



# Towards Case-Based Deviation Management for Flexible Workflows

Lisa Grumbach<sup>1,2</sup>  and Ralph Bergmann<sup>2</sup> 

<sup>1</sup> Trier University of Applied Sciences, Location Birkenfeld, Campusallee,  
55761 Birkenfeld, Germany

[l.grumbach@umwelt-campus.de](mailto:l.grumbach@umwelt-campus.de)

<sup>2</sup> University of Trier, Department of Business Information Systems II,  
54286 Trier, Germany

[bergmann@uni-trier.de](mailto:bergmann@uni-trier.de)

**Abstract** This work introduces potentials for a case-based approach to deviation management in the context of flexible workflows. A constraint-based workflow engine is shortly introduced with some strategies for deviation handling. Three different research works about case-based methods and sequence similarity are described that may be adapted adequately to fit our use case of deviation management. Finally, not only challenges that need to be addressed in future work are sketched, but also potentials of the pursued approach are discussed.

**Keywords:** Case-Based Reasoning · Flexible Workflow Management · Flexibility by Deviation · Sequence Similarity.

## 1 Introduction

Digitalization is advancing in today's business. Small and medium-sized enterprises (SMEs) appear to be lagging behind compared to large companies. Process-aware information systems (PAIS, [1]) are well established for standardized processes, but they lack support for flexibility, which is often required in SMEs and may lead to competitive advantages. Additionally, in SMEs there are often few experts which are responsible for certain processes. The knowledge about how things are done is implicit, but not stored digitally and information is simply shared orally. As stated by da Silva et al. [23], these experts usually deviate and perform processes due to their expertise and experiences without losing control or missing the objective, but rather optimizing the process. Tracing these processes automatically and using them for process control may simplify the transfer and preservation of "best practices". This in turn may lead to an increase of efficiency and enhanced assistance possibilities especially for inexperienced users. Furthermore, bypassing the system and thus a loss of knowledge and transparency is prevented.

---

Copyright © 2019 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

A main characteristic of an adequate approach for SMEs is to allow unexpected deviations and support unforeseen changing circumstances. Thereby, the key challenge is to determine how to continue with the workflow, while still achieving a successful completion. For an automated handling of such deviations without the demand for expert knowledge that needs to be integrated manually, an option is to use past experiences, more specifically completed processes. Applying case-based reasoning (CBR) to solve the described problem seems to be promising, as previous made experiences are exploited for adapting to unknown situations. Our objective is to develop an approach based on case-based reasoning that detects and handles deviations which occur in a flexible workflow, but still guides the user by recommending adequate work items.

The remainder of the paper is organized as follows. The notions of workflow flexibility and deviation management including related work will be introduced in section 2. Our approach of flexible workflow management allowing deviations is presented in section 3, while the proposed case-based deviation management including three promising methods, that could be exploited, are outlined in chapter 4. We conclude with a summary and an insight into our future work.

## 2 Foundations and Related Work

In this section our previous work on workflow flexibility and a specification of the underlying terminology will be presented. Furthermore the notion of deviation will be introduced and related work will be sketched.

### 2.1 Workflow Flexibility

Flexible Workflows have been focussed in research for more than a decade [22]. Schonenberg et al. [22] distinguish four different flexibility principles. *Flexibility by Design*, *Change* and *Underspecification* either require an entire awareness about all possible upcoming situations at design-time in order to manually model all possible alternatives, or a remodeling of the workflow is necessary at run-time. *Flexibility by Deviation* in contrast “is the ability for a process instance to deviate at run-time from the execution path prescribed by the original process without altering its process definition” [22]. Thus, single instances may not fit to the process model. Therefore we explicitly distinguish between modeled workflow, denoted as *de jure*, and executed workflow, called *de facto* [1]. Based on this definition we developed a workflow engine that facilitates flexibility by deviation.

### 2.2 Constraint-Based Workflow Engine

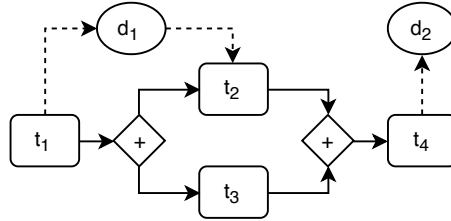
During the SEMAFLEX<sup>3</sup> [12] and the SEMANAS<sup>4</sup> [13] project we developed a flexible workflow engine based on constraints. We presented an approach [10,11],

<sup>3</sup> SEMAFLEX is funded by Stiftung Rheinland-Pfalz für Innovation, grant no. 1158

<sup>4</sup> SEMANAS is funded by the Federal Ministry of Education and Research (BMBF), grant no. 13FH013IX6

that allows flexible deviations from prescribed workflows, but still maintains control and recommends valid work items to a limited extent.

The proposed method is applied to imperatively modeled workflows, restricted to block-oriented ones. The paradigm of regular block structuring introduced by [20] is applied to guarantee model correctness by construction [16] and achieve a simplified handling of workflow models. Block-oriented workflows are constructed through a single root block element, which in turn consists of a single task node, a sequence of two block elements or a control-flow block. Start and end of control-flow blocks are clearly defined through control-flow nodes. An example of a block-oriented workflow is shown in Fig. 1.



**Figure 1.** Example Block-Oriented Workflow

The exemplary workflow consists of four task nodes (rectangles), two data nodes (ovals) and two control-flow nodes (rhombuses), which represent a parallel control-flow block (“+”). Additionally, the edges denote either control-flow (solid lines) or data-flow (dashed lines, input or output relation).

In our approach we transform these imperative workflow models into declarative constraints, which indicate sequential dependencies, to be able to determine task activations and thus possible executions in a specific un-terminated state of the workflow. A constraint satisfaction problem (CSP) is constructed on the basis of these generated constraints and logged task enactments. A solution of the CSP is searched for, which tries to calculate a valid sequential enactment of all already executed and possible future executions of tasks. Thus, with a solution we are able to recommend valid task enactments. Consider the example of Fig. 1, the constraint set as logical formula is constructed as follows:  $t_1 < t_2 \wedge t_1 < t_3 \wedge t_2 < t_4 \wedge t_3 < t_4$ . If task  $t_1$  is executed, a value is assigned, in this case  $t_1 = 1$ , and added to the constraint set. Task recommendations are calculated by regarding possible task assignments of the next sequential value, in this case 2. Considering the constraint set either with  $t_2 = 2$  or with  $t_3 = 2$  a solution is found. Thus,  $t_2$  and  $t_3$  are added as work items to the work list.

An additional advantage of using a CSP is that it is relatively easy and fast to retract violated constraints at run-time in order to restore consistency in case of a deviation. Still, by regarding the remaining constraints valid solutions can be computed. In our work, we therefore described a method, which detects deviations and retracts violated constraints to restore consistency and re-enable the

workflow engine to recommend work items. But, a categorization of deviations and a determination of the cause is not considered until now. Thus, in addition to the restoration of consistency and overcoming a possible deadlock through system failure, no adequate reaction is possible. Different strategies for resolving inconsistencies, that are implemented until now, and their limitations are presented in section 3.1. Due to these limitations, deviation management needs to be elaborated in more detail and we propose a case-based approach.

### 2.3 Deviation Management

The notion of deviation is not clearly defined and often included in the term exception, as a special type. To clarify our used classification of exceptions and deviations several specifications will be quoted. A process deviation is defined by da Silva et al. [23] as mismatch between executed process and process model. In contrast to exceptional behaviour deviations cannot be anticipated. According to Marrella et al. [17] exception handling is implemented manually at design-time, whereas deviation handling requires ad-hoc changes at run-time. Eder et al. [8] describe unexpected exceptions, what we call deviations, as an important aspect which should be handled to gain possible benefits.

Deviation Management in general can be split up into 3 phases: deviation detection, deviation handling and deviation analysis. Most of the related work focus on the detection and handling of deviations by either adapting running workflow instances or proposing corrective measures. Zhu et al. [27] developed a process behaviour space expression based on algebra to identify deviations and provide simple methods like tolerating or adjusting deviations that handle such exceptions to a limited extent.

Several existing approaches utilise ECA rules [4,7,19] to determine a reaction to detected deviations. Da Silva et al. [23] use logical formulae as rule basis for the detection and recommend manually created correction plans. Grambower et al. [9] present a flexible variant of ECA rules, which is enhanced through contextual semantic information and reasoning. Actions to perform are additionally adapted on the basis of semantic knowledge. A similar approach is developed by Adams et al. [3]. The manual handling of deviation is circumvented by so-called worklets, which represent exception handling patterns that are inserted automatically in running workflow instances on the basis of contextual information.

Depaire et al. [6] focus on deviation diagnosis or as they refer to the “managerial view” and investigate and analyse characteristics of deviations on purpose of searching for control weaknesses. Workflow instances are mined and business rules representing the deviations are derived subsequently, reducing the amount of data drastically, which in turn may be analysed.

CBRFlow [25] is a system that exploits conversational case-based reasoning to react to changing circumstances. Business rules are used to model necessary workflow run-time changes and may then be recommended in similar future situations. But still the user has to intervene manually to trigger a deviation and to resolve the issues.

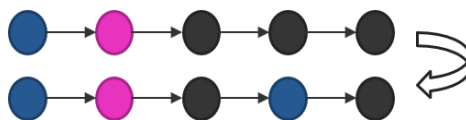
### 3 Analysis of the Deviation Management Problem

All of the presented related work considers manually created knowledge or even requires manual intervention in a running workflow for the management of deviations. As we address SMEs our aim is to automatically adapt to changing circumstances without the need of modeling process fragments or adaptation knowledge in any form. Our objective is to create an approach that relies on experience and thus previously executed workflows and exploits available data to provide further workflow control and recommendations of work items even in unexpected situations.

#### 3.1 Typical Scenarios

Two different and very simple constraint violation scenarios will be presented in combination with strategies, which we already implemented, for restoring consistency in the CSP.

**Sequential Constraint Violation** For handling detected deviations in a simple sequential workflow or in sequential parts of a workflow, we developed two different strategies for restoring consistency. Consider the example in Fig. 2, where the first row shows the initial situation and the second row represents one step further with an additional executed task. Circles indicate tasks and edges represent the execution order. Blue filled nodes were already executed, pink ones



**Figure 2.** Example Sequential Workflow with Deviation

are currently recommended and grey nodes are not activated yet. In the lower row there is a task executed, which is not in a valid order, thus a deviation occurred. The cause of the violation is not explicit without semantic knowledge and needs to be specified by the user. Without necessary user intervention we are able to derive two possible reasons, which are handled in different strategies.

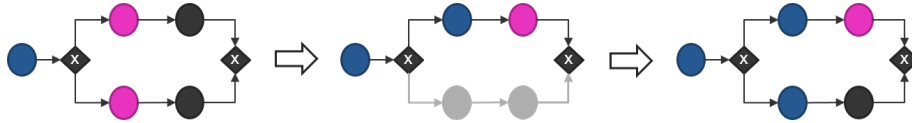
1. **Skipped Tasks.** Tasks may have become obsolete and are therefore skipped or have been executed but without notice of the system (cf. Fig. 3a). All constraints concerning skipped tasks are irrelevant for continuing with the workflow and simply may be omitted.
2. **Changed Order.** Task order may have changed due to unknown reasons. Remaining tasks need to be connected sequentially, as if the current executed task (second blue one) would have been inserted after the last executed one (first blue one, cf. new edges in Fig. 3b).



**Figure 3.** Possible Strategies for Handling Sequential Deviations

Which strategy is applied needs to be determined either prior to the start of a workflow or at run-time. If the applied strategies differ for single deviations of the same type, further user interaction is necessary for each occurrence at run-time, which reduces the advantages of allowed flexibility and an additional obstacle for an easy-to-use system emerges.

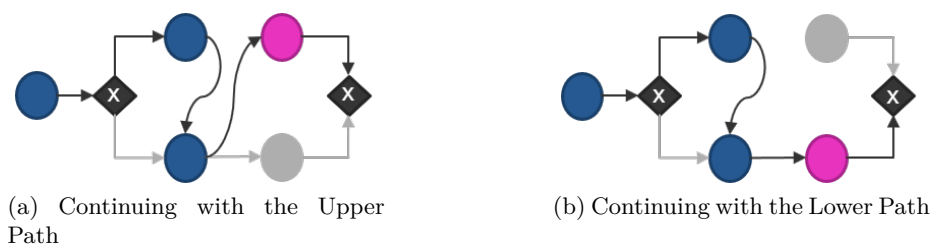
**Exclusive Constraint Violation** Another example scenario of a simple violation that might occur is depicted in Fig. 4. In this simplified sketch at first both exclusive paths might be chosen, followed by the execution of the upper task, which implies the exclusion of the lower tasks. The deviation occurs in the last step, as nevertheless a task of the lower path has been executed.



**Figure 4.** Example Workflow with Exclusive Pattern and Deviation

Which path should be pursued for workflow continuation without further knowledge can only be guessed by the system. Possible reasons are that either some task was completed by mistake or eventually certain circumstances changed that demand an adjustment and caused the execution of an excluded task or rather whole path. Constraints need to be adapted in case of such a violation according to one of the strategies pointed out in Fig. 5a and 5b.

The described scenarios have shown two types of deviations and thus constraint violations with a simple structure and a limited number of resolving possibilities. These strategies are straight-forward and easy to implement, but require user interaction. Besides, deviations might lead to additional deviations or even might be more complex from ground up, which makes it impossible to determine similar strategies to handle all possible deviation scenarios. Furthermore, it is not our aim to construct strategies, which need to be picked by the user, but rather some methods, which can be applied in an automated manner and support the user invisibly without the need of manual intervention.



**Figure 5.** Possible Strategies for Handling Deviations in Exclusive Patterns

In our ongoing work we focus on case-based reasoning as technique to overcome the previously mentioned disadvantages.

## 4 Case-Based Deviation Management

Case-Based Reasoning [2] is a method that exploits previously made experiences to adapt to currently upcoming and partly unforeseen situations. As we pursue an approach where users are allowed to deviate flexibly, but still guidance through recommending work items is to be achieved on the basis of formerly traced workflows, CBR seems promising. De facto workflows may be used as case base, which is searched for similar cases compared to a running workflow instance. Furthermore, retrieved cases may be used to adapt the running workflow instance, either considering the work items or the underlying workflow model. Our aim is to create a learning workflow engine, which automatically handles unforeseen deviations and nevertheless guides the user to a successful completion of the workflow by recommending work items by reuse of previous cases.

### 4.1 Basic Approach for a Case-Based Deviation Management

In the following, we sketch some characteristics of a case-based approach to deviation management for flexible workflows.

**Case** As cases we regard all de facto workflows and their corresponding de jure workflow. The de jure workflow is block-oriented and the de facto workflow is a simple sequence of tasks, which were traced at run-time.

**Query** A running workflow instance, which is not completed yet, will be used as query. The associated de jure workflow is also available. This instance is a subsequence of a future de facto workflow and contains at least one deviation concerning its de jure workflow, that occurred at the time of the request. With case-based reasoning we aim at recommending work items to continue with the running workflow instance of the query, by searching for similar de facto workflows whose subsequences match the current instance. Successive tasks to those subsequences in the case may be reused as possible recommendations.

**Additional Knowledge** Semantic descriptions and contextual information of the task and data nodes are accessible as additional knowledge. In certain cases there might be stored some kind of evaluation, i.e. if the workflow completed successful or failed. Furthermore, a categorization of the deviation might be derived, e.g. skipping of a task. Besides the de jure workflow, the declarative constraints, which are created and processed by the workflow engine, might be considered. Therefore, violated constraints are identifiable as well, if a deviation occurred. All of this additional information may be consulted for similarity calculation and enhance the retrieval process.

**Similarity** The similarity measure has to deal with a subset relation between query and case, as the de facto workflow of the query has not terminated yet, whereas the de facto workflows of the cases are complete. For the comparison of the de facto workflows we propose to use a sequence similarity measure, but local semantic similarity of tasks should be considered as well in a global similarity measure for the whole workflows.

Furthermore, differences in subsequences that are not related directly to the deviation, when comparing query and case, might not be relevant. This relevance might be expressed by the distance to the currently focussed task or the position concerning its level. For example, consider the workflow in Fig. 1, where a parallel control-flow block is included. Valid de facto workflows could be  $f_1 = \langle t_1, t_2, t_3, t_4 \rangle$  as well as  $f_2 = \langle t_1, t_3, t_2, t_4 \rangle$ , where the order of the tasks of the control-flow block may differ. If a deviation occurs in the part after  $t_4$ , which is omitted in the figure, differences in the preceding part of the de facto workflow of case and query, like  $f_1$  and  $f_2$ , might not be relevant for the deviation and should nevertheless be regarded as highly similar.

Additionally, as the technique will be applied during workflow execution, the retrieval and adaptation needs to fulfill real-time constraints, otherwise users will be tempted to bypass the system.

In the following, we will investigate three methods for the retrieval of the most similar cases and adaptations of the query concerning their applicability to the described case-based deviation management.

## 4.2 Trace-Based Reasoning

Trace-based Reasoning (TBR), which was first introduced by Mille [18], is a special form of case-based reasoning, where traces serve as case base. An important aspect as extension to standard CBR is the introduction of a temporal aspect. In TBR similarity measures are used that base upon temporal sequences. Cordier et al. [5] and Zarka [26] presented an approach with the objective of finding contextual recommendations for users that interact with web applications. Therefore an algorithm was developed that is based on the Smith-Waterman-Algorithm (SWA) and was adapted for retrieving similar traces. The SWA is used for local sequence alignment and compares traces in all possible lengths. This characteristic perfectly matches the requirements of trace-based reasoning as terminated



instances have to be compared to ongoing and not completed processes which are only subsequences and will not match exactly considering the length. Several local similarity measures are used for each attribute of the observed elements, that are part of the traces. For each observed element a global similarity value is computed on the basis of local values with use of a weighted average function. These similarity values of single elements are furthermore used in the SWA to calculate the overall similarity of traces. This method can be transferred to our pursued deviation management, if de facto workflows are considered as traces, for retrieving the best matching cases.

As adaptation method, Zarka [26] proposes a simple extraction method, that takes a subsequence of actions from the most similar retrieved case, which follows the part of the matched trace. Some values of this subsequence are adapted by more adequate ones on the basis of contextual information. As last step in the adaptation process, actions are filtered, e.g. invalid actions or unnecessary repetitions.

In our case of deviation management this subsequence can be extracted from the retrieved de facto workflow and first tasks of this subsequence can then be recommended as work items. Furthermore, if case and query do not match exactly, edit steps, which are necessary for aligning query and case, are part of the solution of the SWA. These additional differences relate to previously happened deviations, but might be required for a successful completion of the workflow as well. Therefore, these edit steps possibly might be transformed adequately into additional work items to ultimately provide further workflow control.

### 4.3 Alternative Similarity Measures

**Dynamic Time Warping** Dynamic Time Warping (DTW, [21]) may be used as alternative similarity measure for sequence comparison. DTW is already applied in various CBR approaches, for example for patient case matching in the medical domain [24]. It is a method which is used for aligning two time series, in our case de facto workflows can be regarded as such, by finding the best possible matching. For local comparison of different elements a cost function is defined. An optimal warping path is then searched for in the constructed cost matrix. The algorithm follows the same principle compared to the SWA with the main difference of values that are used in the matrix. A main advantage of both methods is the resistance to noise.

**Vector Similarity Measure for Sequences** A completely different approach is adopted by Gundersen [14]. In his research work, he presents a similarity measure for sequences on the basis of vectors. The main objective is to create a similarity measure which is suitable for real-time analysis relying on the assumption that events that happened near-time are more important than events from longer time ago. This is realized by a weighting function, that considers the position of events, related to the start and end of a sequence. The nearer the occurrence of an event is to the end of a sequence, the higher the weight

value. For each event type this weight is computed as sum of single weights of single event occurrences. These weights of event types are then combined into a vector, that in turn may be compared to other vectors with common methods, like cosine similarity.

The approach is lacking semantic similarity between single tasks, as only whole sequences are compared considering their properties of task occurrences. Only self-referential characteristics of tasks can be included in the single weights of tasks, like e.g. importance or severity.

The vector similarity measure is developed on the basis of requirements for recognizing failure causes and recommending adequate solutions in the oil-well drilling domain. As specific characteristics are addressed, cf. [15], the approach is not generally applicable, but some properties might be transferable, especially for the relevance of task similarity dependent on the distance to the currently focused task.

#### 4.4 Challenges and Potentials

To fit the needs of our proposed case-based deviation management the presented similarity measures and adaptation methods need to be investigated in detail and evaluated against our requirements. But as described previously some characteristics of each of the approaches might be adopted and combined for an adequate handling of deviations. Until now, we consider TBR as the technique with the most potential for an application to our proposed deviation management, as the similarity measure includes semantics of tasks and is also resistant to noise. Furthermore, adaptation opportunities are already available. We will consider to slightly adjust the similarity function with weights, indicating the distance to the deviation, analogously to the vector similarity measure of Gundersen [14].

Additionally to these challenging requirements some opportunities arise. An important aspect that should not be neglected is the performance of the applied algorithm as our approach will be used in real-time. To improve retrieval time, an option may be to filter the case base during execution, as the beginning of the trace is already available and enhanced step by step.

Another potential which will be investigated is, if the work list can be optimized through using CBR methods even if no deviation occurs, e.g. through a prioritization and ranking of work items on the basis of previous traces. Perhaps even deviations could be explicitly triggered. Consider a best practice, and thus optimized, workflow, which is executed through experienced employees, will be traced and enhance the case base. This workflow might not be an exact mapping of the designed workflow model, but may still be used for workflow control with the use of CBR methods.

## 5 Conclusion

In this paper we presented our vision of a case-based approach to deviation management in the context of flexible workflows. We described our ongoing research

work about constraint-based workflow flexibility and the need for an integration of a thorough deviation management. We investigated the potential of applying methods of case-based reasoning and sequence similarity and therefore introduced three existing techniques, that may be used after an adequate adoption. Finally we sketched some challenges and potentials of the approach, that we will face in our future work.

## References

1. van der Aalst, W.M.P.: Business Process Management - A Comprehensive Survey. *ISRN Software Engineering* **2013**, 1–37 (2013)
2. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Commun.* **7**(1), 39–59 (1994)
3. Adams, M., ter Hofstede, A.H.M., van der Aalst, W.M.P., Edmond, D.: Dynamic, extensible and context-aware exception handling for workflows. In: *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS, OTM Confederated International Conferences, Vilamoura, Portugal, November 25–30, 2007, Proceedings, Part I*. pp. 95–112 (2007)
4. Casati, F., Ceri, S., Paraboschi, S., Pozzi, G.: Specification and implementation of exceptions in workflow management systems. *ACM Trans. Database Syst.* **24**(3), 405–451 (1999)
5. Cordier, A., Lefèvre, M., Champin, P., Georgeon, O.L., Mille, A.: Trace-based reasoning - modeling interaction traces for reasoning on experiences. In: *Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2013, St. Pete Beach, Florida, USA, May 22–24, 2013*. (2013)
6. Depaire, B., Swinnen, J., Jans, M., Vanhoof, K.: A process deviation analysis framework. In: *Business Process Management Workshops - BPM 2012 International Workshops, Tallinn, Estonia, September 3, 2012*. pp. 701–706 (2012)
7. Döhring, M., Zimmermann, B., Godehardt, E.: Extended workflow flexibility using rule-based adaptation patterns with eventing semantics. In: *Informatik 2010: Service Science - Neue Perspektiven für die Informatik, Beiträge der 40. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Band 1, 27.09. - 1.10.2010, Leipzig, Deutschland*. pp. 195–200 (2010)
8. Eder, J., Liebhart, W.: The workflow activity model WAMO. In: *CoopIS*. pp. 87–98 (1995)
9. Grambow, G., Oberhauser, R., Reichert, M.: Event-driven exception handling for software engineering processes. In: *Business Process Management Workshops - BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part I*. pp. 414–426 (2011)
10. Grumbach, L., Bergmann, R.: Using constraint satisfaction problem solving to enable workflow flexibility by deviation (best technical paper). In: *Artificial Intelligence XXXIV - 37th SGAI International Conference on Artificial Intelligence, AI 2017, Cambridge, UK, December 12–14, 2017, Proceedings*. pp. 3–17 (2017)
11. Grumbach, L., Bergmann, R.: SEMAFLEX: A novel approach for implementing workflow flexibility by deviation based on constraint satisfaction problem solving. *Expert Systems* (03 2019)
12. Grumbach, L., Rietzke, E., Schwinn, M., Bergmann, R., Kuhn, N.: SEMAFLEX - Semantic Integration of Flexible Workflow and Document management. In: *Proceedings of the Conference “Lernen, Wissen, Daten, Analysen”, Potsdam, Germany, September 12–14, 2016*. pp. 43–50 (2016)

13. Grumbach, L., Rietzke, E., Schwinn, M., Bergmann, R., Kuhn, N.: SEMANAS - semantic support for grant application processes. In: Proceedings of the Conference “Lernen, Wissen, Daten, Analysen”, LWDA 2018, Mannheim, Germany, August 22-24, 2018. pp. 126–131 (2018)
14. Gundersen, O.E.: Toward measuring the similarity of complex event sequences in real-time. In: Case-Based Reasoning Research and Development - 20th International Conference, ICCBR 2012, Lyon, France, September 3-6, 2012. Proceedings. pp. 107–121 (2012)
15. Gundersen, O.E.: Enhancing the Situation Awareness of Decision Makers by Applying Case-Based Reasoning on Streaming Data. Ph.D. thesis, Norwegian University of Science and Technology, Trondheim, Norway (2014)
16. Kiepuszewski, B., ter Hofstede, A.H.M., Bussler, C.: On structured workflow modelling. In: Advanced Information Systems Engineering, 12th International Conference CAiSE, Stockholm, Sweden, June 5-9, 2000, Proceedings. pp. 431–445 (2000)
17. Marrella, A., Mecella, M., Sardiña, S.: Intelligent process adaptation in the smartpm system. *ACM TIST* **8**(2), 25:1–25:43 (2017)
18. Mille, A.: From case-based reasoning to traces-based reasoning. *Annual Reviews in Control* **30**(2), 223–232 (2006)
19. Müller, R., Greiner, U., Rahm, E.: Agent<sup>work</sup>: a workflow system supporting rule-based workflow adaptation. *Data Knowl. Eng.* **51**(2), 223–256 (2004)
20. Reichert, M.: Dynamische Ablaufänderungen in Workflow-Management-Systemen. Ph.D. thesis, University of Ulm, Germany (2000), <http://d-nb.info/960862684>
21. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* **26**, 43–49 (02 1978)
22. Schonenberg, H., Mans, R., Russell, N., Mulyar, N., van der Aalst, W.M.P.: Towards a taxonomy of process flexibility. In: Proceedings of the Forum at the CAiSE’08 Conference, Montpellier, France, June 18-20, 2008. pp. 81–84 (2008)
23. da Silva, M.A.A., Bendraou, R., Robin, J., Blanc, X.: Flexible deviation handling during software process enactment. In: Workshops Proceedings of the 15th IEEE International Enterprise Distributed Object Computing Conference, EDOCW 2011, Helsinki, Finland, August 29 - September 2, 2011. pp. 34–41 (2011)
24. Tsevas, S., Iakovidis, D.K.: Dynamic time warping fusion for the retrieval of similar patient cases represented by multimodal time-series medical data. Proceedings of the 10th IEEE International Conference on Information Technology and Applications in Biomedicine pp. 1–4 (2010)
25. Weber, B., Wild, W., Breu, R.: Cbrflow: Enabling adaptive workflow management through conversational case-based reasoning. In: Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings. pp. 434–448 (2004)
26. Zarka, R.: Trace-based reasoning for user assistance and recommendations. (Raisonnement à partir de l’expérience tracée pour l’assistance à l’utilisateur et les recommandations). Ph.D. thesis, INSA Lyon, Lyon - Villeurbanne, France (2013)
27. Zhu, R., Dai, F., Mo, Q., Yu, Y., Lin, L., Li, T.: An approach to handling software process deviations. *Lecture Notes on Software Engineering* **3**, 238–244 (01 2015)