Trends in business process analysis from verification to process mining

Conference Paper · January 2007

Source: DBLP

CITATIONS
36

READS
1,554

1 author:

Wil Van der Aalst
RWTH Aachen University
1,563 PUBLICATIONS **100,497** CITATIONS

SEE PROFILE

TRENDS IN BUSINESS PROCESS ANALYSIS
From Verification to Process Mining

Wil M.P. van der Aalst
Department of Mathematics and Computer Science, Eindhoven University of Technology,
P.O. Box 513, 5600 MB, Eindhoven, The Netherlands
w.m.p.v.d.aalst@tue.nl

Keywords: Business Process Management, Process Mining, Petri nets, Verification, Simulation, Workflow Management.

Abstract: Business process analysis ranges from model verification at design-time to the monitoring of processes at runtime. Much progress has been achieved in process verification. Today we are able to verify the entire reference model of SAP without any problems. Moreover, more and more processes leave their "trail" in the form of event logs. This makes it interesting to apply process mining to these logs. Interestingly, practical applications of process mining reveal that reality is often quite different from the idealized models, also referred to as "PowerPoint reality". Future process-aware information systems will need to provide full support of the entire life-cycle of business processes. Recent results in business process analysis show that this is indeed possible, e.g., the possibilities offered by process mining tools such as ProM are breathtaking both from a scientific and practical perspective.

1 INTRODUCTION: THE ROLE OF MODELS

Models play an important role in information systems and it is clear that the importance of models will increase. Models can be used to specify systems and processes and can be used for their analysis. Some of today's information systems are even driven by models (cf. workflow management systems). Although the general vision of a "Model Driven Architecture" (MDA) is appealing, it is not yet realistic/practical for many applications. Only in specific niches such as workflow technology, MDA is already a reality and has proven to be valuable. In the context of Enterprise Resource Planning (ERP) systems, i.e., the world of SAP, PeopleSoft, Oracle, etc., models play a less prominent role. These systems offer a workflow component, but most of their functionality is still hard-coded. The well-know reference model of SAP (Curran and Keller, 1997) contains 604 Event-driven Process Chains (EPCs) modeling the different business processes supported by the R/3 system. However, these EPC models are not used for enactment and serve more as background information. It seems vital that ERP systems like SAP start using models as a starting point, rather than just as a means to document things afterwards. It seems particularly interesting to use *configurable* process models (Aalst et al., 2006a) as a starting point. For example, one can make EPCs configurable as shown in (Rosemann and Aalst, 2006; Jansen-Vullers et al., 2006) and use models to configure the system to fit a certain business context.

Although models are highly relevant for the enactment of systems, this paper will not focus on this aspect of modeling. The interested reader may consult workflow literature to learn more about this (Aalst and Hee, 2004; Aalst, 2004; Leymann and Roller, 1999; Georgakopoulos et al., 1995) and play with an open-source workflow management system like YAWL (Aalst and Hofstede, 2005). Instead we focus on the *analysis of business processes* and try to provide an overview of recent developments in this area. We will focus on two types of analysis: (1) analysis at *design-time* and (2) analysis at *run-time*. At design time, the only basis for analysis is a model, e.g., a workflow (re)design (Netjes et al., 2006). At runtime, one can also observe the actual behavior and use this as input for analysis.

Figure 1 shows an overview of the different types of analysis discussed in this paper. To explain the

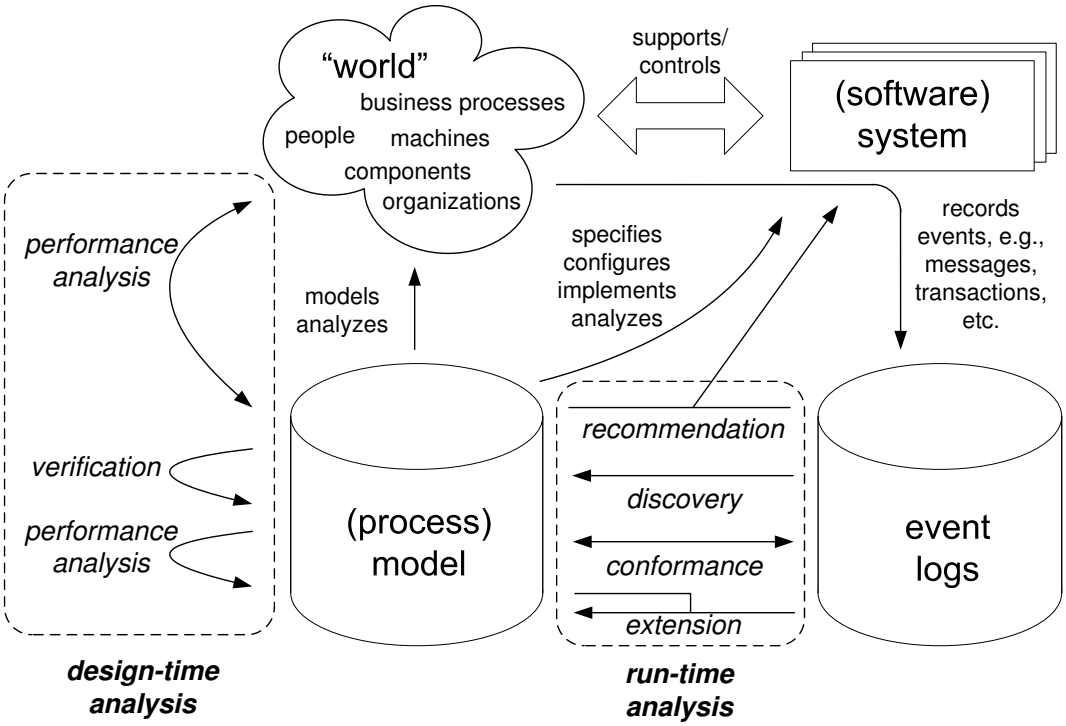

Figure 1: The relationships between reality, systems, logs, and models and the different types of design-time and run-time analysis.

diagram let us first consider the top part showing the interaction between the "world" and some (software) system. Any information system ultimately interacts with some physical environment; otherwise it serves no purpose. The system may support or control all kinds of processes taking place in the real world. Moreover, most systems also record events taking place inside and outside the system as indicated by the arrow connecting the "world" to event logs via the (software) system. Today's information systems log enormous amounts of events. Classical 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. Even embedded systems are connected to the Internet today, thus allowing for unprecedented streams of data. On the other hand, models play a prominent role as indicated in Figure 1. Examples of models are process models such as BPMN diagrams, EPCs, Petri nets, BPEL specifications, UML activity diagrams, but also other types of models such as social networks, organizational charts, data models, etc. These models can be used to model the "world". However, they can also be used to model the system. In this context it is important to note that most information systems have a model of reality, i.e., a software system that has no "mental image" of the organizational context and the processes it should support is of limited use. It is often remarkable to see the resemblance between simulation models and workflow models. This supports the earlier observation that information systems need to have a model of reality. In an MDA or workflow setting, models are used to configure the information system as shown in Figure 1. However, *in this paper we focus on the analysis aspect and not on enactment*.

Figure 1 clearly shows the two types of analysis: (1) analysis at *design-time* and (2) analysis at *run-time*. In the remainder, we will elaborate on the different types of analysis. On the one hand, this provides a comprehensive overview of business process analysis. On the other hand, we will use this to express our views on the interesting research questions in this area. In particular, we would like to make the following statements:

- *Verification of real-life processes has become a reality!* It is possible to verify large sets of complicated models and these efforts pay off because of-

ten many errors are found. For example, the 604 EPCs of the SAP reference models can be easily analyzed and many design errors are uncovered by doing so (Mendling et al., 2006a; Mendling et al., 2006b).

- *The abundance of event logs allows for new and exciting forms of process analysis. Process mining* can use this information in various ways. It may be used to discover the way that people really work, it may be used to find out where there are deviations, it may be used to support people in performing their duties, and ultimately it can be used for all kinds of process improvement.
- Using the slogan *"The World is NOT a Petri Net"* we would like to emphasize that reality is often very different from what is modeled or what people think. Whatever representation is used (Petri nets or any other modeling language), the model is an abstraction (i.e., things are left out) and may not reflect reality. The first observation (abstraction) is unavoidable and should be accepted as a fact of life. However, the second observation is more problematic and should be addressed urgently. As long as managers and system designer take a "PowerPoint" reality as starting point, information systems will remain to have serious alignment problems.
- There is a need for the academic community to share software and to build *mature business process analysis tools*. With the ProM framework we try to do so. ProM provides an open-source environment where people can plug-in their own analysis tools. Currently, ProM has more than 150 plug-ins covering the whole spectrum of process analysis, but with a clear emphasis on process mining and model translations.

In the remainder of this paper, these statements are explained and put into context.

2 ANALYSIS AT DESIGN-TIME

This section elaborates on model-based analysis at design time. Note that we focus on process models but do not limit ourselves to the control-flow perspective.

2.1 Different Types of Analysis

The correctness, effectiveness, and efficiency of the business processes supported by the information system are vital to the organization. A process definition which contains errors may lead to angry customers, back-log, damage claims, and loss of goodwill. Flaws in the design of a process definition may also lead to high throughput times, low service levels, and a need for excess capacity. This is why it is important to analyze a process before it is put into production. As shown in Figure 1, there are three types of analysis:

- *validation*, i.e., testing whether the process behaves as expected,
- *verification*, i.e., establishing the correctness of a process definition, and
- *performance analysis*, i.e., evaluating the ability to meet requirements with respect to throughput times, service levels, and resource utilization.

Validation can be done by interactive simulation: a number of fictitious cases are fed to the system to see whether they are handled well. For verification and performance analysis more advanced analysis techniques are needed. Fortunately, many powerful analysis techniques have been developed and some of the corresponding tools have become mature in recent years. As an example, consider the Petri-net-based techniques and tools available for the modeling and analysis of workflows (Aalst and Hee, 2004; Aalst, 2004; Verbeek et al., 2001). Linear algebraic techniques can be used to verify many properties, e.g., place invariants, transition invariants, and (non-)reachability. Coverability graph analysis, model checking, and reduction techniques can be used to analyze the dynamic behavior of a Petri net. Simulation and Markov-chain analysis can be used for performance evaluation.

2.2 Verification Techniques Have Become Mature!

A complete overview of the different types of design-time analysis is outside the scope of this paper. However, we would like to emphasize that already today there are mature tools and techniques to verify large collections of non-trivial models. Tools such as Woflan (Verbeek et al., 2001) have played a pioneering role in this domain.[1] Using Woflan it is possible to extract models from all kinds of systems and check these models for deadlocks, etc. To illustrate the maturity of process verification we briefly describe the analysis of the entire *SAP reference model* (Curran and Keller, 1997) described in (Mendling et al., 2006a; Mendling et al., 2006b).

The SAP reference model contains more than 600 non-trivial process models expressed in terms of

[1] A substantial part of the functionality of Woflan in embedded in ProM (Dongen et al., 2005).

Event-driven Process Chains (EPCs). We have automatically translated these EPCs into YAWL models (Aalst and Hofstede, 2005) and analyzed these models using WofYAWL, a verification tool based on Petri nets extending Woflan (Verbeek et al., 2001). The translation from EPCs to YAWL is trivial because EPCs can be considered a subset of the much more expressive YAWL language (Aalst and Hofstede, 2005). The YAWL models are then translated to Petri nets and these Petri nets are analyzed using the well-known Petri-net invariants. This approach does not find all errors. However, all errors found are real errors, i.e., the approach is sound but not necessarily complete. Nevertheless, we discovered that *at least 34 of these EPCs contain errors* (i.e., at least 5.6% is flawed). We analyzed which parts of the SAP reference model contain most errors. Moreover, based on 15 characteristics (e.g., the size of the model), we used logistic regression to find possible predictors for these errors showing that complexity of EPCs has a significant impact on error probability. This systematic analysis of the SAP reference model illustrates the maturity of process verification.

The interested reader is referred to (Mendling et al., 2006a; Mendling et al., 2006b) for details and pointers to the analysis of similar datasets covering thousands of EPCs. Moreover, using improved techniques we are even able to find much more errors. In this work we also predict the occurrence of errors based on features such as size, complexity, modeling constructs used, etc. As indicated above, logistic regression can be used to develop useful error predictors. This is related to the complexity metrics presented in (Cardoso, 2006). The hypothesis is that people are more likely to make errors if they use particular techniques and/or constructs.

3 ANALYSIS AT RUN-TIME

After discussing techniques to be used at design-time, we now focus on run-time analysis (cf. Figure 1). As indicated in the introduction, today's systems provide detailed event logs. *Process mining* has emerged as a way to analyze systems and their actual use based on the event logs they produce (Aalst et al., 2003; Aalst et al., 2004; Agrawal et al., 1998; Datta, 1998; Dongen and Aalst, 2004; Herbst, 2000; Rozinat and Aalst, 2006a; Weijters and Aalst, 2003). Note that, unlike classical data mining, the focus of process mining is on concurrent processes and not on static or mainly sequential structures. Also note that commercial "Business Intelligence" (BI) tools are not doing any process mining. They typically look at aggregate data seen from an external perspective (frequencies, averages, utilization, service levels, etc.). Unlike BI tools, process mining looks "inside the process" (What are the causal dependencies?, Where is the bottleneck?, etc.) and at a very refined level. In the context of a hospital, BI tools focus on performance indicators such as the number of knee operations, the length of waiting lists, and the success rate of surgery. Process mining is more concerned with the paths followed by individual patients and whether certain procedures are followed or not.

The omnipresence of event logs is an important enabler of process mining, i.e., analysis of run-time behavior is only possible if events are recorded. As indicated earlier all kinds of information systems provide such logs, e.g., classical 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. These systems provide very detailed information about the activities that have been executed. However, also all kinds of embedded systems increasingly log events. An embedded system is a special-purpose system in which the computer is completely encapsulated by or dedicated to the device or system it controls. 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. The logging capabilities of the machines of PMS illustrate the way in which embedded systems produce event logs.

The goal of process mining is to extract information (e.g., process models) from these logs, i.e., process mining describes a family of *a-posteriori* analysis techniques exploiting the information recorded in the event logs. Typically, these approaches assume that it is possible to sequentially record events such that each event refers to an activity (i.e., a well-defined step in the process) and is related to a particular case (i.e., a process instance). Furthermore, some mining techniques use additional information such as the performer or originator of the event (i.e., the person / resource executing or initiat-

ing the activity), the timestamp of the event, or data elements recorded with the event (e.g., the size of an order).

3.1 Process Discovery

Traditionally, process mining has been focusing on *discovery* (cf. Figure 1), i.e., deriving information about the original process model, the organizational context, and execution properties from enactment logs. An example of a technique addressing the control flow perspective is the α-algorithm, which constructs a Petri net model (Desel and Esparza, 1995; Reisig and Rozenberg, 1998) describing the behavior observed in the event log. However, process mining is not limited to process models (i.e., control flow) and recent process mining techniques are more and more focusing on other perspectives, e.g., the organizational perspective or the data perspective. For example, there are approaches to extract social networks from event logs and analyze them using social network analysis (Aalst et al., 2005). This allows organizations to monitor how people, groups, or software/system components are working together.

There are many different types of discovery. However, to understand the basic idea let us consider the problem of constructing a Petri net from an event log.

Let A be a set of *activity* names. For every *process instance* (often referred to as *case*), a sequence of activities is recorded. Such a sequence of activities is called as a *trace*. A^* is the set of all possible traces given a set of activities A. An *event log* is a set of process instances. Since only the corresponding traces are of interest here, an event log L is described by a bag (i.e., a multi-set) of traces. In other words: $L \in A^* \rightarrow \mathbb{N}$, i.e., for any possible $\sigma \in A^*$, $L(\sigma)$ denotes the number of process instances having a sequence of activities σ.

As an example consider the process of handling customer orders. An example of a trace would be *(register, ship, send_bill, payment, accounting, approved, close)*. This trace represents a sequence of 7 activities. To represent the log we use more convenient (i.e., shorter) names: *r=register*, *s=ship*, *sb=send_bill*, *p=payment*, *ac=accounting*, *ap=approved*, *c=close*. Moreover, there are additional activities such as *em=express_mail*, *rj=rejected*, and *rs=resolve*. Using this shorter notation we now consider a log L where each of the following traces occurs once (r,s,sb,p,ac,ap,c), (r,sb,em,p,ac,ap,c), (r,sb,p,em,ac,rj,rs,c), (r,em,sb,p,ac,ap,c), (r,sb,s,p,ac,rj,rs,c), (r,sb,p,s,ac,ap,c), and (r,sb,p,em,ac,ap,c). It is easy to imagine that such traces could be extracted from the logs of some information system (e.g., a SAP R/3 system). It should be noted that all traces start with *r* (register order) and end with *c* (close order). Moreover, some of the other activities also appear in all traces, e.g., *sb* (send bill), *p* (payment), and *ac* (accounting). In all traces either *s* (ship) or *em* (express mail) occurs. If ap (approved) occurs in this small set of traces, then rj (rejected) and *rs* (resolve) do not occur (and vice versa). Note that for the human eye it is difficult to make these conclusions and construct a process model that corresponds to the behavior recorded in event log L. Therefore, many process discovery algorithms have been proposed to automatically construct process models (e.g., a Petri net) (Aalst et al., 2003; Aalst et al., 2004; Agrawal et al., 1998; Datta, 1998; Dongen and Aalst, 2004; Herbst, 2000; Weijters and Aalst, 2003). Figure 2 shows a Petri net discovered using the α-algorithm (Aalst et al., 2004), i.e., based on the event log L automatically a model is constructed. In Section 5, when we discuss tool support for process analysis, we will provide some more examples.

3.2 Conformance Checking

The second type of analysis based on event logs is *conformance checking* (cf. Figure 1). Unlike process discovery, it is assumed that there is an a-priori model. This model is used to check if reality conforms to the model. For example, there may be a process model indicating that purchase orders of more than one million Euro require two checks. Another example is the checking of the four-eyes principle. Conformance checking may be used to detect deviations, to locate and explain these deviations, and to measure the severity of these deviations.

In (Rozinat and Aalst, 2006a) it is shown how a process model (e.g., a Petri net) can be evaluated in the context of a log using metrics such as "fitness" (Is the observed behavior possible according to the model?) and "appropriateness" (Is the model "typical" for the observed behavior?). However, it is also possible to check conformance based on organizational models, predefined business rules, temporal formulas, Quality of Service (QoS) definitions, etc.

Note that conformance checking seems particularly interesting in the context of web services (Aalst et al., 2006b; Cardoso et al., 2004).

3.3 Extension

The third type of process mining assumes again both a log and a model (cf. Figure 1). However, the model is not checked for correctness, instead it is used as

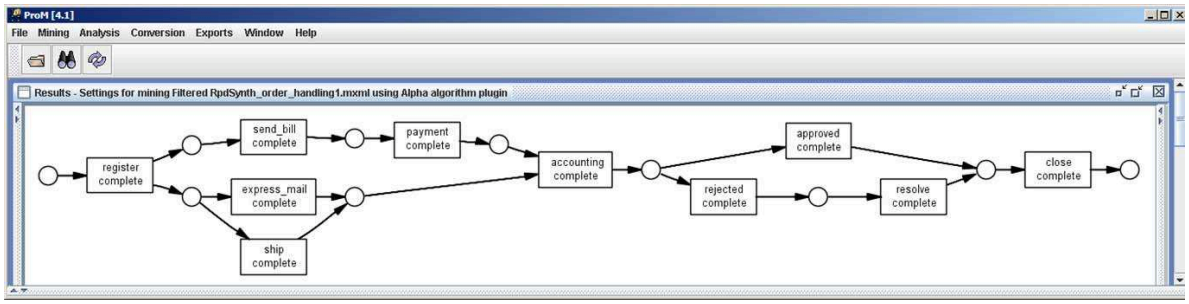

Figure 2: A Petri net discovered by ProM using the α-algorithm.

a basis, i.e., the model is extended with a new aspect or perspective. There are different ways to *extend* a given process model with additional perspectives based on event logs, e.g., decision mining, performance analysis, and user profiling. Decision mining, also referred to as decision point analysis, aims at the detection of data dependencies that affect the routing of a case (Rozinat and Aalst, 2006b). Starting from a process model, one can analyze how data attributes influence the choices made in the process based on past process executions. Classical data mining techniques such as decision trees can be leveraged for this purpose. Similarly, the process model can be extended with timing information (e.g., bottleneck analysis).

3.4 Process Mining and Simulation

Simulation is typically used at design-time. However, given the abundance of data about the actual process, it is interesting to combine process mining and simulation. In our research we envision at least two interesting approaches to link both topics:

- *Simulation model discovery* (Rozinat et al., 2006). Process mining is not limited to the control-flow perspective. Using discovery and extension it is possible to build process models adequately describing multiple perspectives (control-flow, data, resources, timing, etc.). This model can directly be used for simulation purposes as demonstrated in (Rozinat et al., 2006). The possibility to extract simulation model from event logs opens-up all kinds of application possibilities and significantly lowers the threshold for using simulation. Moreover, most simulation tools are able to generate event logs. Therefore, it is also possible to use process mining tools to analyze simulation results. Therefore, it is possible to make a very tight connection between process mining and simulation.
- *Short-term simulation* (Reijers and Aalst, 1999). One can think of short-term simulation as a quick look in the near future, i.e., a kind of "fast forward" button. By pushing this button, it is possible to see what happens if the current situation is extrapolated. It is also possible to see the effect of certain decisions (e.g. hiring additional employees or renounce new orders) in the near future. In this way, short-term simulation becomes a powerful tool for management control and operation control. In order to apply this idea three ingredients are essential: (1) a parameterized process model that can be simulated, (2) historic data to estimate parameters, (3) the current state of the workflows (i.e., the states of all cases in the pipeline). In the context of a workflow management system all this information is available as shown in (Reijers and Aalst, 1999). Process mining can be used to extract the required information from historic data, parameterize the simulation model, and to analyze the simulation results.

3.5 Recommendation

Process mining is not limited to analyzing processes. The results can also be used to act. One of the ideas we are working on is the development of so-called *recommendation services*. The idea is as follows. Using process mining one can analyze which strategies are successful and which are not, i.e., every case (process instance) that was executed can be rated in terms of flow time, costs, quality, etc. Therefore, it is possible to build a model predicting the performance of a case with respect to the selected metric. This can be used to recommend particular executions, i.e., given a partial execution of a case (the sequence of steps executed so far), the recommendation service suggests the next step. Different variants are possible and implemented in ProM.

4 THE WORLD IS NOT A PETRI NET!

The title of this section "The World is Not a Petri Net!" is intended to provoke designers, manager, and academics that actually think that real-life processes behave as modeled in some process model. Note that this applies to any process model having some kind of semantics, i.e., not just Petri nets but also BPMN models, EPCs, BPEL specifications, UML activity diagrams, process algebraic specifications, etc.[2] This is not a notational problem, i.e., it is not caused by the modeling language but by the idealized views people have when describing processes. Real-life processes turn out to be less structured than people tend to believe. Unfortunately, traditional process mining approaches have problems dealing with unstructured processes. The discovered models are often "spaghetti-like" showing all details without distinguishing what is important and what is not.

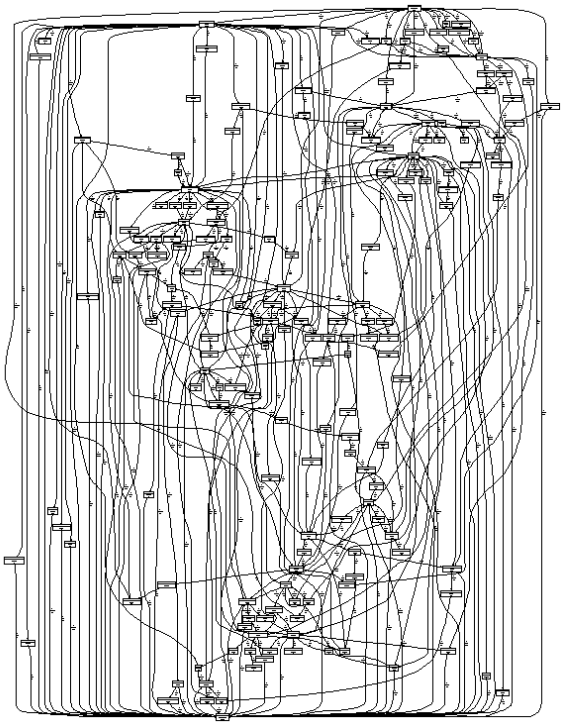

Figure 3: A process model based on the flow of 874 patients having heart surgery.

To illustrate this consider Figure 3 showing a process model obtained using the process mining technique described in (Weijters and Aalst, 2003) which is supported by the heuristics miner of ProM (see Section 5). The model is based on an event log with 10478 events. These event were recorded by the information system of a Dutch hospital for a group of 874 patients. This was a relatively homogeneous group of patients: each patient had a heart surgery. There are 181 different events, i.e., event types corresponding to possible activities in the context of heart surgery. As can be seen, the model is "spaghetti-like". One may think that Figure 3 suggest a poor performance of the process mining technique. However, this is not the case, Figure 3 reflects reality and reality is often "spaghetti-like" and not as structured as people want to believe.

Over the last couple of years we obtained much experience in applying the tried-and-tested set of mining algorithms to real-life processes. Existing algorithms tend to perform well on structured processes but often fail to provide insightful models for less structured processes, cf. Figure 3. The problem is not that existing techniques produce incorrect results. In fact, some of the more robust process mining techniques guarantee that the resulting model is "correct" in the sense that reality fits into the model. The problem is that the resulting model shows all details without providing a suitable abstraction. This is comparable to looking at the map of a country where all cities and town are represented by identical nodes and all roads are depicted in the same manner. The resulting map is correct but not very suitable. Therefore, the concept of a roadmap can used as a metaphor to visualize the resulting models. Based on an analysis of the log the importance of activities and relations among activities are taken into account. Activities and their relations can be clustered or removed depending on their role in the process. Moreover, certain aspects can be emphasized graphically just like a roadmap emphasizes highways and large cities over dirt roads and small towns. The fuzzy miner in ProM is using the roadmap metaphor to present more meaningful process models.

5 TOWARDS COMPREHENSIVE TOOL SUPPORT

In this paper, we already referred to analysis tools such as Woflan (Verbeek et al., 2001) and ProM (Dongen et al., 2005).

The focus on *Woflan* is on design-time analysis and in particular on verification. Woflan was designed to address the problem that today's workflow products have no support for workflow verification. Errors made at design-time are not detected and re-

[2] Models without any semantics are "safe" in this respect but provide no information.

sult in very costly failures at run-time. Woflan analyzes workflow process definitions downloaded from commercial workflow products using state-of-the-art Petri-net-based analysis techniques. Recently, part of the functionality of Woflan was embedded into ProM.

Initially the focus of *ProM* was on run-time analysis and in particular on process mining. However, over the last couple of years the scope of ProM has been extended to all kinds of process analysis as shown in Figure 1. For example, ProM offers several ways to verify models and can export to all kinds of tools, e.g., tools for simulation or performance analysis. It also allows for all kinds of model transformations, e.g., an EPC discovered via process mining can be converted into a Petri net or YAWL model. ProM also offers a recommendation service and allows for different types of conformance checking. Currently, ProM contains 150 plug-ins and it is impossible to give a complete overview here. Therefore, we only elaborate a bit on the plug-ins related to process discovery.

ProM implements about 20 different process discovery algorithms. Since a complete review of the different algorithms is outside the scope of this paper, we just show three examples. In Section 3.1 we already showed a Petri net discovered using the α-algorithm (Aalst et al., 2004). Figure 2 shows a process model generated by ProM based on the event log L described in Section 3.1. It is easy to see that the traces in the log can indeed be reproduced by this Petri net. Note that the α-algorithm "discovers" choices and concurrency. Although the example does not contain any loops, the α-algorithm can also discover iterations. The α-algorithm is rather sensitive to noise and exceptional behavior and has problems handling more advanced control-flow patterns. Figure 4 shows two alternative techniques that are more robust. The multi-phase miner always produces a model that can replay the log (Dongen and Aalst, 2004). It uses Event-driven Process Chains (EPCs) as a default representation as shown on the left-hand-side of Figure 4. However, the EPCs can be converted in other formats such as various types of Petri nets, YAWL models, BPEL specifications, etc. The drawback of the technique used by the multi-phase miner is that it has a tendency to overgeneralize, i.e., sometimes the model allows for too much behavior. The model shown on the right-hand-side of Figure 4 is produced by the heuristics miner (Weijters and Aalst, 2003). The heuristics miner represents processes in a notation dedicated to process mining. However, its results can be converted to other notations. The heuristics miner specializes in dealing with noise and exceptional situations, e.g., situations such as depicted in Figure 3.

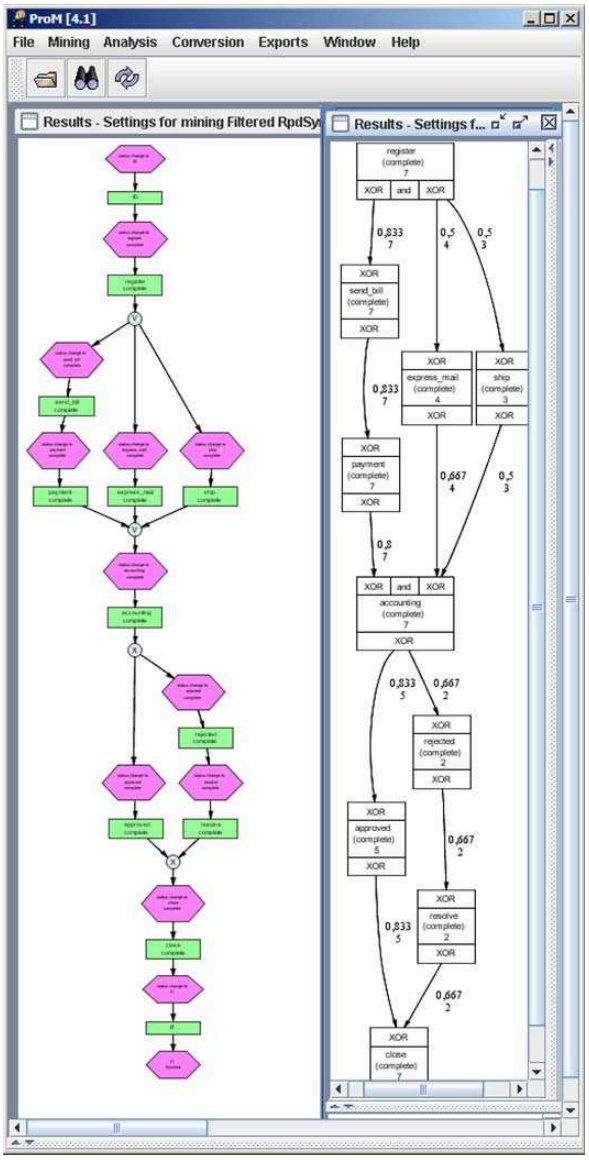

Figure 4: Two process models discovered using the multi-phase miner (left) and the heuristics miner (right).

It is not necessary to understand the three discovered models shown in figures 2 and 4. However, it is important to note that there are various process mining algorithms that perform well on structured processes. ProM offers a wide variety of process discovery techniques. Using ProM the discovered models can be converted to the desired format (Petri nets, EPCs, YAWL, etc.).

ProM is an open-source framework that can be downloaded from `www.processmining.org`.

6 CONCLUSION

In this paper, we provided a, rather personalized, overview of interesting trends in business process analysis. This overview is far from complete and the main purpose of this paper is to make a number of specific statements related to business process analysis. First of all, we argued that the practical verification of real-life processes has become a reality. The verification of the entire SAP reference model is a nice illustration of this (Mendling et al., 2006a; Mendling et al., 2006b). Second, we motivated that the abundance of event logs allows for new and exciting forms of process analysis. We provided an overview of the different types of analysis offered by process mining techniques. Third, we showed, using the slogan *"The World is NOT a Petri Net"*, that reality is often very different from what is modeled or what people think. Figure 3 shows a model describing the flow of patients having heart surgery. It shows a "spaghetti-like" model and for many other real-life processes similar-looking models are obtained. This does not imply a poor performance of the corresponding process mining techniques. It merely reflects that reality is often "spaghetti-like" and not as structured as people want to believe, i.e., reality does not fit onto a PowerPoint slide. Finally, we showed some of the analysis tools developed at Eindhoven University of Technology. We focused on the ProM framework. ProM provides an extensive set of analysis techniques which can be applied to real-life logs while supporting many parts of the spectrum depicted in Figure 1. We encourage people developing and/or using business process analysis tools to join this initiative and download the toolset from `www.processmining.org`.

ACKNOWLEDGEMENTS

This research is supported by EIT, NWO-EW, the Technology Foundation STW, and the SUPER project (FP6). Moreover, we would like to thank the many people involved in the development of ProM.

REFERENCES

Van der Aalst, W. (2004). Business Process Management Demystified: A Tutorial on Models, Systems and Standards for Workflow Management. In Desel, J., Reisig, W., and Rozenberg, G., editors, *Lectures on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 1–65. Springer-Verlag, Berlin.

Van der Aalst, W., Dreiling, A., Gottschalk, F., Rosemann, M., and Jansen-Vullers, M. (2006a). Configurable Process Models as a Basis for Reference Modeling. In Bussler et al., C., editor, *BPM 2005 Workshops (Workshop on Business Process Reference Models)*, volume 3812 of *Lecture Notes in Computer Science*, pages 512–518. Springer-Verlag, Berlin.

Van der Aalst, W., Dumas, M., Ouyang, C., Rozinat, A., and Verbeek, H. (2006b). Choreography Mining and Conformance Checking. In Leymann, F., Reisig, W., Thatte, S., and van der Aalst, W., editors, *The Role of Business Processes in Service Oriented Architectures*, number 6291 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany.

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

Van der Aalst, W. and ter Hofstede, A. (2005). YAWL: Yet Another Workflow Language. *Information Systems*, 30(4):245–275.

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

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

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

Agrawal, R., Gunopulos, D., and Leymann, F. (1998). Mining Process Models from Workflow Logs. In *Sixth International Conference on Extending Database Technology*, pages 469–483.

Cardoso, J. (2006). Process control-flow complexity metric: An empirical validation. In *IEEE International Conference on Services Computing (SCC 2006)*, pages 167–173. IEEE Computer Society.

Cardoso, J., Sheth, A., Miller, J., Arnold, J., and Kochut, K. (2004). Quality of service for workflows and web service processes. *Journal of Web Semantics*, 1(3):281–308.

Curran, T. and Keller, G. (1997). *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model*. Upper Saddle River.

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

Desel, J. and Esparza, J. (1995). *Free Choice Petri Nets*, volume 40 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, UK.

Van Dongen, B. and van der Aalst, W. (2004). Multi-Phase Process Mining: Building Instance Graphs. In Atzeni, P., Chu, W., Lu, H., Zhou, S., and Ling, T., editors, *International Conference on Conceptual Modeling (ER*

2004), volume 3288 of *Lecture Notes in Computer Science*, pages 362–376. Springer-Verlag, Berlin.

Van Dongen, B., Medeiros, A., Verbeek, H., Weijters, A., and Van der Aalst, W. (2005). The ProM framework: A New Era in Process Mining Tool Support. In Ciardo, G. and Darondeau, P., editors, *Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454. Springer-Verlag, Berlin.

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

Herbst, J. (2000). 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.

Jansen-Vullers, M., van der Aalst, W., and Rosemann, M. (2006). Mining Configurable Enterprise Information Systems. *Data and Knowledge Engineering*, 56(3):195–244.

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

Mendling, J., van der Aalst, W., Dongen, B., and Verbeek, E. (2006a). Errors in the SAP Reference Model. *BP-Trends*, 4(6):1–5.

Mendling, J., Moser, M., Neumann, G., Verbeek, H., Dongen, B., and van der Aalst, W. (2006b). Faulty EPCs in the SAP Reference Model. In Dustdar, S., Faideiro, J., and Sheth, A., editors, *International Conference on Business Process Management (BPM 2006)*, volume 4102 of *Lecture Notes in Computer Science*, pages 451–457. Springer-Verlag, Berlin.

Netjes, M., Vanderfeesten, I., and Reijers, H. (2006). "Intelligent" Tools for Workflow Process Redesign: A Research Agenda. In Bussler, C. and Haller, A., editors, *Business Process Management Workshops (BPM 2005)*, volume 3812 of *Lecture Notes in Computer Science*, pages 444–453. Springer-Verlag, Berlin.

Reijers, H. and van der Aalst, W. (1999). Short-Term Simulation: Bridging the Gap between Operational Control and Strategic Decision Making. In Hamza, M., editor, *Proceedings of the IASTED International Conference on Modelling and Simulation*, pages 417–421. IASTED/Acta Press, Anaheim, USA.

Reisig, W. and Rozenberg, G., editors (1998). *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin.

Rosemann, M. and van der Aalst, W. (2006). A Configurable Reference Modelling Language. *Information Systems*, 32(1):1–23.

Rozinat, A. and van der Aalst, W. (2006a). Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models. In Bussler et al., C., editor, *BPM 2005 Workshops (Workshop on Business Process Intelligence)*, volume 3812 of *Lecture Notes in Computer Science*, pages 163–176. Springer-Verlag, Berlin.

Rozinat, A. and van der Aalst, W. (2006b). Decision Mining in ProM. In Dustdar, S., Faideiro, J., and Sheth, A., editors, *International Conference on Business Process Management (BPM 2006)*, volume 4102 of *Lecture Notes in Computer Science*, pages 420–425. Springer-Verlag, Berlin.

Rozinat, A., Mans, R., and van der Aalst, W. (2006). Mining CPN Models: Discovering Process Models with Data from Event Logs. In Jensen, K., editor, *Proceedings of the Seventh Workshop on the Practical Use of Coloured Petri Nets and CPN Tools (CPN 2006)*, volume 579 of *DAIMI*, pages 57–76, Aarhus, Denmark. University of Aarhus.

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

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