Discovering Simulation Models

A. Rozinat, R.S. Mans, M. Song, and W.M.P. van der Aalst

Eindhoven University of Technology
P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
{a.rozinat,r.s.mans,m.s.song,w.m.p.v.d.aalst}@tue.nl

Abstract. *Process mining* is a tool to extract non-trivial and useful information from process execution logs. These so-called *event logs* (also called audit trails, or transaction logs) are the starting point for various discovery and analysis techniques that help to gain insight into certain characteristics of the process. In this paper we use a combination of process mining techniques to discover multiple perspectives (namely, the control-flow, data, performance, and resource perspective) of the process from historic data, and we integrate them into a comprehensive simulation model. This simulation model is represented as a Coloured Petri net (CPN) and can be used to analyze the process, e.g., evaluate the performance of different alternative designs. The discovery of simulation models is explained using a running example. Moreover, the approach has been applied in two case studies; the workflows in two different municipalities in the Netherlands have been analyzed using a combination of process mining and simulation. Furthermore, the quality of the CPN models generated for the running example and the two case studies has been evaluated by comparing the original logs with the logs of the generated models.

1 Introduction

Computer simulation is a useful and versatile tool to gain insight into the operation of systems. Next to, e.g., natural systems also human systems can be subject to simulation. Generally, a model that represents certain key characteristics or behaviors of the system is analyzed to show the eventual real effects of alternative conditions and courses of action. The strength of simulation is that it enables precisely this “what if” analysis, i.e., it allows to “look into the future” under certain assumptions. In this paper, we focus on simulation models of operational processes. For example, in the context of a workflow management system simulation can help to estimate the benefit of an anticipated process redesign, or to predict flow times for an increasing number of incoming cases, a reduced number of specialists for a certain task etc.

Traditionally, such *simulation models* are created manually (cf. Figure 1). Documentation, interviews and close observation help to get an understanding of the *real-world process* of interest. This is a time-consuming activity, which is likely to be error-prone as it is based on human perception of reality rather than on reality itself. Therefore, we propose to use *process mining* techniques to (semi-)automatically discover a simulation model based on information that was recorded during process enactment. In this way, we can much quicker arrive at a first simulation model (to be further evaluated and potentially modified) than with the traditional approach. In addition, it is likely to better represent reality as it is based on objective information. Note that good knowledge of the observed operational process is inevitable for drawing conclusions from a simulation run and, therefore, such a generated model does not make domain and modeling expertise obsolete. However, no modeling efforts are needed to generate an initial model. Furthermore, this can be easily repeated in an iterative manner as soon as the process changes.

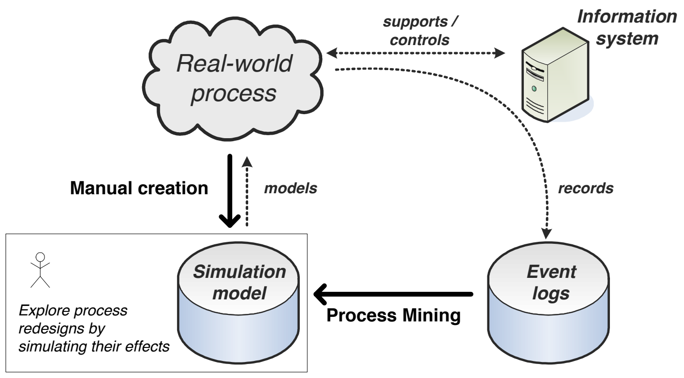


Fig. 1. Traditionally, simulation models are created manually. In this paper we aim at the (semi-)automatic discovery of simulation models using process mining techniques

Nowadays, many business processes are supported by *information systems* that help coordinating the steps that need to be performed in the course of the process. Workflow systems, for example, assign work items to employees according to their roles and the status of the process. Typically, these systems record events related to the activities that are performed, e.g., in audit trails or transaction logs [3]. These *event logs* are the starting point for process mining techniques, which, for example, construct a process model which reflects the causal relations that have been observed among the activities. Another example is decision mining [32], which analyzes which properties (i.e., valuations of data attributes) of a case might lead to taking certain paths in the process. In this paper we use a combination of process mining techniques to discover and integrate multiple perspectives of the process under consideration, namely control-flow, data, performance, and organizational perspective. In principle, further characteristics can be incorporated (cf. Figure 2). Colored Petri Nets (CPNs) [23] are then used as a representation for the integrated model because of their expressiveness and the strong simulation capabilities of CPN Tools [38].

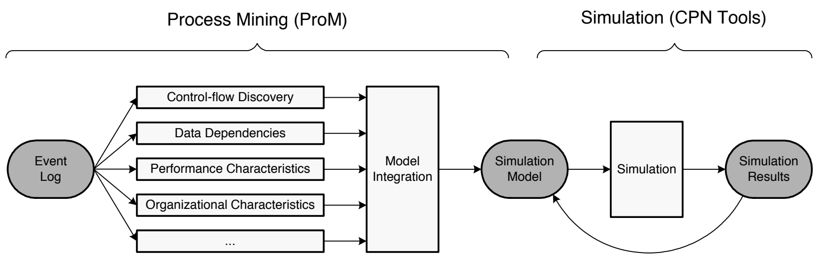


Fig. 2. A combination of process mining techniques is used to discover and integrate multiple perspectives of a process in a comprehensive simulation model

The applicability of the approach presented in this paper stands or falls with the availability of suitable logs. Fortunately, it can be observed that more and more events are being logged in a wide variety of systems. Obviously, information systems such as workflow management systems (e.g. Staffware), ERP systems (e.g. SAP), case handling systems (e.g. FLOWer), PDM systems (e.g. Windchill), CRM systems (e.g. Microsoft Dynamics CRM), middleware (e.g., IBM's WebSphere), hospital information systems (e.g., Chipsoft), etc. provide very detailed information about the activities that have been executed. Moreover, one can also find logs in situations where the information system is not directly noticeable. Examples are medical systems (e.g., X-ray machines), mobile phones, car entertainment systems, production systems (e.g., wafer steppers), copiers, sensor networks, etc. Software plays an increasingly important role in such systems and, already today, many of these systems record events. An example is the "CUSTOMerCARE Remote Services Network" of Philips Medical Systems (PMS). This is a worldwide internet-based private network that links PMS equipment to remote service centers. An event that occurs within an X-ray machine (e.g., moving the table, setting the deflector, etc.) is recorded and analyzed. Another example is the event logging infrastructure developed by ASML. ASML manufactures chip-making equipment such as wafer scanners. Its products are also connected to the internet. This connection is used to distribute the events logged. Already during the testing phase of a wafer scanner in the factory, thousands of events are recorded via an internet connection. This can be used for improving the service, improving the products, and for improving the test processes. The logging capabilities of the machines of PMS and ASML illustrate the omnipresence of event logs, making process mining a reality. Moreover, it is an important motivation for the work in this paper: Why construct simulation models by hand based on assumptions if there is so much real data around?

To directly support the generation of a simulation model from event logs we have implemented a *CPN Tools 2.0 Export* plug-in (in the remainder of this paper referred to as CPN Export plug-in) in the context of the ProM framework[1], which

[1] Both documentation and software (including the source code) of ProM [1] can be downloaded from *www.processmining.org*.

offers a wide range of tools related to process mining and process analysis. In [34] we described the chosen CPN representation for business processes in detail, and we presented the CPN Export plug-in in ProM. In this paper, we build on this work and show how a comprehensive simulation model including characteristics from the data, performance, and organizational perspective can be discovered from an event log. Furthermore, we evaluate how well the generated CPN model approximates these process characteristics by generating event logs during the simulation of the model in CPN Tools, and the repeated analysis of this "second pass", i.e., the behavior of the simulation model is compared with reality.

The paper is organized as follows. First, we review related work in Section 2. Then, Section 3 introduces the notion of an event log and presents a simple example process that is used throughout this paper. Afterwards, process mining techniques discovering different perspectives of a business process are described in Section 4. Then, we show how these perspectives can be integrated and represented as a CPN in Section 5. Section 6 presents a set of plug-ins in the ProM framework that can be used to discover and integrate all these perspectives. Section 7 describes two case studies based on real-life log data. Finally, we highlight some of the challenges for future research in Section 8, and the paper is concluded by Section 9.

2 Related Work

The work reported in this paper is related to earlier work on process mining, i.e., discovering a process model based on some event log. The idea of applying process mining in the context of workflow management was first introduced by Agrawal et. al in [6]. A similar idea was used by Datta in [13]. Cook and Wolf have investigated similar issues in the context of software engineering processes using different approaches [10, 9]. Note that the results presented in [10, 9] are limited to sequential behavior. Cook and Wolf extend their work to concurrent processes in [11]. They propose specific metrics (such as entropy, event type counts, and causality) and use these metrics to discover models out of event streams. However, they do not provide an approach to generate explicit process models. In [12] Cook and Wolf provide a measure to quantify discrepancies between a process model and the actual behavior as registered using event-based data. Herbst and Karagiannis also address the issue of process mining in the context of workflow management using an inductive approach [21]. They use stochastic task graphs as an intermediate representation and generate a workflow model described in the ADONIS modeling language. The α algorithm [5, 39] was one of the first algorithms to truly capture concurrency. In [5] it is shown that this algorithm can be proven to be correct for a large class of processes. Over time many variants and extensions of the α algorithm have been proposed. In [39] a heuristic approach using rather simple metrics is used to construct so-called "dependency/frequency tables" and "dependency/frequency graphs". This is used as input for the α algorithm. As a result it is possible to tackle the problem of noise. For more information on process mining we refer to a special issue of Computers in Industry on process mining [4] and a survey paper [3]. Examples of more recent work are [7], [27], and [35]. In [7] the authors show how to apply region based methods for the synthesis of Petri nets from languages to process mining. [27] and [35] specifically look at process discovery in the context of services and software processes, respectively.

Existing process mining approaches have been mainly focusing on the control-flow perspective. In this paper, we build on some initial work that has been done for discovering social networks in the context of process mining [2] and decision point analysis [32, 31]. The work reported in [32, 31] is closely related to [18], where the authors describe the architecture of the *Business Process Intelligence* (BPI) tool suite on top of the *HP Process Manager* (HPPM). Whereas they outline the use of data mining techniques for process behavior analysis in a broader scope, in [31] we show in detail how a decision point analysis can be carried out also in the presence of duplicate and invisible activities. Note that—while not an issue in the context of HPPM (as the node names of the corresponding workflow model are logged as well)—dealing with duplicate activities is often needed for real-life logs. Similar to [18], in [8] the authors propose methods to monitor and predict the behavior of a workflow instance in terms of pre-defined metrics. The work reported in [37] assumes the presence of a block-structured process model with given input and output data for each node and aims at discovering the earliest positions for decision points to avoid redundant activity executions and decrease the uncertainty. We also would like to mention [19] and [16], where the focus is on monitoring (also in the context of HPPM) and exception handling, respectively. In [25] decision trees are used to analyze staff assignment rules. Additional information about the organizational structure (i.e., a given organizational model is assumed) is incorporated to derive higher-level attributes (i.e., roles) from the actual execution data (i.e., performers). However, all these techniques do not provide an integrated view and do not aim at simulation. This paper is mostly related to [34] where the CPN export of ProM is described. However, the focus of [34] is not on process mining and no case studies are given. According to our knowledge, the integration of various mining results to automatically generate a complete simulation model including multiple perspectives is a novel approach and has not been done before. Furthermore, we focus on the validation aspect as the quality of a simulation model (whether generated or hand-made) is crucial for drawing conclusions from a simulation run. Finally, we highlight challenges that are faced when discovering simulation models from event logs, and creating simulation models for business processes in general.

There is quite some work on the automatic generation of Petri net models for workflows. A typical example is described in [17] where the authors present a translation of Protos simulation models to CPN Tools. In addition, three types of data collector monitors (measuring the total flow time per case, the waiting time per task, and the resource availability/utilization per resource type), and configuration features enabling the dynamic elimination of unnecessary parts of the process model are generated. Besides the work in [17], we are not aware of further attempts to export business process models to CPN Tools. The work reported in this paper has a different starting point as it is not limited by the simulation information present in a Protos model, but aims at discovering the process characteristics to be simulated from the event logs of real process executions.

3 Event Logs

Most information systems, e.g., WFM, ERP, CRM, SCM, and B2B systems, provide some kind of *event log* (also referred to as transaction log or audit trail) [3]. It contains log entries about activities which were executed for a business process that is supported by the information system. Every event refers to a case (i.e., process instance) and an activity, whereas activities in real business scenarios are often logged at a more fine-grained level than the atomic execution of that activity—they record, for example, the *scheduling*, the *start*, and the *completion* of an activity[2]. Furthermore, most systems also register a time stamp, a performer, and some additional data.

Figure 3 depicts a part of an event log in MXML[3] format, which is used as a running example in the remainder of this paper. The logged process reflects the medical examination flow in an outpatient clinic for gynecological oncology, whereas each process instance describes the examination history of one particular patient. Note that this artificial example is based on a real-life outpatient clinic process of the AMC hospital in the Netherlands. The example has been simplified for illustration purposes.

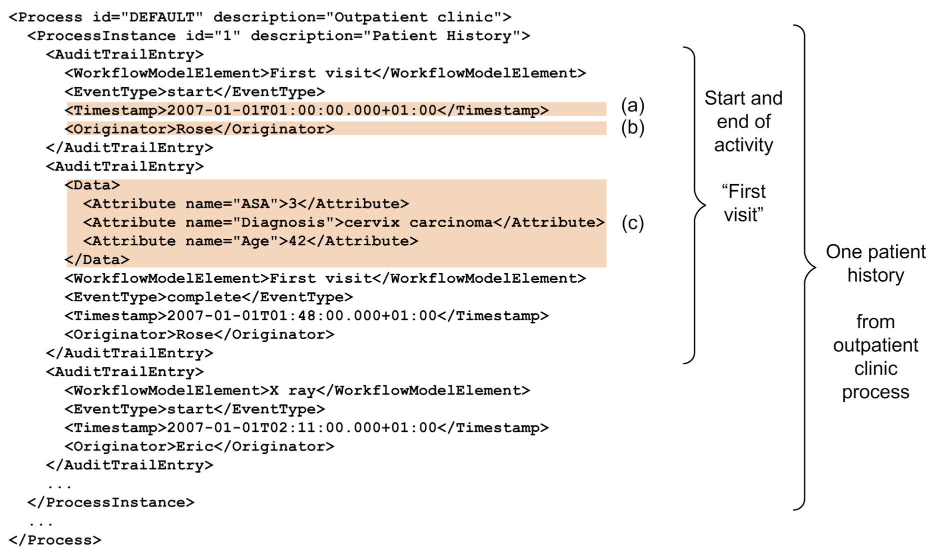
In Figure 3 we show the beginning of the first patient's history. Note that the event log starts with the start and completion of the "First visit" to the gynecologist, and the start of a subsequent "X ray" examination of this patient. One can observe that for each of the audit trail entries (i.e., events) the *time stamp* is recorded (a), and that the involved personnel from the outpatient clinic is registered in the *originator* field (b). Furthermore, the *complete* event of activity "First visit" captures the "ASA", which is a rating of anesthetic risk ranging from 1 (low) to 5 (high), the "Diagnosis", and the "Age" of the patient in additional *data attributes* (c).

4 Process Mining from Different Perspectives

Based on the example log introduced in the previous section, we will now apply a number of process mining algorithms to gain insight into different perspectives of the outpatient clinic process. The aim is to extract key characteristics that can be used for the creation of a simulation model.

[2] The life cycle of an activity has been standardized by the MXML format for workflow logs, which is used by the ProM framework.

[3] Both the corresponding schema definition and the ProM*import* framework [20], which converts logs from existing (commercial) process-aware information systems to the MXML format used by ProM, can be downloaded from *www.processmining.org*.



```
<Process id="DEFAULT" description="Outpatient clinic">
  <ProcessInstance id="1" description="Patient History">
    <AuditTrailEntry>
      <WorkflowModelElement>First visit</WorkflowModelElement>
      <EventType>start</EventType>
      <Timestamp>2007-01-01T01:00:00.000+01:00</Timestamp>
      <Originator>Rose</Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <Data>
        <Attribute name="ASA">3</Attribute>
        <Attribute name="Diagnosis">cervix carcinoma</Attribute>
        <Attribute name="Age">42</Attribute>
      </Data>
      <WorkflowModelElement>First visit</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>2007-01-01T01:48:00.000+01:00</Timestamp>
      <Originator>Rose</Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <WorkflowModelElement>X ray</WorkflowModelElement>
      <EventType>start</EventType>
      <Timestamp>2007-01-01T02:11:00.000+01:00</Timestamp>
      <Originator>Eric</Originator>
    </AuditTrailEntry>
    ...
  </ProcessInstance>
  ...
</Process>
```

(a)
(b)
(c)
Start and end of activity "First visit"
One patient history from outpatient clinic process

Fig. 3. Log fragment in MXML format. The running example reflects the examination flow in an outpatient clinic and contains in total 1000 cases (i.e., patient histories)

Figure 4 visualizes the dependencies between the used process mining techniques. First, a control-flow discovery algorithm is applied to automatically create a process model that reflects the causal relations between the examination activities in the log (Section 4.1). Second, a decision point analysis is performed based on the process model and the event log to discover decision rules for the choice points in the outpatient clinic process (Section 4.2). Third, a performance analysis is carried out to enhance the process model with information about execution times and waiting times for the activities, and probabilities for taking alternative paths (Section 4.3). Fourth, a role discovery algorithm is applied to the event log to group resources into roles, and to associate the discovered roles with the activities in the process (Section 4.4).

Finally, in Section 5, the mining results enhanced with data, performance, and organizational characteristics are integrated in one comprehensive simulation model. During the simulation of this model in CPN Tools we generate execution logs (similar to the original event log), and again apply the different process mining algorithms to see whether we can rediscover the previously discovered information in this "second pass". Note that this "second pass" is mainly relevant for the evaluation of our approach. In real-life applications of our approach, the simulation model is used to gain insights and to evaluate and compare different redesigns, i.e., the focus is not on the quality of the discovered simulation model. However, since such a "what if" analysis using simulation is common practise, we will not elaborate on this and assume that the reader is familiar with the role of simulation in business process analysis and redesign.

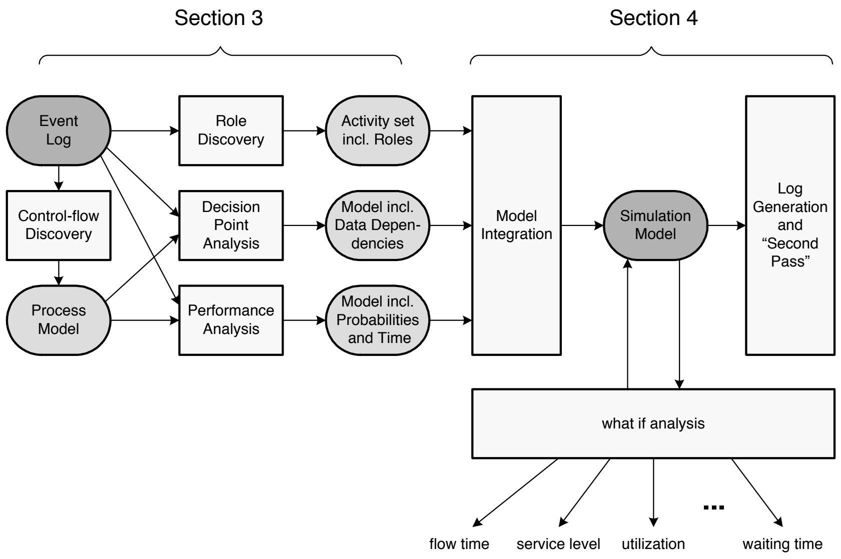


Fig. 4. Overview about the process mining techniques used in this paper

4.1 Control-flow Discovery

Control-flow discovery aims at the automatic extraction of a process model from an event log, i.e., the inference of a structural representation of the underlying process based on historic data.

Typically, events in these logs are only expected to (i) refer to an activity from the business process, (ii) refer to a case (i.e., process instance), and (iii) be totally ordered [3, 4]. Therefore, the event log from Section 3 can be considered as a set of event sequences as shown in Figure 5(a)[4].

Based on this information, the α-algorithm [5] automatically constructs the Petri net model depicted in Figure 5(b). A Petri net is a dynamic structure that consists of a set of *transitions*, which are indicated by boxes and relate to some activity/task, or action that can be executed, a set of *places*, which are indicated by circles and may hold one or more *tokens* (indicated as black dots), and a set of *directed arcs* that connect these transitions and places with each other in a bipartite manner. Transitions are *enabled* as soon as all of their input places (places connected to this transition via an incoming arc) contain a token. If a transition is enabled, it may *fire* whereas it consumes a token from each of its input places and produces a token for each of its output places (places connected to this transition via an outgoing arc). In this way, the firing of a transition may change the *marking* of a net, and therefore the state of the process, which is defined by the distribution of tokens over the places.

[4] The letters *A*–*I* have been introduced as a short-hand for the activity names in the process.

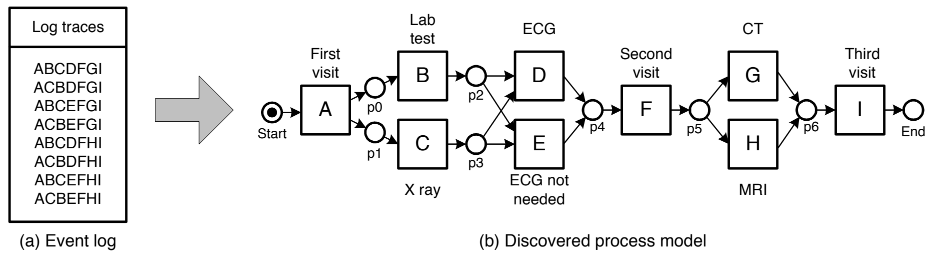


Fig. 5. Control-flow discovery of the outpatient clinic example. A process model reflecting the dependencies among the activities in the process is automatically created

In a nutshell, the α-algorithm works by deriving log-based ordering relations from the event log. Based on these, it infers causal dependencies that can be used to construct the Petri net. For readers interested in further details we refer to the original publication [5]. Note that the α-algorithm is just one of the many control-flow discovery algorithms that have been developed over time (and other algorithms might be used instead). It has been proven to be able to rediscover a large class of workflow processes [5]. Other algorithms have been developed to address challenges such as noise [39] or duplicate tasks [14]. However, this is not relevant for demonstrating the general idea of our simulation model generation approach.

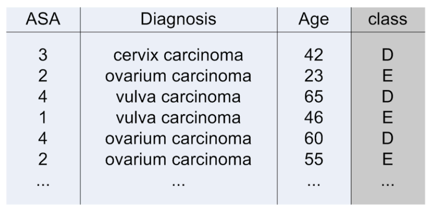
The discovered model in Figure 5(b) visualizes the flow of examinations in the outpatient clinic. To be more precise, the figure shows the *diagnostic process* for gynaecological oncology patients, in which a diagnosis is made for the patient and further investigated through a number of examinations. As soon as the diagnostic process is completed, the treatment phase can be started. The diagnostic process starts with the first visit of the patient to the outpatient clinic. During this “First visit”, a blood sample is taken from the patient, which is tested afterwards in the lab. In parallel to the “Lab test”, the patient undergoes an “X ray” examination, i.e., “Lab test” and “X ray” may occur in any order. After completing the “Lab test” and the “X ray”, the patient is sent to make an appointment for an electrocardiogram, i.e., an “ECG”, if this is needed. After all these examinations, the patient has a follow-up visit in the outpatient clinic. During this “Second visit”, the doctor decides whether either a magnetic resonance imaging (“MRI”) examination, or a computed tomography (“CT”) scan needs to be made. After the “MRI” or “CT”, the diagnostic process is finished with a “Third visit” of the patient to the outpatient clinic.

It is important to keep in mind that the model in Figure 5(b) is constructed fully automatically based on the log referred to in figures 3 and 5(a).

4.2 Decision Point Analysis

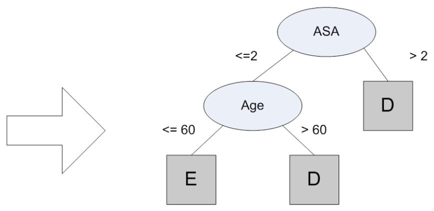
Now that we have discovered a process model reflecting the causal relations between the activities in the outpatient clinic process, we want to gain more insight into the data perspective of that process. More precisely, we want to discover data dependencies that influence the routing of a case. To analyze the choices in a process we first need to identify those parts of the model where the process splits into alternative branches, also called *decision points*. Based on data attributes associated to the events in the log (cf. Figure 3(c)), we subsequently want to find rules for following one route or the other [32].

In terms of a Petri net, a decision point corresponds to a place with multiple outgoing arcs. Since a token can only be consumed by one of the transitions connected to these arcs, alternative paths may be taken during the execution of a process instance. The process model in Figure 5(b) exhibits three such decision points: *p5* (if there is a token, either *G* or *H* can be performed), *p2*, and *p3* (seen from the latter two places, either *D* or *E* may be carried out). The idea is to convert every decision point into a *classification problem* [26, 29, 40], where the *classes* are the different decisions that can be made. As training examples we use the process instances in the log (for which it is already known which alternative path they followed with respect to the decision point). The attributes to be analyzed are the case data attributes contained in the log, and we assume that all attributes that have been written *before* the choice construct under consideration are relevant for the routing of a case at that point[5].



ASA	Diagnosis	Age	class
3	cervix carcinoma	42	D
2	ovarium carcinoma	23	E
4	vulva carcinoma	65	D
1	vulva carcinoma	46	E
4	ovarium carcinoma	60	D
2	ovarium carcinoma	55	E
...	...	...	...

(a) Training examples for decision point “p2”



(b) Decision tree for decision point “p2”

Fig. 6. Decision point represented as a classification problem. Data values associated with process instances in the log are classified and used as training examples, based on which a decision tree can be derived

To illustrate this, we show the classification problem for decision point *p2* in Figure 6(a). The data attributes that are in the scope of this decision point are “ASA”, “Diagnosis” and “Age”, which are recorded during the first visit of the patient (cf. Figure 3). Each row contains the data values for one process instance, i.e., Figure 6(a) shows the data items associated to the first six process instances out of the 1000 cases in the event log. These rows correspond to the training examples that will be provided as input to the classification algorithm later on.

[5] We also allow the user to set other scoping rules, e.g., only the data set in a directly preceding activity, or all case data including the data that is set later.

However, because there is no explicit information in the log about which decision was made at a decision point for some process instance, we first have to infer this information from the log. Starting from the identification of a choice in the process model (i.e., a decision point) a decision can be detected if the execution of an activity in the respective alternative branch of the model has been observed, which requires a mapping from that activity to its "occurrence footprint" in the event log. So, if a process instance contains the given "footprint", this means that there was a decision for the associated alternative path in the process. For simplicity we examine the occurrence of the *first* activity per alternative branch to classify the possible decisions. This is sufficient to fill the "class" column for our running example in Figure 6(a) either with the decision D ("ECG") or with the decision E ("ECG not needed"). However, to successfully conduct decision mining for real-life business processes several challenges posed by, for example, *invisible activities*, *duplicate activities*, and *loops* need to be addressed. We refer the interested reader to our technical report [31], where these issues are discussed in detail.

After identifying a decision point in a business process and classifying the decisions of all the process instances in the log, the next step is to determine whether decisions might be influenced by case data, i.e., whether cases with certain properties typically follow a specific route. To solve the formulated classification problem, various algorithms are available [26, 40]. We decided to use an algorithm based on decision trees (the C4.5 algorithm [29] to be precise). Decision trees are a popular tool for inductive inference and the corresponding algorithms have been extended in various ways to increase practical applicability. For example, they are able to deal with continuous-valued attributes, missing attribute values, and they include effective methods to avoid *over-fitting* the data (i.e., that the tree is too much tailored towards the particular training examples).

Figure 6(b) depicts the decision tree that was derived for decision point *p2* by the J48 algorithm (which is the C4.5 implementation in the Weka machine learning library [40]) with default parameters. Note that decision tree algorithms are able to select those attributes that have influence on the classification (e.g., the data attribute "Diagnosis" has been omitted in the resulting tree as it does not affect the decision between D and E). Furthermore, they are able to automatically find split points in continuous-valued attributes such as the numeric "Age" attribute (i.e., to split at the point > 60 and $<= 60$).

From a decision tree like the one shown in Figure 6(b) we can now read off rules that can be related to the decisions in the process. This way, we can use decision point analysis to extract knowledge about decision rules as shown in Figure 7. Each of the three discovered decision points corresponds to one of the choices in the running example. Note that, because the choices taken at *p2* and *p3* are dependent on each other, they yield the same rules. We can conclude that the "ECG" is only needed for patients that are older than 60 years, or have an ASA greater than 2. Furthermore, the "CT" is only required for patients with the diagnosis "corpus carcinoma" or "ovarium carcinoma", while an "MRI" is needed for patients with the diagnosis "vulva carcinoma" or "cervix carcinoma".

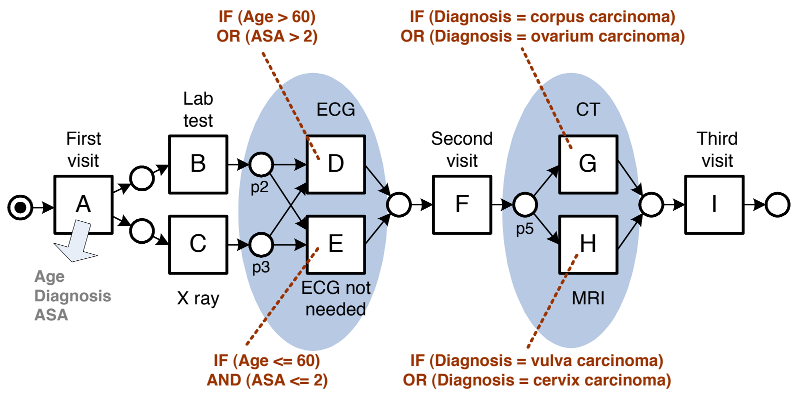


Fig. 7. Example model enhanced with the data perspective. Decision rules for the choice points in the process were derived based on data attributes provided during the "First visit" of the patient

4.3 Performance Analysis

In the previous subsection we have discovered data-based rules for the decision points in the process. Now, we want to gain more insight into the performance perspective of the process. More precisely, we want to enhance the process model with information about *execution times* and *waiting times* for the activities. The execution time is the time between the start and the completion of the activity. The waiting time is the time between the point at which the last activity that is a direct predecessor of this activity was completed and the moment at which the execution of the activity itself is started. Moreover, we also want to enhance the process model with *probabilities for taking alternative paths*, and with information about the *case generation scheme*. This scheme determines the arrival process, e.g., how many new cases arrive per time unit (on average) at the process.

Extracting this information from the log is relatively easy because for each 'start' and 'complete' event in the event log the exact time stamp is given. Together with the discovered Petri net we can *replay*[6] each process instance in the Petri net so that information about execution and waiting times is collected for the activities in the process [22]. Furthermore, for each decision point we can derive the probabilities of alternative paths based on how often each path was followed during log replay. Finally, the arrival rate of cases can be easily derived from the start times of the first activity in each process instance.

During the replay of the log several statistics can be collected for the execution and waiting times, and for how many cases arrive at the process per time unit. These statistics are values like minimum, maximum, mean, variance, etc.

[6] We assume that each process instance can be replayed correctly in the discovered Petri net (see [33] for further details). If, e.g. due to noise, not all instances fit the model, then the non-fitting instances can easily be ignored.

In general, we do not know the underlying distribution for the obtained execution and waiting times, and we assume that they follow a normal distribution. However, we could easily have selected another distribution. Similarly, we do not know the "real" distribution of the case generation scheme, and we assume a negative exponential distribution for the interarrival process (i.e., a so-called Poisson arrival process). To specify the execution and waiting times in terms of a normal distribution, we need to calculate their mean and variance values for each activity. The intensity parameter of the exponential distribution is approximated by the mean value of all measured inter-arrival times (i.e., the times between the start of two new cases). All the values are measured in minutes.

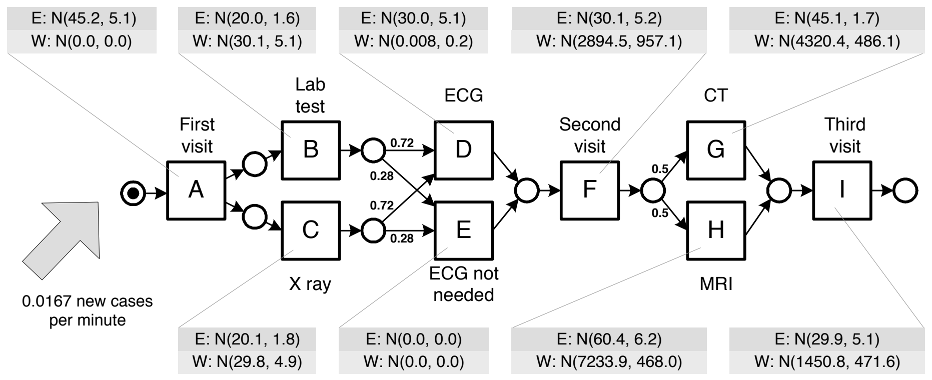


Fig. 8. Example model enhanced with the performance perspective. For each activity the execution time (E) and waiting time (W) are given as a normal distribution N(arithmetic mean, standard deviation). Moreover, the probabilistic values for selecting an alternative path at a decision point and the case arrival rate are provided

The collected values for the outpatient clinic example are shown in Figure 8. For example, one can see that the execution time of activity "CT" has a mean of 45.1 minutes and a standard deviation of 1.7. Similarly, the waiting time of activity "CT" has a mean of 4320.4 minutes and a standard deviation of 486.1. For the case generation scheme we have an intensity of about 0.0167 new cases arriving per minute, i.e., on average one case arrives per hour. Moreover, in Figure 8 one can also see the probabilistic values for selecting an alternative path at a decision point. For example, based on the observations from the log it seems more likely that an "ECG" is needed (72% of the cases) than that it is not needed (only 28%).

4.4 Role Discovery

Now we move our focus onto the organizational perspective of the process. Organizational mining aims at discovering both the *organizational model* (i.e., the relationships between resources and their roles or functional units) and *assignment rules* (i.e., the relationships between roles or functional units and activities) [36]. An organizational model usually contains organizational units (e.g., functional units), roles (e.g., duty), resources, and their relationships (i.e., who belongs to which functional unit, who plays what roles, hierarchy among organizational units). By just using an event log, it is difficult to discover the differences between all of these notions. However, it is possible to derive resource groups in which the people execute similar activities. From a "profile" describing how frequently individuals conduct specific activities, we can derive groups. Such a discovered group of resources may correspond to an organizational unit or a union of people who perform the same roles in real life. In this paper, we will mostly use the term "role" to refer to such a group of resources that have a similar activity profile.

In this paper, we applied the *metrics based on joint activities* proposed in [2] to derive roles. Metrics based on joint activities focus on the activities that resources perform. We assume that people doing similar things are more closely linked than people doing completely different activities. Each individual has a "profile" in the originator by activity matrix based on how frequently it conducts specific activities. Table 1 shows a part of the originator by activity matrix derived from the log in Section 3.

Table 1. A part of the originator by activity matrix, where it is shown how often each resource performed each activity

originator	First visit	Lab test	X ray	ECG	ECG not needed	Second visit	CT	MRI	Third visit
...	.	.	.	.	.	.	.	.	.
Claire	0	310	0	0	0	0	0	0	0
Jan	260	0	0	0	0	239	0	0	129
Jane	0	0	154	0	0	0	47	45	0
Jo	0	349	0	0	0	0	0	0	0
Maria	0	0	158	0	0	0	40	41	0
Martin	244	0	0	0	0	248	0	0	138
Nigel	0	0	178	0	0	0	52	42	0
Ralph	0	0	188	0	0	0	46	47	0
Rose	239	0	0	0	0	250	0	0	140
...	.	.	.	.	.	.	.	.	.

From this matrix, we can measure the "distance" between the profiles of different originators by comparing the corresponding row vectors. We calculated *Pearson's correlation coefficient* to quantify this "distance". Pearson's correlation coefficient produces values ranging from -1.0 to 1.0. Since the positive values imply positive linear relationships between variables, we applied the threshold value of 0.0 and removed negative relationships. In this way, five clusters are derived, namely {Jan, Martin, Rose, Vanessa}, {Claire, Jo, Valentine}, {Fred, Wilma, Vic}, {Alex, Eric, Jane, Maria, Nigel, Ralph}, and {Nobody}. These clusters correspond to roles. Note that the last role ({Nobody}) is a bit artificial and stems from tasks that do not require any resource to be executed. Then, we assigned the clusters to activities based on an entity assignment method. If an originator executed an activity, the activity is assigned to the cluster to which the originator belongs. For example, according to the log fragment depicted in Figure 3 *Nigel* executed the activity "X ray", and, therefore, "X ray" is assigned to the role of *Nigel*, i.e., "Radiology department".

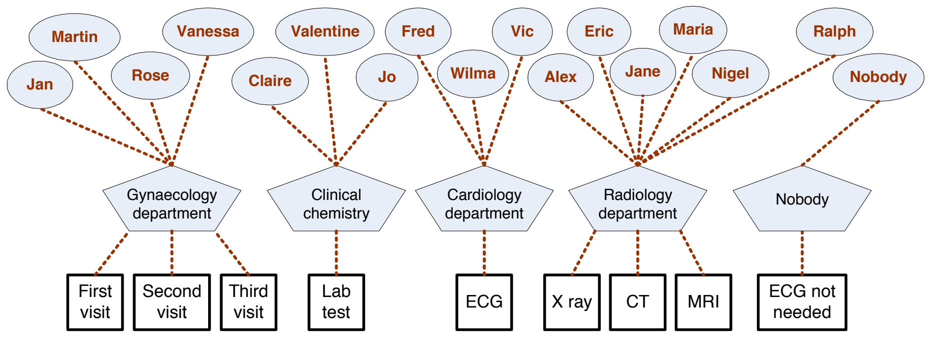


Fig. 9. Example model enhanced with the organizational perspective. The boxes represent activities, the pentagons represent roles, and the circles represent originators

Figure 9 shows the derived roles, and the relationships between roles and activities for the outpatient clinic example. The detected clusters correspond to the "Gynaecology department" (Jan, Martin, Rose, and Vanessa), "Clinical chemistry" (Claire, Jo, and Valentine), "Cardiology department" (Fred, Wilma, and Vic), and "Radiology department" (Alex, Eric, Jane, Maria, Nigel, and Ralph), respectively. The "Gynaecology department" is involved in the "First visit", "Second visit", and "Third visit" of the patient, while the "Lab test" is done by the "Clinical chemistry". The "Cardiology department" is in charge of the "ECG" activity, and the "Radiology department" is related to the medical imaging activities such as "X ray", "CT", and "MRI". The cluster "Nobody" does not contain a real resource from the outpatient clinic, but only a dummy originator that was used for activity "ECG not needed" (in the case that activity "ECG" can be skipped).

5 Model Integration and Evaluation

In the previous section we have seen how different characteristics of a process can be extracted from an event log. These characteristics can now be used to construct a simulation model. In this section, we briefly show how the different perspectives are joined into a single model (Section 5.1) and represented as a

Colored Petri net (Section 5.2). Moreover, we will evaluate how good the CPN model approximates these characteristics by generating event logs during the simulation of the model in CPN Tools, and the repeated analysis of this "second pass" (Section 5.3). For this, the same algorithms will be used as in Section 4, and the results of the "first pass" and "second pass" will be compared to each other.

5.1 Merging Perspectives

To get a better view on the process as a whole, it is useful to integrate the discovered perspectives in one holistic model. This is fairly easy as long as the discovered process characteristics are orthogonal to each other (i.e., there is no conflicting information). They can be simply merged together. If there are conflicting characteristics, then this becomes mainly a technical challenge. Imagine, for example, two performance analysis algorithms that fit execution time distributions in a different way. To integrate these two results we would need to either select one of them on a per-activity basis, or to average them etc.

In the case of the running example, the discovered characteristics are largely orthogonal. Note that although the decision rules and the mined probabilities are potentially conflicting as they both relate to the decision-making behavior of the analyzed process, we still want to preserve them as they reside on different levels of abstraction (i.e., the probabilities are influenced by the data value distributions) and might want to use either of them when configuring the simulation model later on. Figure 10 shows the integrated model for the example process. For example, activity "ECG" is only needed for patients who are older than 60 or have an ASA greater than 2, the corresponding execution time is on average 30 minutes (with a standard deviation of 5.1), and the "Cardiology department" is in charge of the activity.

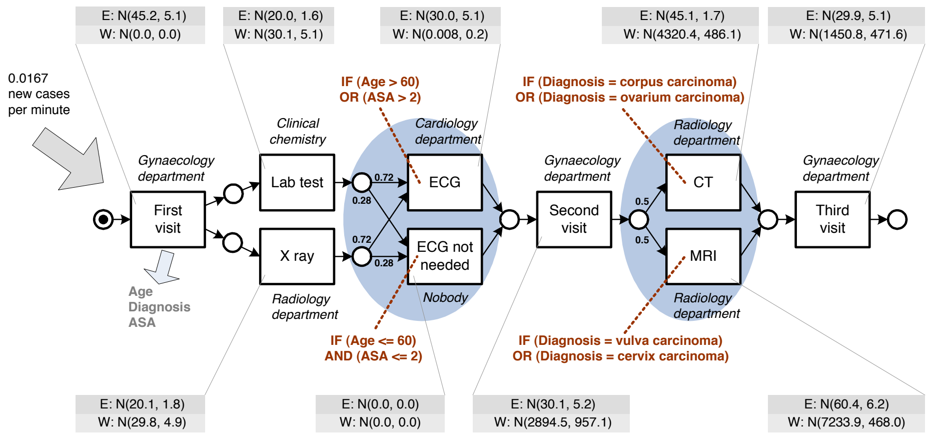


Fig. 10. Example model with integrated data, organizational and performance view

5.2 CPN Model

The integrated model now contains key characteristics of the outpatient clinic process from different perspectives. As a next step, we want to represent these characteristics in an actual, executable simulation model. It is clear that a simulation model can only capture certain aspects of a process, and that simplifying assumptions must be made to approximate the real behavior.

We selected Colored Petri Nets (CPNs) [23, 24] as a representation because of their expressiveness and the strong simulation capabilities of CPN Tools [38]. Furthermore, the hierarchy concept allows for the composition of a CPN model in a modular way (and, therefore, for different levels of abstraction). The time concept and the availability of many probability distributions in CPN Tools allow for the modeling of performance aspects. Moreover, by introducing resource tokens also organizational and work distribution aspects can be modeled. Finally, data-related aspects can be modeled by introducing data tokens.

In [34], we proposed a CPN representation for business processes that is able to capture different perspectives of a process. This CPN representation is generic and suitable for automatic generation, while it remains readable for a human analyst. We developed a CPN Export plug-in, which can generate such CPN models. To make use of the simulation facilities of CPN Tools, the actual process model is provided together with a simulation environment, which generates cases, initializes data etc. Furthermore, for each activity in the process, a sub-page is created containing the actual simulation information. Depending on the selected process characteristics, these activity sub-pages may look very different.

As an example, the sub-page of activity "CT" is depicted in Figure 11. It covers information from all the perspectives that were previously discovered:

- The sub-page for the "CT" activity contains *schedule*, *start* and *complete* transitions to incorporate the waiting time and execution time of the activity. According to the time delay that is generated by the normal distribution for the waiting time (cf. outgoing arc of the "CT_schedule" transition), the case token remains in place W (i.e., waiting state) while the model time progresses. Similarly, a case token is forced to reside in place E until the corresponding execution time delay has passed (cf. outgoing arc of the "CT_start" transition).
- The "CT" activity may only be executed by people of the *Radiology department*. This is modeled by defining a separate color set with the name RADIOLOGY_DEPARTMENT, which contains only the people that belong to this department. Then, a variable "Radiology_department", which is of type RADIOLOGY_DEPARTMENT, can be used to match the required role for the activity. In this way, only the resources that belong to this color set can be consumed by transition "CT_start", i.e., the selection of a token from place "Resources" is limited by the type of this variable. As soon as transition "CT_start" is fired, the corresponding resource token resides in the place E and it is not available for concurrent executions of further activities, until transition "CT_complete" fires and puts the token back.

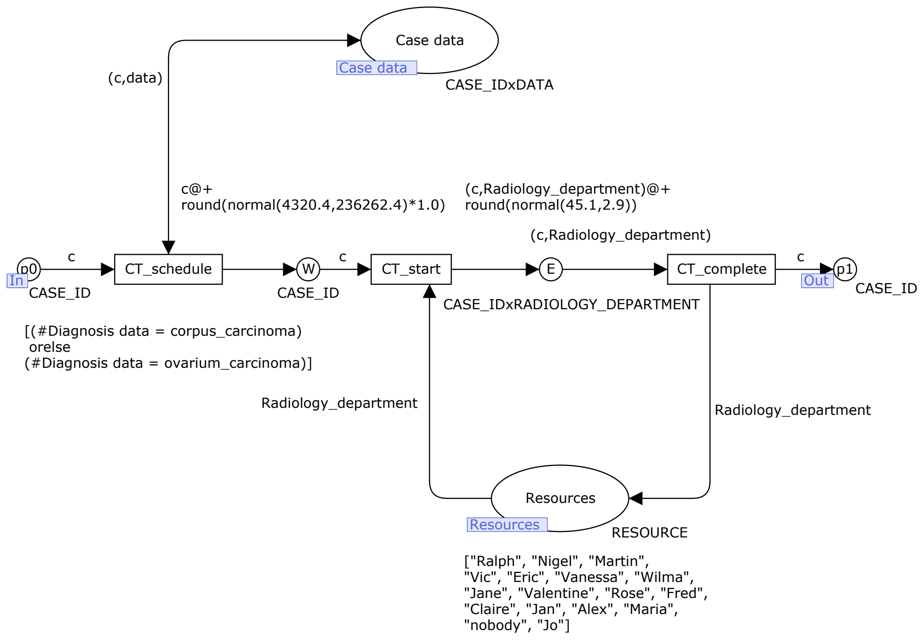

Case data
Case data
CASE_IDxDATA
(c,data)
c@+
round(normal(4320.4,236262.4)*1.0)
(c,Radiology_department)@+
round(normal(45.1,2.9))
(c,Radiology_department)
p0
In
c
CT_schedule
W
c
CT_start
E
CT_complete
c
p1
Out
CASE_ID
CASE_ID
CASE_IDxRADIOLOGY_DEPARTMENT
CASE_ID
[(#Diagnosis data = corpus_carcinoma)
orelse
(#Diagnosis data = ovarium_carcinoma)]
Radiology_department
Radiology_department
Resources
Resources
RESOURCE
["Ralph", "Nigel", "Martin",
"Vic", "Eric", "Vanessa", "Wilma",
"Jane", "Valentine", "Rose", "Fred",
"Claire", "Jan", "Alex", "Maria",
"nobody", "Jo"]

Fig. 11. Sub-page for the "CT" activity, showing how the simulation information related to the "CT" activity is represented in the CPN model

– The "CT" is only required for patients with diagnosis "corpus carcinoma" or "ovarium carcinoma". This is represented in the CPN by the guard that belongs to the "CT_schedule" transition. If this transition is enabled from a control-flow perspective, it additionally needs to satisfy the given guard condition to be fired. To check the value for the "Diagnosis" data attribute, a double arc is modeled between the "CT_schedule" transition and the "Case data" place, which contains all data attributes.

Note that in the combination of different process characteristics we have to make choices. For example, in Figure 11 one can see that the decision to perform a CT (instead of an MRI) is based on the value of a data attribute (i.e., the diagnosis of the patient). However, we could have also chosen to base the decision on stochastic values (i.e., probabilities of alternative paths).

Note that the integrated process and the CPN fragment shown in figures 10 and 11 are generated automatically without any human intervention or modeling. However, users can influence the model generation by changing the settings or adding explicit information.

5.3 Evaluation "Second Pass"

The discovered simulation model can be used for all kinds of analysis (e.g., what-if analysis for estimating the effects of some redesign). However, we assume that the reader is aware of the practical relevance of simulation. Therefore, this section focuses on the validation of the approach.

It is clear that the value of a simulation-based analysis largely depends on the quality and validity of the simulation outcomes. The validity of the representation of the selected key characteristics is one important aspect that needs to be ensured when approximating a real-life process by a simulation model. Therefore, we want to evaluate how good our simulation model captures the discovered process characteristics. Other aspects, such as the validity of the recorded log data (which is used as input for the described process discovery techniques), are beyond the scope of this paper, and we assume that the event log contains representative behavior for the original process.

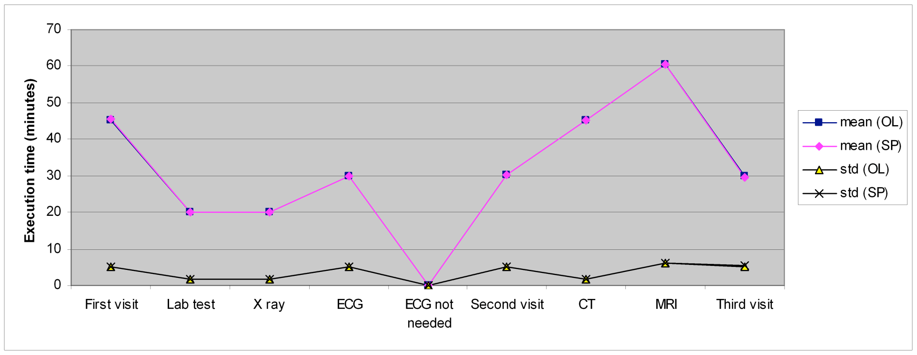
The following approach has been taken: For the CPN model described in the previous section, a simulation was run for 1000 cases. During the simulation in CPN Tools, separate event logs were generated for each case. These event logs can be created by extending the CPN model with monitors, which log the occurrence of each transition on the sub-page of an activity, as discussed in [15] and [34]. Note that our CPN Export plug-in automatically adds monitors to the CPN models, so that they produce simulation output that can be converted into MXML using the CPN Tools plug-in of the ProM*import* framework. The resulting MXML file can again be used as input for the mining algorithms. The analysis of the simulated event log based on the discovered model is called the "second pass". In the "second pass", the same algorithms as those used in Section 4 to discover the control-flow, data, performance and resource perspective are deployed, but now for the newly generated MXML file.

The results obtained for the control-flow, data, and resource perspective are exactly the same as in the previous analysis, i.e., they can be completely rediscovered. The results obtained for the performance perspective are also very similar. In the second pass, we have an intensity of about 0.0163 (instead of 0.0167) new cases arriving per minute, and we get the same probability values for the alternative branches. Figure 12 depicts the execution time and waiting time values obtained from both the original log (OL) and the second pass (SP), which are almost identical.

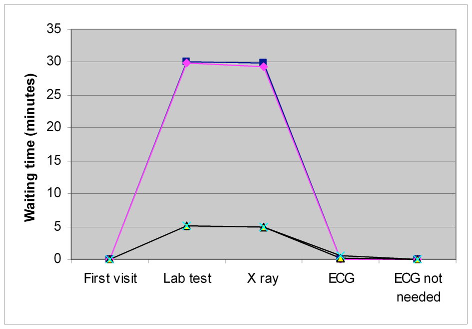
With regard to our artificial example, the discovered CPN model coincides with the original model, i.e., it is possible to completely rediscover the model from the event logs. This serves as a proof-of-concept, i.e., it is possible to discover simulation models. In Section 7, we will investigate this further by evaluating the quality of the generated simulation models for two real-life examples. However, we first present the software supporting our approach.

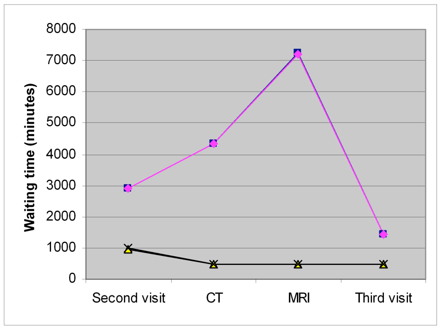
6 Simulation Models in the ProM Framework

The *Pro*cess *M*ining (*ProM*) framework [1] is an extensible tool suite that supports a wide variety of process mining techniques in the form of plug-ins. In this section, we describe how the presented discovery approach is supported by ProM. A set of plug-ins is available to discover and integrate different charac-



(a) execution time





(b-1) waiting time (b-2) waiting time

Fig. 12. Execution time and waiting time results obtained for the original log (OL) and the second pass (SP) of the running example. One can see that both the arithmetic mean (mean) and the standard deviation (std) of first and second pass are very close. In fact, it is difficult to distinguish the lines relating to OL and SP because they coincide in each of the three graphs

teristics of a process from an event log, and to generate a CPN model that can be directly used for simulation. We shortly explain the functionality, and show screenshots using the running example[7].

As can be seen in Figure 4, we start with an event log. To discover the control-flow perspective a number of different algorithms are available. We choose the `Alpha algorithm` plug-in, which constructs a process model in terms of a Petri net. Based on this process model and the log, we can discover the performance perspective of the process by applying the `Performance Analysis with Petri net` plug-in. A screenshot of the analysis results for the example process is shown in Figure 13(a). The plug-in evaluates the time stamps in the log and projects the extracted performance information on places and transitions. It graphically shows the bottlenecks by coloring places according to the time that is spent in

[7] Note that the event log of the running example used in this paper can be downloaded together with ProM from *www.processmining.org*.

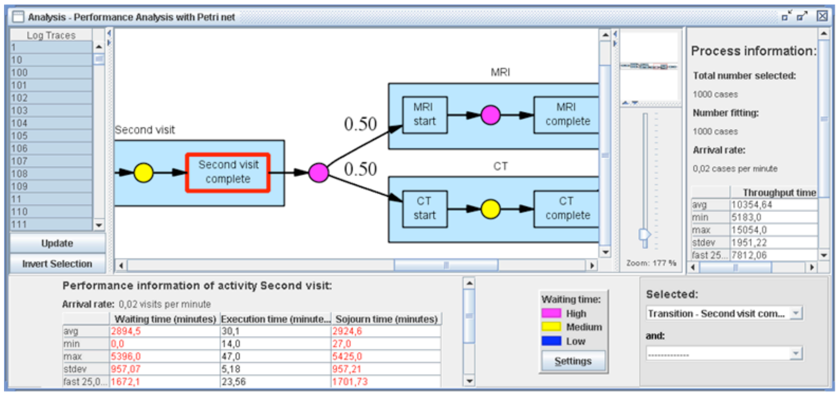
this part of the process (red indicates a high waiting time, while blue means less waiting time). Furthermore, it provides performance indicators, such as average, variance etc. of the execution time for an activity, or the time spent at a certain place, or between two selected activities. For example, in Figure 13(a), we can see several statistics for the waiting, execution and sojourn time of the "Second visit" activity, which we have selected.

The data perspective of the process can be discovered using the `Decision Point Analysis` (also called Decision Miner) plug-in. This plug-in analyzes the data attached to events and using classical decision tree analysis it is possible to add decision rules to the Petri net. These decision rules are presented as conditions on arcs. A screenshot of the plug-in is shown in Figure 13(b). For example, in Figure 13(b), we can see the decision tree that has been discovered for the choice between "ECG" and "ECG not needed" in the outpatient clinic process.

The resource perspective can be discovered by applying the `Organizational Miner` plug-in. This plug-in analyzes information about resources in the event log and returns information about the relationship between activities and originators, which can also be visualized. Among others, resources that perform similar activities or are working together can be discovered and grouped together. In this way, allocation rules can be added to activities. A screenshot of the plug-in is shown in Figure 13(c), where the people that are working on activities in the "Gynaecology department" are shown (represented as *minedGroup3*).

These three discovery plug-ins now offer each a simulation model with additional information about the process (e.g., the arrival rate of new cases), or specific activities in the process (e.g., by which group of people it was performed). But before we can integrate the different perspectives into one model, we first have to transform the discovered Petri net model with the `Combine Low-level Activities` plug-in. This step is necessary because we start with a log that contains transactional information (i.e., the start and completion of an activity). As a consequence, each 'start' and 'complete' event in the log is represented as a separate activity in the discovered process model (for example, in Figure 13(a) you can see both activity "CT start" and activity "CT complete"). However, because we want to link the obtained information to activities as a whole, we combine these low-level tasks that belong to one activity into a single transition in the Petri net model. Since this is rather a technicality and not essential for understanding the functionality, we will not elaborate on this.

As a next step, the different, now activity-based simulation models can be integrated into one model with the help of the `Merge Simulation Models` plug-in. Note that, although we use Petri nets, a simulation model may be based on any kind of process model and ProM supports different model conversions. For example, in Event-driven Process Chains (EPCs) activity-related information could be mapped to functions, decision point-related information to XOR connectors etc. For this, a *reference model* is chosen among the input models as a "template" for the output model (and the activities from the different input models are mapped on their corresponding activities in the output model). For



Process information:

Total number selected: 1000 cases

Number fitting: 1000 cases

Arrival rate: 0,02 cases per minute

	Throughput time
avg	10354,64
min	5183,0
max	15054,0
stdev	1951,22
fast 25...	7812,06

Performance information of activity Second visit:

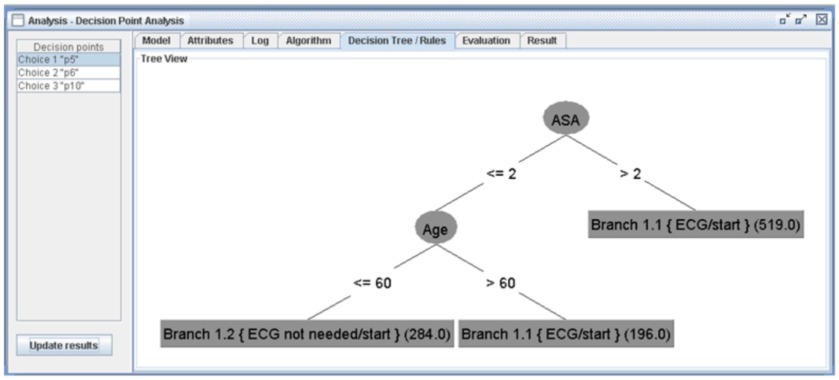
Arrival rate: 0,02 visits per minute

	Waiting time (minutes)	Execution time (minute...	Sojourn time (minutes)
avg	2894,5	30,1	2924,6
min	0,0	14,0	27,0
max	5396,0	47,0	5425,0
stdev	957,07	5,18	957,21
fast 25,0...	1672,1	23,56	1701,73

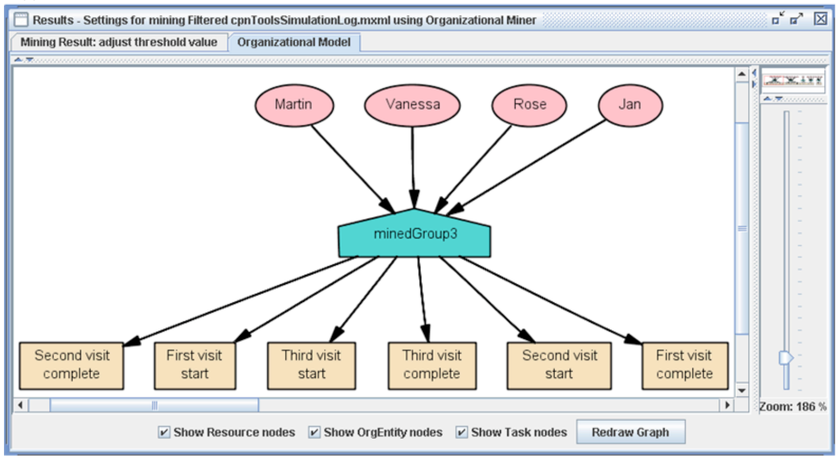
Selected: Transition - Second visit com...

and: -------------

(a) The Performance Analysis plug-in replays the process instances in the Petri net process model and calculates, among others, execution times and waiting times for activities

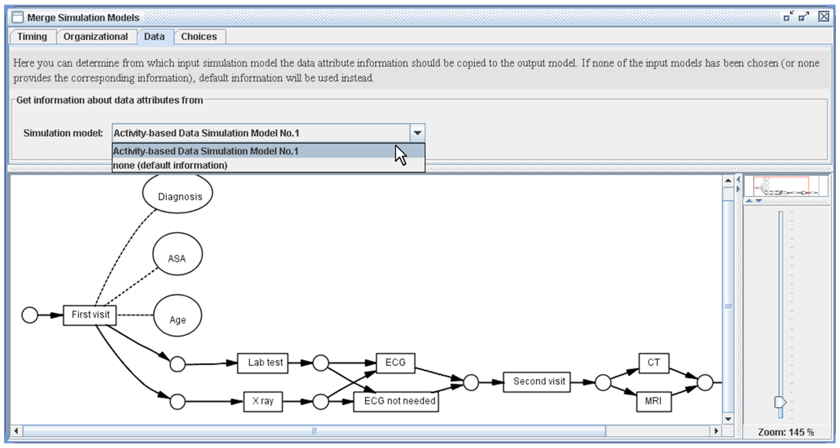


(b) The Decision Point Analysis plug-in discovers decision rules for the alternative branches based on data attributes. Here, the choice between "ECG" and "ECG not needed" is analyzed by constructing a decision tree



(c) The Organizational Miner discovers resources that perform similar tasks, and groups them together. Here, the group of people working in the "Gynaecology department" is depicted

Fig. 13. Discovery plug-ins in the ProM framework



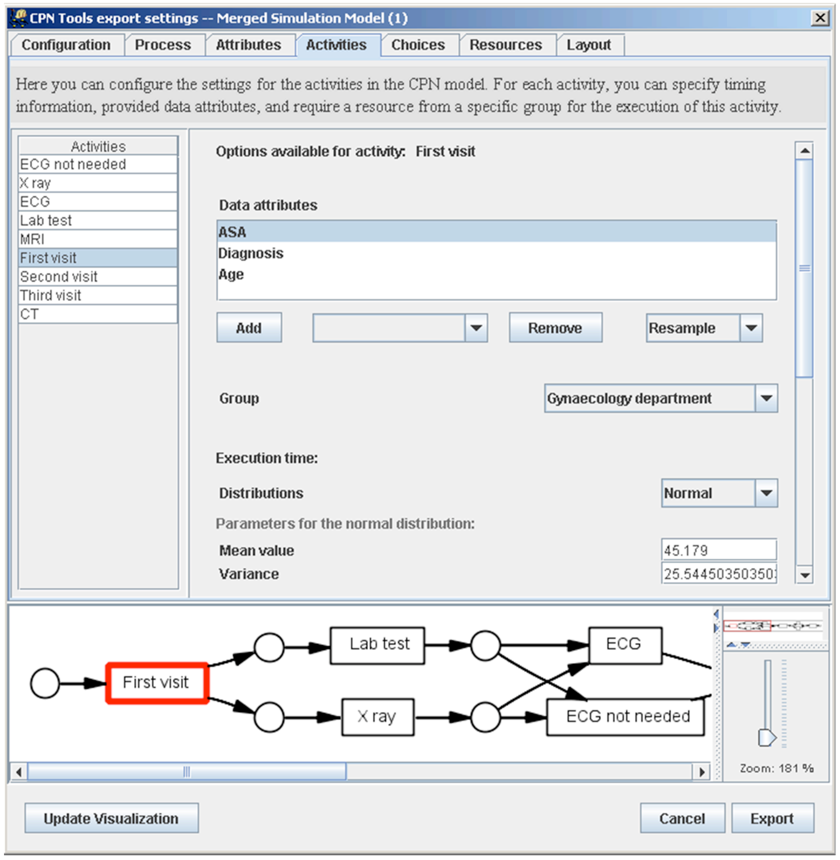
Here you can determine from which input simulation model the data attribute information should be copied to the output model. If none of the input models has been chosen (or none provides the corresponding information), default information will be used instead.

Get information about data attributes from

Simulation model: Activity-based Data Simulation Model No.1

Activity-based Data Simulation Model No.1
none (default information)

(a) The Merge Simulation Models plug-in integrates high-level information from different perspectives into one model. Here, the data attributes obtained from the Decision Miner are depicted



Here you can configure the settings for the activities in the CPN model. For each activity, you can specify timing information, provided data attributes, and require a resource from a specific group for the execution of this activity.

Activities
ECG not needed
X ray
ECG
Lab test
MRI
First visit
Second visit
Third visit
CT

Options available for activity: First visit

Data attributes

ASA
Diagnosis
Age

Group: Gynaecology department

Execution time:

Distributions: Normal

Parameters for the normal distribution:

Mean value: 45.179

Variance: 25.54450350350

(b) The CPN Tools Export plug-in generates a CPN model according to the given high-level information. Here, the discovered and integrated information about activity “First visit” is depicted

Fig. 14. Integration and Export plug-ins in the ProM framework

each perspective, we can then determine from which of the (potentially conflicting) input models the information should be included. For example, in the screenshot in Figure 14(a), we select to retain the characteristics delivered by the Decision Miner for the data attributes in the merged simulation model.

Finally, we want to generate a CPN from the integrated model, which can be read and simulated by CPN Tools. To this end, we use the `CPN Tools 2.0 Export` plug-in, which displays all the discovered process characteristics before the actual export takes place. We can also modify the given settings ("what if" analysis), and provide additional information. For example, we may choose to give meaningful names to the discovered resource clusters, i.e., *minedGroup3* = "Gynaecology department" etc. Furthermore, it is possible to choose different configuration options. For example, each choice in the process may be either based on data rules or stochastic properties, logging monitors can be generated or not, and we can either include or exclude the modeling of explicit waiting time (to include waiting time, a 'schedule' transition will be generated for each sub page as shown in Figure 11, otherwise it will be left out). A screenshot of the plug-in is shown in Figure 14(b), where one can see the provided data attributes, associated resource group, and the execution time settings for activity "First visit". After exporting, CPN Tools can be used to simulate the process without adding any additional information to the generated model.

We have demonstrated that the discovery of a simulation model as presented in this paper is completely supported by the ProM framework. Although a number of different plug-ins must be applied to arrive at the CPN model, sensible default settings and a convenient user interface make it possible to quickly explore different variants.

7 Case Studies

To further validate the approach discussed in this paper, we have performed two case studies. They are based on process logs from two different municipalities in the Netherlands. In these case studies, we focus on the quality of the simulation models rather than on the discovered process characteristics. Similar to the running example, we first perform process mining to discover several perspectives (control-flow, data, performance, and organizational), and generate a CPN from the integrated model. Then, an event log is generated during simulation in CPN Tools, and re-analyzed with ProM to compare the results of this second pass with the initial analysis results. We also investigate the results according to different simulation configurations.

7.1 Case Study I

The case study is conducted based on real-life logs from a municipality in the Netherlands. The municipality uses a workflow system developed using the "Eastman Software Workflow for NT". This systems is, among others, used to handle complaints. We obtained a log from this workflow system for the complaint handling process, which we converted to the MXML format via a plug-in in the ProM*import* framework [20]. When the municipality receives a complaint, they first initiate a process instance. Then it is prepared (the case is investigated) and assigned to a suitable activity out of four actual complaint handling activities. Then, the case is moved to the assigned activity and, after it has been handled, the process is finished. The particular event log we use in this section contains 363 cases. The number of total events is 1,817, and the log has five different activities. Each activity is recorded by logging 'start' and 'complete' events. 13 employees participated in the process execution, and the log contains 15 different data elements such as the ID of the case, queue of activities in the workflow engine, priority of the task, etc.

Figure 15 shows some screenshots of the mining results. We used the Alpha miner in ProM to discover the process model. The resulting process model has 100% fitness (i.e., every trace in the log complies with the discovered model). The average waiting time of activities in this complaint handling process is 45.7 hours, while the average execution time of activities is only 5.1 hours. The overall processing time is 72.6 hours, and a new case is generated every 12.4 hours (on average). The decision point analysis result shows that the choice is based on the value of the data attribute "queue" (which determines the activity that is scheduled next in the workflow engine). From the discovered rules, it seems as if the most frequent branch "AG08_GBA_afnemer"[8], which is executed in 88% of the cases, is also the default branch if none of the four complaint handling activities has been explicitly scheduled (in this case the "queue" still contains one of the previous activities). By the Organizational Miner, 6 roles are derived and 13 originators are assigned to clusters based on the execution history in the log.[9]

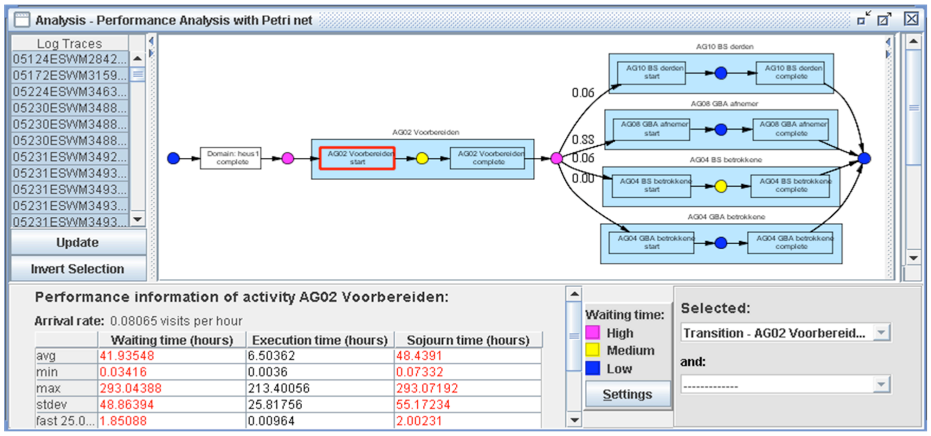
Table 2. Configurations of the three simulation models

	M0	M1	M2
Control flow	include	include	include
Organizational model	include	include	include
Decision Rule	data attribute	data attribute	probability
Performance info	execution time no waiting time	execution time 95% waiting time	execution time 95% waiting time

From the mining results, three models are generated with different configurations on decision rules and waiting time. Table 2 summarizes the configurations of the three models. The first model (M0) and the second model (M1) combine

[8] Note that, because there is no need to understand the process in detail, we did not translate the Dutch activity names.

[9] Note that in Figure 15(c) and Figure 18(c) the names of the originators were erased to ensure confidentiality and privacy.



Performance information of activity AG02 Voorbereiden:

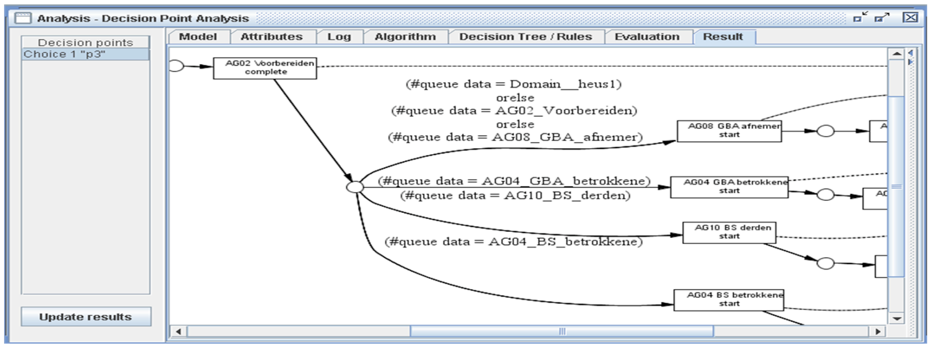
Arrival rate: 0.08065 visits per hour

	Waiting time (hours)	Execution time (hours)	Sojourn time (hours)
avg	41.93548	6.50362	48.4391
min	0.03416	0.0036	0.07332
max	293.04388	213.40056	293.07192
stdev	48.86394	25.81756	55.17234
fast 25.0...	1.85088	0.00964	2.00231

Selected: Transition - AG02 Voorbereid...

and: ------------

(a) Performance values: performance values of the process and activities are discovered

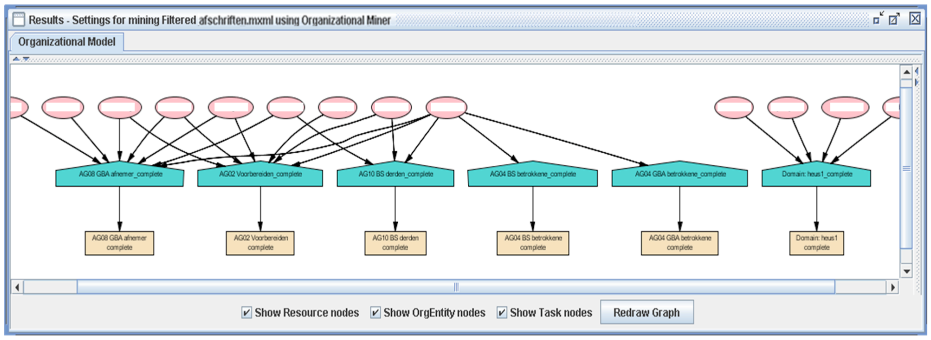


(#queue data = Domain__heus1)
orelse
(#queue data = AG02_Voorbereiden)
orelse
(#queue data = AG08_GBA_afnemer)

(#queue data = AG04_GBA_betrokkene)
(#queue data = AG10_BS_derden)

(#queue data = AG04_BS_betrokkene)

(b) Decision rules: the choice is performed based on “queue” value


Results - Settings for mining Filtered afschriften.mxml using Organizational Miner
Organizational Model
AG08 GBA afnemer_complete
AG02 Voorbereiden_complete
AG10 BS derden_complete
AG04 BS betrokkene_complete
AG04 GBA betrokkene_complete
Domain: heus1_complete
AG08 GBA afnemer complete
AG02 Voorbereiden complete
AG10 BS derden complete
AG04 BS betrokkene complete
AG04 GBA betrokkene complete
Domain: heus1 complete
Show Resource nodes
Show OrgEntity nodes
Show Task nodes
Redraw Graph

(c) Organizational model: the 6 roles and 13 originators are discovered

Fig. 15. Mining results: (a) performance values (b) decision rules (c) organizational model

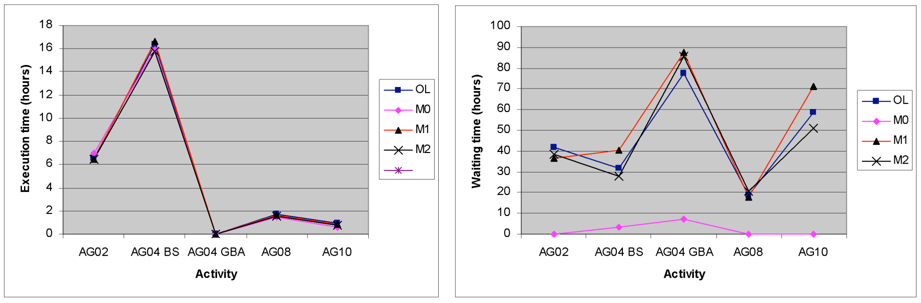
all the four mining results and use the data perspective from the decision miner for the decision rule. In contrast, the third model (M2) uses probabilities (i.e., a stochastic approximation) to make the choice of the subsequent activity. Furthermore, the waiting time is determined only by resources in M0 (if all resources of a certain type are occupied, an activity might have to wait until a resource becomes available again and it can be started). In contrast, M1 and M2 contain extra waiting time (95% of the observed waiting time) in addition to the waiting time that results from the unavailability of resources.

Note that waiting time may result from competing for resources with other cases *inside* the same process and competition *between* processes. For example, an employee may be involved in multiple processes that are all competing for the employee's attention. Although the CPN model only considers the complaint handling process, the employees of the municipality also work on other processes. Therefore, it is necessary to add waiting time in addition to the simulated queueing time resulting from the competition with other complaints. Since the total waiting time is measured by ProM, it is only natural to add part of this time. Therefore, we added 95%[10] of the observed waiting time to M1 and M2. Note that this is a necessary approximation when analyzing a process in isolation.

We perform simulations and analyze the generated process logs; each of them contains 400 cases. We obtain the same process models and organizational models. The same decision rules are derived from the logs from M0 and M1 (since M2 does not contain the data perspective, it has no information about data). Figure 16 shows the performance analysis results including the performance values from the original log (OL). Figure 16(a) shows the execution times of the activities. The values for all the three models are comparable to each other. Figure 16(b) shows the waiting times. The values from M0 are much smaller than the others. In the real world, resources deal with several activities in the organization and perform their work based on their own schedule. However, since we handle not all processes but only one process, we cannot take this situation into account. Thus, the number of resources obtained from process logs is generally large enough to immediately execute activities. To compensate this, we introduce extra waiting time. If we add an additional delay of 95% of the observed waiting time (M1, M2), waiting times are similar to the values from the original log. Note that the waiting time generated by the simulation model is the sum of the delays resulting from the queuing of complaints and the added extra waiting time.

Figure 17 shows the probability values for choosing a particular path when visiting the decision point shown in Figure 15(b). The names on the horizontal axis refer to the activities following the decision point. When we use probabilities to make a decision (M2), the resulting probability values are almost identical to

[10] We find the right degree of external waiting time by experimenting and evaluating the results of the second pass as described in this paper. That is, we try to match the simulated results with the original ones to find out to which degree the waiting time stems from competition within the process according to our model (here, this was approx. 5%) and to which degree it is influenced by other factors (95%).



(a) execution time (b) waiting time

Fig. 16. Performance analysis results based on the original log (OL) and the three discovered simulation models

the original values. However, we can see that if we use decision rules based on data attributes (M0, M1), the results are different. This is not desirable since the probability of an alternative branch can, for example, influence the throughput time of the process (e.g., through the increase of the probability of executing an activity which has a long execution time, or because one branch has more steps to be executed than the other). We found out that the differences stem from the fact that the values of the (nominal) decision variable "queue" are randomly generated by the model, rather than being sampled from the actual data value range distribution. Note that in the running example the probabilities did match because we generated the log with decisions based on randomly distributed (nominal) data in the first place. However, in this section we use a real-life log where this is not the case, which enabled us to detect this flaw. This demonstrates the value of performing experiments based on real-life data.

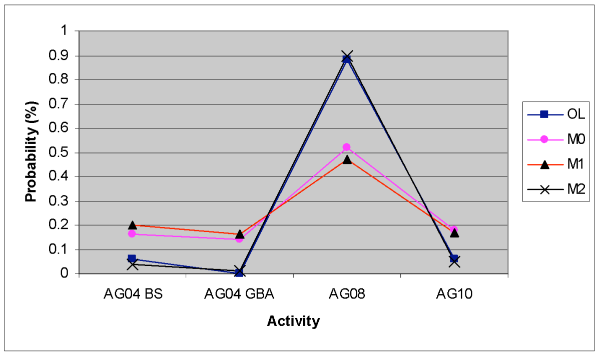


Fig. 17. Probabilities for the decision point shown in Figure 15(b)

7.2 Case Study II

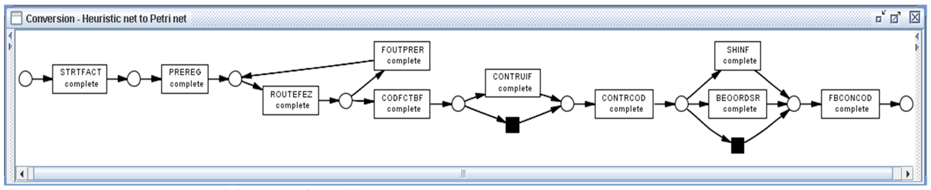
The second case study deals with a log that we obtained from the Urban Management Service of a local municipality of 90,000 citizens, situated in the northern part of the Netherlands. They have implemented their own custom-made workflow system. From the workflow system, we extracted process logs and converted them into the MXML format. We use the log of the handling of invoices in 2005. The log data contains 570 cases. The number of total events is 6,616. The process consists of 10 activities, and 110 employees participated in the process execution. For the first activity only 'complete' events are recorded, but the other activities record both 'start' and 'complete' events. The general procedure is that an invoice is scanned and subsequently sent by the workflow management system to the central financial department. A clerk registers the invoice, after that it is sent to the proper local financial office. Depending on the kind of invoice, there are various checks that need to take place: the person responsible for the budget that is used for the purchase must approve (the budget keeper); the fit between the purchase with the supplier's contract (if any) must be established; various managers may be required to authorize the invoice depending on the amount of money involved etc. Eventually, a purchase may be paid by the central financial office.

From the process logs we derived a model for the control-flow perspective and extended the model with characteristics from the performance and organizational perspectives. Note that, since the original log does not contain any data, we cannot perform the decision point analysis. After integrating the discovered models, we generate simulation models with different configurations of the waiting time.

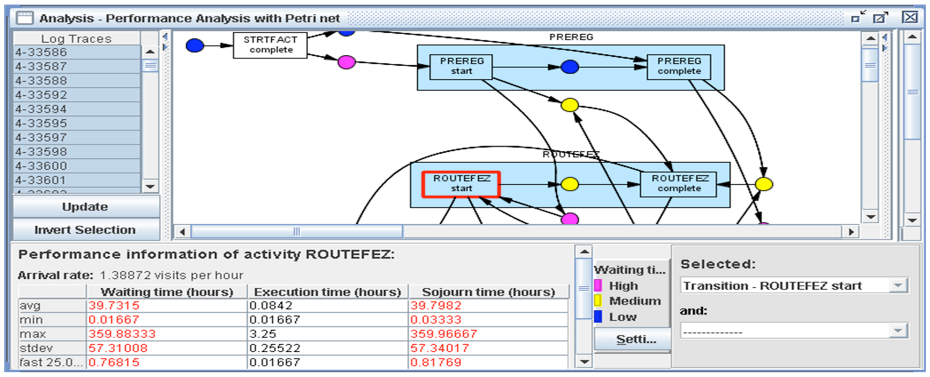
Figure 18 shows the mining results. We generate the process model with the Heuristic miner in ProM. Figure 18(a) shows the generated model in terms of a Petri net[11]. The generated model has 100% fitness with respect to the log (i.e., it completely captures the behavior from the log). To calculate performance information such as execution time and waiting time, we create a Petri net model including *start* and *complete* tasks and use the Performance analysis plug-in. Figure 18(b) shows the results of this performance analysis. The overall processing time is 182 hours (about a week), and a new case is started every 50 minutes, on average respectively. We also generate an organizational model as shown in Figure 18(c). 10 roles are derived and employees are assigned to the roles. Table 3 shows the performance analysis results for the five major activities in the process. Similar to the previous case study, one can observe that the waiting times are by a multiple higher than the execution times in the process. This is very typical for a real-life process, where much time is spent on administration, synchronization, etc. The table also shows organizational mining results. It shows the number of resources who are involved in the execution of each activity.

From the mining results, two models are generated with different configurations on the waiting time. In the first model (M0), the waiting time is determined

[11] Note that the ProM framework supports various conversions between modeling languages, such from Petri nets to Heuristic nets, EPCs, YAWL and vice versa.



(a) control flow: 10 tasks and their dependencies are discovered



Performance information of activity ROUTEFEZ:

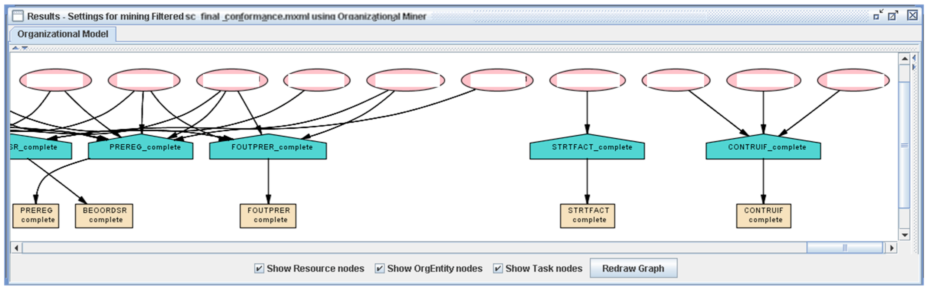
Arrival rate: 1.38872 visits per hour

	Waiting time (hours)	Execution time (hours)	Sojourn time (hours)
avg	39.7315	0.0842	39.7982
min	0.01667	0.01667	0.03333
max	359.88333	3.25	359.96667
stdev	57.31008	0.25522	57.34017
fast 25.0...	0.76815	0.01667	0.81769

Selected: Transition - ROUTEFEZ start

and: -------------

(b) performance values: performance values of the process and activities are discovered



(c) Organizational model: the 10 roles and 110 originators are discovered

Fig. 18. Mining results: (a) control flow (b) performance values (c) organizational model

Table 3. Organizational mining and performance analysis results for the five main activities in the process

	execution time (minutes)	waiting time (hours)	# of resources
CODFCTBF	3.71	29.7	76
CONTRUIF	1.75	17.9	3
ROUTEFEZ	5.05	39.7	25
CONTRCOD	2.71	22	24
FBCONCOD	2.85	66.5	7

only by resources, while the second model (M1) contains extra waiting time (100% of the observed waiting time). Figure 19 shows the generated simulation model (M0) in CPN Tools. Note that the Petri net includes invisible activities (i.e., transitions that do not correspond to an event in the log and that are only executed to enable "real activities"). They are denoted as small tasks filled with black color.

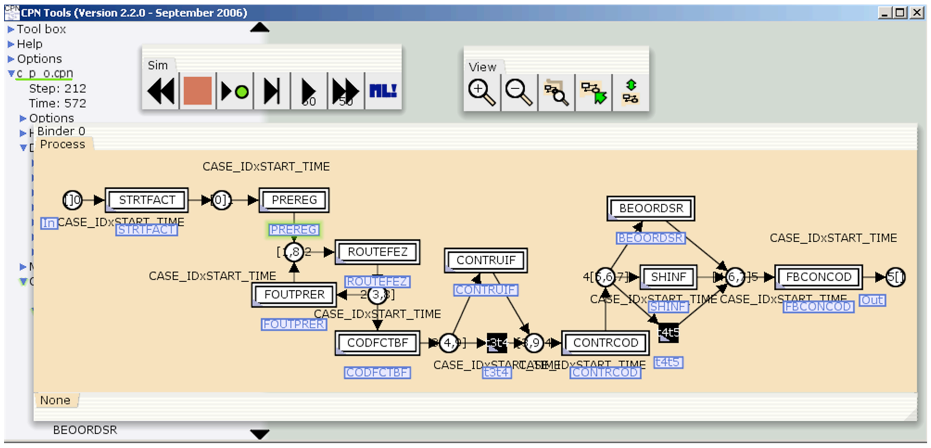


Fig. 19. Simulation model in CPN Tools

With each simulation model, we generate 600 cases and analyze the generated process logs with ProM. In both cases, the process model discovered by the Heuristic miner from the simulated process log equals to the original model in Figure 18(a). The organizational model is also re-discovered. Figure 20 shows the performance analysis results. Figure 20(a) shows the execution times of the activities. As before, the results are very similar. Figure 20(b) shows the waiting times. In M0, no waiting times are observed due to the small execution times, and the big number of available resources. Again, the reason is that we observe one process in isolation, i.e., we do not see the activities that resources perform for other processes. Hence, utilization of resources is low and there is hardly any queueing due to competition between cases. Since the processing times are negligible compared to the observed waiting times, we add an extra delay of 100% of the observed waiting time (M1). As a result of this intervention, the waiting times are similar to the values from the original log.

In these two case studies, we have generated simulation models from real-life process logs with the method proposed in this paper. We have also investigated the quality of the simulation models by analyzing the generated process logs and comparing the results with the original mining results. The results are very promising as they demonstrate that it is indeed possible to automatically con-

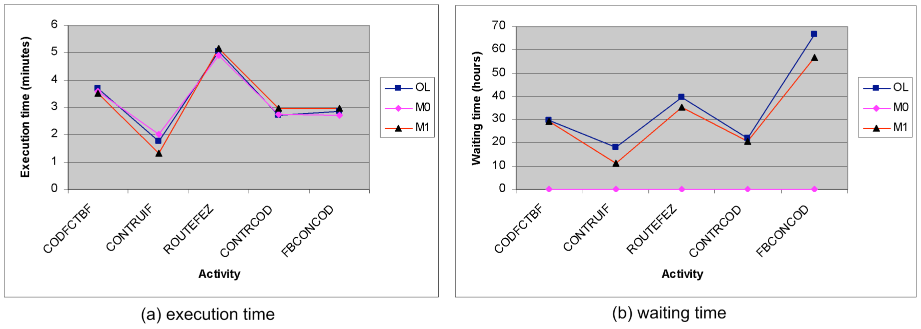


Fig. 20. Performance analysis results for the original log (OL) and the two discovered models (M0 and M1)

struct simulation models based on real-life event logs. Furthermore, with the proper configuration these models could accurately reflect the real situation for all the covered perspectives. However, the results also highlight a number of challenges that must be addressed in the future. They are summarized in the next section.

8 Future Challenges

The goal is to derive a simulation model that reflects the real process as precisely as possible. However, instead of trying to "capture the whole world" (which obviously is an unachievable goal) we must find simplifying but suitable approximations for the desired key characteristics and behaviors. Process mining can help to automatically extract the information that we need for these approximations from log data.

Clearly, the level of detail (e.g., the number of covered perspectives) of a simulation model has an impact on the usefulness of a simulation model. For example, we can approximate the routing behavior at a decision point in the process by probabilities. However, this will not allow us to directly investigate the effect to be expected by an increased proportion of cases with certain characteristics (assuming that these characteristics affect the routing behavior). Similarly, solely modeling the waiting time for an activity is a way to approximate all kinds of delays that prevent the immediate execution of the activity (including the availability of suitable resources). However, with such a coarse approximation we are not able to predict the eventual real effects of an increased case load, or the allocation of more resources to the process. Therefore, a simulation model needs to cover different perspectives, such as control-flow, data, resources, and time.

The organizational perspective is a very important and challenging aspect to be covered in a simulation model. Note that in the case studies people were only working part-time on the processes at hand. As a result, we had to add additional waiting time to the models based on an analysis of the log. However, this is not satisfactory since it does not really capture the way that people actually work (and the high degree of additional waiting time needed to fit the real-life processes shows the relevance of getting more insight into the actual root causes for these delays). Numerous simulation studies, where, e.g., master students model the business processes of organizations, have shown that initial simulation results are typically very optimistic because the resources in the model are too "eager". For example, people work only part-time, or are involved in multiple processes. However, even if we can correctly capture the availability of a person, it does not mean that in reality he or she would start working on a task as soon as it is possible. For example, certain tasks may be continuously delayed due to prioritization issues. As a consequence, we need to find better ways to characterize human behavior without "imitating" the (very complex) reality.

During our experiments we have also identified potential points of improvement for our mining plug-ins. For example, we realized that we need to capture data value range statistics from the execution log not only for numeric but also nominal attributes. This could be easily addressed by recording frequency statistics with respect to the different data values, and by including the corresponding random value generators in the CPN models[12]. Furthermore, it would be good to find ways to extract performance characteristics and decision rules without the need of a (mined) process model as input because this often implies conformance problems (i.e., if not all cases comply with the given process model, we might not be able to use the data from these cases).

Other directions of future work are the automated support of redesign (i.e., suggesting process improvements based on log analysis and simulation of alternatives [28]) and the use of simulation for real-time decision making (i.e., making recommendations for current cases based on a mixture of process mining and short-term simulation [30, 41]).

9 Conclusion

This paper demonstrates that it is possible to automatically construct simulation models based on event logs. These simulation models need to cover different perspectives: control-flow, data, resources, time, etc. Therefore, we showed that each of these perspectives can be discovered using existing process mining techniques and that all these mining results can be merged into a single simulation model.

The approach has been implemented in the context of ProM and has been tested using different examples. In this paper, we used an artificial example and two case studies to evaluate our approach. Based on this we can conclude that it is indeed possible to automatically construct simulation models based on event logs. The discovered models can be exported to CPN Tools which allows for all kinds of simulation. Using the monitoring functionality of CPN it is possible

[12] This and other improvements will be available with the release 5.0 of ProM.

to do all kinds of measurements and to feed the results back to ProM. In this paper, we did not focus on the analysis of the processes of the two municipalities (although extensive simulation studies have been done by students doing their final Master projects in these two municipalities). The reason is that it is obvious that good simulation models have a high practical value. Therefore, we focused on the validation of our approach.

Given the relevance of simulation and the problems people have making good simulation models, we will continue to work on the topics mentioned in this paper. We are exploring various ways to improve the various plug-ins mentioned. Moreover, we would like to improve the modeling of human behavior as indicated in the previous section.

Acknowledgements

This research is supported by the Technology Foundation STW, EIT, SUPER, NWO, and the IOP program of the Dutch Ministry of Economic Affairs. Furthermore, the authors would like to thank all ProM developers for their on-going work on process mining techniques. We would also like to thank Lisa Wells and Kurt Jensen for their support in using CPN Tools.

References

1. W.M.P. van der Aalst, B.F. van Dongen, C.W. Günther, R.S. Mans, A.K. Alves de Medeiros, A. Rozinat, V. Rubin, M. Song, H.M.W. Verbeek, and A.J.M.M. Weijters. ProM 4.0: Comprehensive Support for Real Process Analysis. In J. Kleijn and A. Yakovlev, editors, *Application and Theory of Petri Nets and Other Models of Concurrency (ICATPN 2007)*, volume 4546 of *Lecture Notes in Computer Science*, pages 484–494. Springer-Verlag, Berlin, 2007.
2. W.M.P. van der Aalst, H.A. Reijers, and M. Song. Discovering Social Networks from Event Logs. *Computer Supported Cooperative Work*, 14(6):549–593, 2005.
3. W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering*, 47(2):237–267, 2003.
4. W.M.P. van der Aalst and A.J.M.M. Weijters, editors. *Process Mining*, Special Issue of Computers in Industry, Volume 53, Number 3. Elsevier Science Publishers, Amsterdam, 2004.
5. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
6. R. Agrawal, D. Gunopulos, and F. Leymann. Mining Process Models from Workflow Logs. In *Sixth International Conference on Extending Database Technology*, pages 469–483, 1998.
7. R. Bergenthum, J. Desel, R. Lorenz, and S. Mauser. Process Mining Based on Regions of Languages. In G. Alonso, P. Dadam, and M. Rosemann, editors, *BPM*, volume 4714 of *Lecture Notes in Computer Science*, pages 375–383. Springer, 2007.

8. M. Castellanos, F. Casati, M.C. Shan, and U. Dayal. iBOM: A Platform for Intelligent Business Operation Management. In *ICDE '05: Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*, pages 1084–1095, Washington, DC, USA, 2005. IEEE Computer Society.
9. J.E. Cook and A.L. Wolf. Automating Process Discovery Through Event-Data Analysis. In *International Conference on Software Engineering*, pages 73–82, 1995.
10. J.E. Cook and A.L. Wolf. Discovering Models of Software Processes from Event-Based Data. *ACM Transactions on Software Engineering and Methodology*, 7(3):215–249, 1998.
11. J.E. Cook and A.L. Wolf. Event-Based Detection of Concurrency. In *Proceedings of the Sixth International Symposium on the Foundations of Software Engineering (FSE-6)*, pages 35–45, 1998.
12. J.E. Cook and A.L. Wolf. Software Process Validation: Quantitatively Measuring the Correspondence of a Process to a Model. *ACM Transactions on Software Engineering and Methodology*, 8(2):147–176, 1999.
13. A. Datta. Automating the Discovery of As-Is Business Process Models: Probabilistic and Algorithmic Approaches. *Information Systems Research*, 9(3):275–301, 1998.
14. A.K. Alves de Medeiros. *Genetic Process Mining*. PhD thesis, Eindhoven University of Technology, Eindhoven, 2006.
15. A.K. Alves de Medeiros and C.W. Guenther. Process Mining: Using CPN Tools to Create Test Logs for Mining Algorithms. In K. Jensen, editor, *Proceedings of the Sixth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pages 177–190, 2005.
16. W. Gaaloul and C. Godart. Mining Workflow Recovery from Event Based Logs. In W.M.P. van der Aalst, A.H.M. ter Hofstede, and M. Weske, editors, *International Conference on Business Process Management (BPM 2005)*, pages 169–185, 2005.
17. F. Gottschalk, W.M.P. van der Aalst, M.H. Jansen-Vullers, and H.M.W. Verbeek. Protos2CPN: Using Colored Petri Nets for Configuring and Testing Business Processes. Accepted for the Seventh Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, 2006.
18. D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.-C. Shan. Business Process Intelligence. *Computers in Industry*, 53(3):321–343, 2004.
19. D. Grigori, F. Casati, U. Dayal, and M.C. Shan. Improving Business Process Quality through Exception Understanding, Prediction, and Prevention. In P. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, and R. Snodgrass, editors, *Proceedings of 27th International Conference on Very Large Data Bases (VLDB'01)*, pages 159–168. Morgan Kaufmann, 2001.
20. C.W. Günther and W.M.P. van der Aalst. A Generic Import Framework for Process Event Logs. In J. Eder and S. Dustdar, editors, *Business Process Management Workshops, Workshop on Business Process Intelligence (BPI 2006)*, volume 4103 of *Lecture Notes in Computer Science*, pages 81–92. Springer-Verlag, Berlin, 2006.
21. J. Herbst. A Machine Learning Approach to Workflow Management. In *Proceedings 11th European Conference on Machine Learning*, volume 1810 of *Lecture Notes in Computer Science*, pages 183–194. Springer-Verlag, Berlin, 2000.
22. P.T.G. Hornix. Performance Analysis of Business Processes through Process Mining. Master's thesis, Eindhoven University of Technology, Department of Computer Science, Eindhoven, The Netherlands, 2007.
23. K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use*. Springer-Verlag, 1997.

24. K. Jensen, L.M. Kristensen, and L. Wells. Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems. *International Journal on Software Tools for Technology Transfer*, 9(3-4):213–254, 2007.
25. L.T. Ly, S. Rinderle, P. Dadam, and M. Reichert. Mining Staff Assignment Rules from Event-Based Data. In C. Bussler et al., editor, *Business Process Management 2005 Workshops*, volume 3812 of *Lecture Notes in Computer Science*, pages 177–190. Springer-Verlag, Berlin, 2006.
26. T.M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
27. H.R. Motahari-Nezhad, R. Saint-Paul, B. Benatallah, and F. Casati. Protocol Discovery from Imperfect Service Interaction Logs. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 1405–1409, 2007.
28. M. Netjes, I. Vanderfeesten, and H.A. Reijers. "Intelligent" Tools for Workflow Process Redesign: A Research Agenda. In C. Bussler and A. Haller, editors, *Business Process Management Workshops (BPM 2005)*, volume 3812 of *Lecture Notes in Computer Science*, pages 444–453. Springer-Verlag, Berlin, 2006.
29. J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
30. H.A. Reijers and W.M.P. van der Aalst. Short-Term Simulation: Bridging the Gap between Operational Control and Strategic Decision Making. In M.H. Hamza, editor, *Proceedings of the IASTED International Conference on Modelling and Simulation*, pages 417–421. IASTED/Acta Press, Anaheim, USA, 1999.
31. A. Rozinat and W.M.P. van der Aalst. Decision Mining in Business Processes. BPM Center Report BPM-06-10, BPMcenter.org, 2006.
32. A. Rozinat and W.M.P. van der Aalst. Decision Mining in ProM. In S. Dustdar, J.L. Fiadeiro, and A. Sheth, editors, *BPM 2006*, volume 4102 of *Lecture Notes in Computer Science*, pages 420–425. Springer-Verlag, Berlin, 2006.
33. A. Rozinat and W.M.P. van der Aalst. Conformance Checking of Processes Based on Monitoring Real Behavior. *Information Systems*, 33(1):64–95, 2008.
34. A. Rozinat, R.S. Mans, M. Song, and W.M.P. van der Aalst. Discovering Colored Petri Nets from Event Logs. *International Journal on Software Tools for Technology Transfer (STTT)*, 10(1):57–74, 2008.
35. V. Rubin, C.W. Günther, W.M.P. van der Aalst, E. Kindler, B.F. van Dongen, and W. Schäfer. Process Mining Framework for Software Processes. In Q. Wang, D. Pfahl, and D.M. Raffo, editors, *Software Process Dynamics and Agility, ICSP 2007*, volume 4470 of *LNCS*, pages 169–181, 2007.
36. M. Song and W.M.P. van der Aalst. Towards Comprehensive Support for Organizational Mining. BETA Working Paper Series, WP 211, Eindhoven University of Technology, Eindhoven, 2006.
37. S. Subramaniam, V. Kalogeraki, D. Gunopulos, F. Casati, M. Castellanos, U. Dayal, and M. Sayal. Improving Process Models by Discovering Decision Points. *Information Systems*, 32(7):1037–1055, 2007.
38. A. Vinter Ratzer, L. Wells, H.M. Lassen, M. Laursen, J.F. Qvortrup, M.S. Stissing, M. Westergaard, S. Christensen, and K. Jensen. CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets. In W.M.P. van der Aalst and E. Best, editors, *Applications and Theory of Petri Nets 2003: 24th International Conference, ICATPN 2003*, volume 2679 of *Lecture Notes in Computer Science*, pages 450–462. Springer Verlag, 2003.
39. A.J.M.M. Weijters and W.M.P. van der Aalst. Rediscovering Workflow Models from Event-Based Data using Little Thumb. *Integrated Computer-Aided Engineering*, 10(2):151–162, 2003.
40. I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques, 2nd Edition*. Morgan Kaufmann, 2005.

41. M.T. Wynn, M. Dumas, C.J. Fidge, A.H.M. ter Hofstede, and W.M.P. van der Aalst. Business Process Simulation for Operational Decision Support. In A.H.M. ter Hofstede, B. Benatallah, and H.-Y. Paik, editors, *BPM 2007 Workshops*, volume 4928 of *Lecture Notes in Computer Science*, pages 66–77. Springer-Verlag, 2008.