Beyond Roles: Prediction Model-Based Process Resource Management

Florian Niedermann, Alexandru Pavel, and Bernhard Mitschang

Institute of Parallel and Distributed Systems,
Universität Stuttgart,
Universitätsstraße 38,
D-70569 Stuttgart
{Florian.Niedermann,Alexandru.Pavel,
Bernhard.Mitschang}@ipvs.uni-stuttgart.de

Abstract. The outcome of a business process (e.g., duration, cost, success rate) depends significantly on how well the assigned resources perform at their respective tasks. Currently, this assignment is typically based on a static resource query that specifies the minimum requirements (e.g., role) a resource has to meet. This approach has the major downside that any resource whatsoever that meets the requirements can be retrieved, possibly selecting resources that do not perform well on the task. To address this challenge, we present and evaluate in this paper a model-based approach that uses data integration and mining techniques for selecting resources based on their likely performance for the task or sub-process at hand.

1 Introduction

In this introduction, we will first discuss the general role of resources in business processes. Then, we take a look at how *Business Process Management Systems (BPMS)* typically handle the assignment resources and some of the challenges associated with this approach. Finally, we briefly introduce our *deep Business Optimization Platform (dBOP)* which provides integrated process analysis and optimization facilities and in which the work presented in this paper is embedded.

1.1 Resources in Business Processes

Broadly speaking, resources are the (physical or virtual) entities that execute the activities that make up a business process. Resources are frequently - next to the process structure - the major factor when it comes to deciding the process outcomes such as process duration, cost, success rate or quality.

Consider for instance the mail order return process that will be the subject of the evaluation in Section 4. In the process, an administrative and a technical employee are tasked with assessing both the return information given by the customer as well as the returned item(s). Given that return rates can reach 30% or more [DKDBvdV02], this is a crucial process for any mail order company - the

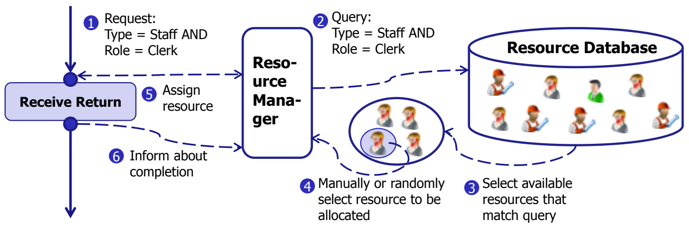

Fig. 1. Standard Resource Assignment

performance of which depends significantly on the employees executing it. As the number of available employees is limited, their employment is associated with considerable cost and their performance can greatly vary, depending on their own attributes and the nature of the returned articles, it is obvious that significant benefits can be realized through optimizing the task allocation in that and other scenarios. There is hence a considerable business need to assign resources in such a way that for a given task and a given process state, the resource that is most likely to create the best outcome (with regards to the process goals) is selected.

1.2 Resource Assignment in Business Process Management Systems

Many of today's business processes are described in a formal language and executed on a so-called *Business Process Management System (BPMS)*. As *BPMS* vendors have recognized the importance of managing resources in business processes, most major *BPMS* engines provide integrated support for the management and assignment of resources to activities and work items. While the individual implementations vary, the general process for resource assignment usually follows the one depicted in Figure 1.

The starting point is the resource (or organizational) database that contains and describes all the resources available to the business process. When a new resource is needed, the activity requiring that resource sends a query describing the resource's properties (typically, the required role) to the resource manager. The resource manager uses this query to fetch a set of resources that match the resource requirements. Finally, one of the resources contained in this set is more or less arbitrarily selected and allocated to the requesting activity (or alternatively, resources can actively claim a new task from the list of available requests).

While well-tested in practice, this approach cannot guarantee that the assigned resource will perform optimally. As the role requirements specified in the resource query are typically quite broad, the set of returned resources is potentially quite large. As the resource manager does not take the likely performance of the resources into account, it will routinely not select the best resource for the task at hand, causing sub-optimal process results.

1.3 Deep Business Optimization Platform

To overcome the limitations of standard resource managers discussed in the previous section, we need a resource manager that is able to predict the likely performance of resources and assign them to resource requests accordingly. For that, we need a platform that is able to gather all the required data, construct a prediction model based on this data and provide its results during process runtime to the resource manager.

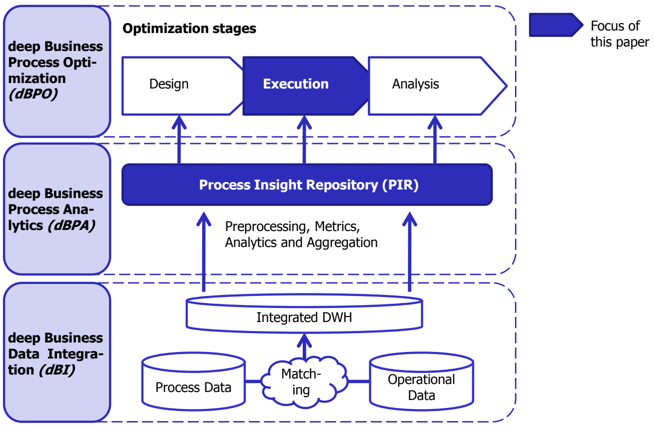
deep Business Process Optimization (dBPO)
Optimization stages
Focus of this paper
Design
Execution
Analysis
deep Business Process Analytics (dBPA)
Process Insight Repository (PIR)
Preprocessing, Metrics, Analytics and Aggregation
deep Business Data Integration (dBI)
Integrated DWH
Process Data
Matching
Operational Data

Fig. 2. dBOP Platform Overview

One example for a platform that provides such facilities is our *deep Business Optimization Platform (dBOP)* [NRM10]. The *dBOP*, as conceptually shown in Figure 2, consists of three architectural layers. The integration layer [RNB10] combines various heterogeneous data sources to ensure that all attributes that are relevant for selecting a resource are available to the analysis layer. In the analysis layer [NRM10], various process- and resource-related metrics are calculated based on the integrated data. Using the metrics and the integrated data as input, the analysis layer further builds prediction and classification models, as used e.g., for resource management or decision support [NMR+11]. The aggregated insights are then stored in the *Process Insight Repository (PIR)*. These insights also include the models used by the resource manager presented in this paper. In the optimization layer, the analysis results are used to ensure that the business process design and execution are optimal with respect to the given goal function(s) using a set of formalized optimization patterns [NRM11]. The model-based resource manager presented in this paper is an example for execution-stage optimization.

In this paper, we will show how the *dBOP* can be used to provide a model-based resource manager during process execution that assigns resources not arbitrarily, but based on their predicted performance. For that purpose, we will introduce the underlying process- and resource model as well as the basic functions of the resource manager in Section 2. Building on this, we discuss the detailed concepts and the design underlying our approach in Section 3, including the information required to specify a model-based query. In Section 4, we evaluate the performance of our resource management approach based on a sample case study. Finally, we discuss related work in Section 5 before concluding the paper in Section 6.

2 Fundamentals

In this section, some of the fundamental concepts underlying this paper are discussed. First, we take a look at the structure of resource models and resource queries. Then, we provide a process meta-model that is geared towards modeling resources usages and dependencies and that is used by the resource manager proposed in the next section.

2.1 Resource Models and Queries

While individual implementations vary, most resource (or organizational) models of *BPMS* characterize a resource with the following information: A resource role *Ro* and several resource attributes *Att* that additionally characterize the properties of the resource. In the resource model employed by the *dBOP* we further add a resource type T that determines the basic nature of the resource (typically, one of the following: staff, machine, service). For the purpose of unique identification in a resource model, a resource is given a unique resource name *Rn*. This resource name is resolved to a concrete resource identifier at runtime. For any given process, the set of all resources that can be retrieved using a resource query is called R.

Building on this resource model, we define a basic resource query Q_B as follows:

Definition. *Basic resource query: A basic resource query* Q_B *is a mapping* $T \times Ro \times Rn \times Rel[\times \wp(AC)] \rightarrow R$*, where*

1. *T, Ro, Rn and R are defined as above.*
2. *$Rel \in \{KEEP, RELEASE\}$ indicates whether to release the resource after execution (or reserve it for further activities of the process).*
3. *$AC \in Att \times COMP \times DOMAIN(Att)$ lists properties the resource attributes have to fulfill, with $COMP \in \{=, <, >, \geq, \leq, \in\}$ where $\wp(AC)$ denotes the power set and hence all possible combinations of these properties.*

2.2 Process Meta-model and Resource Dependency Graph

As the next section will show, improving the outcomes of resource selection requires the resource manager to have awareness of the process model and its execution behaviour. Hence, we present in this section a resource-aware process model that provides adequate expressiveness for the concepts discussed in this paper. It is based on a modified version of the graph-based process model introduced in [LR00]. For this paper, we are only introducing the elements essential for the resource management approach presented - see [NRM11] for a complete version.

Definition. *Simplified process graph: A simplified process graph G is a tuple $(V, N, R, \iota, \rho, E_C, E_R)$, $SUCC_R$ is a mapping and μ is a function, in which*

1. *V is the finite set of process data elements (also called variables).*
2. *N is the finite set of process nodes.*
3. *R is the finite set of available resources as defined above.*
4. *$\iota : N \cup \{G\} \rightarrow \wp(V)$ is the input data map, with $\wp(V)$ being the power set over V.*
5. *$\rho : N \rightarrow Rn$ is the resource usage map.*
6. *$E_C \subseteq N \times N \times C$ is the set of control connectors as explicitly modeled.*
7. *$E_R \subseteq N \times N \times R$ is the set of resource connectors which indicates resource dependencies between activities.*
8. *$SUCC_R : Rn \times N \rightarrow \wp(N)$ retrieves all successors of the given node that use the resource with the given resource name.*
9. *$\mu : \wp(V) \cup \wp(R) \rightarrow \wp(O)$ performs the matching, i.e., integrates operational data with process data, where O is the set of operational data pertaining to the process.*

In this graph, a resource dependency between two nodes exist, if they use a resource with the same unique name Rn. The direction of the resource dependency follows the direction indicated by the control dependencies.

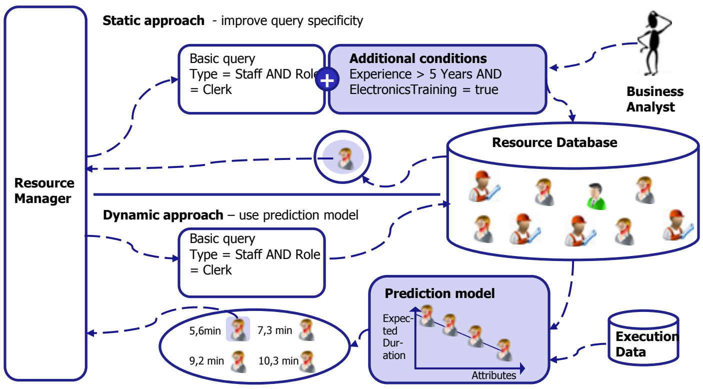
Static approach - improve query specificity
Basic query Type = Staff AND Role = Clerk
Additional conditions Experience > 5 Years AND ElectronicsTraining = true
Business Analyst
Resource Database
Resource Manager
Dynamic approach – use prediction model
Basic query Type = Staff AND Role = Clerk
Prediction model
Expected Duration
Attributes
Execution Data
5,6min
7,3 min
9,2 min
10,3 min

Fig. 3. Alternatives for Improving Resource Assignment Quality

3 Model-Based Resource Assignment

In this section, we first discuss two alternatives for improving the quality of process resource assignment and demonstrate, why a model-based approach will be preferable in most situations to its alternative. Making such a model-based resource manager work depends on the fulfillment of three principal requirements: First, the resource manager's capabilities need to be powerful enough to be readily adaptable to different usage scenarios. Second, the prediction model needs to be designed so that it provides a good selection performance and a high degree of flexibility. Finally, the resource manager needs to be able to offer its services to business processes at runtime. This section will discuss all of these aspects in detail.

3.1 Improving Resource Assignment Quality

In a typical application scenario, a resource query used by any given process activity will return multiple resources that match the given specification. The number of resources returned will thereby depend on the current resource utilization, as the lower the utilization the higher the number of available resources. As discussed in Section 1.2, all of the returned resources will fulfill the necessary attributes required to execute the activity - which, however, does not necessarily mean that all of them will perform equally well.

As Figure 3 shows, there are two basic approaches for improving the likely performance of the assigned resources. In the static approach, the process analyst determines (possibly aided by some analysis tool) the resource attributes that influence the activity performance positively. Then, he refines the query accordingly to make sure that the retrieved resources not only fulfill the basic requirements, but also are likely to perform well at the given activity.

While the static approach has the advantage of not requiring changes to the resource management application, there are several severe disadvantages. First, it requires considerable manual work to determine a well-fitting resource query - especially, since it might have to be updated frequently. Second, to provide an actual improvement, it likely produces enormous and excessively complex queries. Finally, it might actually worsen process performance, as it can significantly reduce the number of resources retrieved, potentially creating a situation where an activity has to wait for a resource to be available (i.e., a resource bottleneck).

In the dynamic (or model-based) approach, the specification contained in the resource query is not fundamentally changed. Instead, a prediction model is constructed based on the recorded process execution history. While this approach requires modification of the resource manager and the presence of execution data, it produces superior results. First, beyond the specification of the model's target parameters, little to no manual work is required. Second, the approach is highly flexible, as the model can include basically arbitrary data into its prediction. Further, it does not have the risk of creating a resource bottleneck through overtly specific queries.

In conclusion, while the model-based approach has higher requirements with regards to the capabilities of the resource manager than the static approach, it can also be expected to yield superior results.

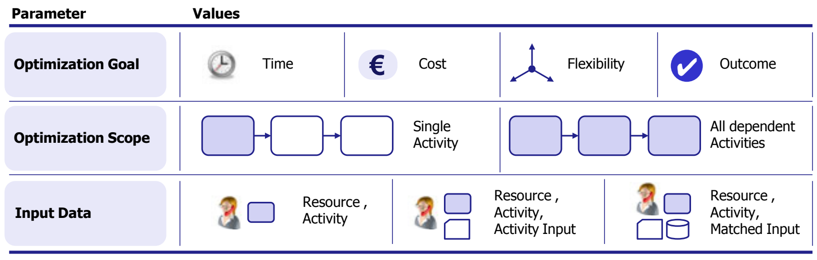

Fig. 4. Resource Manager Parameters

3.2 Resource Manager Capabilities

As the previous section has shown, a dynamic, model-based resource management approach is often likely to yield better results than (static) alternatives. As a monolithic model that can't be parameterized will only perform satisfyingly in a limited set of scenarios, the first step to making this approach work is determining the variability parameters that the resource manager should provide for. We distinguish between parameters that affect the model itself and those that are concerned with the final selection of the resources.

The model parameters need to ensure that the resource manager can provide a process analyst with the optimal model for a given situation. As Figure 4 shows, our resource manager offers three different model parameters. The first parameter concerns the optimization goal Go, such as process time or cost. This is important, as some of the goals are conflicting [RLM05] and it is unlikely that a resource is best along all dimensions (e.g., a highly trained employee might be quite fast, but also expensive). The optimization scope S is the second parameter, which is important when a resource is used by more than one activity. It indicates if the selection optimization only takes the performance of the resource for the current activity or for all activities that it executes into account. The third parameter is the amount of data D to be considered in the determination of the likely optimal resource - either only the resource attributes or also the work item attributes (either as modeled in the process or matched with operational data [RNB10]).

Next to the model parameters, the resource manager further supports additional selection parameters Con, e.g., to specify a maximum resource utilization threshold.

Using the parameters discussed above as well as the foundations introduced in Section 2, we can now define a model-based resource query Q_M as follows:

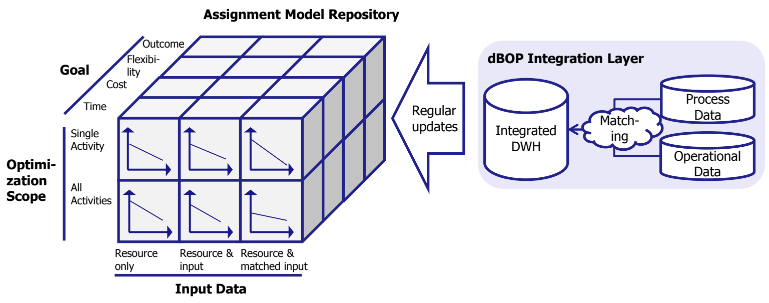

Fig. 5. Multidimensional Resource Assignment Model Repository

Definition. *Model-based resource query: A model-based resource query Q_M is a map $a \times T \times Ro \times Rn \times Rel[\times \wp(AC)] \times Go \times S \times D[\times \wp(CC)] \rightarrow R$, where*

1. *$a \in N$ is the activity initially requesting the resource.*
2. *T, Ro, Rn, Rel, $\wp(AC)$ are defined as in the definition of the basic resource query.*
3. *$Go \in \{TIME, COST, QUALITY, FLEXIBILITY\}$ determines the optimization goal.*
4. *$S \in \{a, SUCC_R(a, Rn)\}$ determines the optimization scope.*
5. *$D \in \{\mu(R) \cup a, \mu(R) \cup \iota(a) \cup a, \mu(R) \cup \mu(\iota(a)) \cup a\}$ indicates which data to use for the model.*
6. *$CC \in Con \times COMP \times DOMAIN(Con)$ lists further constraints to be considered, with $COMP$ defines as in Q_B.*

3.3 Prediction Model Design and Management

To be able to effectively deal with the parameters discussed in the previous section, the resource manager needs to ensure that the prediction model performs well, that the prediction can be done quickly even on large data sets and that the prediction model is current with respect to the recent execution history.

The first choice to make to satisfy these requirements is selecting an appropriate model type. After evaluating neural networks, multilinear regression models, regression trees [HK06] and M5 model trees [Qui92] we have decided to use the latter. M5 model trees are similar to "classical" classification trees [HK06], however, instead of a class label, each leaf node contains a regression model. It offers similar prediction performance to neural networks in most scenarios and superior performance to the other model types. Further, compared to neural networks, the model construction has proven to be considerably faster and more robust in our experiments. Our resource manager hence predicts the resource performance with a M5 model tree with the selected optimization goal as a target variable.

The second choice is how to store and when to update the model. The two extremes for these choice would be to either dynamically construct the model

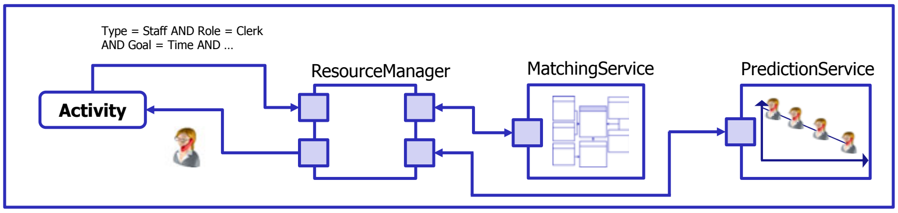

Fig. 6. Resource Manager Implementation

anew whenever a new resource request is or to initially construct a model for all possible model parameters, which is then used to process all future requests. However, the first extreme is impractical due to its high overhead for constructing the models and the second extreme is likely to become outdate. Our resource manager hence uses the third option, which caches and updates the models based on some user-defined criterion (e.g., time elapsed, number of activity executions etc.). To achieve optimal performance, a model is cached for every parameter combination in the multidimensional assignment model repository, as shown in Figure 5. In the context of the architecture shown in Figure 2, this model cache is part of the *Process Insight Repository*.

3.4 Resource Manager Implementation

The final step for making model-based resource assignment work is providing the resource manager's services at run time to business processes. For that, the resource manager is embedded in our *dBOP* platform and implemented as shown in Figure 6. First, the resource manager receives the resource request. Depending on the required input data, the request is further enriched with additional data according to the available mappings to, e.g., a data warehouse [RNB10]. Then, the appropriate model is selected from the model repository (which is implemented based on the WEKA [HFH$^+$09] library). Finally, the performance of the resources that match the query and the constraints is computed and the best resource is returned.

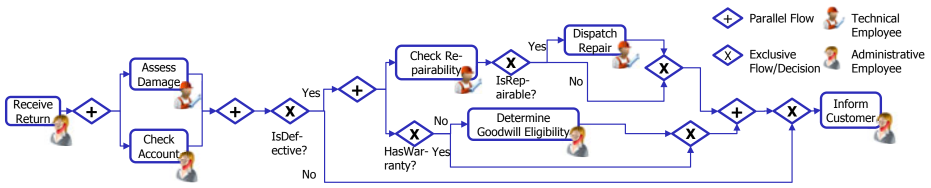

Fig. 7. Mail Order Return Process

4 Evaluation

Using a simplified sample process fragment taken from the retail business domain, we will demonstrate in this section quantitatively the benefit of a prediction model-based resource manager as well as the effects of the different model settings. First, we will briefly introduce the sample scenario and the evaluation setup. Then, we will discuss the evaluation results and their impact on the work presented in this paper. Finally, we discuss further opportunities and limitations of the presented approach.

4.1 Evaluation Design

The evaluation design is based on the sample fragment of the mail order return handling process introduced in Section 1 which is shown in Figure 7. The performance of an employee at each of the depicted tasks can vary greatly, depending on, e.g., experience, (formal) education, familiarity with the particular item being requested or specific received. Hence, this scenario provides a good testing environment for giving a first indication of the benefits that can be realized through a model-based resource assignment approach. As the benefit of using integrated process data has been shown in related work by the authors [NMR+11], this evaluation focuses on two other aspects. First, it measures the impact of different optimization scopes S on the process performance. Second, the impact of different levels of resource utilization are assessed to validate the hypothesis that the benefit of the model-based approach increases with the number of available resources. For that purpose, the evaluation is set up as follows:

- **Scenarios:** Nine different scenarios are measured by combining two dimensions. The first dimension is the level of optimization used, with either no optimization (i.e., a basic resource query Q_B) or with "single activity" or "all activities" optimization scope. The second dimension is the share of resources available during the evaluation, ranging from low (utilization $> 80\%$) to high (utilization $< 20\%$).
- **Data:** In total, we use a set of 1.800 sample processes in the evaluation. The models are built using 1.200 of these sample processes executed using a resource manager without enabled optimization. The evaluation (testing) itself is performed using 600 samples in each of the scenarios listed above.
- **Success measurement:** The success of the respective approach is measured by the performance of the process fragment with respect to the selected goal function (process duration).

4.2 Results

The first observation that can be made from the results shown in Figure 8 is that there is, in this scenario, a clear benefit from using the model-based approach. When using model-based resource allocation, the process performance is on average significantly better than in the non-optimized case. Further, using the

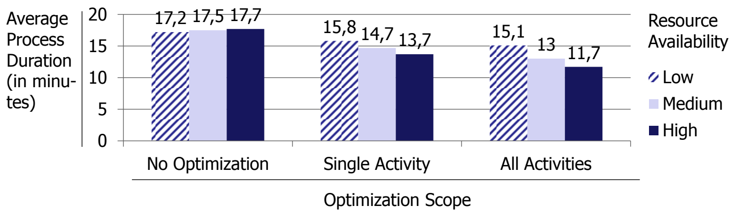

Fig. 8. Evaluation Results

"all activities" optimization scope yields better results than the "single activity" optimization scope.

The second observation is concerned with the impact of different resource utilization rates. The evaluation shows that the lower the utilization rate, the higher the benefit of the model-based approach. This can be attributed to the larger number of resources that are available at any given time, which also increases the likelihood that a resource with a good predicted performance is available.

4.3 Current Limitations and Further Development

Through the case study discussed in this section, we have demonstrated that a model-based resource assignment can be of significant value for improving process performance in certain scenarios. Specifically, the case study has illustrated the usefulness in a scenario where the work to be done can be freely distributed, each process instance or work item is assumed to be of equal value to the process stakeholders and there are no specific resource setup times, e.g., when the type of work item being processed changes. These properties are typical in a "classical" business process management scenario, however, not in a manufacturing scenario.

In a manufacturing context, the work item is typically physical, the priority of work items can vary significantly and there are considerable setup times. To deal with these additional complexities, the presented resource management approach needs to be further extended. Some of these extensions include a cost function for moving work items to a certain resource, a value function for work items and the inclusion of setup times when deciding on which resource to use. All of these extensions are considered in our current work on the topic, which deals with applying the *dBOP* platform to a manufacturing context, as is also discussed in Section 6.

5 Related Work

This paper is part of our work on the *dBOP* platform [NRM10]. The data integration layer is discussed in [RNB10]. The methods employed in the analysis layer are adapted from standard data mining and machine learning literature [HK06] [Qui92]. Examples for their application can be found in [NMR^{+}11]. The optimization layer builds heavily on existing research into business process optimization techniques, such as [RLM05]. Its role within the *dBOP* is the subject of [NRM11].

The approach closest to the one presented in this paper is the classifier-based approach to resource assignment presented in [LWYS08]. However, our approach extends the presented approach in several dimensions. First, the resource manager seems to offer only a single optimization mode. Second, it lacks data integration capabilities, which - as the authors themselves argue - significantly improves classifier performance. Finally, it is unclear how the authors envision the resource manager to be used during an actual process execution. [YJJ07] discusses and validates the feasibility of using prediction models for resource assignment by comparing the results of the model to human judgment in a manufacturing setting. Similarly, [AHM06] show the feasibility of that approach in a software engineering scenario. [JT09] presents an approach that uses association rule mining to assist in the creation of more sophisticated resource assignment rules - basically, a specific variant of the static approach discussed in Section 3.1. [LRDR06] use the mining of workflow data to determine staff assignment rules from the workflow's execution data.

6 Conclusion and Outlook

This contribution introduces a model-based approach to resource assignment that is embedded in the *dBOP*, an integrated platform for analytical process improvement. We have discussed the benefits of such a dynamic approach over the static approach focused on improving the assignment rules themselves. After discussing the details of the model-based resource manager's capabilities and its design, we have illustrated the usefulness of our approach using an order management case study.

As both the core components of the *dBOP* in general and the resource manager in particular have been successfully implemented, our current work focusses on two tasks. First, we are currently working on transferring the tools and concepts of the *dBOP* to the manufacturing application domain. Within this effort, we are also addressing the limitations of the approach discussed in Section 4.3. Second, we are conducting a thorough evaluation of the platform, both from an effectiveness and from a user's perspective. For that purpose, we are currently conducting user studies on the topic and collaborating with several companies for determining additional application scenarios.

References

[AHM06] Anvik, J., Hiew, L., Murphy, G.C.: Who should fix this bug? In: Proceedings of the 28th International Conference on Software Engineering, pp. 361–370. ACM (2006)

[DKDBvdV02] De Koster, R.B.M., De Brito, M.P., van de Vendel, M.A.: How to organise return handling: an exploratory study with nine retailer warehouses. International Journal of Retail & Distribution Management 30(8/9), 407–421 (2002)

[HFH$^+$09] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. ACM SIGKDD Explorations Newsletter 11(1), 10–18 (2009)

[HK06] Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann (2006)

[JT09] Jablonski, S., Talib, R.: Agent assignment for process management: agent performance evaluation. In: Proceedigns FIT 2009 (2009)

[LR00] Leyman, F., Roller, D.: Production Workflow. Prentice-Hall, Englewood Cliffs (2000)

[LRDR06] Ly, L., Rinderle, S., Dadam, P., Reichert, M.: Mining Staff Assignment Rules from Event-Based Data. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 177–190. Springer, Heidelberg (2006)

[LWYS08] Liu, Y., Wang, J., Yang, Y., Sun, J.: A semi-automatic approach for workflow staff assignment. Computers in Industry 59(5), 463–476 (2008)

[NMR$^+$11] Niedermann, F., Maier, B., Radeschütz, S., Schwarz, H., Mitschang, B.: Automated Process Decision Making based on Integrated Source Data. In: Abramowicz, W. (ed.) BIS 2011. LNBIP, vol. 87, pp. 160–171. Springer, Heidelberg (2011)

[NRM10] Niedermann, F., Radeschütz, S., Mitschang, B.: Deep Business Optimization: A Platform for Automated Process Optimization. In: Proceedings BPSC 2010 (2010)

[NRM11] Niedermann, F., Radeschütz, S., Mitschang, B.: Business Process Optimization Using Formalized Optimization Patterns. In: Abramowicz, W. (ed.) BIS 2011. LNBIP, vol. 87, pp. 123–135. Springer, Heidelberg (2011)

[Qui92] Quinlan, J.R.: Learning with continuous classes. In: 5th Australian Joint Conference on Artificial Intelligence (1992)

[RLM05] Reijers, H.A., Liman Mansar, S.: Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics. Omega 33(4), 283–306 (2005)

[RNB10] Radeschütz, S., Niedermann, F., Bischoff, W.: BIAEditor - Matching Process and Operational Data for a Business Impact Analysis. In: Proceedings EDBT (2010)

[YJJ07] Yingbo, L., Jianmin, W., Jiaguang, S.: A machine learning approach to semi-automating workflow staff assignment. In: Proceedings of the 2007 ACM Symposium on Applied Computing (2007)