Supporting the Full BPM Life-Cycle Using Process Mining and Intelligent Redesign

Wil M.P. van der Aalst, Mariska Netjes, and Hajo A. Reijers

Eindhoven University of Technology, Department of Technology Management,
PO Box 513, NL-5600 MB Eindhoven, The Netherlands
w.m.p.v.d.aalst,m.netjes,h.a.reijers@tm.tue.nl

Abstract. Business Process Management (BPM) systems provide a broad range of facilities to enact and manage operational business processes. Ideally, these systems should provide support for the complete BPM life-cycle: (re)design, configuration, execution, control, and diagnosis of processes. However, based on an extensive evaluation of the FileNet P8 BPM Suite, we show that existing BPM tools are unable to support the full life-cycle. There are clearly gaps between the various phases (i.e., users need to transfer or interpret information without any support) and some of the phases (e.g., the redesign and diagnosis phases) are not supported satisfactorily. We selected the FileNet P8 BPM Suite because it is consistently ranked as one of the leading commercial BPM systems and representative for the current generation of BPM products. Based on this evaluation, we show what is needed to close the BPM life-cycle and seamlessly support all phases. We will argue that techniques for process mining and intelligent redesign are needed to support the (re)design and diagnosis phases and thus close the BPM life-cycle. We also briefly report on the work done in the context of the ProM tool which is used as framework to experiment with such techniques.

Keywords: Business Process Management, Workflow Technology, Process Mining, Business Process Simulation, Business Process Intelligence, FileNet.

1 Introduction

Business Process Management (BPM) systems can be seen as successors of Workflow Management (WFM) systems, which became popular in the mid-nineties. However, already in the seventies people were working on office automation systems which are comparable with today's WFM systems. Consider, for example, the OfficeTalk system developed by Ellis et al. at Xerox that was already able to support administrative processes based on Petri-net-based specifications of procedures (Ellis, 1979). Today, many WFM systems are available (Aalst & Hee, 2004a; Jablonski & Bussler, 1996; Lawrence, 1997; Mühlen, 2004). The core functionality of these systems can be described as *the ability to support an operational business process based on an explicit process model*, i.e., automating the "flow of work" without necessarily automating individual activities.

Recently, WFM vendors started to position their systems as BPM systems. We define BPM as follows: *Supporting business processes using methods, techniques, and software to design, enact, control, and analyze operational processes involving humans, organizations, applications, documents and other sources of information* (Aalst, Hofstede, & Weske, 2003). This definition restricts BPM to operational processes, i.e., processes at the strategic level and processes that cannot be made explicit are excluded. It also follows that systems supporting BPM need to be "process aware". After all, without information about the operational processes at hand little support is possible. When comparing classical definitions of WFM (Lawrence, 1997) with the above definition of BPM, one could conclude that the main goal of BPM systems is to offer a broader set of functionalities and support of the whole process life-cycle. This is also the "sales pitch" that many vendors use to market their products. However, analysis of existing BPM systems shows that the functionality of these systems leaves much to be desired. In the first part of this paper we analyze the limitations of today's BPM systems. Based on this analysis problems are identified and in the second part of this paper, we show how to address these problems.

1.1 Step 1: Evaluation of the FileNet P8 BPM Suite

The first goal of this paper is to analyze whether today's BPM systems actually support the BPM life-cycle. To do this we use the BPM life-cycle as depicted in Figure 1. This life-cycle identifies five phases (*design*, *configuration*, *execution*, *control*, and *diagnosis*), which will be described later. The depicted life-cycle is an extension of the life-cycle presented in (Aalst, Hofstede, & Weske, 2003). We will discuss the desired functionality in each of the phases. To make things more concrete, we have evaluated one particular system in detail: FileNet P8 BPM Suite (Version 3.5). We have selected this system because it is considered as one of the leading commercial BPM systems (Gartner, 2003, 2004, 2005). Moreover, the system is explicitly positioned by the vendor as a tool to support the whole BPM life-cycle.

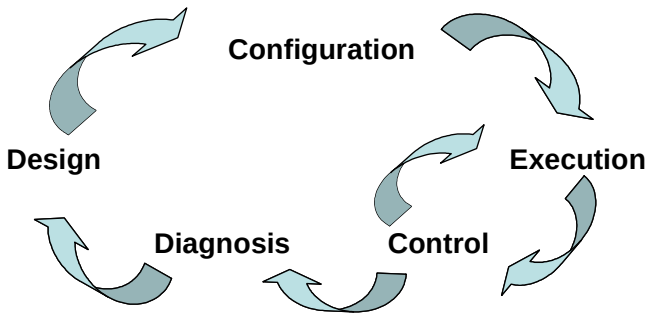

Fig. 1. The BPM life-cycle.

We analyze the support of the FileNet P8 BPM Suite in each of the five phases shown in Figure 1. For our evaluation we performed a full pass through these phases using five realistic workflow scenarios, each including a concrete workflow process and life-cycle context. We have used five workflows to be able to obtain additional insights when necessary. As starting point for our evaluation, we will assume that each workflow has already made one pass through the BPM cycle. The name and the related literature for each of the workflows is provided in Table 1. These particular workflows have been selected because the papers describing them provide a diagnosis of the improvement points and one or more alternative designs. Also, the original workflows and the alternatives have already been tested and the underlying data were available to us.

Table 1. The workflows used in our analysis.

Workflow Name	Reference
Intake_Admin	(Reijers, 2003)
Credit application	(Reijers, 2003)
Intake_Meetings	(Jansen-Vullers & Reijers, 2005; Reijers, 2003)
Bank account	(Netjes, Aalst, & Reijers, 2005)
Mortgage request	(Aalst, 2001; Netjes, Vanderfeesten, & Reijers, 2006)

1.2 Step 2: An Approach Based on Process Mining and Intelligent Redesign

Based on the evaluation of the FileNet P8 BPM Suite using the five workflows mentioned in Table 1, we noted that there was little support for the diagnosis and design phases in the BPM life-cycle. Moreover, the transfer of information from the run-time environment to the design-time environment is hardly supported. When looking at other BPM products we see the same limitations. Therefore, the second part of this paper is concerned with a more detailed analysis of these problems and a proposal for a solution addressing them. Figure 2 sketches the link between Step 1 (Evaluation of systems like the FileNet P8 BPM Suite) and Step 2 (Our proposal to address some of the problems identified). On the left-hand side of Figure 2, we highlight the part of the BPM life-cycle we consider to be most problematic and typically poorly supported by BPM systems. The right-hand side of Figure 2 shows a diagram analyzing the problem and proposing an approach based on process mining and intelligent redesign. We will elaborate on this diagram later in the paper.

Our approach is based on the observation that in current BPM systems there are two core problems:

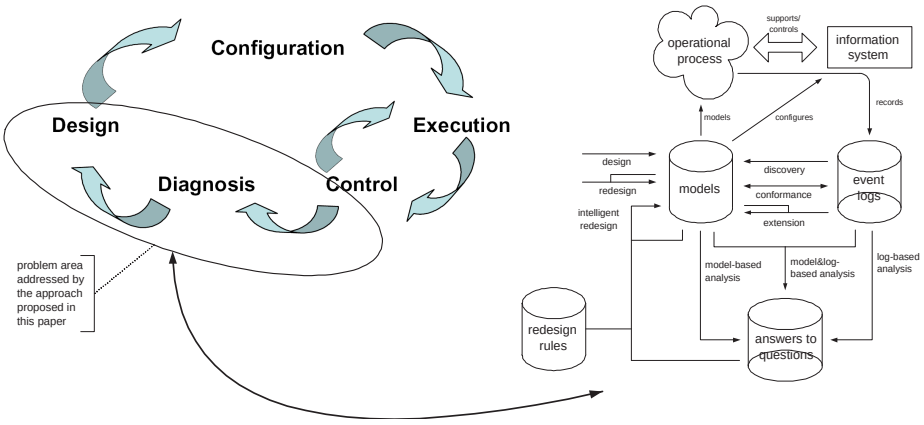

Fig. 2. Problems in the BPM life-cycle related to the approach presented in this paper.

- *Problem 1:* The actual execution of the process supported by the BPM system is completely disconnected from the (re)design of the process. To close the BPM life-cycle it is necessary to automatically interpret information stored in event logs and use this to discover, check, and enhance models describing what is really going on in the business process. To address this problem we propose to use *process mining techniques* (Aalst, Weijters, & Maruster, 2004; Aalst, Reijers, & Song, 2005; Agrawal, Gunopulos, & Leymann, 1998; Cook & Wolf, 1998; Datta, 1998; Weijters & Aalst, 2003).
- *Problem 2:* BPM systems offer graphical editors and sometimes also simple analysis facilities to analyze the design.[1] Several BPM systems (including the FileNet P8 BPM Suite) offer basic simulation facilities. However, these facilities are typically considering a very abstract situation (with many assumptions about reality that do not hold) and provide only "what-if" analysis. This means that it is difficult to use these systems and the designer has to come up with ideas for redesigns. The goal of *intelligent redesign* is twofold. First of all, we want to use the information about the real process obtained through process mining. Second, we want to move beyond "what-if" analysis, i.e., the system should automatically suggest and evaluate different redesign possibilities.

In the context of the *ProM framework* (Dongen, Medeiros, Verbeek, Weijters, & Aalst, 2005) we are developing automated support for process mining and intelligent redesign in an effort to address the two problems mentioned above. The goal of this paper is not to provide detailed solutions but to show the potential of truly closing the BPM life-cycle.

[1] In this context we are not referring to the verification of models, i.e., the goal is not to assess the internal consistency (e.g., absence of deadlocks) but to improve the process from a business perspective.

The remainder of this paper is organized as follows. Section 2 discusses related work. Then, in Section 3, we evaluate the FileNet P8 BPM Suite based on a generic approach which can also be applied to other BPM systems. Based on this evaluation we identify the main problems and propose in Section 4 an approach involving techniques for process mining and intelligent redesign. Section 5 concludes the paper.

2 Related Work

Since the early nineties, workflow technology has matured (Georgakopoulos, Hornick, & Sheth, 1995) and several textbooks have been published, e.g., (Aalst & Hee, 2004a; Dumas, Aalst, & Hofstede, 2005; Jablonski & Bussler, 1996; Leymann & Roller, 1999; Marinescu, 2002; Mühlen, 2004). Most of the available systems use some proprietary process modeling language and, even if systems claim to support some "standard", there are often all kinds of system-specific extensions and limitations. Petri nets have often been used as an abstraction of these languages: both for the purpose of modeling the control-flow aspects workflows as for the analysis of workflows and different workflow languages and systems (Aalst & Hee, 2004a; Aalst, Hofstede, Kiepuszewski, & Barros, 2003; Dumas et al., 2005). The evaluation of workflow and BPM systems has been the domain of consultancy firms such as Gartner (Gartner, 2003, 2004, 2005) and many others. These evaluations are typically not very rigorous and lack objective criteria with respect to the functionality of systems. The *Workflow Patterns* initiative, cf. www.workflowpatterns.com, has been one of the few attempts to evaluate systems and languages in a systematic manner (Aalst, Hofstede, Kiepuszewski, & Barros, 2003). This paper does not use the workflow patterns to evaluate the FileNet P8 BPM Suite because we want to look at the support of the whole BPM life-cycle. Note that the workflow patterns focus on specific aspects of the system such as the control-flow perspective, the data perspective, or the resource perspective. Therefore, we used an approach were we selected five processes (Table 1) and for each of these processes we went through the whole BPM life-cycle.

It is impossible to give a complete overview of process mining here. Therefore, we refer to a special issue of Computers in Industry on process mining (Aalst & Weijters, 2004) and a survey paper (Aalst, Dongen, et al., 2003) for more references. We will show that process mining techniques can be split into three categories: (1) discovery, (2) conformance, and (3) extension. Techniques for *discovery* try to generate a model based on the event logs. This may be a process model (Aalst et al., 2004; Agrawal et al., 1998; Cook & Wolf, 1998; Datta, 1998; Weijters & Aalst, 2003), but also other models focusing on different perspectives (e.g., a social network (Aalst, Reijers, & Song, 2005)) can be discovered using process mining. Techniques for *conformance* checking (Rozinat & Aalst, 2006a) aim at exposing the difference between some a-priori model (e.g., a Petri net describing the control-flow) and the real process observed via the event log.

Process mining techniques for model *extension* take some a-priori model (e.g., a control-flow model) and project other information on it derived from the log, e.g., a Petri net can be extended by providing a decision tree for each choice in the system (Rozinat & Aalst, 2006b). This way data and performance aspects can be projected on some a-priori process model obtained directly from the system or discovered through process mining.

Process mining can be seen in the broader context of Business (Process) Intelligence (BPI) and Business Activity Monitoring (BAM). In (Grigori et al., 2004; Grigori, Casati, Dayal, & Shan, 2001; Sayal, Casati, Dayal, & Shan, 2002) a BPI toolset on top of HP's Process Manager is described. The BPI toolset includes a so-called "BPI Process Mining Engine". In (Mühlen & Rosemann, 2000) the authors describe the PISA tool which can be used to extract performance metrics from workflow logs. Similar diagnostics are provided by the ARIS Process Performance Manager (PPM) (IDS Scheer, 2002). The latter tool is commercially available and a customized version of PPM is the Staffware Process Monitor (SPM) (TIBCO, 2005) which is tailored towards mining Staffware logs.

Literature on "intelligent" redesign is limited (Netjes, Vanderfeesten, & Reijers, 2006). Clearly there are many techniques originating from operations management (e.g., simulation, queueing networks , etc.) that can be applied to workflow processes (Buzacott, 1996). However, these typically only support "what-if" analysis and do not provide suggestions for improvement.

Several researchers have focused on the problem of not meeting certain deadlines in a process. For example, in (Panagos & Rabinovich, 1997) an approach to dynamically adjust deadlines based on costs, expected execution times, and available slack time is described. In (Panagos & Rabinovich, 1998) this approach is refined and supported by simulation experiments. In (Eder, Panagos, & Rabinovich, 1999; Eder, Panagos, Pezewaunig, & Rabinovich, 1999) the topic of capturing time constraints in workflow definitions is considered, and a PERT-like technique for the analysis of the temporal behavior of a workflow is proposed. This technique is similar to the one employed by the "prediction engine" of Staffware (Staffware, 2003). There also several papers that address timing (e.g., determining performance indicators based on simulation or some analytical method) without considering what to do in case deadlines are not met, cf. (Reijers, 2003) for an overview. For example, in (Zhao & Stohr, 1999) different task prioritization policies are compared with respect to turnaround times.

Few of the simulation techniques and systems described in literature use historic data. Even fewer use the current state of a workflow in their analysis. A notable exception is (Reijers & Aalst, 1999) which proposes the concept of a so-called short-term simulation, i.e., a "fast forward" into the near future based on historic and current data.

Most related to the type of intelligent redesign discussed in this paper are the redesign rules defined in (Reijers, 2003). One example of such a rule is the "knock-out rule" which describes how to reorder activities in a sequential or parallel "checking process" consisting of multiple tests (Aalst, 2001). Also related

are the "Process Recombinator" tool developed in the context of the MIT Process Handbook (Bernstein, Klein, & Malone, 1999) and the KOPeR tool described in (Nissen, 1998). The Process Recombinator tool (Bernstein et al., 1999) uses the notions of (i) process specialization and (ii) coordination mechanisms to generate new designs on the basis of a list of core activities. From these process designs the user can select the most promising ones. The KOPeR tool (Nissen, 1998) attempts to automate three activities required for process redesign: process measurement, pathology diagnosis, and transformation matching. The idea is that a limited set of process measures (e.g. process length, process handoffs, etc.) can be used to identify process pathologies in a given process (e.g. a problematic process structure, fragmented process flows, etc.). These process pathologies can then be matched to redesign transformations known to effectively deal with these pathologies. Note that the KOPeR tool only provides ideas for redesign and does not generate new designs.

3 Evaluation of the FileNet P8 BPM Suite

As described in the introduction, we start by analyzing the FileNet P8 BPM Suite.[2] The main goal of this evaluation is not to discuss the specifics of this particular system, but to provide some insights into the functionality of today's BPM systems. The evaluation approach that we will present is based on the BPM life-cycle and can also be applied to other systems. We selected the FileNet P8 BPM Suite for two reasons. First of all, it is consistently ranked as one of the leading commercial BPM systems (Gartner, 2003, 2004, 2005). Second, it is representative for the current generation of BPM products. In fact, when it comes to supporting the full BPM life-cycle we expect most systems to offer less functionality (Gartner, 2003, 2004, 2005).

3.1 Evaluation Approach Based on the BPM Life-Cycle

First, we present our system-independent approach to evaluate BPM systems. Pivotal to our evaluation approach is the BPM life-cycle depicted in Figure 1. Clearly, we want to evaluate the degree to which each phase is facilitated by a BPM system. Moreover, we want to asses the interoperability among phases, i.e., can information obtained or created in one phase be used in another phase? For example, a BPM system may incorporate a simulation tool, but it may be the case that the simulation model and the model used for execution are incompatible, forcing the user to re-create models or to set parameters twice.

First, we focus on the *design phase*. In case of an already existing process the goal of this phase is to create an alternative for the current process. This alternative should remedy the diagnosed weaknesses of the process according to the identified improvement possibilities. As indicated in Figure 1, this phase is in-between the diagnosis phase and the configuration phase, i.e., input from the

[2] This section is based on (Netjes, Reijers, & Aalst, 2006b).

diagnosis phase is used to identify improvement opportunities (e.g., bottlenecks or other weaknesses) and the output is transferred towards the configuration part of the BPM system. The resulting process definition consists of the following elements (Aalst & Hee, 2004a):

- the process structure,
- the resource structure,
- the allocation logic, and
- the interfaces.

We would like to emphasize that a graphical editor by itself does not offer full support for the design phase. In the design phase the designer wants to experiment with designs, evaluate designs, and use input from the diagnosis phase. Some systems offer a simulation tool to support the design phase. Unfortunately, such a tool is often disconnected from the diagnosis phase, i.e., it is impossible to directly use historic data (e.g., to estimate service time distributions or routing probabilities). Moreover, simulation tools typically offer only what-if analysis, i.e., the designer has to come up with ideas for alternative designs and needs to analyze each alternative separately without sufficient tool support (Netjes, Vanderfeesten, & Reijers, 2006).

The *configuration phase* focuses on the detailed specification of the selected design. Note that in the design phase the emphasis is on the performance of the process, while in the configuration phase the emphasis shifts to the realization of the corresponding system. In principle, the design and configuration phase could use a common graphical editor, i.e., the configuration phase details the process definition created in the design phase. However, it is important (a) that the user is not forced to bypass the editor to code parts of the process and (b) that technical details do not need to be addressed in the design phase. If both phases use different tools or concepts, interoperability issues may frustrate a smooth transition from design to configuration.

In the *execution phase* the configured workflow becomes operational by transferring the process definition to the workflow engine. For the workflow execution not only the process definition data is required, but also context data about the environment with which the BPM system interacts. Relevant environmental aspects are:

- information on arriving cases,
- availability and behavior of internal/external resources and services.

The execution part of the BPM system captures the context data and relates it to specific instances of the workflow.

The execution of the operational business process is monitored in the *control phase*. The control part of the BPM system monitors on the one hand individual cases to be able to give feedback about their status and on the other hand, aggregates execution data to be able to obtain the current performance of the workflow. The monitoring of specific cases is done with the data from individual process executions without any form of aggregation, while obtaining the performance indicators requires aggregation of these data. Information about running

cases can be used as input for the diagnosis phase. However, it can also be used to make changes in the process. For example, temporary bottlenecks do not require a redesign of the process, but require the addition of resources or other direct measures (e.g., not accepting new cases). Hence, the control phase also provides input for the execution phase.

In the *diagnosis phase* information collected in the control phase is used to reveal weaknesses in the process. In this phase the focus is usually on aggregated performance data and not on individual cases. This is the domain of process mining (Aalst, Dongen, et al., 2003), business process intelligence (Grigori et al., 2004), data warehousing, and classical data mining techniques. This diagnosis information is providing ideas for redesign (e.g., bottleneck identification) and input for the analysis of redesigns (e.g., historic data) in the design phase.

As indicated, it is not sufficient to support each of the five phases in isolation: interoperability among phases is vital for the usability of a BPM system. Consider for example the role of simulation. In a worst case scenario, a BPM system could offer a simulation tool that, on the one hand, cannot directly read the current workflow design used for execution (or relevant information is lost in some translation) and, on the other hand, cannot use any historic data to extract information about service times, routing probabilities, workloads, resource availability. Such a simulation tool probably offers little support for the BPM life-cycle (Reijers & Aalst, 1999).

3.2 Applying the Evaluation Approach to FileNet

We will evaluate the available BPM support by conducting a full pass through the BPM cycle with the aid of several tools from the FileNet P8 BPM Suite. We have evaluated the FileNet P8 BPM Suite, Version 3.5. The system has been used with Microsoft Windows 2000 as operating system, a Microsoft SQL Server as database, BEA Weblogic as J2EE application server and Microsoft Internet Explorer as browser. The P8 BPM Suite consists of six parts: Workflow Management, process design, process simulation, process tracking, process analysis and document review & approval (www.FileNet.com). The evaluation of FileNet's BPM abilities focuses on the tools supporting the first five parts. Document review & approval is not relevant for the evaluation; it only facilitates process management. In the remainder of this section, we consider FileNet's capabilities for each of the five BPM phases (design, configuration, execution, control, and diagnosis). A detailed illustration of the BPM support offered by FileNet can be found in (Netjes, Reijers, & Aalst, 2006a) where we present the full pass through the BPM life-cycle for one of the five workflow scenarios.

Design We start our evaluation with the design phase. For each of the five workflow scenarios mentioned in Table 1 we would like to create an alternative workflow with help from the FileNet P8 BPM Suite. We assume these workflows have already made one pass through the BPM cycle, meaning that the original workflow model and data from execution are present in the FileNet system.

A workflow model for which an alternative should be made can be loaded in the FileNet *process designer*, which, however, does not support the creation of one or more alternatives. The redesign of the original model to obtain a better performing alternative should be done manually. For each of the workflows we take the alternatives described in the related paper and use the *process designer* to change the original model to the alternative model. One of the alternative designs made with the *process designer* is shown in Figure 3. The depicted design presents a medical process in which a mental patient is registered and assigned to medical employees (intakers), and for which intake meetings are planned. A detailed description of the process is available in (Reijers, 2003). More information on the modeling of workflows with the FileNet *process designer* can be found in (Netjes, Reijers, & Aalst, 2006a).

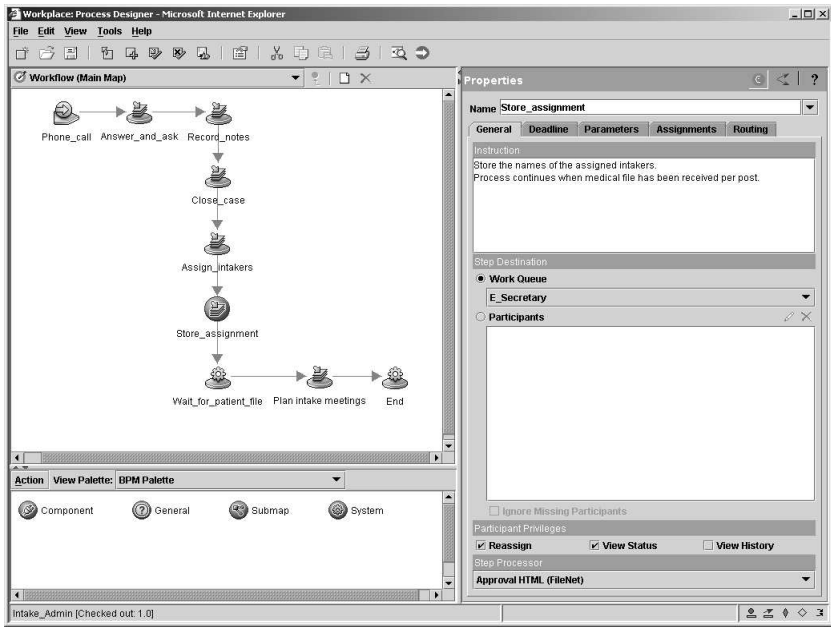

Fig. 3. Workflow model in the *process designer*.

The performance of each of the created alternatives should be evaluated to find the best alternative. For this we use the FileNet *process simulator*. For each alternative we create a simulation scenario for which we import the process steps, their order and the allocation logic defined with the *process designer*. The imported data can not be changed in the *process simulator*, but a replacement can be imported from the *process designer* without the loss of settings. Other process definition data should be added to the simulation scenario manually. *Jobs* are connected to the process steps and assigned to *resources* which are

allocated according to *shifts*. The notion of *shifts* allows for the scheduling of resources over the available working hours. Relating these *jobs*, *resources* and *shifts* to each other is rather complicated, because only one definition window can be open at the time and relations should also be indicated when defining a *job*, *resource* or *shift*.

In addition to the definition data there is context data required to perform a simulation. Historic data is present in the system, but it can only be used in a limited way. Historic information on arriving cases can be transferred to the *process simulator*, but all other data, like processing times and routing probabilities, should be derived from the execution data and included manually. It is only possible to provide constant values for the simulation parameters, so the simulation results will only provide a rough indication for the performance of a scenario. Simulation results are generated fast and with no additional effort. The use of the FileNet *process simulator* is in detail explained in (Netjes, Reijers, & Aalst, 2006a). A simulation scenario with simulation results is depicted in Figure 4. For each of the five workflows we choose the best alternative which we specify in detail in the configuration phase.

Configuration The FileNet *process designer* is also used for the configuration of the chosen alternative workflows and offers interoperability between the design and the configuration phase. In the design phase we already specified the process structure and the mapping of resources to tasks for each workflow with the *process designer*. The more complicated parts of the process structure are detailed out in the configuration phase. Each workflow model contains one or more complex constructs, but besides one construct, we have been able to configure them all with the *process designer*. The resource structure, the allocation rules and the interfaces are defined outside the *process designer*. Defining outside the process designer allows for sharing with other processes, making the resource structure and the allocation rules reusable for other process definitions. All five workflows use the same allocation rules and some workflows have the same resource structure. The complete configuration of the five workflows, both inside and outside the *process designer* has been done in two working days. The configuration phase is strongly supported by the FileNet P8 BPM Suite.

As closure of the configuration phase, the workflow model is checked for completeness by the system and a workflow instance could be launched to pretest the execution of the workflow. Another possible check would have been a check on the correctness of the model, conform the verification of workflow processes provided by the Woflan tool (Verbeek, Basten, & Aalst, 2001), but such a verification is not supported by the FileNet system. The configuration of the workflows is necessary for their execution.

Execution The execution phase is started with the transfer of the workflow configurations to the FileNet *process engine*. All process definition data are transferred to the *process engine* providing interoperability between the configuration and the execution phase. Resources work on the processes in operation

via an inbox. The FileNet P8 BPM Suite offers integration with external applications, document management, integration with content management, and interaction between inter-related processes. The FileNet system supports the execution phase in an excellent way. We expected mature support for execution, because this support has traditionally been the heart of a WFM system and many systems provide extended support for the execution phase. In the execution phase context data is related to each specific instance of a workflow and this combination of definition and context data is used for the control of a workflow.

Control In the control phase, the operational business process is monitored to follow individual cases and to obtain the performance of a workflow. The first way of monitoring is supported by the FileNet *process administrator* and the second by the *analysis engine*, providing a strong support for the control phase.

The execution data for individual cases and other workflow events are logged by the *process engine*. The history of a certain workflow, step or work item can be tracked in the log through the FileNet *process administrator*. For the workflows with conditional routing this gives the opportunity to determine which steps were executed for a specific case. With the *process administrator* it can also be determined how certain decisions were made during execution allowing us to see at which point and why a certain case was rejected.

The performance of a workflow is read from aggregated execution data. The execution data present in the *process engine* is aggregated and parsed to the FileNet *analysis engine*. Interoperability exists between the execution and the control phase, because all execution data necessary for control are available either through the *process engine* or the *analysis engine*. The aggregated performance data resides on a separate engine to not affect the performance of the *process engine*. Reporting and analysis of the aggregated data is facilitated by twenty out-of-the-box reports; each graphically presenting the data related to one performance indicator. It is possible to specify custom reports, but this requires advanced Excel skills. The representation of the data can be manipulated by adjusting the detail level or by filtering the data.

An analysis of the work present in the queues gives insight in the existence of temporary bottlenecks in the process. This information is used as feedback for the execution phase. The feedback, however, is obtained from human interpretation of the analysis results and does not contain suggestions for the removal of the bottleneck. More permanent weaknesses in the process could also be revealed based on the analysis of performance data and this is done in the diagnosis phase.

Diagnosis In the diagnosis phase, problems and improvement possibilities are identified through analysis of the operational process. The *analysis engine* facilitates the control and the diagnosis phase, creating interoperability between the two phases. Analysis reports present an aggregated view on the performance data and weaknesses in the process are derived from this. The derivation, however, is not supported by the FileNet P8 BPM Suite and is based on human

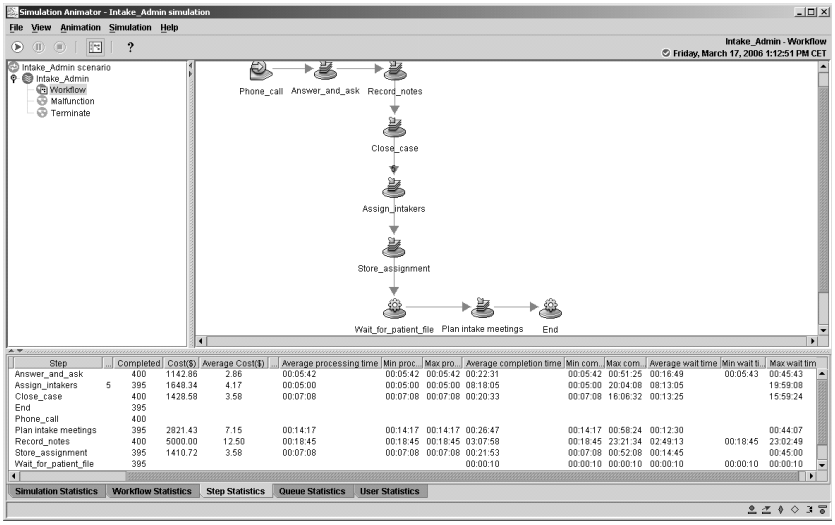

Fig. 4. Simulation results from the *process simulator*.

insights. A system not capable of identifying process weaknesses is certainly unable to provide improvement suggestions for these weaknesses. The FileNet P8 BPM Suite provides limited support for the diagnosis phase and the creation of ideas for process improvement should be done manually.

The ideas for redesign generated in the diagnosis phase could result in another pass through the BPM cycle starting with a new design phase. When we started our pass in the design phase it became clear that historic performance data is necessary to obtain the performance of the created redesigns with simulation. We already mentioned that only historic arrival data could be used, making the interoperability between the diagnosis and the design phase limited. We did not mention yet that data generated with simulation can also be transferred to the *analysis engine* and presented in the performance reports. This provides a comprehensive view on the simulation results. Nevertheless, presenting the correct data becomes problematic when multiple scenarios of the same simulation model have been simulated over the same simulation time. It is not possible to select the data of only one of the scenarios, while the aggregation of all simulation data leads to unusable results. The only solution for this is clearing the *analysis engine* before each new simulation run, which does not only lead to unworkable situations, but will also remove the historic execution data from the *analysis engine*.

3.3 Analysis of the Evaluation Results

The conclusions from the evaluation just described are summarized in Table 2. In Table 2 we present the support required for each phase in the BPM life-cycle and the support provided by the FileNet P8 BPM Suite. From our evaluation

we conclude that FileNet provides strong support for the configuration, the execution and the control phase. In particular,

- The configuration phase is well supported by the *process designer*.
- The execution of the workflow is strongly supported by the *process engine*.
- The control phase is supported by the *process administrator* and the *analysis engine*.

Less explicit support is available for the diagnosis and (re)design phase. Some support in the diagnosis phase is provided by the *process analyzer*, which gives an aggregate view on the data. However, the search for weaknesses in the process is not supported and certainly no improvement suggestions are generated. Moreover, the information provided by the *process analyzer* is limited to simple performance indicators such as flow time and utilization. There is no analysis aiming at identifying structures or patterns in the processes and the organization (e.g., social networks or deviations from the normal execution paths). Furthermore, in the design phase the creation of the alternatives is not supported. Limited support is available through the representation of the alternatives as facilitated by the *process designer* and the selection of the best alternative by the *process simulator*.

The conclusion for our interoperability evaluation is that the interoperability of the FileNet process tools is notably supported in the transitions between the design, the configuration, the execution, the control and the diagnosis phase. At the same time, the interoperability between the diagnosis and the design phase is limited to the use of historic arrival data (present in the *analysis engine*) for the simulation. All other performance data present in the *analysis engine* can not be passed to the *process simulator* and should be copied manually. Although interoperability exists between the execution and control phase, the loop back from control to execution is not supported. In the control phase temporary bottlenecks can be identified, but human intervention is required to interpret the findings and tune the operational process.

These insights are in line with the support that could be expected from a WFM system, as these systems are well-known for their emphasis on the configuration, execution and control phase. Nonetheless, it is also clear that opportunities exist to improve the support that so-called BPM systems offer to execute the entire BPM life-cycle. We consider the FileNet P8 BPM suite as a relevant benchmark for many of the other available systems, because of its broad range of features and market dominance. Yet the system is far from supporting the whole BPM life-cycle. Therefore, in the remainder, *we focus on the main problems identified, i.e., limited support for the diagnosis and (re)design phase and the problematic integration of components in this part of the BPM life-cycle.* In the next section we will argue that an approach based on state-of-the-art techniques for *process mining* and *intelligent redesign* is needed to address these problems.

Table 2. Summary of the evaluation.

Phase	Required support	FileNet support
Design	Make redesign	-
	Model designs	Process designer
	Evaluate designs	Process simulator
	Compare designs	-
	Input from diagnosis phase available	- (only arrival data)
	Output for configuration phase available	Through process designer
Configuration	Model detailed designs	Process designer
	Input from design phase available	Through process designer
	Output for execution phase available	Transfer of process definition
Execution	Workflow engine	Process engine
	Capture context data	Process engine
	Input from configuration phase available	Transfer to process engine
	Output for control phase available	Transfer from process engine
Control	Monitor specific cases	Process administrator
	Aggregation of execution data	Analysis engine
	Monitor performance	Process analyzer
	Input from execution phase available	Transfer to analysis engine
	Output for diagnosis phase available	Through analysis engine
	Output for execution phase available	-
Diagnosis	Reveal weaknesses	Process analyzer
	Identify improvement points	-
	Input from control phase available	Through analysis engine
	Output for design phase available	- (only arrival data)

- : not supported by FileNet, should be done manually.

4 An Approach Based on Process Mining and Intelligent Redesign

Based on the evaluation and analysis presented in the previous section, we now zoom in onto the two problems mentioned in the introduction. First, we describe the two problems in more detail. Based on this we present our ideas with respect to using process mining and intelligent redesign. Finally, we briefly discuss the ProM framework as a platform for addressing the two problems.

4.1 Linking Design and Reality

To clarify the two problems mentioned in the introduction and to link our proposal to the evaluation presented in the previous section, we use Figure 5.

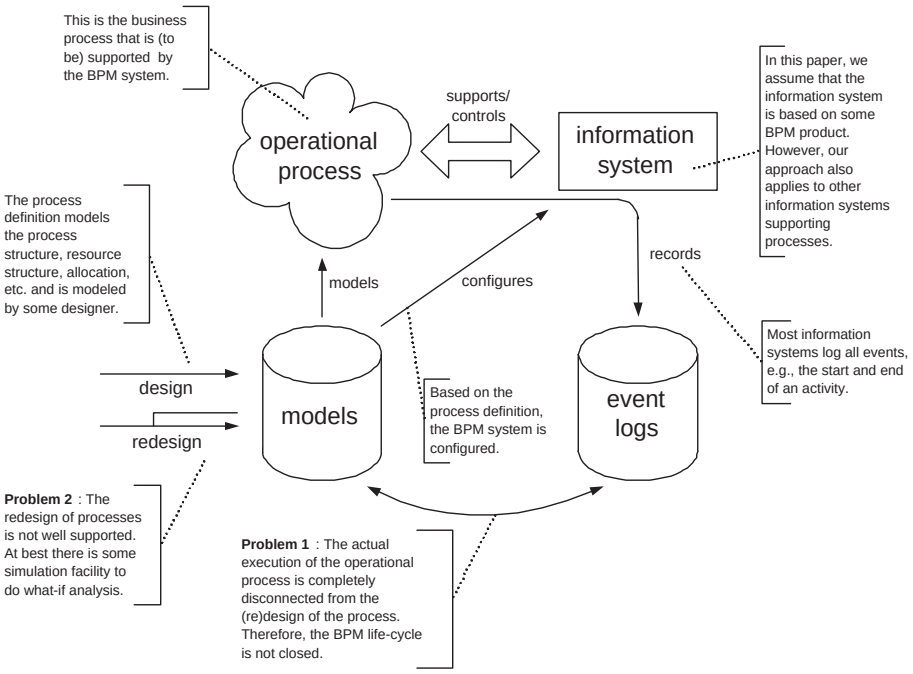

Fig. 5. Two problems not addressed by contemporary BPM systems: (1) event logs are not used to feed the (re)design of processes and (2) the redesign is at best supported by "what-if" analysis (e.g., simulation).

Figure 5 shows the operational process (e.g., the flow of patients in a hospital, the handling of insurance claims, the procurement process of a multinational, etc.) that is interacting with some information system (e.g., an ERP, CRM, PDM, BPM, or WFM system). Clearly the information system and the

operational process exchange information, e.g., the system may support and/or control the processes at hand. Related to the information system and processes it supports are models and event logs. Many systems log events related to the processes they support (cf. the arrow labeled *records* in Figure 5). The role of models is more involved. Clearly, process models can be used to model the operational process for a variety of reasons. Process models can be used to analyze and optimize processes but can also be used for guidelines, training, discussions, etc. (cf. the arrow labeled *models* in Figure 5). However, increasingly information systems are configured on the basis of models (cf. the arrow labeled *configures* in Figure 5). For example, consider process-aware systems (Dumas et al., 2005) ranging from workflow and BPM systems such as FileNet, Staffware and COSA to ERP systems like SAP R/3 and PeopleSoft. Models can be prescriptive or descriptive. If they are used for configuration, they tend to be prescriptive. If they are used for other purposes, they are often descriptive. In the first part of the paper, we were focusing on prescriptive models (i.e., the process definition used by FileNet). However, it is important to note that also descriptive models play an important role when it comes to process support and improvement.

Using Figure 5 we can further detail the two problems mentioned in Section 1.2:

- *Problem 1:* As Figure 5 shows, the actual execution of the process supported by the BPM system is completely disconnected from the (re)design of the process. In BPM systems such as FileNet the event logs are not related to the initial process design. Moreover, there is no support to extract knowledge from these logs to aid the redesign.
- *Problem 2:* Redesign activities are typically only supported by a graphical editor which allows the designer to modify the existing process. This implies that the designer has to come up with ideas for process improvement. These are not suggested by the system. Moreover, at best there is a simple simulation facility that is completely disconnected from the real operational process, i.e., no information extracted from the real process is automatically used in the analysis of different redesign alternatives.

We would like to argue that the only way to address these problems is by linking *design* and *reality*, i.e., as long as information about the real process (event logs) is not used by the design tool it is not possible to close the BPM life-cycle. Therefore, we propose to use recent results achieved in the domain of *process mining* (Aalst & Weijters, 2004; Aalst, Dongen, et al., 2003).

4.2 Process mining

Figure 6 shows that there are three classes of process mining techniques. This classification is based on whether there is an a-priori model and, if so, how it is used.

- *Discovery*: There is no a-priori model, i.e., based on an event log some model is constructed. For example, using the alpha algorithm (Aalst et al., 2004)

a process model can be discovered based on low-level events. There exist many techniques to automatically construct process models (e.g., in terms of a Petri net) based some event log (Aalst et al., 2004; Agrawal et al., 1998; Cook & Wolf, 1998; Datta, 1998; Weijters & Aalst, 2003). Recently, process mining research also started to target the other perspectives (e.g., data, resources, time, etc.). For example, the technique described in (Aalst, Reijers, & Song, 2005) can be used to construct a social network.

- *Conformance*: There is an a-priori model. This model is compared with the event log and discrepancies between the log and the model are analyzed. For example, there may be a process model indicating that purchase orders of more than 1 million euro require two checks. Another example is the checking of the so-called "four-eyes" principle. Conformance checking may be used to detect deviations, to locate and explain these deviations, and to measure the severity of these deviations. An example is the conformance checker described in (Rozinat & Aalst, 2006a) which compares the event log with some a-priori process model expressed in terms of a Petri net.
- *Extension*: There is an a-priori model. This model is extended with a new aspect or perspective, i.e., the goal is not to check conformance but to enrich the model. An example is the extension of a process model with performance data, i.e., some a-priori process model is used to project the bottlenecks on. Another example is the decision miner described in (Rozinat & Aalst, 2006b) which takes an a-priori process model and analyzes every choice in the process model. For each choice the event log is consulted to see which information is typically available the moment the choice is made. Then classical data mining techniques are used to see which data elements influence the choice. As a result, a decision tree is generated for each choice in the process.

Note that the FileNet P8 BPM Suite does not support any form of process mining. This illustrates the missing link between the various models used in the context of a BPM system (e.g., process definitions, organizational models, simulation models, etc.) and the real processes as observed by the information system through its event logs. Clearly all three classes of process mining techniques are valuable in a setting where BPM systems are used. Discovery techniques look at the process under a specific angle and reveal what is really going on. For example, by creating a social network one can see how people work together. The construction of a process model (e.g., a Petri net) may be helpful if the process is not enforced by the information system or if one is collecting information about a process that is not yet supported by the BPM system (e.g., to generate an initial design). Conformance checking is useful to detect deviations. This information can be used to redesign the process where it does not fit or to improve the control of the system to make sure that reality follows the desired process. Finally, model extension based on event logs can be very valuable because it combines a-priori knowledge with real observations. This can be used to "upgrade" process models with information relevant for simulation purposes (e.g.,

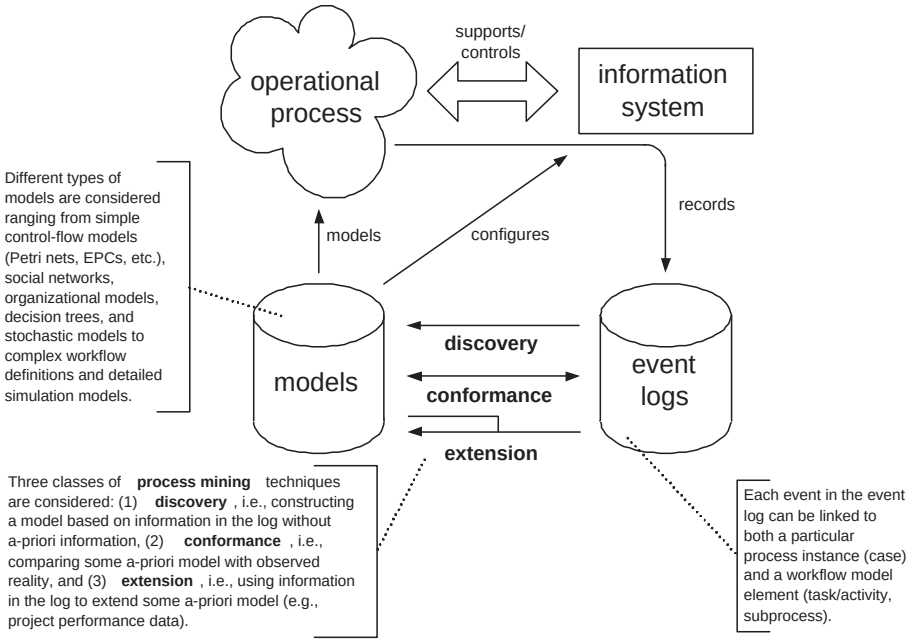

Fig. 6. Process mining as a means to link models and event logs.

routing probabilities, service times, inter-arrival times, etc.) or decision support (e.g., decision analysis).

4.3 Intelligent redesign

Although process mining is useful by itself, we believe that it is particularly powerful if it is combined with *intelligent redesign*. To illustrate this, consider Figure 7 where intelligent redesign is linked to process mining. The goal of intelligent redesign is to come up with suggestions for redesigns based on the evaluation of all kinds of redesign possibilities. The starting point for intelligent redesign is a set of *redesign rules* (in the sense of (Reijers, 2003)). Each rule specifies:

– A *precondition* describing under which circumstances the rule is applicable. For example, a redesign rule intended to make a sequential process more parallel has as a precondition that there should be a sequence of tasks where the tasks are potentially executed by different people (otherwise it makes no sense to put things in parallel). One can think of this precondition as a kind of pattern matching: potentially suboptimal structures in the process definition (in the broadest sense, i.e., also interaction with other parties and organizational structures are considered) are identified using a repository of redesign rules. Note that the patterns do not have to be of a strictly

structural nature, i.e., they may also refer to performance indicators as work-in-progress, flow time, utilization, service levels, etc.
- A *transformation* describing how the design should be changed. For example, the redesign rule intended to make a sequential process more parallel should specify how to transform a sequential process fragment into a parallel one. The transformation is typically a replacement of the pattern described in precondition.
- A set of *intended effects*, i.e., the redesign rule aims at improving some aspect of the process. The different aspects are captured in so-called performance indicators. The goal of the rule may be to improve the flow time, to reduce resource utilization, to improve quality, etc. Note that redesign rules often provide a tradeoff, e.g., parallel processing may lead to a reduction in flow time but at the same time increase overhead.

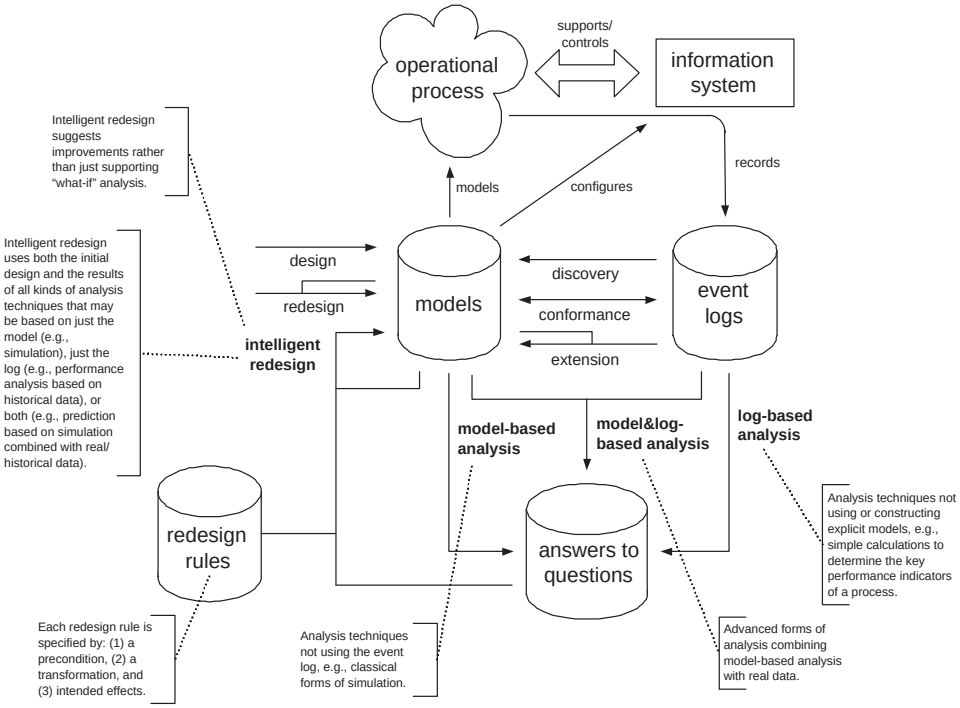

Fig. 7. Intelligent redesign linked to process mining.

As an example consider the *knock-out rule* defined in (Aalst, 2001). The knock-out rule applies to so-called "knock-out processes". The goal of a knock-out process is to decide whether the case should be accepted or rejected. To make this decision, a sequence of tasks needs to be executed. Each task has two possible results: OK or NOK (i.e., not OK). If for a specific case (i.e., process instance)

a task results in NOK, the case is rejected immediately. Only if all tasks have a positive result, the case is accepted. Many business processes have parts that can be viewed as knock-out processes. Handling an insurance claim, a request for a loan, a job application, and the reviewing of paper for publication in a journal are typical examples of processes with a knock-out structure. Many knock-out processes are characterized by the fact that the degree of freedom with respect to the order in which tasks can be executed is quite high. Most tasks correspond to checks which can be executed in any order. A task is *selective* if the reject probability is high. A task is *expensive* if the average processing time is high. Clearly, it is wise to start with selective tasks that are not expensive and postpone the expensive tasks which are not selective as long as possible. This rule of thumb is captured in the following redesign rule: *Tasks sharing the same resource class should be ordered in descending order using the ratio $rp(t)/pt(t)$ to obtain an optimal process with respect to resource utilization and maximal throughput.*[3] It is fairly straightforward to define the knock-out rule in terms of a precondition, a transformation, and its intended effects. The precondition is the presence of a sequential process fragment consisting of different tests executed by potentially the different sets of people (preferably different roles, departments, etc.). The transformation is a reordering of activities based on the ratio $rp(t)/pt(t)$. The intended effects are a reduction of flow time and a lower resource utilization.

Driven by a set of redesign rules, the following procedure is followed to support intelligent redesign:

- *Step 1: Determine applicable redesign rules.* In this first step the preconditions of all redesign rules are checked in the context of the concrete process at hand. Note that the preconditions may also refer to the process as it is being executed, i.e., the pattern matching is not limited to the process design and extends to e.g. the detection of bottle-necks, etc.
- *Step 2: Select applicable redesign rules based on intended effects.* Based on the opportunities identified in the first step a first selection is made. This selection is based on the expected effects of the application of the rule and what the perceived problem is. Note that the same redesign rule may be applicable at multiple places in the process. For each possible application of the rule, a selection decision needs to be made. The selection of applicable redesign rules is an interactive process, i.e., the designer can guide the selection process.
- *Step 3: Create redesigns.* After selecting the applicable redesign rules, different redesigns are created. Typically, each selected redesign rule results in a redesign, i.e., the rules are applied incrementally (cf. Step 5). To do this the transformation part of the redesign rule is used.
- *Step 4: Evaluate redesigns.* After applying the redesign rules, each of the resulting redesigns is evaluated using simulation. Using simulation the effects of applying the redesign rule in this particular context are predicted. Note that the performance indicators used to describe the intended effects of a

[3] In this ratio, t refers to a task, $rp(t)$ the rejection probability (i.e., the percentage of cases for which t results in a NOK), and $pt(t)$ the average processing time of task t.

redesign are of particular interest for this step, because now the predicted effect can be measured.
- *Step 5: Select the redesigned processes.* For each redesign evaluated through simulation, two decisions need to be made: (1) Will the redesign be used for further investigation? and/or (2) Will the redesign be presented to the designer? In this step the most promising redesigns are selected. For each redesign selected for further investigation steps 1-5 are repeated. Moreover, all redesigns selected for presentation are collected as input for Step 6.
- *Step 6: Present the selected redesigns and their expected effects.* In the final step the results are presented to the designer. It is the designer that selects the redesign to be used (if any). Moreover, the designer may refine a selected redesign or restart the procedure with different input.

It is important to note that both in Step 1 and in Step 4 the event logs play an important role, i.e., the pattern matching in Step 1 may be partially based on performance indicators derived from the real process (e.g., a redesign rule only applies if there is some sort of problem) and the simulation in Step 4 should be based on data derived from the actual process (e.g., the arrival patterns of new cases). This is illustrated by the lower half of Figure 7. As shown, intelligent redesign is driven by the existing process design, the redesign rules, and answers to various questions. These questions relate to just the models (e.g., the process definition or the organizational model), just the event logs (e.g., flow times, frequencies, etc.), or both (e.g., the process definition embedded in a real-life context). Figure 7 refers to answering these questions as *model-based analysis*, *log-based analysis*, and *model&log-based analysis*. An example of model-based analysis is the classical form of simulation, i.e., based on some model completely disconnected from the run-time environment performance indicators are estimated. Log-based analysis typically uses simple queries or calculations one some database with run-time/historic information. This is the area commonly referred to as data-warehousing, business activity monitoring, or business intelligence. Model&log-based analysis tries to combine some a-priori model with relevant run-time/historic information, e.g., a simulation model using run-time/historic information as described in (Reijers & Aalst, 1999). It should be noted that model&log-based analysis is closely related to model extension (the third form of process mining). Note that the same information, e.g., inter-arrival times of cases, may be used to extend the model (fit some stochastic distribution and add this to the simulation model) or to directly answer a question without extending the model (directly feed a log with historic inter-arrival times to the simulation engine as supported by the FileNet simulator).

Existing BPM systems provide limited support for the lower half of Figure 7. At best they offer basic model-based analysis (e.g., simple simulation engine) and log-based analysis (e.g., basic information on the key performance indicators). There is typically no support for process mining and model&log-based analysis. As shown in Section 3, leading BPM systems such as the FileNet P8 BPM Suite only provide a basic *process designer* and a *process simulator*. Historic information on arriving cases can be transferred to the *process simulator*, but

all other data, like processing times and routing probabilities, should be derived from the execution data and included manually. It is only possible to provide constant values for the simulation parameters, so the simulation results will only provide a very rough indication for the performance of a scenario. Moreover, there is no form of intelligent redesign, i.e., it is impossible to use redesign rules in a systematic way. The user needs to come up with redesigns and and is not supported in any way.

4.4 The ProM Framework

The primary goal of this paper is not to provide detailed solutions but to sketch how process mining and intelligent redesign could be used to close the BPM life-cycle and to enhance systems like the FileNet P8 BPM Suite. We have been developing process mining techniques in the context of the ProM framework for several years now (Dongen et al., 2005) and recently started adding functionality to support intelligent redesign to ProM. Therefore, we briefly describe the ProM framework and some of its plug-ins.

Starting point for ProM are event logs in MXML format. The MXML format is system-independent and using ProMimport it is possible to extract logs from a wide variety of systems, i.e., systems based on products such as SAP, Peoplesoft, Staffware, FLOWer, WebSphere, YAWL, ADEPT, ARIS PPM, Caramba, InConcert, Oracle BPEL, Outlook, etc. and tailor-made systems. It is also possible to load and/or save a variety of models, e.g., EPCs (i.e., event-driven process chains in different formats, e.g., ARIS, ARIS PPM, EPML, and Visio), BPEL (e.g., Oracle BPEL, Websphere), YAWL, Petri nets (using different formats, e.g., PNML, TPN, etc.), CPNs (i.e., colored Petri nets as supported by CPN Tools), and Protos. Currently, there are no interfaces with the FileNet P8 BPM Suite but it would be fairly easy to make this connection (both at the model level and the event log level) since it is similar to many of the other systems already supported.

The ProM framework is open-source and plug-able, i.e., people can plug-in new pieces of functionality. Some of the plug-ins are related to model transformations and various forms of model analysis (e.g., verification of soundness, analysis of deadlocks, invariants, reductions, etc.). Most of the plug-ins, however, focus on a particular process mining technique. Currently, there are more than 100 plug-ins of which about half are mining and analysis plug-ins. In this paper, we distinguished three types of process mining techniques: discovery, conformance, and extension. Below we briefly mention some of the plug-ins present in ProM to support these three types of process mining.

ProM supports many discovery algorithms. ProM currently offers eight plug-ins for discovering processes. These plug-ins use different "target formats", i.e., different languages to represent the result of the discovery algorithm (e.g., Petri nets, EPCs, heuristics nets). Figure 8 shows four process models obtained using various plug-ins for control-flow discovery present in ProM. The models are based on a event log with about 20000 events relating to about 1000 patients.

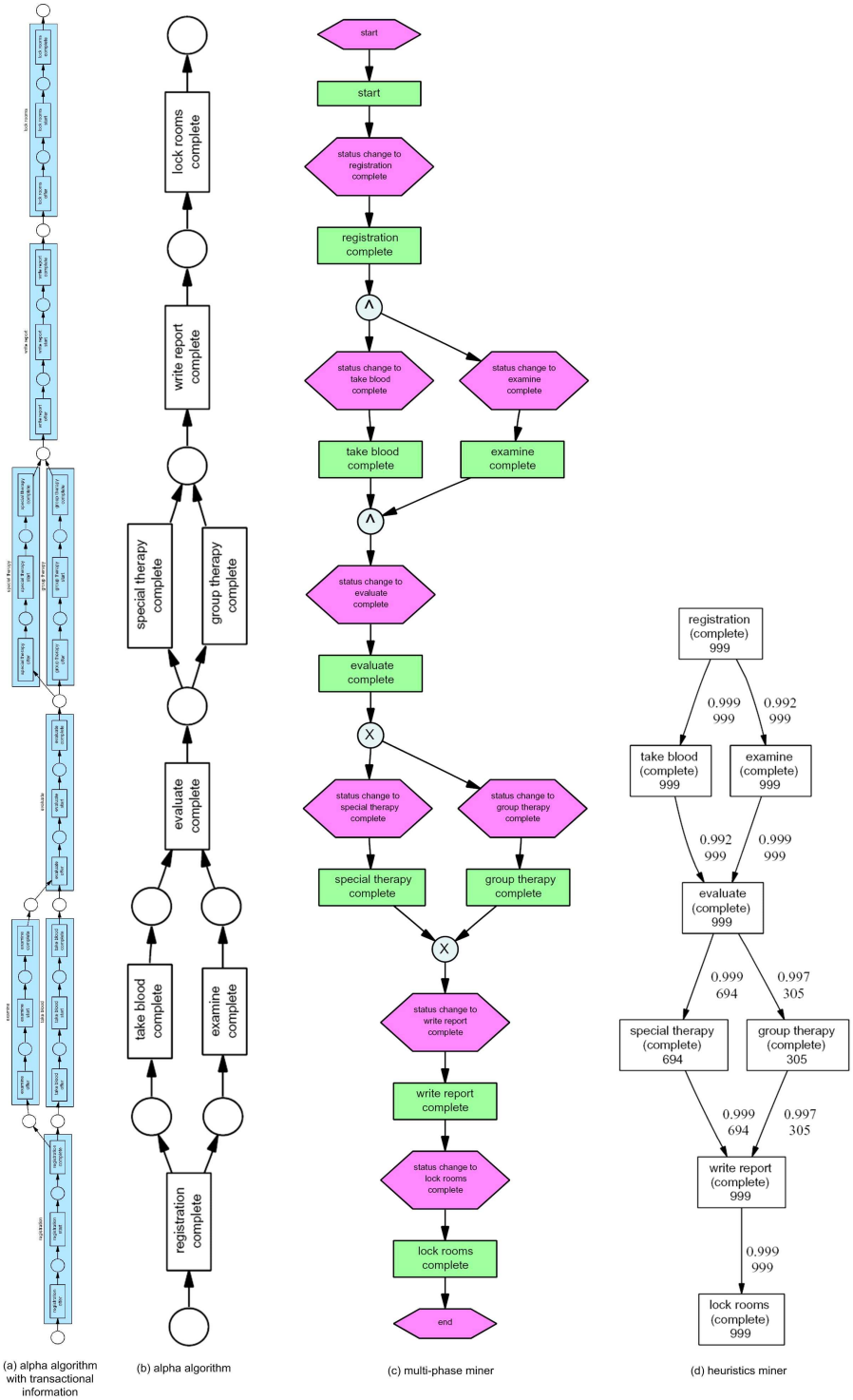

Fig. 8. Based on an analysis of an event log with data on 1000 patients, various mining plug-ins are able to discover the underlying process.

It is important to note that the process models have been generated automatically without any a-priori information. Figure 8(a) shows the result obtained by applying the α algorithm (Aalst et al., 2004) to the log with the full transactional information, i.e., each activity characterized by three events in the log: *offer*, *start*, and *complete*. Hence, in the resulting Petri-net each activity is represented by three transitions. Figure 8(b) is also generated by the α algorithm but now by only considering the *complete* events. Figure 8(c) shows the result of applying the multi-phase miner. This approach first builds a model for every instance and then starts aggregating these instance models. The native format of the multi-phase miner is the EPC language. However, the multi-phase miner can also display its results as Petri nets. In fact, in ProM any Petri net can be converted into an EPC, YAWL model, or heuristics net and vice versa. Figure 8(d) shows a heuristics net discovered by the heuristics miner (Weijters & Aalst, 2003). The format is also used by the genetic miner and both are able to cope with noise. Discovery in ProM is not limited to the control-flow perspective. It is also possible to discover social networks (Aalst, Reijers, & Song, 2005) or staff assignment rules. However, a complete overview of all discovery techniques supported by ProM is outside the scope of this paper.

ProM also has two plug-ins for conformance: the conformance checker (Rozinat & Aalst, 2006a) and the LTL checker (Aalst, Beer, & Dongen, 2005). Both are able to discover deviations between some model and the real behavior captured in the event log.

The decision miner described in (Rozinat & Aalst, 2006b) is an example of a plug-in that takes a process model without data and extends it with data and information about decisions. More precisely, it takes an a-priori process model and analyzes every choice in the process model. For each choice the event log is consulted to see which information is typically available the moment the choice is made. Then classical data mining techniques are used to see which data elements influence the choice. As a result, a decision tree is generated for each choice in the process.

In the context of intelligent redesign it is relevant to note that it is possible to export process models in ProM to CPN Tools (CPN Group, University of Aarhus, Denmark, n.d.). The control-flow, data-flow, performance and organizational perspectives can be incorporated in a single Colored Petri Net (CPN) model (Jensen, 1996) that can be exported to CPN Tools. CPN Tools allows for various types of analysis including state-space analysis and simulation. Using the monitoring facilities of CPN Tools it is easy to detect bottlenecks and collect all kinds of performance indicators. We believe that the combination of automatic discovery of process models using ProM and the simulation capabilities of CPN Tools offers an innovative way to improve business processes. The initially discovered process model describes reality better than most hand-crafted simulation models. Moreover, if there is an a-priori model it can be verified whether reality fits the model (conformance checker) and the model can also be extended to incorporate historic information (e.g., arrival processes and service times).

The simulation models are constructed in such a way that it is easy to explore various redesigns.

ProM is available as open source (under the Common Public License, CPL) and can be downloaded from www.processmining.org. It has been applied to various real-life processes, ranging from administrative processes and healthcare processes to the logs of complex machines and service processes. While the process mining capabilities of ProM are very mature, the support of intelligent redesign is still in its infancy. Currently, we are adding redesign rules to ProM to support intelligent redesign.

5 Conclusion

In this paper we focused to supporting the whole BPM life-cycle consisting of the following phases: *(re)design*, *configuration*, *execution*, *control*, and *diagnosis*. The contribution of this paper is twofold.

First of all, we analyzed the limitations of today's BPM systems using the FileNet P8 BPM Suite as a representative example. We presented a generic approach to analyze the capabilities of a BPM product with respect to its support of the BPM life-cycle. We discovered several problems. It was observed that systems such as the FileNet P8 BPM Suite provide limited support for the diagnosis and (re)design phase and that the integration of the different components in this part of the BPM life-cycle is problematic (i.e., results obtained in one component are not used in other components and users need to rekey and reinterpret results).

Second, based on these observations, we zoomed in onto the problematic part of the BPM life-cycle and discussed techniques that can alleviate the problems. We identified two key problems: (1) the complete disconnect between the (re)designs and reality as observed by the system through its event logs and (2) the lack of support when it comes to automatically generating redesigns based on an analysis of the current situation (i.e., current design and event logs) and evaluating these results. To address these problems we argued that techniques for process mining and intelligent redesign are needed to support the (re)design and diagnosis phases and thus close the BPM life-cycle.

We briefly presented ProM framework and some of its plug-ins. Currently, ProM already provides mature support for a wide variety of process mining techniques (aiming at discovery, conformance, and extension). Future work aims at strengthening the support of intelligent redesign by adding redesign plug-ins and a repository of redesign rules.

Acknowledgements

We would like to thank the consultancy and IT support staff from FileNet for their kind assistance in carrying out this study. We also thank the people involved in the development of the ProM framework: Ton Weijters, Boudewijn van Dongen, Ana Karla Alves de Medeiros, Anne Rozinat, Christian Günter,

Minseok Song, Laura Maruster, Eric Verbeek, Monique Jansen-Vullers, Huub de Beer, Ronny Mans, Peter van den Brand, Andriy Nikolov, Irene Vanderfeesten, et al. This research is supported by EIT and the Technology Foundation STW, applied science division of NWO and the technology program of the Dutch Ministry of Economic Affairs.

References

Aalst, W. van der (2001). Reengineering Knock-out Processes. *Decision Support Systems*, *30*(4), 451–468.

Aalst, W. van der, Beer, H., & Dongen, B. van (2005). Process Mining and Verification of Properties: An Approach based on Temporal Logic. In R. Meersman & Z. T. et al. (Eds.), *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2005* (Vol. 3760, pp. 130–147). Springer-Verlag, Berlin.

Aalst, W. van der, Dongen, B. van, Herbst, J., Maruster, L., Schimm, G., & Weijters, A. (2003). Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering*, *47*(2), 237–267.

Aalst, W. van der, & Hee, K. van (2004a). *Workflow Management: Models, Methods, and Systems*. MIT press, Cambridge, MA.

Aalst, W. van der, Hofstede, A. ter, Kiepuszewski, B., & Barros, A. (2003). Workflow Patterns. *Distributed and Parallel Databases*, *14*(1), 5–51.

Aalst, W. van der, Hofstede, A. ter, & Weske, M. (2003). Business Process Management: A Survey. In W. Aalst, A. Hofstede, & M. Weske (Eds.), *International Conference on Business Process Management (BPM 2003)* (Vol. 2678, pp. 1–12). Springer-Verlag, Berlin.

Aalst, W. van der, Reijers, H., & Song, M. (2005). Discovering Social Networks from Event Logs. *Computer Supported Cooperative work*, *14*(6), 549–593.

Aalst, W. van der, & Weijters, A. (Eds.). (2004). *Process Mining.* Elsevier Science Publishers, Amsterdam.

Aalst, W. van der, Weijters, A., & Maruster, L. (2004). Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, *16*(9), 1128–1142.

Agrawal, R., Gunopulos, D., & Leymann, F. (1998). Mining Process Models from Workflow Logs. In *Sixth international conference on extending database technology* (pp. 469–483).

Bernstein, A., Klein, M., & Malone, T. (1999). The Process Recombinator: A Tool for Generating New Business Process Ideas. In *Icis* (p. 178-192).

Buzacott, J. (1996). Commonalities in Reengineerd Business Processes: Models and Issues. *Management Science*, *42*(5), 768–782.

Cook, J., & Wolf, A. (1998). Discovering Models of Software Processes from Event-Based Data. *ACM Transactions on Software Engineering and Methodology*, *7*(3), 215–249.

CPN Group, University of Aarhus, Denmark. (n.d.). *CPN Tools Home Page.* (http://wiki.daimi.au.dk/cpntools/)

Datta, A. (1998). Automating the Discovery of As-Is Business Process Models: Probabilistic and Algorithmic Approaches. *Information Systems Research*, *9*(3), 275–301.

Dongen, B. van, Medeiros, A., Verbeek, H., Weijters, A., & Aalst, W. van der (2005). The ProM framework: A New Era in Process Mining Tool Support. In G. Ciardo & P. Darondeau (Eds.), *Application and Theory of Petri Nets 2005* (Vol. 3536, pp. 444–454). Springer-Verlag, Berlin.

Dumas, M., Aalst, W. van der, & Hofstede, A. ter (2005). *Process-Aware Information Systems: Bridging People and Software through Process Technology.* Wiley & Sons.

Eder, J., Panagos, E., Pezewaunig, H., & Rabinovich, M. (1999). Time Management in Workflow Systems. In W. Abramowicz & M. Orlowska (Eds.), *Third International Conference on Business Information Systems (BIS'99)* (pp. 265–280). Poznan, Polen: Springer-Verlag, Berlin.

Eder, J., Panagos, E., & Rabinovich, M. (1999). Time Constraints in Workflow Systems. In M. Jarke & A. Oberweis (Eds.), *Proceedings of the 11th International Conference on Advanced Information Systems Engineering (CAiSE '99)* (Vol. 1626, pp. 286–300). Springer-Verlag, Berlin.

Ellis, C. (1979). Information Control Nets: A Mathematical Model of Office Information Flow. In *Proceedings of the Conference on Simulation, Measurement and Modeling of Computer Systems* (pp. 225–240). Boulder, Colorado: ACM Press.

Gartner. (2003). *Gartner's Magic Quadrant for Pure-Play BPM.* (http://www.gartner.com)

Gartner. (2004). *Gartner's Magic Quadrant for Pure-Play BPM.* (http://www.gartner.com)

Gartner. (2005). *Gartner's Magic Quadrant for Pure-Play BPM.* (http://www.gartner.com)

Georgakopoulos, D., Hornick, M., & Sheth, A. (1995). An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, *3*, 119–153.

Grigori, D., Casati, F., Castellanos, M., Dayal, U., Sayal, M., & Shan, M. (2004). Business Process Intelligence. *Computers in Industry*, *53*(3), 321–343.

Grigori, D., Casati, F., Dayal, U., & Shan, M. (2001). Improving Business Process Quality through Exception Understanding, Prediction, and Prevention. In P. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, & R. Snodgrass (Eds.), *Proceedings of 27th international conference on very large data bases (VLDB'01)* (pp. 159–168). Morgan Kaufmann.

IDS Scheer. (2002). *ARIS Process Performance Manager (ARIS PPM): Measure, Analyze and Optimize Your Business Process Performance (whitepaper).* (IDS Scheer, Saarbruecken, Gemany, http://www.ids-scheer.com)

Jablonski, S., & Bussler, C. (1996). *Workflow Management: Modeling Concepts, Architecture, and Implementation.* International Thomson Com-

puter Press, London, UK.

Jansen-Vullers, M., & Reijers, H. (2005). Business Process Redesign at a Mental Healthcare Institute: A Coloured Petri Net Approach. In K. Jensen (Ed.), *Proceedings of the Sixth Workshop on the Practical Use of Coloured Petri Nets and CPN Tools (CPN 2005)* (Vol. 576, pp. 21–38). Aarhus, Denmark: University of Aarhus.

Jensen, K. (1996). *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use.* Springer-Verlag, Berlin.

Lawrence, P. (Ed.). (1997). *Workflow Handbook 1997, Workflow Management Coalition.* John Wiley and Sons, New York.

Leymann, F., & Roller, D. (1999). *Production Workflow: Concepts and Techniques.* Prentice-Hall PTR, Upper Saddle River, New Jersey, USA.

Marinescu, D. (2002). *Internet-Based Workflow Management: Towards a Semantic Web* (Vol. 40). Wiley-Interscience, New York.

Mühlen, M. (2004). *Workflow-based Process Controlling: Foundation, Design and Application of workflow-driven Process Information Systems.* Logos, Berlin.

Mühlen, M., & Rosemann, M. (2000). Workflow-based Process Monitoring and Controlling - Technical and Organizational Issues. In R. Sprague (Ed.), *Proceedings of the 33rd hawaii international conference on system science (HICSS-33)* (pp. 1–10). IEEE Computer Society Press, Los Alamitos, California.

Netjes, M., Aalst, W. van der, & Reijers, H. (2005, October). Analysis of Resource-Constrained Processes with Colored Petri Nets. In K. Jensen (Ed.), *Proceedings of the Sixth Workshop on the Practical Use of Coloured Petri Nets and CPN Tools (CPN 2005)* (Vol. 576, pp. 251–266). Aarhus, Denmark: University of Aarhus.

Netjes, M., Reijers, H., & Aalst, W. van der (2006a). *FileNet's BPM life-cycle support.* BPM Center Report BPM-06-07. BPMcenter.org.

Netjes, M., Reijers, H., & Aalst, W. van der (2006b). Supporting the BPM Lifecycle with FileNet. In T. Latour & M. Petit (Eds.), *Proceedings of the EMMSAD Workshop at the 18th International Conference on Advanced Information Systems Engineering (CAiSE'06)* (pp. 497–508). Namur University Press.

Netjes, M., Vanderfeesten, I., & Reijers, H. (2006). "Intelligent" Tools for Workflow Process Redesign: A Research Agenda. In C. Bussler & A. Haller (Eds.), *Business Process Management Workshops (BPM 2005)* (Vol. 3812, pp. 444–453). Springer-Verlag, Berlin.

Nissen, M. (1998). Redesigning Reengineering Through Measurement-Driven Inference. *MIS Quarterly*, *22*(4), 509–534.

Panagos, E., & Rabinovich, M. (1997). Escalations in Workflow Management Systems. In *Proceedings of the workshop on databases: Active and real time (dart-96)* (pp. 25–28). ACM Press.

Panagos, E., & Rabinovich, M. (1998). Reducing Escalation-Related Costs in WFMSs. In *Workflow management systems and interoperability* (pp. 107–

127). Springer-Verlag, Berlin.

Reijers, H. (2003). *Design and Control of Workflow Processes: Business Process Management for the Service Industry* (Vol. 2617). Springer-Verlag, Berlin.

Reijers, H., & Aalst, W. van der (1999). Short-Term Simulation: Bridging the Gap between Operational Control and Strategic Decision Making. In M. Hamza (Ed.), *Proceedings of the iasted international conference on modelling and simulation* (pp. 417–421). IASTED/Acta Press, Anaheim, USA.

Rozinat, A., & Aalst, W. van der (2006a). Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models. In C. Bussler et al. (Ed.), *BPM 2005 Workshops (Workshop on Business Process Intelligence)* (Vol. 3812, pp. 163–176). Springer-Verlag, Berlin.

Rozinat, A., & Aalst, W. van der (2006b). Decision Mining in ProM. In S. Dustdar, J. Faideiro, & A. Sheth (Eds.), *International Conference on Business Process Management (BPM 2006)* (Vol. 4102, pp. 420–425). Springer-Verlag, Berlin.

Sayal, M., Casati, F., Dayal, U., & Shan, M. (2002). Business Process Cockpit. In *Proceedings of 28th international conference on very large data bases (VLDB'02)* (pp. 880–883). Morgan Kaufmann.

Staffware. (2003). *Staffware Process Suite Version 2 – White Paper.* Maidenhead, UK.

TIBCO. (2005). *TIBCO Staffware Process Monitor (SPM).* (http://www.tibco.com)

Verbeek, H., Basten, T., & Aalst, W. van der (2001). Diagnosing Workflow Processes using Woflan. *The Computer Journal*, *44*(4), 246–279.

Weijters, A., & Aalst, W. van der (2003). Rediscovering Workflow Models from Event-Based Data using Little Thumb. *Integrated Computer-Aided Engineering*, *10*(2), 151–162.

Zhao, J., & Stohr, E. (1999). Temporal Workflow Management in a Claim Handling System. In *Proceedings of the international joint conference on Work activities coordination and collaboration (WACC'99)* (p. 187-195). ACM.