A "Case Mining"-based Recommender System for Knowledge Workers

Sebastian Huber[1]

[1] Friedrich-Alexander-Universität Erlangen-Nürnberg,
Lange Gasse 20,
90403 Nürnberg, Germany
Sebastian.Huber@fau.de

Abstract. Knowledge workers solve problems, which are usually non-routine and where no standardized solutions can be applied. During the problem-solving process, the overarching task is broken down into a sequence of activities. To support the coordination of these steps Adaptive Case Management (ACM) emerged as a novel paradigm in Business Process Management research. However, current ACM systems lack in supporting the decision making of knowledge workers to prioritize their activities. The objective of this research in progress is to consolidate different approaches in process mining literature and integrate them as an activity recommender system into an existing ACM tool. Based on 25 sample cases, different predictive models are created to recommend actions to shorten case running time, to mitigate deadline violations, and to support case goals. A preliminary evaluation of the models indicates moderate positive results regarding their accuracy. In addition, starting points for further improvement are discussed.

1 Motivation

The nature of knowledge work is neither routine nor predictable, and therefore requires human judgment by the knowledge worker [1]. Therefore, knowledge workers face different challenges in their daily working context. For example, questions like what actions they should perform next, what actions contribute in achieving their project goals and to whom a certain task should be assigned.

Without guidance, knowledge workers face a high complexity in such decisions. They may choose a non-optimal action leading to delays, deadline violations, or an inefficient utilization of resources. To reduce this complexity and to guide knowledge workers, decision support is essential. A common approach is to provide prediction and recommendation capabilities to help the knowledge workers determining the optimal order of actions for a certain case.

In recent years, process mining emerged from process model discovery to operational support. Originally, process mining "aims to discover, monitor, and improve real processes by extracting knowledge from event logs readily available in today's information systems" [2]. However, the original approach can only be used in offline settings, i.e., after process execution. New types of process mining suggest an

operational deployment. That includes types of process mining that check the conformity of currently running process instances, predict their future state, and provide recommendations for appropriate next step actions [3].

Research already has shown the applicability of process mining for generating predictions and recommendations with promising results. These concepts may be applied to the field of knowledge work as well. The goal of this research is to consolidate current process mining approaches and integrate them in an executable ACM prototype that gives predictions and recommendations based on running case instances. That means that this happens in a flexible environment with no predefined processes, i.e., where the process outcome is highly depended on the executing subjects, the knowledge workers.

2 Research Methodology

To support the achievement of case goals in the domain of ACM, this paper provides an approach for optimizing the execution of cases, based on a next step recommender system. It follows the idea that a complex system (with non-linear relations between its parts), like multiple knowledge workers executing multiple cases collaboratively, cannot be optimized globally. Therefore, it follows a divide-and-conquer inspired approach: By emphasizing the knowledge worker's importance and the enhancement of their task sequences, all cases are more likely to get completed in time.

For its realization, a recommendation system is utilized. Contrary to other application areas, e.g., the recommendation of products in a web shop [4], the recommended items are steps to be executed next. Therefore, the underlying mechanisms are located in the data and process mining field of research.

First predictions have to be made, how long cases are estimated to run, which case goals may not be achieved, and which deadlines may be violated. Based on these predictions, recommendations can be derived, which tasks should be handled immediately. Finally, the presented approach offers the possibility to shape personalized processes dynamically, based on the knowledge workers' interactions.

Summarizing, the research methodology follows a three-step approach: In the following, publications from different research areas regarding predictions and recommendations are discussed (1). Based upon the literature review, the conceptual approach for a software prototype is depicted and implemented (2). Finally, the quality of the prototype is evaluated (3).

3 Related Work

There is only sparsely literature available on mining case associated data for predictions and recommendations, due to the youth of ACM as a field of research. Therefore, different relevant research areas are presented in the following: case management related work, workflow and process mining approaches as well as concepts regarding case-based reasoning (CBR). Finally, the applicability of the available approaches is discussed shortly.

The only case management system that offers recommendations is the IT Support Conversation Manager (ITSCM) found in [1]. It allows the recommendation on best next steps and experts to invite in current running cases by analyzing previous case resolutions. The ITSCM architecture consists of two major components: annotated model discovery and steps/experts recommendation. The annotated model discovery component analyzes previous cases in a repository and constructs an annotated step flow model. The steps recommendation component takes a currently running case and tries to match it into the annotated step flow models. Afterwards, it creates an ordered list of recommended steps and suggests IT staff members that have performed such task in the past. Performance tests reveal an advantage of pre-building the model offline, and using this model for generating recommendations at run-time, rather than building the model ad-hoc. The approach focuses of next item recommendations in the case management domain. Evaluation has shown that adding case-related information, such as type, title, or tags, lead to more relevant recommendations compared to adding only process-information. Performance tests reveal an advantage of pre-building the model offline, and using this model for generating recommendations at run-time, rather than building the model ad-hoc.

Contrary, mining previously executed process instances is extensively discussed in the process and workflow mining literature. [5] propose a service that gives recommendations on possible next steps during process execution of a partial case and process instance respectively. This is done by considering specific optimization goals of similar process executions. Following this approach each business process is described as a process model loaded into a process-aware information system (PAIS). The PAIS runs single process instances and records information about executed activities in event logs during run-time. Those event logs are later used by the service to generate recommendation results. For this purpose, the service requires information about the currently executed partial case, i.e., currently enabled activities and the partial trace of executed activities in the past. With this information, the recommendation decides what activities to perform. The final recommendation for an executed activity is then calculated by a “do value”, the expected outcome of the target value when an enabled activity is executed, and a “don’t value”, the expected target value of not executing the enabled activity. Finally, differently executed activities are collected in an ordered recommendation result and returned back to the user. Experiments on the recommendation service have shown that the more heuristic information are used, the better the quality of the recommendation. The performance of their system depends on process characteristics and the degree of abstraction. For instance, by comparing the cycle times of traces generated by the different abstractions and randomly created traces, the prefix abstraction of events outperforms random samples, set abstractions and multi-set abstractions. In general, it has been shown that goals, such as cycle times, can be reduced through the support of recommendations [5]. [5] investigated different cycle time predictors using historic information from event logs. Their goal is to answer questions about the completion time of a process instance by using the average cycle time and deducting the remaining cycle time considering the past time of the case. However, this approach is not sufficiently accurate. Hence, they additionally define different non-parametric regressions: Activity occurrence estimator, activity duration estimator, case attribute estimator and a combined regression estimators. The activity occurrence estimator determines the remaining cycle time based on the occurrence of activities. The activity duration estimator uses duration of each activity for remaining cycle time calculation. The case attribute estimator applies case data attributes from event logs as ordinal variables to estimate

the remaining cycle time. The combined regression estimators multiplies all three estimator's kernel functions that is then used in the regression. Experiments show that regression estimators from [5] outperform the average cycle time estimator with case attribute and combined estimators as the best-performing ones.

[7] coined the term predictive workflow management. They argue that the integration of predictions into workflow enactment benefits in improved workflow-related decisions, such as decisions on setting execution time of single workflow steps or completion time of the whole workflow. They demonstrate a concept that can assign a deadline, i.e., the completion time, to workflows through predictions, and is able to escalate when a deadline violation is foreseeable. A similar framework can be found in [8]. This approach ensures that process deadlines are met and no time constraints are violated by computing activity deadlines. More precise, it checks at build time for time constraints and constructs a timed activity graph that includes deadline ranges. This graph is extended on process instantiation times with concrete deadlines. It monitors on run-time whether a given time constraint for remaining activities is satisfied based on already executed activities and activity completion times. On deadline or time constraint violation it triggers exception-handling activities (escalation).

CBR is a research topic that has gained attraction in the 1990s. According to [9] "a case-based reasoner solves new problems by adapting solutions that were used to solve old problems". By this definition, a case-based reasoner finds cases that solved similar problems in the past and adapts previous solutions to the current problem. [10] developed a CBR cycle. First one or more past cases are retrieved to solve the problem. Then, the information and knowledge in the previous case are reused to solve the problem. Next, the proposed solution is revised and confirmed. Finally, experiences made by applying the solution are retained for future problem-solving. CBR suggests solutions to knowledge workers, which they can use as a guidance for next steps. The integration of CBR in operational systems, like workflow management systems, has been introduced [10]. In their prototype a workflow is extended with conversational CBR, an interactive system that retrieve a similar case for problem solving and exception handling through question-answering sequences. That allows dynamic changes of the predefined process model and provides learning capabilities for process model improvements.

Overall, only one solution focuses on collaborative knowledge-intensive processes. Most other approaches are tailored to scenarios that are characterized by pre-definable business process models that are run by process engines. Such solutions are suitable for executing predictable routine work, but only restricted in unpredictable knowledge-intensive processes. No existing approach is meeting requirements of a context where flexibility and autonomy of the executing knowledge workers is essential.

4 Concept

The objective of this paper is to adapt existing ideas in the literature to knowledge-intensive processes in ACM. Due to the unique challenges in the context of ACM, this chapter will introduce a concept that uses combined multiple approaches for considering various perspectives for suggesting next step predictions and recommendations for guiding knowledge workers during case handling.

4.1 Architecture

The prototype consists of three main parts: The event log generator, the model builder, and the predictor/recommender (cf. Figure 1). The event log generator produces log entries, which represent the subject's activities in a standardized XES event log format. Those logs are based on historic events and operational data from the ACM system. The event logs are used by the model builder to construct underlying models based on different approaches. Those models are merged into a composite model, which is capable of predicting possible future states and recommending items based on various process and data mining techniques.

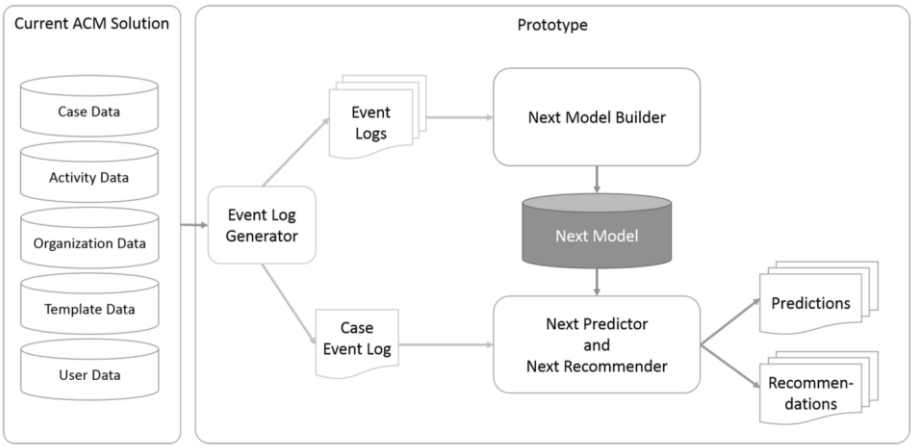

Fig. 1. Architecture overview

Event Log Generation: The event logs are generated based on different data sources within the application. Therefore a classical extract, transform and load (ETL) process is used before the event logs are finally stored in a standardized format. A XML-based standard, the Extensible Event Stream (XES) is used as a data format to store the generated event logs. In the first step for the event log generation, data is extracted from case, template, organization, user and activities data. Most relevant is the activities data store. There, all activities which occurred on the platform are stored, inspired by the official standardized activity streams schema. In the second step data is transformed for operational needs. Activities are transformed into events and assigned to traces, which are stored according to the XES format. Optionally, filtering may happen in cases where redundant actions on business objects were recorded. Finally, in the third step, the data is loaded to be used by the model builder.

Model Building: The model builder uses the stored activities to derive different models. Four models covering various aspects (time, deadline, decision, and goals) are created by applying different algorithms. Separate results are generated by every model, before they are combined into a composite model:

1. Time-based: The first component of the composite model is a time-based model, based on the timestamps of the recorded log-file entries. The time-based model can look for similar events and return the remaining time (or

average remaining time) as a prediction. Based on the history of events in other cases, the following event with the smallest expected remaining time is selected and then recommend to the user. Thereby, the likelihood for delays regarding the running time of case is considerably reduced. The model is represented as an annotated transition system according to [3].

2. Deadline-based: The deadline-based model is an extension to the time-based model. It takes the possible violation of case deadlines into consideration. Predictions are made to identify possible deadline violations. Then recommendations are made which tasks should be completed to avoid a deadline violation.
3. Decision-based: During the progress of a case, knowledge workers have to make decision how to proceed. Decision mining, introduced by [4], aims to extract rules to explain why certain choices are made. Decision points in progress of a case are characterized by multiple outgoing paths with different sequences of activities. Two challenges have to be mastered with regard to decision mining: identifying decision points and finding meaningful rules [4]. With knowledge workers shaping their daily activities, the first problem is irrelevant, because every recorded event is the triggered result of a decision. The second problem is a classification problem. Here decision trees can be applied [4]. Case attributes can be used as attributes for classification to identify class labels. For the construction of the decision tree, the algorithm of [5] is used. The tree's class labels as recommendations are activities which were chosen in other cases with similar labels and therefore should be performed by the knowledge worker in the current case.
4. Goal-based: A case always follows one or multiple goals. Tasks in a case are linked with the goals. Therefore, the goal-based model provides next step recommendations of activities that favor the attainment of particular case goals. The goal-based model is based on the approach of [5]. For every stored trace of events in the log, a target value for a case goal can be computed. "Do calculations" are performed to determine the expected target value if an activity is executed. "Don't calculations" compute the expected target value in the case that the activity is not executed. Using the results of the calculations for events and their traces in the logs, goal-supporting activities can be recommended.

Prediction and Recommendation: The predictor and recommender is using the models described above, based on the current state of the running case. As each underlying predictor and recommender outputs their respective suggestions, this component combines the outputs into a final prediction and recommendation set by realizing the ensemble method according to [12]. It considers the four different aspects for a recommendation by weighting them individually. Finding the correct weights is not a trivial task and discussed as part of the evaluation.

4.2 Prototype

The previously described concept is integrated in an ACM software solution called Collaborative Case Management (CoCaMa) (more details can be found in [13]). It is applied as a plug-in to extend the range of its functionality. The core of CoCaMa, which directly supports collaboration, is the case workspace. With an intuitive "drag and drop" behavior it is possible to create different lists and freely arrange all objects that are used in the case. In that way hierarchical relationships between tasks (milestones, subtasks) or the assignment of documents to discussions, etc. can be realized. This is the primary part where the knowledge workers designs the different views on a respective case. In Figure 2 an example of a task board is depicted that is used in agile software development (SCRUM).

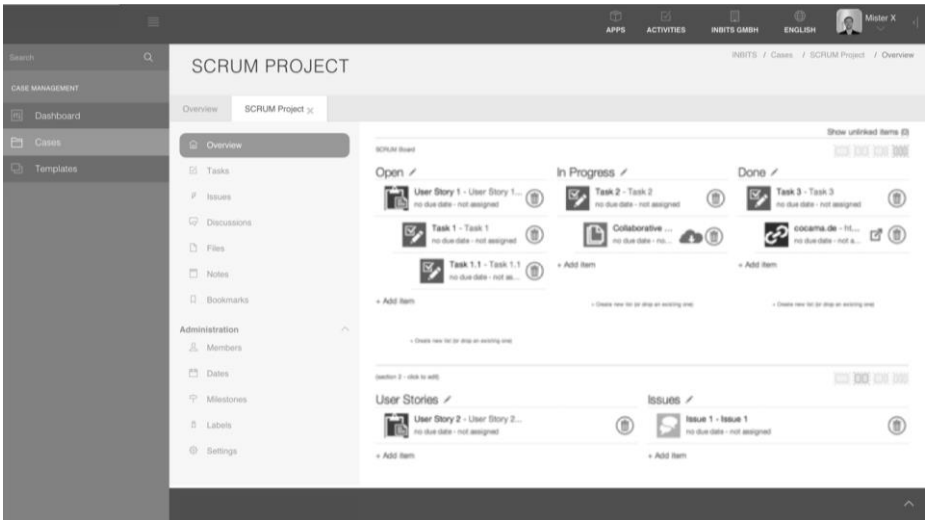

Fig. 2. Case Workspace

Based on that principle, two views on recommendations are available. First a general view, recommending what steps should be executed next, without being limited to a single case, but providing an overview of next step recommendations on all cases. This guides the knowledge worker by generating and suggesting a subject focused process dynamically, based on past personal actions as well as historic actions of others. The prototype's general view can be seen in Figure 3.

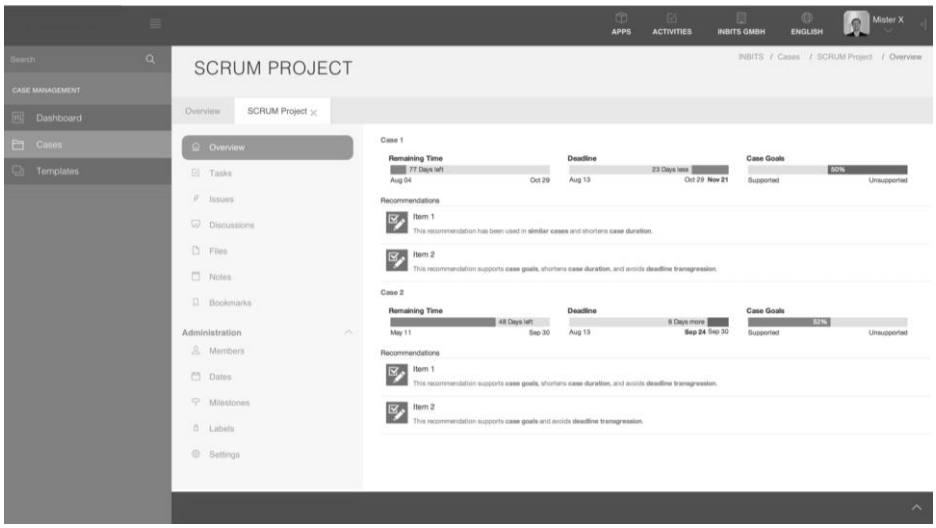

Fig. 3. Case-spanning Predictions and Recommendations

Predictions are presented by a progress bar that visualizes the remaining time. The progress in green indicates the elapsed time in proportion to the total time. The remaining space in the progress bar stands for the remaining time in proportion to the total time. A duration indicator on the left side shows the remaining case time in days. The two dates below the progress bar represent the starting data of the case and the estimated completion date. The second visualization shows whether a deadline violation occurs or not. A deadline violation is indicated by a red bar, whereas a non-violation bar is shown in green. The duration indicator inside the visualization demonstrate whether the case is estimated to finish earlier than the specified due date (X days less) or to finish later than the specified due date (X days more). The dates below correspond to today's date, due date, and estimated completion date. The due date is emphasized by a bold typeface. The third visualization shows the proportion of case goals that are supported and unsupported by the current case progress. A case goal is supported when similar cases in the past share the same goals. A case goal is unsupported when those similar cases do not include those goals. The higher proportion is further emphasized in green (supported) or red (unsupported) and by a percentage value.

Recommendations are visualized in rows. A recommendation includes the recommended item and reasons why this case item is recommended. A recommendation row presents the case item title and its type through its icon. The reason is shown below the title. Four reasons are possible, based on the underlying models of the composite model: The recommendation shortens the case duration, avoids deadline violations, supports case goals, or has been used in similar cases. With the help of the reasons, the users can determine the benefits of the recommendation. Besides the general view, predictions and recommendations can be used case specific within currently running cases (cf. Figure 4).

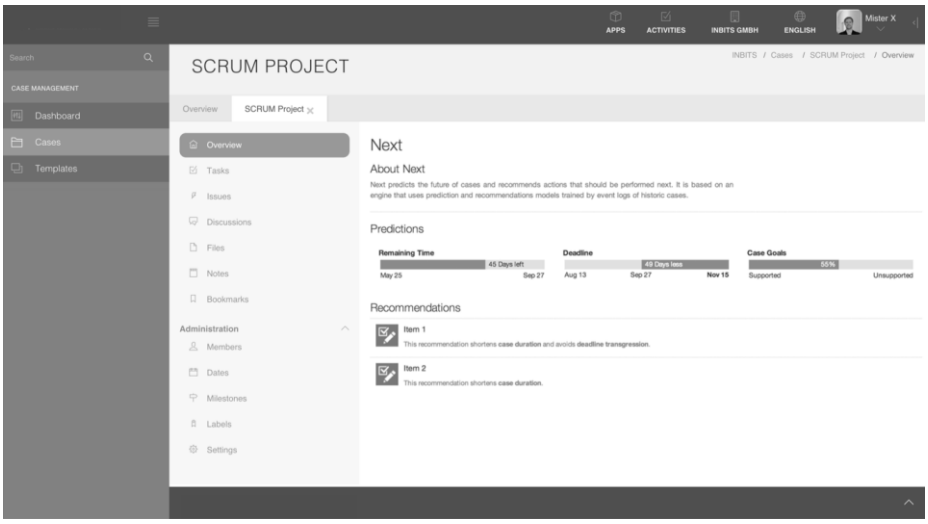

Fig. 4. Case-specific Predictions and Recommendations

In addition, a settings view allows the user to set the prioritization and weights of the algorithm parameters. Currently, the model is not build and updated continuously to avoid performance consumption while CoCaMa is used during the day. The models are used to make situation depended recommendations, but the update frequency can be controlled in the settings view as well.

5 Evaluation

The evaluation of the predictive and recommendation quality is based on a knowledge-intensive customer inquiry process. Altogether, 25 historic cases and five running cases have been defined that follow the predefined inquiry process. The process itself is derived from a real-world example, but the execution path has been fictively defined with the help of experts. Those test cases are manually populated into the database of the case management system. The prototype evaluation focuses on three different aspects:

1. Event Log Generation Evaluation: Goal of this evaluation is to determine whether the prototype can construct an event log. Further, the event log should include information from the given cases and activities. The evaluation of the event log generation is defined in four tests. The first test creates an organization event log, the second test two template event logs, the third test four user event logs, and a forth test five case event logs.
2. Model Building Evaluation: The evaluation of model building tests whether all models can be built from event logs. Based on the data from the event log generation this test builds all underlying models (time, deadline, decision, and goal) and a combined version.

3. Prediction and Recommendation Evaluation: The built models are used in this evaluation phase to examine the predictive and recommendation outputs of the models. Case event logs of partial traces are used with a set of current case items to generate predictions and recommendations out of each model. The partial traces are constructed in a way that some predictions are known beforehand. Ten major prediction or recommendation tests are provided that describes the major scenarios in the customer inquiry process.

As a first result, the prototype is able to build all composite models and the underlying models. As for the time-based and decision-based models, the log files reveal that the annotation transition system contained all required nodes and edges. The nodes hold state information and the elapsed time and remaining time annotations. The edges contained information about preceding and succeeding states and events leading from one state to another. Overall, the model included all information necessary to describe the time annotated transition system. Regarding the decision-based model building, the decision tree could be constructed by the used Java library for machine learning, WEKA. The evaluation on the basis of WEKA, however, revealed low accuracy of the tree. In fact, it could only predict a low amount of the class labels correctly. This result may be due to limited availability of training data. For goal-based models, terminal printouts show traces along with their supporting goals.

The time-based predictor generates all expected predictions. It estimates proper remaining times of the partial event log traces and could provide correct elapsed- and total times. The time-based recommender provides the right recommended items that are expected to complete the case in the fastest way. The deadline-based predictor is able to detect a deadline violation when the due date was set prior to the expected remaining time of the case. It also successfully detects the non-violation state of the case. The deadline-based recommender generates some recommendations that mitigate a deadline violation by offering items that finish the case as soon as possible. It suggests the same recommendations as the time-based model, which indicates redundant recommendations to some extent. The decision-based recommender only provides one single recommendation. In the evaluation, it suggested an event that occurred in all previous historical cases. Finally, the goal-based predictor is able to output the expected support and unsupported number of goals. It suggests items that have been used in related past traces following the same goal.

6 Conclusion

In this research in progress, the design of a next-best action recommender system for knowledge workers in the context of ACM is illustrated. Activities during a case's proceeding are extracted successfully by analyzing the related log-files. Predictive models for predicting actions to shorten the case running time, to mitigate deadline violations and to support case goals are created based on artificial test data. According to preliminary results, these models showed moderate positive results in their prediction accuracy.

Within future research, the developed prototype is to be tested with case-data from real-word knowledge work scenarios. Beside the accuracy of the predicted next activities, the general user acceptance of the recommendations has to be evaluated, as well as their impact on the productivity of the knowledge workers. Another limitation that has been not addressed yet is scalability in cases that are characterized by a lot of members, many tasks and a high degree of flexibility.

References

[1] Motahari-Nezhad, H. R. & Bartolini, C. (2011). Next best step and expert recommendation for collaborative processes in it service management. In: Business process management, 6896(7), 50–61.

[2] Aalst, W. M. P. van der. (2012). Process mining: overview and opportunities. ACM Transactions on Management Information Systems, 3(2), 1–17.

[3] Aalst, W. M. P. van der, Pesic, M., & Song, M. (2010). Beyond process mining: from the past to present and future. In: Advanced information systems engineering, 6051, 38-52.

[4] Ricci, F., Rokach, L., Shapira, B. (2011): Introduction to Recommender Systems Handbook. In: Recommender Systems Handbook, Springer US.

[5] Schonenberg, H., Weber, B., Dongen, B. van, & Aalst, W. M. P. van der. (2008). Supporting flexible processes through recommendations based on history. In: Business process management, 5240, 51-66.

[6] Dongen, B. F. van, Crooy, R. A., & Aalst, W. M. P. van der. (2008). Cycle time prediction: when will this case finally be finished? In: On the move to meaningful internet systems, 5331, 319-336.

[7] Panagos, E. & Rabinovich, M. (1997). Predictive workflow management. In: Proceedings of the 3rd international workshop on next generation information technologies and systems, 193-197.

[8] Eder, J., Panagos, E., & Rabinovich, M. (1999). Time constraints in workflow systems. In: Advanced information systems engineering, 1626, 286-300.

[9] Riesbeck, C. K. & Schank, R. C. (1989). Inside case-based reasoning. Hillsdale, NJ, USA, L. Erlbaum Associates Inc.

[10] Aamodt, A. & Plaza, E. (1994). Case-based reasoning: foundational issues, methodological variations, and system approaches. AI Commun. 7(1), 39–59.

[11] Weber, B., Wild, W., & Breu, R. (2004). CBRFlow: enabling adaptive workflow management through conversational case-based reasoning. In: Advances in case-based reasoning, 3155, 434-448.

[12] Han, J., Kamber, M., & Pei, J. (2011). Data mining: concepts and techniques (3rd). San Francisco, CA, USA, Morgan Kaufmann Publishers Inc.

[13] Huber, S., Lederer, M., Bodendorf, F. (2014): IT-enabled Collaborative Case Management: Principles and Tools. In: Proceedings of the 2014 International Conference on Collaboration Technologies and Systems, Minneapolis, Minnesota, USA, IEEE by The Printing House, Inc., Stoughton, 259 - 266.