeling cycles and earlier delivery of productive systems.

To cope with these risks and to ensure that the modeled process schemes closely reflect a company's business processes, short iteration cycles must be supported. Instead of starting with a completely defined process schema, the process engineer can either develop a preliminary one with a clear focus on core functionality (e.g., covering the standard process, but not all alternative paths) or apply process mining techniques to extract process fragments from event logs.

When starting with an inital process model our framework can be used to iteratively evolve the model over time. Whenever necessary, extensions to the processs schema can be performed in an ad-hoc way by using adaptive process management. CCBR is applied to document these deviations, to memorize them and to support their reuse. Case reuse in combination with execution/change logs is applied to trigger process type changes and to incorporate these deviations into respective process schemes (cf. Fig. 5).

The framework also provides support for evolving process fragments which have been extracted by process mining techniques. Initially, new process fragments may only reflect a partial view on a process, i.e., the derived models are not mature. However, they already represent meaningful process knowledge which can be evolved over time and be reused in a different context (e.g., to automate routine work or to derive more comprehensive process templates). For example, engineers may want to customize fragments to individual needs or combine them to come up with more complete process views reflecting complex processes. When adapting or extending process fragments in such a way, new process knowledge is created, which again could be reused, harvested, and linked with existing fragments.

In this scenario fragment creation, storage, reuse, composition, and execution are important tasks, our framework provides fundamental support to deal with them. Initially, process fragments are derived by applying process mining techniques; the resulting models can then be semantically annotated using CCBR before they are stored in the process repository. When a user wants to retrieve a fragment, he is guided through a question-answering sequence by the CCBR sub-system. When a fragment is selected, its reuse counter is increased. Additionally, quality ratings from users are aggregated in a reputation score. A special challenge is the execution of incomplete fragments which are to be completed or adapted; for this, adaptive process management technology is key to success. Altogether, our framework provides the needed adaptation and mining techniques to support such evolving and emergent processes.

5 Conclusion and Outlook

We have introduced a framework for the agile mining of business processes based on CCBR, adaptive PM and process mining. The main focus has not been put on the development of new concepts in these domains, but on the integration of existing methods, concepts and prototypes (ADEPT and CBRFlow) and on the support of business process intelligence. We have sketched the basic relationships between the different components, discussed technical requirements for providing an integrated solution, and drafted the approach we take. Currently we work on the implementation of the framework presented, starting with the integration of ADEPT and CBRFlow. In this context we have also developed concepts for process evolution and the migration of related case-bases [20,21]. In the next step we want to incorporate process mining tools into our framework.

References

1. Guth, V., Oberweis, A.: Delta analysis of petri net based models for business processes. In: Proceedings of International Conference on Applied Informatics. (1997) 23–32
2. v.d. Aalst, W.: Inheritance of business processes: A journey visiting four notorious problems. In: Petri Net Technology for Communication Based Systems. LNCS 2472 (2003) 383–408

3. v.d. Aalst, W., van Dongen, B., Herbst, J., Maruster, L., Schimm, G., Weijters, A.: Workflow mining: A survey of issues and approaches. Data and Knowledge Engineering **27** (2003) 237–267
4. Reichert, M., Dadam, P.: ADEPT$_{flex}$ - supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems **10** (1998) 93–129
5. Rinderle, S., Reichert, M., Dadam, P.: Flexible support of team processes by adaptive workflow systems. Distributed and Parallel Databases **16** (2004) 91–116
6. Aha, D.W., Breslow, L., Muñoz-Avila, H.: Conversational case-based reasoning. Applied Intelligence **14** (2001) 9–32
7. Kolodner, J.L.: Case-Based Reasoning. Morgan Kaufmann (1993)
8. Golani, M., Pinter, S.S.: Generating a process model from a process audit log. In: Proceedings of International Conference on Business Process Management (BPM'03), Eindhoven (2003) 136–151
9. Herbst, J.: An inductive approach to adaptive workflow systems. In: Proc. Workshop Towards Adaptive Workflow Systems (CSCW'98), Seattle (1998)
10. de Medeiros, A., van der Aalst, W., Weijters, A.: Workflow mining: Current status and future directions. In: Proceedings of International Conference on Cooperative Information Systems (CoopIS'03). LNCS 2888, Berlin (2003) 389–406
11. van Dongen, B., van der Aalst, W.: Multi-phase process mining: Building instance graphs. In: Proceedings of International Conference on Conceptual Modeling (ER 2004). LNCS 3288, Berlin (2004) 362–376
12. van der Aalst, W., Song, M.: Mining social networks. uncovering interaction patterns in business processes. In: Proceedings of International Conference on Business Process Management (BPM'04), Potsdam, Germany (2004) 244–260
13. van der Aalst, W., van Dongen, B.: Discovering workflow performance models from timed logs. In: International Conference on Engineering and Deployment of Cooperative Information Systems (EDCIS 2002). LNCS 2480, Berlin (2003) 45–63
14. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations and system approaches. AI Communications **7** (1994) 39–59
15. Aha, D.W., Muñoz-Avila, H.: Introduction: Interactive case-based reasoning. Applied Intelligence **14** (2001) 7–8
16. Weber, B., Wild, W., Breu, R.: CBRFlow: Enabling adaptive workflow management through conversational case-based reasoning. In: Proceedings of European Conference on Case–based Reasoning (ECCBR'04), Madrid (2004) 434–448
17. Rinderle, S., Reichert, M., Dadam, P.: On dealing with structural conflicts between process type and instance changes. In: Proceedings of International Conference on Business Process Management (BPM'04), Potsdam (2004) 274–289
18. Rinderle, S., Reichert, M., Dadam, P.: Correctness criteria for dynamic changes in workflow systems – a survey. Data and Knowledge Engineering **50** (2004) 9–34
19. Rinderle, S., Reichert, M., Dadam, P.: Disjoint and overlapping process changes: Challenges, solutions, applications. In: Proceedings of International Conference on Cooperative Information Systems (CoopIS'04), Agia Napa (2004) 101–120
20. Weber, B., Rinderle, S., Wild, W., Reichert, M.: CCBR–driven business process evolution. In: Proceedings of International Conference on Case-Based Reasoning (ICCBR'05), Chicago (2005) 610–624
21. Rinderle, S., Weber, B., Reichert, M., Wild, W.: Integrating process learning and process evolution - a semantics based approach. In: Proceedings of International Conference on Business Process Management (BPM'05). (2005) 252–267
22. Process Mining Research: www.processmining.org (2005)