Available online at www.sciencedirect.com
ScienceDirect
Information Systems 32 (2007) 713–732
www.elsevier.com/locate/infosys

Business process mining: An industrial application

W.M.P. van der Aalst[a,*], H.A. Reijers[a], A.J.M.M. Weijters[a], B.F. van Dongen[a], A.K. Alves de Medeiros[a], M. Song[a,b], H.M.W. Verbeek[a]

[a]*Department of Technology Management, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands*
[b]*Department of Industrial Engineering, Pohang University of Science and Technology, San 31 Hyoja-Dong, Nam-gu, Pohang 790-784, South Korea*

Received 28 July 2005; accepted 31 May 2006
Recommended by F. Carino Jr.

Abstract

Contemporary information systems (e.g., WfM, ERP, CRM, SCM, and B2B systems) record business events in so-called *event logs*. *Business process mining* takes these logs to discover process, control, data, organizational, and social structures. Although many researchers are developing new and more powerful process mining techniques and software vendors are incorporating these in their software, few of the more advanced process mining techniques have been tested on real-life processes. This paper describes the application of process mining in one of the provincial offices of the Dutch National Public Works Department, responsible for the construction and maintenance of the road and water infrastructure. Using a variety of process mining techniques, we analyzed the processing of invoices sent by the various subcontractors and suppliers from three different perspectives: (1) the process perspective, (2) the organizational perspective, and (3) the case perspective. For this purpose, we used some of the tools developed in the context of the ProM framework. The goal of this paper is to demonstrate the applicability of process mining in general and our algorithms and tools in particular.

Keywords: Process mining; Social network analysis; Workflow management; Business process management; Business process analysis; Data mining; Petri nets

1. Introduction

Today, many enterprise information systems store relevant events in some structured form. For example, Workflow Management Systems (WfMSs) typically register the start and completion of activities [1]. ERP systems like SAP log all transactions, e.g., users filling out forms, changing documents, etc. Business-to-business (B2B) systems log the exchange of messages with other parties. Call center packages but also general-purpose CRM systems log interactions with customers. These examples show that many systems have some kind of *event log* often referred to as "history", "audit trail", "transaction log", etc [2–5]. The event log typically contains information about events referring to an *activity* and a *case*. The case (also named *process instance*) is the "thing" which is being handled, e.g., a customer order, a job application, an insurance claim, a building permit, etc. The

*Corresponding author. Tel.: +31 40 2474295.
E-mail address: w.m.p.v.d.aalst@tm.tue.nl (W.M.P. van der Aalst).

0306-4379/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.is.2006.05.003

activity (also named *task*, *operation*, *action*, or *work-item*) is some operation on the case. Typically, events have a *timestamp* indicating the time of occurrence. Moreover, when people are involved, event logs will characteristically contain information on the person executing or initiating the event, i.e., the *performer*.

Besides the availability of event logs there is an increased interest in monitoring business processes. On the one hand, new legislation such as the Sarbanes–Oxley (SOX) Act [6] and increased emphasis on corporate governance are forcing organizations to follow their business activities more closely [7]. On the other hand, there is a constant pressure to improve the performance and efficiency of business processes. This requires more fine-grained monitoring facilities as is illustrated by today's buzzwords such as business activity monitoring (BAM), business operations management (BOM), and business process intelligence (BPI). However, the functionality offered by tools such as Cognos and BusinessObjects is limited to simple performance indicators such as flow time and utilization. Unfortunately, most of these systems do not focus on causal and dynamic dependencies in processes and organizations. One of the few commercial software tools adopting a more process-oriented view on monitoring is the ARIS Process Performance Monitor (ARIS PPM) [8].

Business process mining, or *process mining* for short, aims at the automatic construction of models explaining the behavior observed in the event log. For example, based on some event log, one can construct a process model expressed in terms of a Petri net. Over the last couple of years many tools and techniques for process mining have been developed [2,3,5,8–15]. Although process mining is very promising, most of the techniques make assumptions which do not hold in practical situations. For example, some techniques assume that there is no noise and have difficulties dealing with exceptions. Other approaches are limited to processes having a particular structure. Therefore, it is important to confront existing tools and techniques with event logs taken from real-life applications. In this paper we describe a case study based on a log of the *process of handling invoices in a provincial office of the Dutch National Public Works Department*. This office is one of 12 offices, employing about 1000 civil servants. The office is responsible for the construction and maintenance of the road and water infrastructure in its province, and in order to do this it subcontracts work to various parties such as road construction companies, cleaning companies, and environmental bureaus. Also, the provincial office purchases services and products to support its construction, maintenance, and administrative activities. We have used an event log containing information on more than 14,000 invoices as a starting point for mining the process perspective (How?), the organizational perspective (Who?), and the case perspective (What?). (These perspectives are presented in detail in Section 3.) The results are reported in this paper and demonstrate the applicability of our tools in an industrial setting.

The remainder of this paper is organized as follows. Section 2 briefly discusses related work. Section 3 introduces the concept of business process mining and Section 4 discusses the ProM framework used for the case study. Section 5 describes the case study. Sections 6–8 present the results of mining the process, organizational, and case perspective of the invoice handling process. Section 9 reflects on these results and concludes the paper.

2. Related work

In our case study we analyze the event log generated by the WfMS of the organization involved. Clearly, most of the workflow literature [1,16,17] has been focusing on modeling, verification, simulation, and enactment rather than process mining. The idea of applying process mining in the context of workflow management was first introduced in [3]. Cook and Wolf [12] have investigated similar issues in the context of software engineering processes using different approaches. Herbst [13] and Karagiannis also address the issue of process mining in the context of workflow management using an inductive approach. They use stochastic task graphs as an intermediate representation and generate a workflow model described in the ADONIS modeling language. The α algorithm [11] can be seen as the first algorithm to truly capture concurrency in business processes. This algorithm was proved to be correct for a large class of processes [11], but like most other techniques it has problems in dealing with noise and incompleteness. Therefore, we developed the heuristic approach used in this paper [18,15].

The focus of this paper is not limited to the control-flow perspective. In the case study we also investigate the organizational perspective. This work uses social network analysis (SNA) [19,20]

techniques and tools. SNA can be seen as part of sociometry, the roots of which can be found in the early work of Moreno [21]. Although SNA has been around for a long time, recently its application increased because of the availability and widespread use of electronic communication and information facilities. For example, several studies have generated sociograms from email logs [22–26] to analyze the communication structure inside or between organizations. Such studies have resulted in the identification of relevant, recurrent aspects of interaction in organizational contexts [27,24]. However, these studies are unable to relate the derived social networks to a particular business process, as the analyzed data does not reveal to what activity or case it applies. This paper builds on the results presented in [28,10] where SNA is related to process mining.

As indicated in the Introduction, business process mining can be seen in the broader context of BPI and BAM. In [4,5,29] a BPI toolset on top of HP's process manager is described. The BPI toolset includes a so-called "BPI process mining engine". In [14] zur Mühlen describes the PISA tool which can be used to extract performance metrics from workflow logs. Similar diagnostics are provided by the ARIS Process Performance Manager (PPM) [8]. The latter tool is commercially available and a customized version of PPM is the Staffware Process Monitor (SPM) [30] which is tailored towards mining Staffware logs.

For more information on process mining we refer to a special issue of Computers in Industry on process mining [31] and a survey paper [2]. Given the scope of this paper, we are unable to provide a complete listing of the many papers on process mining published in recent years. However, in the next section we give a brief overview of the field.

3. Business process mining: an overview

The goal of process mining is to extract information about processes from transaction logs [2]. We assume that it is possible to record events such that (i) each event refers to an *activity* (i.e., a well-defined step in the process), (ii) each event refers to a *case* (i.e., a process instance), (iii) each event can have a *performer* also referred to as *originator* (the person executing or initiating the activity), and (iv) events have a *timestamp* and are totally ordered. Table 1 shows an example of a log involving 19 events, five activities, and six originators. In addition to the information shown in this table, some event logs contain more information on the case itself, i.e., data elements referring to properties of the case. For example, the case handling system FLOWer [32] logs every modification of every data element.

Table 1
An event log

Case id	Activity id	Originator	Timestamp
Case 1	Activity *A*	John	9-3-2004:15.01
Case 2	Activity *A*	John	9-3-2004:15.12
Case 3	Activity *A*	Sue	9-3-2004:16.03
Case 3	Activity *B*	Carol	9-3-2004:16.07
Case 1	Activity *B*	Mike	9-3-2004:18.25
Case 1	Activity *C*	John	10-3-2004:9.23
Case 2	Activity *C*	Mike	10-3-2004:10.34
Case 4	Activity *A*	Sue	10-3-2004:10.35
Case 2	Activity *B*	John	10-3-2004:12.34
Case 2	Activity *D*	Pete	10-3-2004:12.50
Case 5	Activity *A*	Sue	10-3-2004:13.05
Case 4	Activity *C*	Carol	11-3-2004:10.12
Case 1	Activity *D*	Pete	11-3-2004:10.14
Case 3	Activity *C*	Sue	11-3-2004:10.44
Case 3	Activity *D*	Pete	11-3-2004:11.03
Case 4	Activity *B*	Sue	14-3-2004:11.18
Case 5	Activity *E*	Clare	17-3-2004:12.22
Case 5	Activity *D*	Clare	18-3-2004:14.34
Case 4	Activity *D*	Pete	19-3-2004:15.56

Event logs such as the one shown in Table 1 are used as the starting point for mining. We distinguish three different perspectives: (1) the process perspective ("How?"), (2) the organizational perspective ("Who?"), and (3) the case perspective ("What?").

The *process perspective* focuses on the control flow, i.e., the ordering of activities. The goal of mining this perspective is to find a good characterization of all possible paths, e.g., expressed in terms of a Petri net [33] or event-driven process chain (EPC) [34].

The *organizational perspective* focuses on the originator field, i.e., which performers are involved and how are they related. The goal is to either structure the organization by classifying people in terms of roles and organizational units or to show relations between individual performers (i.e., build a social network [19–21,35]).

The *case perspective* focuses on properties of cases. Cases can be characterized by their path in the process or by the originators working on a case. However, cases can also be characterized by the values of the corresponding data elements. For example, if a case represents a replenishment order,

it may be interesting to know the supplier or the number of products ordered.

To illustrate the first two perspectives consider Fig. 1. The log shown in Table 1 contains information about five cases (i.e., process instances). The log shows that for four cases (1–4) the activities *A*–*D* have been executed. For the fifth case only three activities are executed: activities *A*, *E*, and *D*. Each case starts with the execution of *A* and ends with the execution of *D*. If activity *B* is executed, then also activity *C* is executed. However, for some cases activity *C* is executed before activity *B*. Based on the information shown in Table 1 and by making some assumptions about the completeness of the log (i.e., assuming that the cases are representative and a sufficient large subset of possible behaviors is observed), we can deduce the process model shown in Fig. 1(a). The model is represented in terms of a Petri net [33]. The Petri net starts with activity *A* and finishes with activity *D*. These activities are represented by transitions. After executing *A* there is a choice between either executing *B* and *C* in parallel or just executing activity *E*. Note that for this example we assume that two activities are in parallel if they appear in any order. By distinguishing between start events and complete events for activities it is possible to explicitly detect true parallelism, i.e., concurrent execution of tasks.

Fig. 1(a) does not show any information about the organization, i.e., it does not use any information on the people executing activities. However, Table 1 shows information about the performers. For example, we can deduce that activity *A* is executed by either John or Sue, activity *B* is executed by John, Sue, Mike, or Carol, *C* is executed by John, Sue, Mike, or Carol, *D* is executed by Pete or Clare, and *E* is executed by Clare. We could indicate this information in Fig. 1(a). The information could also be used to "guess" or "discover" organizational structures. For example, a guess could be that there are three roles: *X*, *Y*, and *Z*. For the execution of *A* role *X* is required and John and Sue have this role. For the execution of *B* and *C* role *Y* is required and John, Sue, Mike, and Carol have this role. For the execution of *D* and *E* role *Z* is required and Pete and Clare have this role. For five cases these choices may seem arbitrary but for larger data sets such inferences capture the dominant roles in an organization. The resulting "activity–role–performer diagram" is shown in Fig. 1(b). The three "discovered" roles link activities to performers. Fig. 1(c) shows another view on the

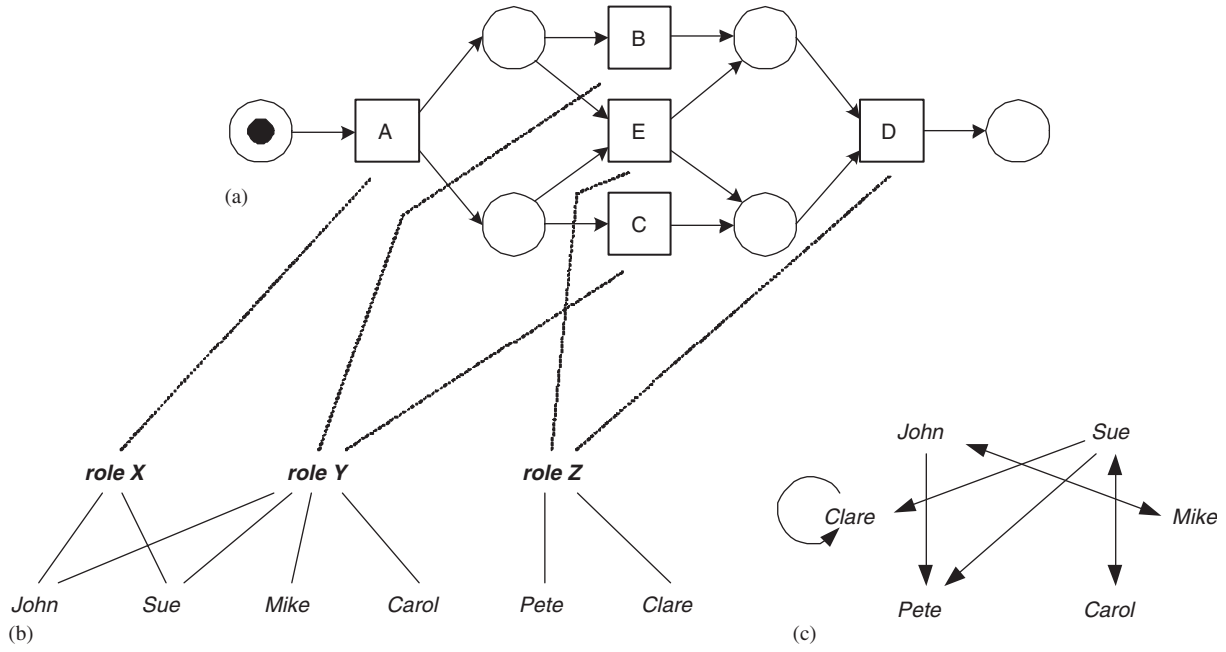

Fig. 1. Some mining results for the process perspective (a) and organizational (b and c) perspective based on the event log shown in Table 1: (a) the control-flow structure expressed in terms of a Petri net, (b) the organizational structure expressed in terms of an activity–role–performer diagram, and (c) a sociogram based on transfer of work.

organization based on the transfer of work from one individual to another, i.e., not focusing on the relation between the process and individuals but on relations among individuals (or groups of individuals). Consider, for example, Table 1. Although Carol and Mike can execute the same activities (*B* and *C*), Mike is always working with John (cases 1 and 2) and Carol is always working with Sue (cases 3 and 4). Probably Carol and Mike have the same role but based on the small sample shown in Table 1 it seems that John is not working with Carol and Sue is not working with Mike.[1] These examples show that the event log can be used to derive relations between performers of activities, thus resulting in a sociogram. For example, it is possible to generate a sociogram based on the transfers of work from one individual to another as is shown in Fig. 1(c). Each node represents one of the six performers and each arc represents that there has been a transfer of work from one individual to another. There is a "transfer of work from *A* to *B*" if, for the same case, an activity executed by *A* is directly followed by an activity executed by *B*. For example, both in cases 1 and 2 there is a transfer from John to Mike. Fig. 1(c) does not show frequencies. However, for analysis purposes these frequencies can be added. The arc from John to Mike would then have weight 2. (Typically, we do not use absolute frequencies but weighted frequencies to get relative values between 0 and 1.) Fig. 1(c) shows that work is transferred to Pete but not vice versa. Mike only interacts with John and Carol only interacts with Sue. Clare is the only person transferring work to herself.

Besides the "How?" and "Who?" question (i.e., the process and organization perspectives), there is the case perspective that is concerned with the "What?" question. Fig. 1 does not address this. In fact, focusing on the case perspective is most interesting when also data elements are logged but these are not listed in Table 1. The case perspective looks at the case as a whole and tries to establish relations between the various properties of a case. Note that some of the properties may refer to the activities being executed, the performers working on the case, and the values of various data elements linked to the case. Using clustering algorithms it would, for example, be possible to show a positive correlation between the size of an order or its handling time and the involvement of specific people.

Orthogonal to the three perspectives (process, organization, and case), the result of a mining effort may refer to *logical* issues and/or *performance* issues. For example, process mining can focus on the logical structure of the process model (e.g., the Petri net shown in Fig. 1(a)) or on performance issues such as flow time. For mining the organizational perspectives, the emphasis can be on the roles or the social network (cf. Fig. 1(b) and (c)) or on the utilization of performers or execution frequencies.

4. ProM: a process mining framework

As indicated in the Introduction the functionality of commercial tools is typically limited to measuring and analyzing performance indicators such as flow times, failure rates, and frequencies. As shown in the previous section, process mining involves the construction of models and is not limited to simple metrics. Although buzzwords such as BAM, BOM, and BPI suggest differently, commercial systems are typically unable to discover non-trivial models. In the last five years, several mining tools have been developed at Eindhoven University of Technology, e.g., EMiT [9], Thumb [15], and MiSoN [10]. These tools refer to different perspectives and use different mining techniques. However, they work on the same type of event logs and may create similar types of models. Therefore, these tools have been integrated in the ProM framework [36].

The ProM framework has been developed as a completely plug-able environment. It can be extended by simply adding plug-ins, i.e., there is no need to know or recompile the source code. Currently, more than 90 plug-ins have been added. The most interesting plug-ins are the mining plug-ins and the analysis plug-ins. The architecture of ProM allows for five different types of plug-ins:

Mining plug-ins which implement some mining algorithm, e.g., mining algorithms that construct a Petri net based on some event log.

Export plug-ins which implement some "save as" functionality for some objects (such as graphs). For example, there are plug-ins to save EPCs, Petri nets, spreadsheets, etc.

Import plug-ins which implement an "open" functionality for exported objects, e.g., load instance EPCs from ARIS PPM.

[1] Clearly, the number of events in Table 1 is too small to establish these assumptions accurately. However, for the sake of argument we assume that the things that did not happen will never happen.

Analysis plug-ins which typically implement some property analysis on some mining result. For example, for Petri nets there is a plug-in which constructs place invariants, transition invariants, and a coverability graph.

Conversion plug-ins which implement conversions between different data formats, e.g., from EPCs to Petri nets.

Earlier tools such as EMiT [9], Thumb [15], and MiSoN [10] have been refactored as plug-ins in the ProM framework. Fig. 2 shows a screenshot of ProM. Note that one plug-in shows the discovered process model in terms of a Petri net. This Petri net is identical to the one shown in Fig. 1(a). The other plug-in shows the sociogram also depicted in Fig. 1(c). Both models have been constructed automatically from the event log shown in Table 1.

The ProM*import* can be used to import event logs from various systems (e.g., Staffware and FLOWer) such that they can be analyzed using ProM. ProM uses a standard XML format, named *MXML* [37]. In our case study the proprietary format of the WfMS used by provincial office was mapped onto the XML format. Therefore, we discuss the format in some more detail. Understanding the format is also important for understanding the applicability of ProM.

Fig. 3 illustrates the standard XML format. The *Source* element contains the information about software or system that was used to record the log. The *Process* element represents one process holding multiple cases. The *ProcessInstance* elements correspond to cases. One *ProcessInstance* element may hold multiple *AuditTrailEntry* elements. Each of these elements represents an event, i.e., one line in a table like Table 1. Each *AuditTrailEntry* element may contain *WorkflowModelElement*, *EventType*, *Timestamp*, and *Originator* elements. The *WorkflowModelElement* and *EventType* are mandatory elements as shown in Fig. 3. The *WorkflowModelElement* element refers to an activity, a subprocess, or some other routing element in the process model. The *EventType* element can be used to record the type of event (e.g., the start or completion of an activity or some exceptional behavior like the cancellation of a case). Table 1 does not show any event types. However, one can always use the default event type *complete*. The *Timestamp* element can be used to record the time of occurrence. The *Originator* element refers to the performer, e.g., the person executing the corresponding activity. To make the format more expressive, we define *Data* element that can be used at various levels (i.e., *WorkflowLog*, *Process*, *ProcessInstance*, and *AuditTrailEntry* level). If users want to specify additional information, this can be recorded using the *Data* element (e.g., data elements linked to cases).

In the subsequent sections we will provide more information on the mining tools we are using in this case study but first we provide more information on the case study itself.

5. The Public Works Department

The industrial application in this paper involves one of the 12 provincial offices of the Dutch

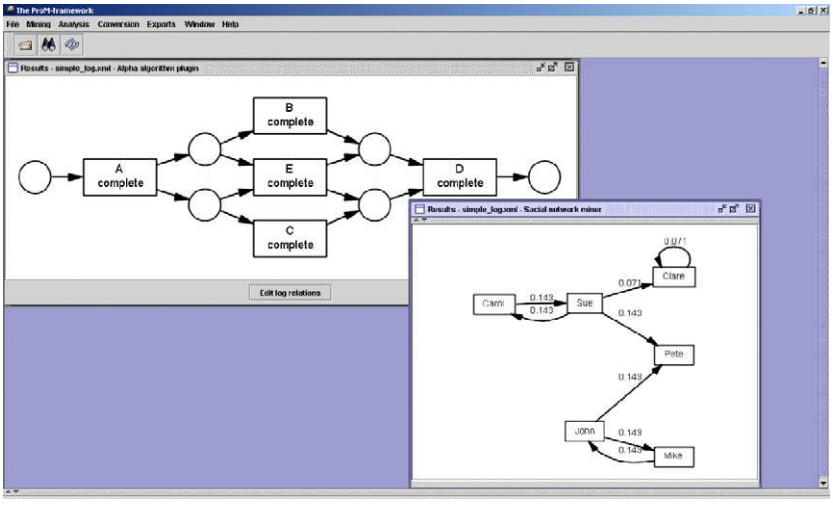

Fig. 2. Screenshot of ProM showing two plug-ins applied to the event log shown in Table 1.

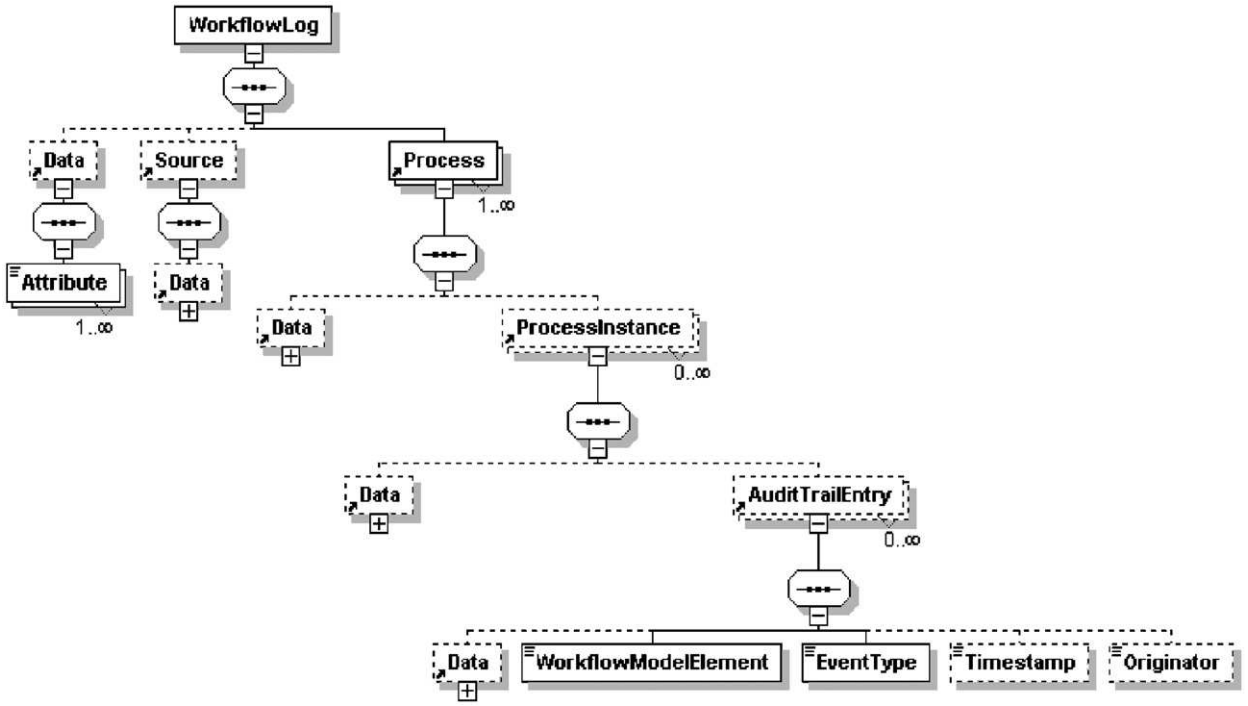

Fig. 3. The MXML format for process mining (XML schema).

National Public Works Department. In The Netherlands this department is referred to as ''Rijkswaterstaat'', abbreviated as ''RWS''. Like all other RWS offices, the particular office studied here is primarily responsible for the construction and maintenance of the road and water infrastructure in its province. About 1000 civil servants work for this office. To perform its functions, the RWS office subcontracts various parties such as road construction companies, cleaning companies, and environmental bureaus. Also, it purchases services and products to support its construction and maintenance activities on the one hand (e.g., mechanical tools, fuel, and rasters) and its administrative activities on the other (e.g., office supplies). In 2001, the RWS office processed 20,000 invoices from various subcontractors and suppliers.

Before 2001, the 12 provincial offices maintained 18 different process versions to handle the various invoices. This diversity made it difficult and time consuming to update all local payment processes to adhere to changing national regulations. In addition, the performance of all these different process versions varied. An important performance indicator for the processing of invoices is the timeliness of payment. For legitimate invoices hold that payment should take place within 31 days from the moment the invoice was received. After this period, the creditor is entitled (on the basis of Dutch law) to receive interest over the outstanding sum. Clearly, a slack payment attitude negatively influences the financial position of an RWS office. For the office studied in this particular case, it became clear that the norms for payment timeliness were not met (see Table 2). As shown, the norm states that 90% or more of the invoices must be paid before the appointed time of 31 days, a maximum of 5% should be paid within 31–62 days, and again up to a maximum of 5% may be paid after 62 days. In the second column, the actual performance of the RWS office in question is given.

Table 2
The time until invoices are paid (i.e., norms and actual performance before the implementation of a workflow management system)

Payment duration (days)	Norm (%)	Actual (%)|
0–31	90	70
32–62	5	22
63	5	8

In response to these issues, the national RWS management decided to unify invoice processing

across the various provincial offices in search of efficiency gains. The momentum of change was seized to develop a proprietary WfMS to support and further optimize the processing of invoices. One of the main promises of workflow technology is that it supposedly speeds up the processing, by liberating workers from routine work they need for coordination and by handing out work to exactly the right resources at the right time [38]. In 2002, the WfMS was implemented at the RWS office involved in our case study.

The contact with the RWS office was established in 2001, when Eindhoven University of Technology in a joint effort with Deloitte management consultancy initiated a longitudinal study into the effectiveness of WfMSs. The aim of the study—which is still running—is to quantify the contribution of workflow technology to improved business process performance with respect to lead time, wait time, service time, and utilization of resources. More information on this study including its preliminary results can be found in [39].

The RWS office expressed its interest in the mentioned study and participated as one of the 10 Dutch organizations where workflow management effectiveness would be measured. During the years 2001 and 2002, information was gathered for comparison purposes on the performance of the invoice processing, both before and after the implementation of the WfMs. The data which is analyzed and mined in this paper involves the situation *after* which the WfMs was implemented. This data was gathered after the system had been running for a number of months (to avoid any startup effects). The management of the provincial office supported this analysis and was interested to see how mining techniques could contribute to a better understanding of the performance of the process and perhaps identify opportunities for improvement. Therefore, RWS provided us with the logs of the invoice payment process.

The invoice process analyzed in this paper consists of 17 real activities, aside from logistic steps and splits. The RWS event log (or "RWS log" for short) contains 14,279 cases. The total number of logged events is 147,579 and 487 employees participated in the process execution. Fig. 4 shows a snippet of the RWS log. The left-hand side shows the native format of the WfMs used by the RWS office. The right-hand side shows the same log in the MXML format described in Section 4.

6. Mining the process perspective

As indicated before, we will analyze the RWS log from three perspectives. In this section, we focus on the *process perspective*, also known as the *control-flow* perspective. Before presenting the results of applying process mining in the case study, we first need to tell more about the particular process mining technique being used.

For the process perspective, we only consider the case and activity attributes of an event log, e.g., in Table 1 we only need to consider the first two

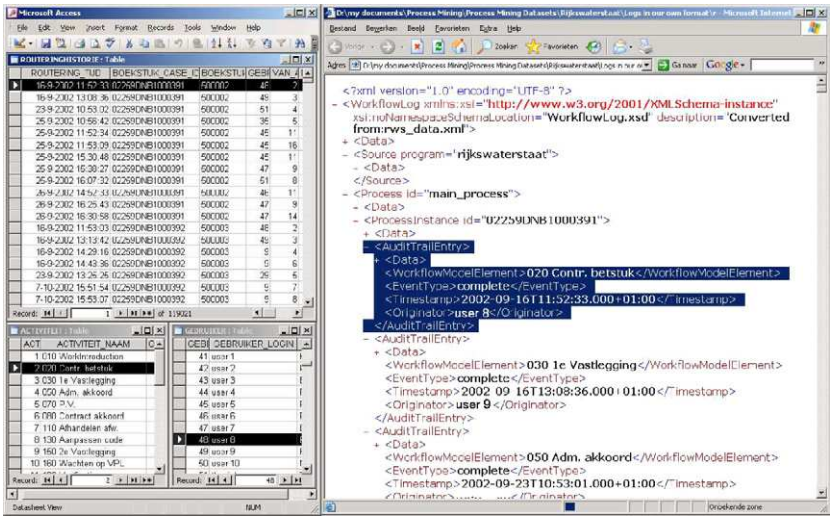

Fig. 4. A snippet of the RWS log in its native format (left) and the MXML format (right).

columns. To construct a process model like in Fig. 1(a), we need to be able to discover causal dependencies and decide on the types of splits and joins. $A \rightarrow_W B$ denotes the casual dependency between activities A and B, i.e., A is (directly) followed by B but B is not (directly) followed by A. This is indeed the case in Table 1. Moreover, $A \rightarrow_W C$, $A \rightarrow_W E$, $B \rightarrow_W D$, $C \rightarrow_W D$, and $E \rightarrow_W D$. For this simple example it is easy to discover the causal dependencies, however, for more realistic logs (such as the RWS log) there are two complicating factors:

Completeness: For larger or more complicated processes the log will typically not contain all possible routes. Consider 10 activities which can be executed in parallel. The total number of interleavings is $10! = 3,628,800$. It is not realistic that each interleaving is present in the log. Moreover, certain paths through the process model may have a low probability and therefore remain undetected. As a result the log is not complete in the sense that it does not capture all possible behavior.

Noise: Parts of the log may be incorrect, incomplete, or refer to exceptions. Events can be logged incorrectly because of human or technical errors. Events can be missing in the log if some of the activities are manual or handled by another system/organizational unit. Events can also refer to rare or undesired events. Clearly, exceptions which are recorded only once should not automatically become part of the regular process model.

To tackle these problems we have chosen to use the *heuristic approach* described in [15,18]. This approach is relatively robust (i.e., it can deal with noise and incompleteness) and has options to focus on the main process instead of trying to model the full details of the behavior reported in the event log. For a better understanding of the approach we shortly discuss the ideas to discover causal dependencies in the presence of noise.

We use a frequency-based metric to indicate how certain we are that there is a dependency relation between two activities A and B (notation $A \Rightarrow_W B$). The basic idea is that if activity A is often directly followed by activity B, but the opposite (B directly followed by A) never occurs, then there is a high probability that there is a dependency relation between A and B. Below, we first define the $\Rightarrow_W$ metric. After that we will illustrate how we can use this metric in a simple heuristic in which we search for reliable dependency relations (the $A \rightarrow_W B$ relation).

Let T be a set of activities, W be an event log over T, and $a, b \in T$:

- $|a >_W b|$ is the number of times $a >_W b$ occurs in W (i.e., the number of times event a is directly followed by event b),
- $a \Rightarrow_W b = \left(\frac{|a >_W b| - |b >_W a|}{|a >_W b| + |b >_W a| + 1} \right)$.

First, note that the value of $a \Rightarrow_W b$ is always between -1 and 1. Some simple examples demonstrate the rationale behind this definition. If we use this definition in the situation that, in five cases, activity A is directly followed by activity B but the other way around never occurs, the value of $A \Rightarrow_W B = \frac{5}{6} = 0.833$ indicating that we are not completely sure of the dependency relation. After all, there are only five observations and these may correspond to noise. However, if there are 50 cases in the event log in which A is directly followed by B but the other way around never occurs, the value of $A \Rightarrow_W B = \frac{50}{51} = 0.980$ indicates that we are more confident about the causality relation. If there are 50 traces in which activity A is directly followed by B and noise caused B to follow A once, the value of $A \Rightarrow_W B$ is $\frac{49}{52} = 0.94$ indicating that we are pretty sure of a causal relation.

A high $A \Rightarrow_W B$ value strongly suggests that there is a causal relation between activities A and B. But what is a high value? What would be a good threshold to take the decision that B truly depends on A (i.e., $A \rightarrow_W B$ holds)? Such a threshold appears sensitive for the amount of noise, the degree of concurrency in the underlying process, and the frequency of the involved activities.

On closer inspection, it appears unnecessary to use a threshold value. After all, we know that each non-initial activity must have at least one other activity that is its cause, and each non-final activity must have at least one dependent activity. Using this information in a heuristic approach we can limit the search and take *the best* candidate (with the highest $A \Rightarrow_W B$ score).[2] This simple heuristic helps us enormously in finding reliable causality relations even if the event log contains noise.

[2]Or the best candidate plus all candidates with an $A \Rightarrow_W B$ score close (default 95%) to the value of the best candidate.

Although the heuristic formulated above is not complete and has to be extended to recognize (i) recursion, (ii) short loops, and (iii) the type of joins and splits (i.e., AND or XOR) we can now understand the meaning of a statement like "$A \Rightarrow_W B = 0.98$" (i.e., the dependency value between activity A and B calculated equals 0.98).

Applying the approach described above (with default parameter settings) to the RWS log results in the dependency graph of Fig. 5. Each node in the dependency graph represents an activity. Note that the activity names are in Dutch. Since these are the names that appear in the log, we cannot change them without changing the entire log. (Fig. 5 is generated automatically on the basis of the event log containing 147,579 events.) The arcs in the graph represent the causal dependencies.

Activities *bBb* and *eEe* in Fig. 5 are artificially added *begin* and *end* activities. Adding the extra end activity *eEe* makes it clear that not all instances end with the intended end activity 220_Afsluiten; also, activities 030_1e_Vastlegging, 070_PV, and 050_Adm_akkoord appear as end activities. The number in the activity box indicates the frequency of that activity (e.g., the frequency of the first real activity 020_Contr_betstuk is 14,280). The string 1.000|14029| close to the arrow from activity 020_Contr_betstuk to activity 030_1e_Vastlegging means that in the RWS log there are 14,029 registrations of activity 020_Contr_betstuk directly followed by activity 030_1e_Vastlegging; for this reason the calculated dependency value between these two activities is very high (i.e., 1.000).

A close observation of Fig. 5 reveals that there are many loops in the model, specifically around the activity 170_Parkeer. The way this activity is connected with other activities indicates the special status of this activity; after some discussion with the owners of the process it appears that 170_Parkeer is not really an activity but a way to suspend the processing of the case temporarily. With respect to the other loops, many of these reflect how cases are at times classified incorrectly at the start of the process, which then after some time requires a reevaluation of the case. For example, the case is routed to a department supposedly managing the involved contract. Furthermore, this process involves a number of checks (e.g., activity 180_Verificatie, 080_Contract_akkoord) that lead to reiterations in case the quality of processing is not satisfactory.

After discussing the various issues with the process owners of the RWS the decision was taken to concentrate on the main process and to ignore the suspension facility 170_Parkeer and all low-frequent activities (i.e., activities with a frequency below 1% of the total number of events (147,579)). This results in ignoring the following six activities: 190_Wachten_op_PV (frequency = 17), 200_Wachten_op_CF (129), 160_Wachten_op_VPL

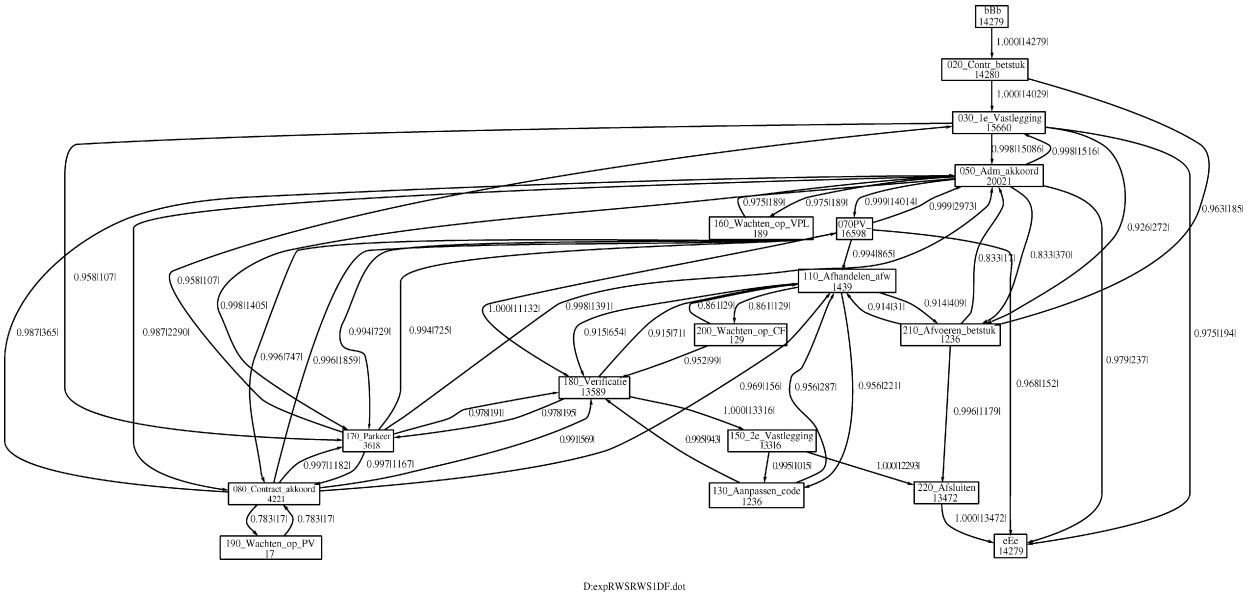

Fig. 5. The resulting dependency graph of applying the mining algorithm with default parameter settings.

(189), 210_Afvoeren_betstuk (1226), 130_Aanpassen_code (1236), and 110_Afhandelen_afw (1439). The dependency graph resulting from this abstraction is presented in Fig. 6.

Fig. 6 gives a good impression of the main flow (the vertical axis of the graph) in the RWS process. The relatively high values between vertical bars (i.e., the "direct followed by" measure) indicate that the process has a strong sequential character with some alternative paths and some loops. During the mining of the process also the type of each split and join is discovered. For instance, the first split (i.e., after 020_Contr_betstuk) is an XOR-split and the last join (i.e., before 220_Afsluiten) is an XOR-join. This information is not shown in Fig. 6 but is present and can be used to generate a process model in terms of a Petri net or EPC. Using the information on splits and joins we can easily check the quality of the model by trying to parse the material in the log. An incomplete model (i.e., a model with missing causal dependencies) or errors in the type of splits and joins will result in parsing errors. The mined model partly presented in Fig. 6 (the AND/XOR information is not presented in the figure) is able to parse 13,465 of the 14,279 cases completely correct. With 814 cases there are parsing problems (i.e., 812 of them leave some enabled activities and in 101 cases there are activities that cannot be parsed). Note that the total sums up to a higher value than 814 because a case can have more than one parsing problem.

From the perspective of RWS, the mining analysis delivered a highly informative process model. In comparison with the predefined process model that the WfMS uses to operate, the mined model clearly shows the same main flow of invoices being handled. In this way, infrequently executed activities can be left out for a better understanding

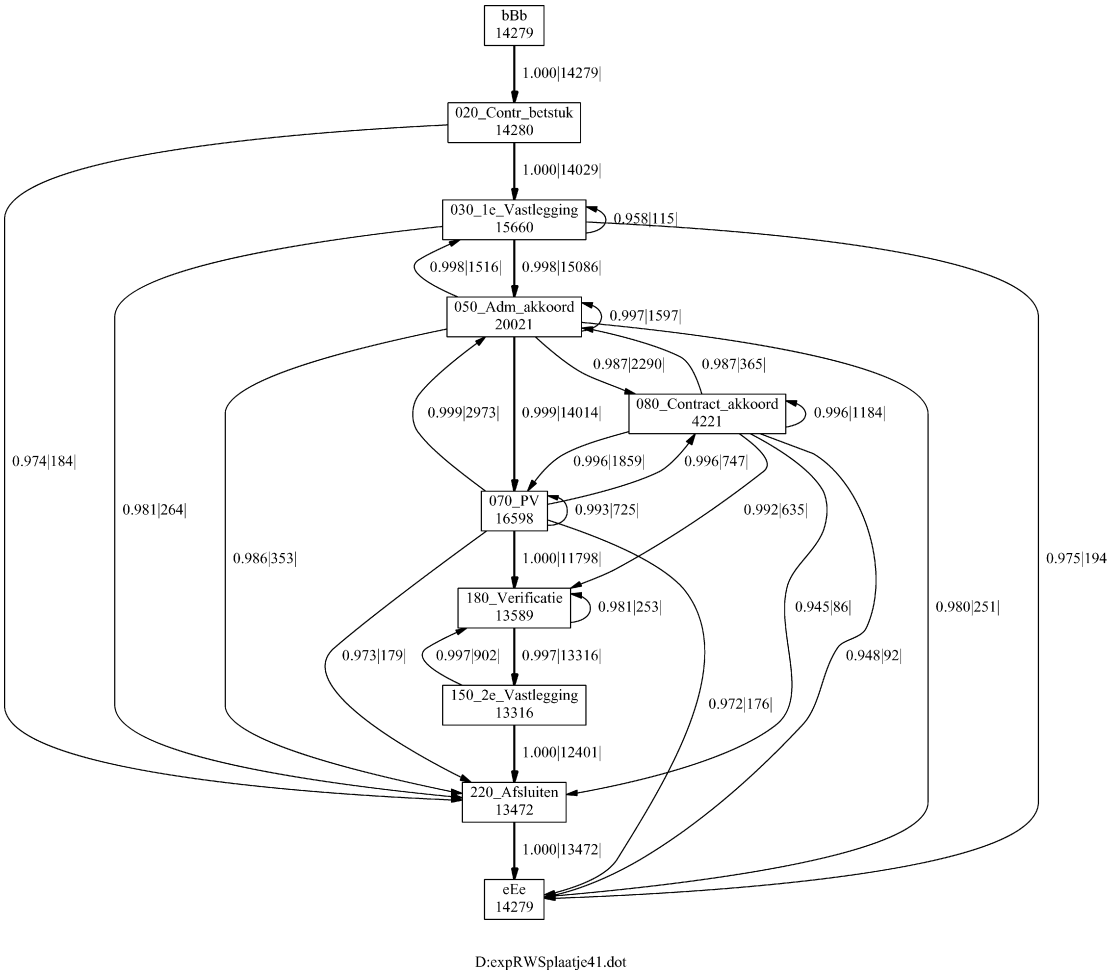

Fig. 6. The resulting dependency graph after ignoring the artificial activity 170_Parkeer and six other low-frequent activities.

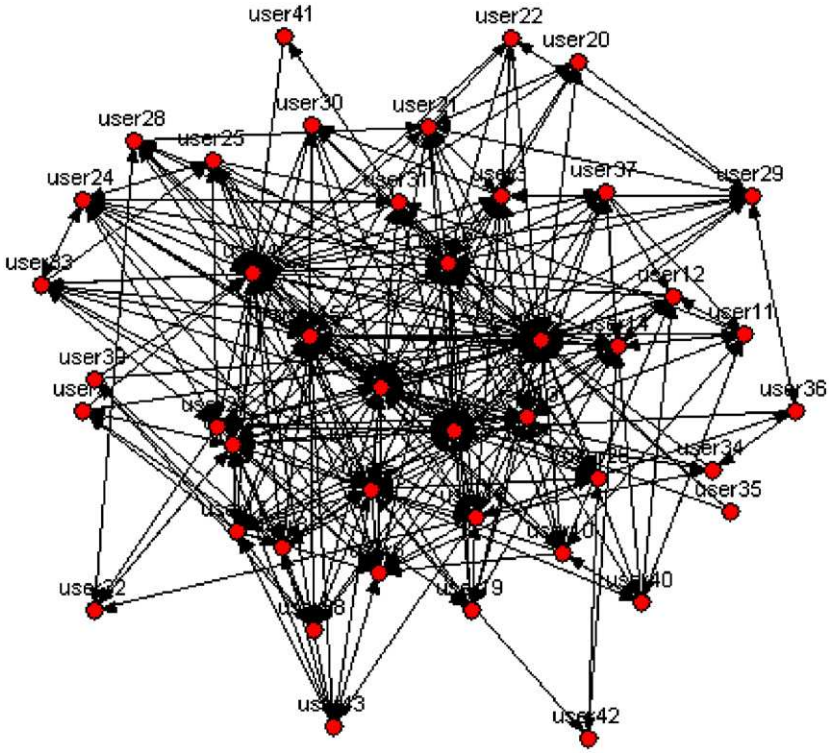

Fig. 7. Social network based on the handover of work metric.

of the process. Furthermore, the mined model indicates that many cases will go through loops during their life-cycle. This in particular cannot be deduced from the predefined process model which, by its very nature, lacks information on actual behavior.

In the remainder, we no longer focus on the process perspective and direct our attention towards the organizational perspective and the case perspective. We show how we used additional mining analyses to provide more insight into the nature of the loops shown in Fig. 6 and their effect on the performance of the process.

7. Mining the organizational perspective

In this section, we examine the *organizational perspective*. In other words, we focus on "who" performs the different steps and how performers are related. One of the basic ideas is that relationships between workers may be derived from the frequency of passing a case from one performer to another. For analyzing such relationships, the tool MiSoN has been developed [10]. The functionality of MiSoN has been embedded in the ProM framework (see Fig. 2 for a screenshot). Based on the event logs extracted from these systems, our tool constructs sociograms that can be used as a starting point for SNA [19,20]. The derived relationships can be exported in a matrix format and used by most SNA tools, such as AGNA and NetMiner. With an SNA tool, several techniques can be applied to analyze social networks, e.g., find interaction patterns, evaluate the role of an individual in an organization, etc.

When we consider the RWS log, an obvious way to start is to look for direct handovers of work within cases between performers (see the discussion of Fig. 1(c) in Section 3). From an analysis of the RWS log, we can derive a social network as shown in Fig. 7. A directed arc between *user32* and *user28*, for example, represents that on some occasion a case was handed over from *user32* to *user28*.[3] It can easily be verified that the presented network contains no isolated nodes.

[3]Note that, throughout the paper, the real user names are changed into anonymous identifiers like *userXX* to ensure confidentiality and privacy.

Note that the represented network contains only 43 nodes, all representing human users, while the original RWS log contains 487 performers. For the purpose of clarity, we have decided to select the users that are responsible for the core of the process. The 43 users as shown take care of 15 out of 17 activities. The activities not considered in this initial analysis are 070_PV and 170_Parkeer. The former of the two is only performed by project leaders, whose only responsibility in the project is to approve project-related invoices. This is not a major part of their regular work, while the inclusion of these dozens of project leaders would make the diagram unreadable. The latter activity, 170_Parkeer, is not a real activity, as we explained in the previous section.

Some SNA diagnostics look at the social network as a whole while others focus on a single node (i.e., performer). For example, if all other individuals are in short distance to a given node and all geodesic paths (i.e., the shorted paths in the graph between two nodes) visit this node, clearly the node is very central—like a spider in the web. There are different metrics for this intuitive notion of *centrality*, such as betweenness, in and out closeness, and power [40] of each node. Table 3 shows both the top and bottom ranked performers based on the *betweenness* metric (note that the bottom performers in the table all share the same value). Betweenness expresses the extent to which a node lies between all other pairs of nodes on their geodesic paths. Formally, for a node i:

$$betweenness(i) = \sum_{j=1,k=1}^{g} \frac{GPATHS_{j \to i \to k}}{GPATHS_{j \to k}},$$

where g is the size of the network, $GPATHS_{j \to k}$ is the total number of geodesic paths from node j to node k and $GPATHS_{j \to i \to k}$ is the number of geodesic paths from node j to node k involving node i. In other words, the betweenness value for a node becomes higher when it is visited more often on a shortest path between two other points.

From the responses of the RWS process owners to this analysis, we learned that, typically, performers with high scores (e.g., *user1* and *user4* in Table 3) work for the administrative department in supportive functions. This confirms a general insight that highly connected people often are assistants. Because the administrative department is responsible for both the preparation and completion of the handling of each invoice, its staff is involved in the handling of each case, giving them strong ties with other performers. The managers of RWS indicated, however, that not all of the people in these positions were present in the top of the lists, indicating that having a supportive function is not sufficient in itself to become highly connected.

The performers with bottom scores could be categorized as follows. First of all, project leaders were highly represented in the bottom of the lists (e.g., *user9*). They play an isolated role in the handling of invoices, normally they are not involved in any steps other than approving invoices related to their own projects. Second, performers with limited formal verification responsibilities could be identified as well (e.g., *user22*).

The second category of relatively unconnected performers could be traced back to auxiliary logins (e.g., *user30*), used by system administrators and management to deal with exceptional circumstances. An example of an exception is an invoice that is being withdrawn while its processing has already started. The isolated "participation" of this category of users is therefore not very surprising. It did, however, make the managers conscious of the visibility of this type of irregular interference. One manager remarked: "So, auditors can derive this type of information too...".

Table 3
Performers having high and low values for betweenness when analyzing the social network shown in Fig. 7

Ranking	Name	Betweenness
1	user1	0.152
2	user4	0.141
3	user23	0.085
4	user5	0.079
5	user16	0.065
6	user13	0.057
7	user18	0.052
8	user2	0.049
9	user7	0.04
..	..	..
35	user9	0
36	user20	0
37	user22	0
38	user30	0
39	user35	0
40	user36	0
41	user39	0
42	user41	0
43	user42	0

The third category turned out to be more surprising, as it involved senior positions in the contractual and financial departments (e.g., *user41*). At least nominally, they are expected to be actively involved in the process. Their low position could indicate that a large amount of work being executed with a WfMS is delegated to their juniors. Also, one of these performers was about to retire in a couple of weeks, explaining her current low centrality.

So far we only looked at sociograms based on the transfer of work metric (i.e., the frequency of passing work from one performer to another). However, we can also use the *subcontracting* metric which is related to the transfer of work metric. The main idea behind the subcontracting metric is to count the number of times individual j executed an activity in between two activities executed by individual i. In a sense, there is a loop of work between such individuals. This may indicate that work was subcontracted from i to j. To find subcontracting relationships between people, we analyzed the occasions where a direct succession of contractor and subcontractor takes place with only one event in between the steps performed by the contractor. Fig. 8 shows the resulting social network. In this network, the direction of arcs is important. The start node of an arc represents a contractor, while the end node of an arc represents a subcontractor.

The displayed network has 43 nodes and 146 links. It also contains eight nodes that are isolated from the network, which means they are not involved in a direct subcontracting relation.

Triggered by the many loops uncovered in the process mining analysis (see Section 6) and the social network based on the subcontracting metric, the RWS process owners were interested to learn more about the places in the process where work seemed to circle. After some discussion, they selected four places in the process where going "back-and-forth" is particularly undesirable. The four loops of interest, which can all be clearly distinguished in Fig. 6, are as follows:

- 030_1e_Vastlegging → 050_Adm_akkoord → 030_1e_Vastlegging,
- 050_Adm_akkoord → 080_Contract_akkoord → 050_Adm_akkoord,
- 050_Adm_akkoord → 070_PV → 050_Adm_akkoord,
- 080_Contract_akkoord → 070_PV → 080_Contract_akkoord.

Each occurrence of this pattern is highly undesirable, as it slows down the processing of the invoice without any progress being made. Note that from an organizational perspective, it is just as undesirable when the work package is routed back to the

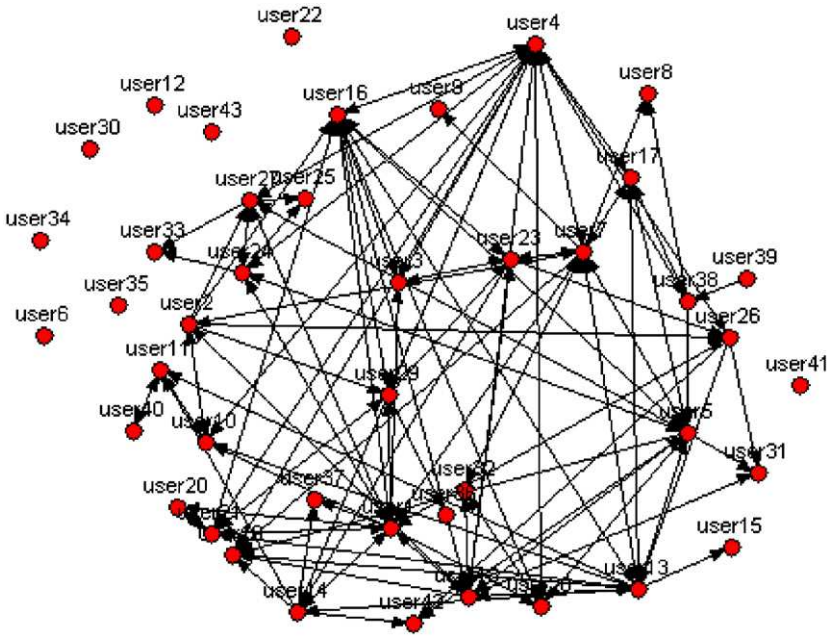

Fig. 8. Social network based on subcontracting metric.

original contractor as to a colleague with a similar organizational role.

In extending our analysis, it turned out that in the handling of over 17% of all invoices, at least once an undesired subcontracting takes place at either of the four identified places in the process. The exact distribution is shown in Fig. 9. As can be seen, there are cases where 10 or more erroneous routings take place. As this figure seemed rather excessive, we carried out a further analysis.

As it turned out, from the 4592 cases of undesired loops/subcontracting it happened 2087 times (45%) that the routing back was initiated by the WfMS *without* any human performer initiating this action. This could be seen in the RWS log, since no human performer of an activity was registered, but instead a time-out message appeared. The WfMS had been configured in such a way that whenever a step becomes ready to be executed because a preceding step is completed, it starts keeping track of the time the case is idle. When this time exceeds a time period of two weeks, the system passes the case back to the last performer being active on the case. In the extreme case of the loop 050_Adm_akkoord → 070_PV → 050_Adm_akkoord, this active involvement of the WfMS was responsible for 57% of the iterations. So, at least in part, the loops can be explained by an overly long passivity of performers in combination with the particular configuration of the WfMS to act on that automatically. It is interesting to note here that the performers responsible for 070_PV are the project leaders, who normally operate outside the main RWS building where the invoice processing takes place. As we mentioned before, the approval of invoices amounts to only a very small part of their work, which is focused on the execution of infrastructural projects.

Based on these insights, the RWS management took steps to improve the process. Before we discuss these steps, we will first show the additional analyses from a case perspective in the following section. As will be shown, the case perspective helped to develop an even better understanding of the looping behavior.

8. Mining the case perspective

In this section we will illustrate business process mining from the third and last perspective: the *case perspective*. In other words, the emphasis is on the "what" question in the handling of invoices. Focusing on the case perspective is most interesting when different properties of individual cases are available. Some properties may refer to the activities being executed and the performers working on a specific case. Other properties are directly linked to a case (i.e., independent of the way the case has been processed in the WfMS), such as, for instance, the amount of money of an invoice. However, there are possibly interesting correlations between the practical processing of a case and properties directly linked to case. As the mining of data already has a long tradition, standard knowledge discovering techniques can be used to search for this kind of relations.

Information that could be directly distilled from the RWS log relates to the timeliness of payments now that a WfMS is operational (see Table 4). The third column reports the invoice payment results as registered in the RWS log. The other information

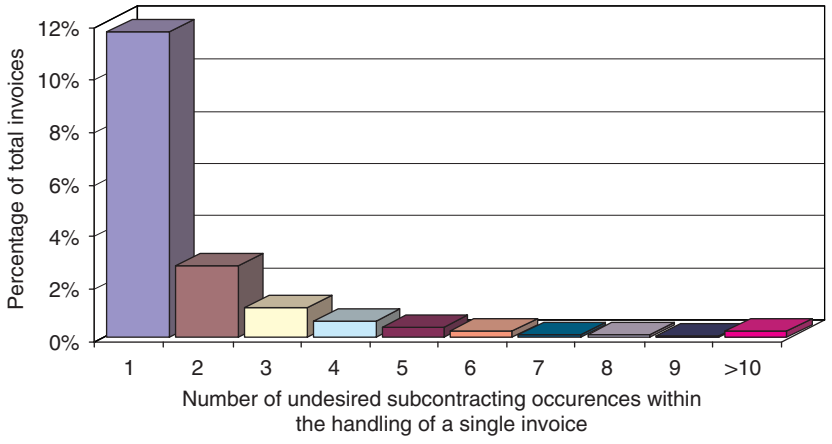

Fig. 9. Distribution of undesired loops within the handling of invoices.

has been already presented in Table 2, but is repeated for clarity.

As can be seen, the implementation of the WfMS has accelerated the payment of invoices, but the norms are still not met. This triggers the following question: "Is there a relation between the time that an invoice is being paid and the amount of money being involved with the invoice?". Suppose that the payment of invoices with a small amount of money are the ones that are delayed the most, perhaps because they are considered as less important. If this is the case, the negative financial effects would not be so bad for the financial position of the RWS office. After all, the interest is proportional to the invoice sum. To address this specific question, the SPSS tool Answer Tree was used, resulting in five categories (automatically generated by Answer Tree) with values as reported in Table 5.

Unfortunately for the RWS office in question, it appears that specifically invoices with a high payment sum—the ones above 1618K euro—are delayed *more strongly* than others. It follows from the table that 20% of the cases between 1618K and 8377K euro are paid too late and this is the case for even 24% of the cases above 8377K euro. For the categories with lower payment sums (below 1618K euro), overly late payments are between 13% and 14%. An explanation for this phenomenon may be that people are reluctant to accept the responsibility for approving an invoice involving a high sum of money. Be it as it may, this clearly illustrates the kind of insight that can be gained from a case analysis. It also once more emphasizes the need to improve the invoice handling process.

Here, we present the further steps taken in the case analysis of the RWS log that follow up on the results of the analyses as presented before. As we identified from the mining of the process control flow, various loops are a significant part of the main process, as many cases follow these paths (see Section 6). Furthermore, we took a closer look at the performers responsible for the various steps involved in these loops, in particular considering four specific parts in the process (see Section 7).

By taking a case perspective, in particular by combining the information on the execution history of specific cases and their corresponding processing times, we try to gain a deeper understanding of the *impact* of these loops. We will once more focus on the loop behavior involving the four activities 030_1e_Vastlegging, 050_Adm_akkoord, 070_PV, and 080_Contract_akkoord. Note that the loops prioritized by the RWS process owners themselves are composed of precisely these activities. Again, we used Answer Tree with the three payment classes ($A: \leqslant 31$, $B: \geqslant 32$ and $\leqslant 62$, $C: \geqslant 63$) as the target classification on the one hand and the number of times a case visits one of the four activities as predictors on the other. All other Answer Tree parameters are set to default values. Below, we discuss the information in the automatically generated decision tree.

Table 4
The time until invoices are paid (i.e., norms and actual performance before and after the implementation of a workflow management system)

Payment duration (days)	Norm (%)	Before WfMS (%)	After WfMS (%)
0–31	90	70	84
32–62	5	22	12
63	5	8	4

Table 6
The time of payment distribution related to the number of times activity 050_Adm_akkoord is processed

Time of payment	Overall	0	1	2	>2
0–31	84%	99%	91%	77%	36%
32–62	12%	1%	8%	18%	40%
63	4%	0%	1%	5%	24%
#*n*	14,043	364	10,680	1782	1218

Table 5
The time of payment related to the amount of money

Time of payment	Overall	$\leqslant 24$	$>24 \leqslant 172$	$>172 \leqslant 1618$	$>1618 \leqslant 8377$	>8377
0–31	84%	86%	87%	86%	80%	76%
32–62	12%	10%	11%	11%	15%	15%
63	4%	3%	2%	3%	5%	9%
#*n*	14,043	1403	2810	5619	2807	1404

First of all, activity 050_Adm_akkoord appears the best predictor for the payment classification (see Table 6). This is the activity where the registration takes place of the invoice's payment sum, the relevant budgets for paying it, and the required distribution over these budgets. The actual verification whether the payment should be done needs then yet to take place. Normally, such a verification is either done by a project leader in the case that the payment involves a specific project or by a clerk if the invoice falls within the conditions of a long-running contract.

From Table 6 it clearly follows that when activity 050_Adm_akkoord is executed twice for a particular case, the probability of the involved payment being delayed beyond the 31 day limit is much higher (23%) than when this activity is not executed at all (1%) or just once (9%). If the activity 050_Adm_akkoord is processed more than two times, the probability of a payment being too late even increases to 64%. Note that considering the four places of undesired subcontracting identified earlier, the repeated execution of activity 050_Adm_akkoord may indicate occurrences of either the loop 050_Adm_akkoord → 080_Contract_akkoord → 050_Adm_akkoord or 050_Adm_akkoord → 070_PV → 050_Adm_akkoord.

It is now interesting to find out whether it is activity 080_Contract_akkoord or activity 070_PV that is most related to late payments. These are the two alternatives for a loop involving 050_Adm_akkoord, being the two principal ways that the actual verification of the invoice can take place. By examining the lower leaves of the decision tree, we found that it is in fact activity 070_PV that best explains these delays in the case that 050_Adm_akkoord is processed more than two times. It is exactly this activity that needs to be executed by the project leaders, who often work remotely from the RWS main office. If the activity 070_PV is processed more than two times, the probability of the payment being too late increases to 75%.

The analysis presented in this section clearly identifies how multiple executions of specific activities contribute to the late processing of invoices. In response to this insight, the RWS process owners informed the project leaders responsible for executing 070_PV on how their purported passivity affected the overall performance of the payment process. Recall that the analysis from the organizational perspective already indicated that the WfMS often acted by itself to hand back a case to a previous performer, because the intended performer of an activity did not respond in time. As it turned out, the project leaders were not aware of the impact of their actions. They agreed to give the invoice approval a higher priority in their work load, even though it is not a core part of their daily work. This nicely illustrates the phenomenon that performers often do not have a good insight into the wider context of a business process [41].

9. Reflection and conclusions

In this paper, we presented a case study illustrating the practical application of process mining. To present the case study we used three perspectives: the process, organization, and case perspective. As a starting point, we used an industrial event log extracted from an operational WfMS. This proprietary WfMS is being used (amongst others) to support the process of handling invoices at a local RWS office in The Netherlands. We described how one can use a mixture of standard and specific mining tools to carry out this analysis.

The most important outcome from a process mining perspective was the discovery of the *main flow* in the invoice handling process. Because the generated process model incorporated information on the execution frequencies of activities, it could clearly be seen that loops in this process are far from exceptional. Inspired by this observation, process mining from an organizational perspective focused on the subcontracting metric in particular, leading to the identification of places in the process where the circling of work is undesirable. By inspecting these places more closely, the specific roles of two kinds of process performers were identified: the WfMS itself and the project leaders. Additional mining from the case perspective revealed that the various loops indeed have a great impact on the process performance. It confirmed that the activity that needs to be executed by project leaders, who often work remotely, is strongly affecting the performance of the process in terms of timeliness.

Based on this industrial application of process mining, we can make *three important observations*. First of all, the practical application of business process mining is already feasible using the techniques embedded in the ProM framework. Nonetheless, it seems necessary to take realistic characteristics of event logs into account, such as e.g., the existence of noise. The successful application of our heuristics-based mining approach

illustrates this point. Second, business process mining is a promising and potentially effective way to deal with real organizational challenges. The results from our analyses enabled the involved management to formulate and target specific organizational measures. The mining results could be used as an objective support for these measures. As a side effect, process mining made the management aware of the visibility of irregular behavior. Finally, the case study showed that it is worthwhile to combine different mining perspectives to reach a richer understanding of the process. In this case, for example, the control flows revealed the various loops, but it took an organizational perspective to identify the key players, and a case-oriented analysis to understand the impact of these loops on the process performance.

In performing this business process mining project, we learned an *important lesson*. It seems crucial to be closely involved with the people of the organization itself to carry out a meaningful analysis. As a small illustration of this point, it would have been impossible to determine the real value of the oddly connected activity 170_Parkeer on the basis of the event log itself (i.e., without direct involvement of the process owner). This activity turned out not to be an activity at all, but rather a WfMS facility to suspend an operation. More importantly, it took the input of the RWS process owners to identify and prioritize four locations of the process that seemed of interest to subject to a closer analysis. This certainly helped to speed up the identification of relevant results.

Furthermore, the case study also showed that process mining also has a number of obvious *limitations*. First of all, we can only monitor the events that are actually logged. This implies that some interactions may not be visible. Moreover, people may find ways to work around the system. Second, the system may enforce certain interaction patterns. If workers are completely controlled by the system, the discovered process models and sociograms reflect the system rather than the organization. Fortunately, most systems offer a lot of flexibility when it comes to the selection and ordering of work-items. Even WfMSs allow for a pull mechanism where workers select work-items from a shared pool in any order. Moreover, other types of process-aware information systems (e.g., ERP, CRM, PDM systems) tend to allow for even more flexibility. Finally, it is clear that there are privacy issues that may complicate the application of process mining (in particular the organizational perspective). It would have been relatively easy on the basis of the logged information to determine which specific individual performers are responsible for the slow processing of invoices. Although this might be important information to optimize the process further, we did not carry out this analysis. Privacy issues, labor contracts, and other agreements are crucial to determine to what level practical mining analyses may proceed.

To conclude this paper, we briefly discuss our plans for future work. We are currently extending and improving our mining techniques and continue to do so given the many challenges and open problems. For example, we are developing genetic algorithms to improve the mining of noisy logs. Recently, we added a lot of new functionality to the ProM framework.[4] Therefore, our first priority is to apply process mining in a wide variety of practical situations. Interesting application domains that we would like to target (and where we already have some limited experiences) are:

Health care: The flow of patients in a hospital and other health-care-related processes could be discovered using process mining. Just measuring performance indicators such as flow times and frequencies does not provide enough insight in the actual process. Effective health-care management requires more fine-grained information that is truly process-oriented. Hence process mining could really help to improve processes here.

Web services: Cross-organizational workflows are enabled by web-services technology, but also require an agreement on a common process. However, one party cannot force the other party to work in a certain way. Therefore, one can make agreements and check these or try to discover the behavior of the other parties involved. Process mining can be used to discover the "choreography" of web services and detect deviations. This is a way to operationalize the notion of abstract processes in BPEL [42] and choreographies in [43].

Case handling: Case handling systems such as FLOWer [32] extend existing workflow technology with more flexibility. As a result, users can deviate from the "normal" flow of work. Process mining can be used to discover and quantify these deviations. ProM is already able to mine from FLOWer logs and we applied this in a Dutch social security

[4]See www.processmining.org for more information and to download the tools used in this paper.

agency. However, we would also like to apply our tools to change the logs of systems such as ADEPT [44].

Related to the application of the ProM framework in real-life situations is the transfer of ideas to commercial tools. In our collaboration with IDS Scheer some of our results have been incorporated in ARIS PPM, e.g., the "Organizational Analysis" module of ARIS PPM borrowed several ideas from the Social Network Miner in ProM. We hope that more software vendors will adopt more of the ideas presented in this paper.

Acknowledgments

The authors would like to thank Andriy Anikolov, Laura Maruster, Anne Rozinat, Christian Günther, Huub de Beer, Monique Jansen-Vullers, Michael Rosemann, and Peter van den Brand for their on-going work on process mining techniques and tools at Eindhoven University of Technology. Minseok Song visited Eindhoven University of Technology with funding by the BK21 program. He would like to thank the Ministry of Education of Korea for its financial support through the BK21 program.

References

[1] W.M.P. van der Aalst, K.M. van Hee, Workflow Management: Models, Methods, and Systems, MIT press, Cambridge, MA, 2002.

[2] W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, A.J.M.M. Weijters, Workflow mining: a survey of issues and approaches, Data Knowl. Eng. 47 (2) (2003) 237–267.

[3] R. Agrawal, D. Gunopulos, F. Leymann, Mining process models from workflow logs, in: Sixth International Conference on Extending Database Technology, 1998, pp. 469–483.

[4] D. Grigori, F. Casati, U. Dayal, M.C. Shan, Improving business process quality through exception understanding, prediction, and prevention, in: P. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, R. Snodgrass (Eds.), Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01), Morgan Kaufmann, Los Alamitos, CA, 2001, pp. 159–168.

[5] M. Sayal, F. Casati, U. Dayal, M.C. Shan, Business process cockpit, in: Proceedings of 28th International Conference on Very Large Data Bases (VLDB'02), Morgan Kaufmann, Los Alamitos, CA, 2002, pp. 880–883.

[6] P. Sarbanes, G. Oxley, et al., Sarbanes–Oxley act of 2002, 2002.

[7] T. Hoffman, Sarbanes–Oxley sparks forensics apps interest: vendors offer monitoring tools to help identify incidents of financial fraud, Comput World 38 (2004) 14.

[8] IDS Scheer, ARIS process performance manager (ARIS PPM): measure, analyze and optimize your business process performance (whitepaper), IDS Scheer, Saarbruecken, Germany. ⟨http://www.ids-scheer.com⟩, 2002.

[9] W.M.P. van der Aalst, B.F. van Dongen, Discovering Workflow Performance Models from Timed Logs, in: Y. Han, S. Tai, D. Wikarski (Eds.), International Conference on Engineering and Deployment of Cooperative Information Systems (EDCIS 2002), Lecture Notes in Computer Science, vol. 2480, Springer, Berlin, 2002, pp. 45–63.

[10] W.M.P. van der Aalst, M. Song, Mining social networks: uncovering interaction patterns in business processes, in: J. Desel, B. Pernici, M. Weske (Eds.), International Conference on Business Process Management (BPM 2004), Lecture Notes in Computer Science, vol. 3080, Springer, Berlin, 2004, pp. 244–260.

[11] W.M.P. van der Aalst, A.J.M.M. Weijters, L. Maruster, Workflow mining: discovering process models from event logs, IEEE Trans. Knowl. Data Eng. 16 (9) (2004) 1128–1142.

[12] J.E. Cook, A.L. Wolf, Discovering models of software processes from event-based data, ACM Trans. Software Eng. Methodol. 7 (3) (1998) 215–249.

[13] J. Herbst, A machine learning approach to workflow management, in: Proceedings of the 11th European Conference on Machine Learning, Lecture Notes in Computer Science, vol. 1810, Springer, Berlin, 2000, pp. 183–194.

[14] M. zur Mühlen, M. Rosemann, Workflow-based process monitoring and controlling—technical and organizational issues, in: R. Sprague (Ed.), Proceedings of the 33rd Hawaii International Conference on System Science (HICSS-33), IEEE Computer Society Press, Los Alamitos, CA, 2000, pp. 1–10.

[15] A.J.M.M. Weijters, W.M.P. van der Aalst, Rediscovering workflow models from event-based data using little thumb, Integrated Computer-Aided Eng. 10 (2) (2003) 151–162.

[16] S. Jablonski, C. Bussler, Workflow Management: Modeling Concepts, Architecture, and Implementation, International Thomson Computer Press, London, UK, 1996.

[17] F. Leymann, D. Roller, Production Workflow: Concepts and Techniques, Prentice-Hall PTR, Upper Saddle River, NJ, USA, 1999.

[18] A.J.M.M. Weijters, W.M.P. van der Aalst, Workflow mining: discovering workflow models from event-based data, in: C. Dousson, F. Höppner, R. Quiniou (Eds.), Proceedings of the ECAI Workshop on Knowledge Discovery and Spatial Data, 2002, pp. 78–84.

[19] J. Scott, Social Network Analysis, Sage, Newbury Park, CA, 1992.

[20] S. Wasserman, K. Faust, Social Network Analysis: Methods and Applications, Cambridge University Press, Cambridge, 1994.

[21] J.L. Moreno, Who Shall Survive?, Nervous and Mental Disease Publishing Company, Washington, DC, 1934.

[22] S. Farnham, S.U. Kelly, W. Portnoy, J.L.K. Schwartz, Wallop: designing social software for co-located social networks, in: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04), IEEE Computer Society Press, Los Alamitos, CA, 2004.

[23] S. Farnham, W. Portnoy, A. Turski, Using email mailing lists to approximate and explore corporate social networks, in: D.W. McDonald, S. Farnham, D. Fisher (Eds.),

Proceedings of the CSCW'04 Workshop on Social Networks, 2004.
[24] D. Fisher, P. Dourish, Social and temporal structures in everyday collaboration, in: E. Dykstra-Erickson, M. Tscheligi (Eds.), Proceedings of the 2004 Conference on Human Factors in Computing Systems (CHI2004), ACM Press, New York, NY, USA, 2004, pp. 551–558.
[25] B.A. Nardi, S. Whittaker, E. Isaacs, M. Creech, J. Johnson, J. Hainsworth, Integrating communication and information through contactMap, Communi. ACM 45 (2) (2002) 89–95.
[26] H. Ogata, Y. Yano, N. Furugori, Q. Jin, Computer supported social networking for augmenting cooperation, Comput. Supported Coop. Work 10 (2) (2001) 189–209.
[27] J. Begole, J. Tang, R. Smith, N. Yankelovich, Work rhythms: analyzing visualizations of awareness histories of distributed groups, in: C. Neuwirth, T. Rodden (Eds.), Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work, ACM Press, New York, NY, USA, 2002, pp. 334–343.
[28] W.M.P. van der Aalst, H.A. Reijers, M. Song, Discovering social networks from event logs, Comput. Supported Coop. Work 14 (6) (2005) 549–593.
[29] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, M.C. Shan, Business process intelligence, Comput. Ind. 53 (3) (2004) 321–343.
[30] TIBCO, TIBCO Staffware Process Monitor (SPM). ⟨http://www.tibco.com⟩, 2005.
[31] W.M.P. van der Aalst, A.J.M.M. Weijters (Eds.), Process Min., Special Issue of Comput. Ind. 53(3) (2004).
[32] P. Athena, Flower User Manual, Pallas Athena BV, Apeldoorn, The Netherlands, 2002.
[33] W. Reisig, G. Rozenberg (Eds.), Lectures on Petri Nets I: Basic Models, Lecture Notes in Computer Science, vol. 1491, Springer, Berlin, 1998.
[34] G. Keller, T. Teufel, SAP R/3 Process Oriented Implementation, Addison-Wesley, Reading MA, 1998.
[35] H. Nemati, C.D. Barko, Organizational Data Mining: Leveraging Enterprise Data Resources for Optimal Performance, Idea Group Publishing, Hershey, PA, USA, 2003.
[36] B.F. van Dongen, A.K. Alves de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, W.M.P. van der Aalst, The ProM framework: a new era in process mining tool support, in: G. Ciardo, P. Darondeau (Eds.), Application and Theory of Petri Nets 2005, Lecture Notes in Computer Science, vol. 3536, Springer, Berlin, 2005, pp. 444–454.
[37] B.F. van Dongen, W.M.P. van der Aalst, A meta model for process mining data, in: J. Casto, E. Teniente (Eds.), Proceedings of the CAiSE'05 Workshops (EMOI-INTEROP Workshop), vol. 2, FEUP, Porto, Portugal, 2005, pp. 309–320.
[38] P. Lawrence (Ed.), Workflow Handbook 1997, Workflow Management Coalition, Wiley, New York, 1997.
[39] H.A. Reijers, W.M.P. van der Aalst, The effectiveness of workflow management systems: predictions and lessons learned, Int. J. Inf. Manage. 25 (5) (2005) 458–472.
[40] P. Bonacich, Power and centrality: a family of measures, Am. J. Sociol. 92 (1987) 1170–1182.
[41] M. Hammer, J. Champy, Reengineering the Corporation, Nicolas Brealey Publishing, London, 1993.
[42] A. Arkin, S. Askary, B. Bloch, F. Curbera, Y. Goland, N. Kartha, C.K. Liu, S. Thatte, P. Yendluri, A. Yiu, Web Services Business Process Execution Language Version 2.0, WS-BPEL TC OASIS, 2005.
[43] N. Kavantzas, D. Burdett, G. Ritzinger, T. Fletcher, Y. Lafon, Web Services Choreography Description Language Version 1.0 (W3C Working Draft 17 December 2004). ⟨http://www.w3.org/TR/2004/WD-ws-cdl-10-20041217/⟩, 2004.
[44] S. Rinderle, M. Reichert, P. Dadam, Correctness criteria for dynamic changes in workflow systems: a survey, Data and Knowl. Eng. 50 (1) (2004) 9–34.