Chapter 2
Process Mining: The Missing Link

Information systems are becoming more and more intertwined with the operational processes they support. As discussed in the previous chapter, multitudes of events are recorded by today's information systems. Nevertheless, organizations have problems extracting value from these data. The goal of *process mining* is to use event data to extract process-related information, e.g., to automatically *discover* a process model by observing events recorded by some enterprise system. A small example is used to explain the basic concepts. These concepts will be elaborated in later chapters.

2.1 Limitations of Modeling

Process mining can be viewed as the missing link between *data science* and *process science*, as demonstrated in the previous chapter using Fig. 1.7. Another way to characterize process mining is shown in Fig. 2.1. The diagram shows that process mining starts from *event data* and uses *process models* in various ways, e.g., process models are discovered from event data, serve as reference models, or are used to project bottlenecks on. Figure 2.1 shows that event data and process models can be viewed as "yin and yang" in process mining. Like in Chinese philosophy, we aim for a duality (yin and yang). *Data-driven forces* and *process-centric forces* are viewed as complementary, interconnected, and interdependent in process mining. Figure 2.1 also provides examples of questions that can be answered using process mining. These questions can be grouped into *performance* and *conformance* related questions. Clearly, such questions cannot be answered using a spreadsheet program. We later show concrete examples. However, before introducing process mining using a concrete event log, we first discuss the limitations of modeling when it comes to process analysis and process improvement.

To discuss the limitations of modeling, in particular the use of hand-made models, we briefly introduce *Petri nets* as an example language. A plethora of notations exists to model operational (business) processes (next to Petri nets there are languages like BPMN, UML, and EPCs), some of which will be discussed in the next

W. van der Aalst, *Process Mining*, DOI 10.1007/978-3-662-49851-4_2

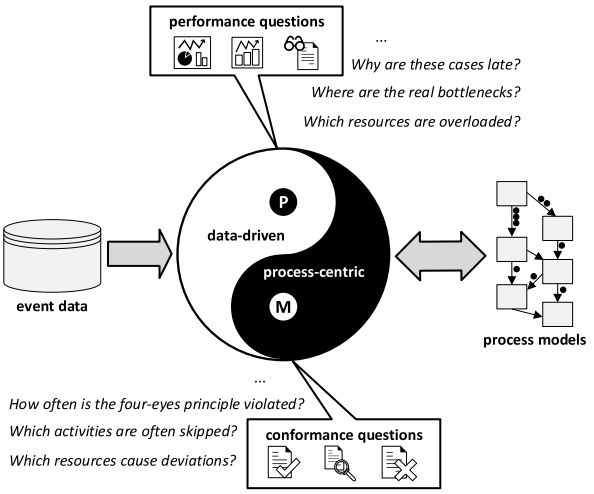

Fig. 2.1 Process mining is both data-driven and process-centric: Using a combination of event data and process models a wide range of conformance and performance questions can be answered

chapter. We refer to all of these as *process models*. The notations mentioned have in common that processes are described in terms of activities (and possibly subprocesses). The ordering of these activities is modeled by describing casual dependencies. Moreover, the process model may also describe temporal properties, specify the creation and use of data, e.g., to model decisions, and stipulate the way that resources interact with the process (e.g., roles, allocation rules, and priorities).

Figure 2.2 shows a process model expressed in terms of a *Petri net* [46]. The model describes the handling of a request for compensation within an airline. Customers may request compensation for various reasons, e.g., a delayed or canceled flight. As Fig. 2.2 shows the process starts by registering the request. This activity is modeled by transition *register request*. Each *transition* is represented by a square. Transitions are connected through *places* that model possible states of the process. Each place is represented by a circle. In a Petri net a transition is *enabled*, i.e., the corresponding activity can occur, if all input places hold a *token*. Transition *register request* has only one input place (*start*) and this place initially contains a token to represent the request for compensation. Hence, the corresponding activity is enabled and can occur. This is also referred to as *firing*. When firing, the transition consumes one token from each of its input places and produces one token for each of its output places. Hence, the firing of transition *register request* results in the removal of the token from input place *start* and the production of two tokens: one for output place *c1* and one for output place *c2*. Tokens are shown as black dots. The configuration of tokens over places—in this case the state of the request—is referred to as *marking*. Figure 2.2 shows the initial marking consisting of one token in place *start*. The marking after firing transition *register request* has two tokens: one in place *c1* and one in place *c2*. After firing transition *register request*, three transitions are enabled. The token in place *c2* enables transition *check ticket*. This transition models an administrative check to see whether the customer is eligible to issue a request. For example, while executing *check ticket* it is verified whether the customer indeed has

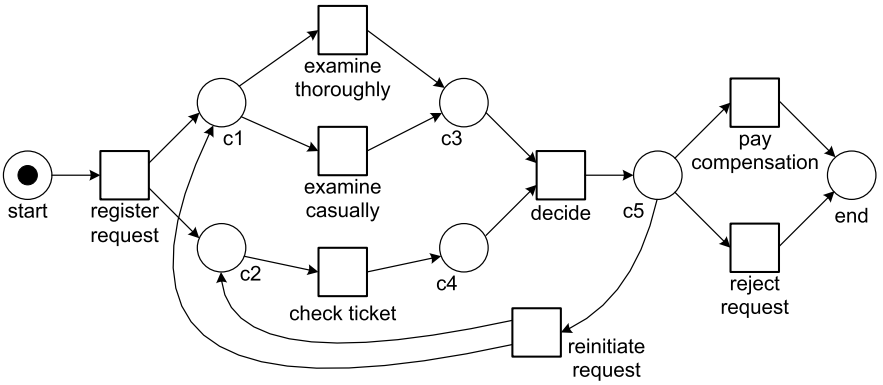

Fig. 2.2 A Petri net modeling the handling of compensation requests

a ticket issued by the airline. In parallel, the token in *c1* enables both *examine thoroughly* and *examine casually*. Firing *examine thoroughly* will remove the token from *c1*, thus disabling *examine casually*. Similarly, the occurrence of *examine casually* will disable *examine thoroughly*. In other words, there is a choice between these two activities. Transition *examine thoroughly* is executed for requests that are suspicious or complex. Straightforward requests only need a casual examination. Firing *check ticket* does not disable any other transition, i.e., it can occur concurrently with *examine thoroughly* or *examine casually*. Transition *decide* is only enabled if both input places contain a token. The ticket needs to be checked (token in place *c4*) and the casual or thorough examination of the request has been conducted (token in place *c3*). Hence, the process synchronizes before making a decision. Transition *decide* consumes two tokens and produces one token for *c5*. Three transitions share *c5* as an input place, thus modeling the three possible outcomes of the decision. The requested compensation is paid (transition *pay compensation* fires), the request is declined (transition *reject request* fires), or further processing is needed (transition *reinitiate request* fires). In the latter case the process returns to the state marking places *c1* and *c2*: transition *reinitiate request* consumes a token from *c5* and produces a token for each of its output places. This was the marking directly following the occurrence of *register request*. In principle, several iterations are possible. The process ends after paying the compensation or rejecting the request.

Process-Aware Information Systems

Process-Aware Information Systems (PAISs) include all software systems that support processes and not just isolated activities [49]. For example, ERP (Enterprise Resource Planning) systems (SAP, Oracle, etc.), BPM (Business Process Management) systems (Pegasystems, Bizagi, Appian, IBM BPM, etc.), WFM (Workflow Management) systems, CRM (Customer Relationship Management) systems, rule-based systems, call center software, high-end middle-

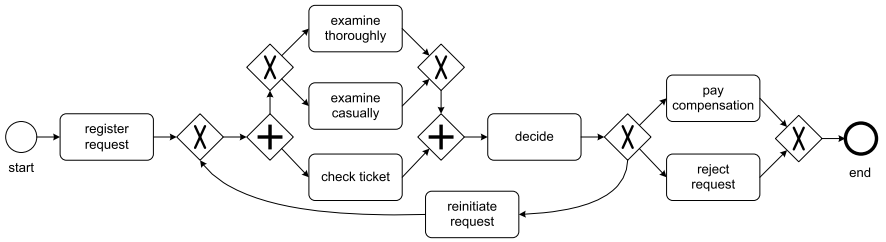

Fig. 2.3 The same process modeled in terms of BPMN

ware (WebSphere), etc. can be seen as process-aware. What these systems have in common is that there is a process notion present in the software (e.g., the completion of one activity triggers another activity) and that the information system is aware of the processes it supports (e.g., collecting information about flow times). This is very different from a database system, e-mail program, text editor, spreadsheet program, or currency transfer application. The latter set of example tools may be used to execute steps in some business process. However, these tools are not "aware" of the processes they are used in. Therefore, they cannot be actively involved in the management and orchestration of the processes they are used for.

A particular class of PAISs is formed by generic systems that are *driven by explicit process models*. Examples are BPM and WFM systems. WFM primarily focuses on the automation of business processes [76, 92, 151], whereas BPM has a broader scope: from process automation and process analysis to process management and the organization of work [50, 143, 187]. However, both BPM and WFM systems start from process models in a Petri-net or BPMN-like language. These models can be executed for any number of process instances. Changing the model corresponds (in theory) to automatically changing the process. This way flexibility and adaptability are support.

Note that ERP systems are often hybrid. They provide a WFM subsystem, but also support processes driven by (configurable) code rather than process models. What all PAISs have in common is that they can be configured in some way (through an explicit process model, via predefined settings, or using customization).

Figure 2.2 models the process as a Petri net. There exist many different notations for process models. Figure 2.3 models the same process in terms of a so-called BPMN diagram [110, 187]. The *Business Process Modeling Notation* (BPMN) uses explicit *gateways* rather than places to model the control-flow logic. The diamonds with a "×" sign denote XOR split/join gateways, whereas diamonds with a "+" sign denote AND split/join gateways. The diamond directly following activity *register request* is an XOR-join gateway. This gateway is used to be able to "jump back"

after making the decision to reinitiate the request. After this XOR-join gateway there is an AND-split gateway to model that the checking of the ticket can be done in parallel with the selected examination type (thorough or casual). The remainder of the BPMN diagram is self explanatory as the behavior is identical to the Petri net described before.

Figures 2.2 and 2.3 show only the *control-flow*, i.e., the ordering of activities for the process described earlier. This is a rather limited view on business processes. Therefore, most modeling languages offer notations for modeling other perspectives such as the organizational or resource perspective ("The decision needs to be made by a manager"), the data perspective ("After four iteration always a decision is made unless more than 1 million Euro is claimed"), and the time perspective ("After two weeks the problem is escalated"). Although there are important differences between the various process modeling languages, we do not elaborate one these in this book. Instead, we refer to the systematic comparisons in the context of the *Workflow Patterns Initiative* [155, 191]. This allows us to focus on the role that process models play in process science, rather than worrying about notation. Although process mining can be used in a variety of applications domains, we often assume a BPM context for clarity. However, the techniques in this book can be used for *all* types of (discrete) events (e.g., in healthcare logistics, luggage handling systems, software analysis, smart maintenance, website analytics, and customer journey analysis).

What are process models used for?

- *insight*: while making a model, the modeler is triggered to view the process from various angles;
- *discussion*: the stakeholders use models to structure discussions;
- *documentation*: processes are documented for instructing people or certification purposes (cf. ISO 9000 quality management);
- *verification*: process models are analyzed to find errors in systems or procedures (e.g., potential deadlocks);
- *performance analysis*: techniques like simulation can be used to understand the factors influencing response times, service levels, etc.;
- *animation*: models enable end users to "play out" different scenarios and thus provide feedback to the designer;
- *specification*: models can be used to describe a PAIS before it is implemented and can hence serve as a "contract" between the developer and the end user/management; and
- *configuration*: models can be used to configure a system.

Clearly, process models play an important role in larger organizations. When redesigning processes and introducing new information systems, process models are used for a variety of reasons. Typically, two types of models are used: (a) *informal models* and (b) *formal models* (also referred to as "executable" models). Informal models are used for discussion and documentation whereas formal models

are used for analysis or enactment (i.e., the actual execution of process). On the one end of the spectrum there are "PowerPoint diagrams" showing high-level processes whereas on the other end of the spectrum there are process models captured in executable code. Whereas informal models are typically ambiguous and vague, formal models tend to have a rather narrow focus or are too detailed to be understandable by the stakeholders. The lack of alignment between both types of models has been discussed extensively in BPM literature [49, 70, 132, 137, 154, 187, 193]. Here, we would like to provide another view on the matter. Independent of the kind of model—informal or formal—one can reflect on the alignment between model and reality. A process model used to configure a workflow management system is probably well-aligned with reality as the model is used to force people to work in a particular way. Unfortunately, most hand-made models are disconnected from reality and provide only an idealized view on the processes at hand. Moreover, also formal models that allow for rigorous analysis techniques may have little to do with the actual process.

The value of models is limited if too little attention is paid to the alignment of model and reality. Process models become "paper tigers" when the people involved cannot trust them. For example, it makes no sense to conduct simulation experiments while using a model that assumes an idealized version of the real process. It is likely that—based on such an idealized model—incorrect redesign decisions are made. It is also precarious to start a new implementation project guided by process models that hide reality. A system implemented on the basis of idealized models is likely to be disruptive and unacceptable for end users. A nice illustration is the limited quality of most *reference models*. Reference models are used in the context of large enterprise systems such as SAP [37] but also to document processes for particular branches, cf. the NVVB (Nederlandse Vereniging Voor Burgerzaken) models describing the core processes in Dutch municipalities. The idea is that "best practices" are shared among different organizations. Unfortunately, the quality of such models leaves much to be desired. For example, the SAP reference model has very little to do with the processes actually supported by SAP. In fact, more than 20 percent of the SAP models contain serious flaws (deadlocks, livelocks, etc.) [101]. Such models are not aligned with reality and, thus, have little value for end users.

Given (a) the interest in process models, (b) the abundance of event data, and (c) the limited quality of hand-made models, it seems worthwhile to relate event data to process models. This way the actual processes can be discovered and existing process models can be evaluated and enhanced. This is precisely what process mining aims to achieve.

2.2 Process Mining

To position process mining, we first describe the so-called *BPM life-cycle* using Fig. 2.4. The life-cycle describes the different phases of managing a particular business process. In the *design* phase, a process is designed. This model is transformed

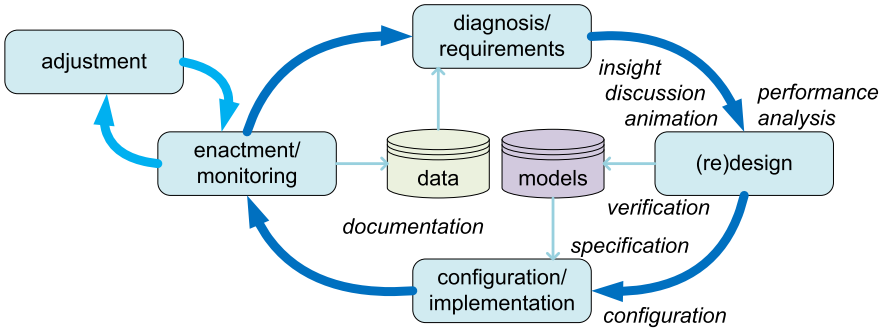

Fig. 2.4 The BPM life-cycle showing the different uses of process models

into a running system in the *configuration/implementation* phase. If the model is already in executable form and a WFM or BPM system is already running, this phase may be very short. However, if the model is informal and needs to be hard-coded in conventional software, this phase may take substantial time. After the system supports the designed processes, the *enactment/monitoring* phase starts. In this phase, the processes are running while being monitored by management to see if any changes are needed. Some of these changes are handled in the *adjustment* phase shown in Fig. 2.4. In this phase, the process is not redesigned and no new software is created; only predefined controls are used to adapt or reconfigure the process. The *diagnosis/requirements* phase evaluates the process and monitors emerging requirements due to changes in the environment of the process (e.g., changing policies, laws, competition). Poor performance (e.g., inability to meet service levels) or new demands imposed by the environment may trigger a new iteration of the BPM life-cycle starting with the *redesign* phase.

As Fig. 2.4 shows, process models play a dominant role in the (re)design and configuration/implementation phases, whereas data plays a dominant role in the enactment/monitoring and diagnosis/requirements phases. The figure also lists the different ways in which process models are used (as identified in Sect. 2.1). Until recently, there were few connections between the data produced while executing the process and the actual process design. In fact, in most organizations the diagnosis/requirements phase is not supported in a systematic and continuous manner. Only severe problems or major external changes will trigger another iteration of the life-cycle, and factual information about the current process is not actively used in redesign decisions. Process mining offers the possibility to truly "close" the BPM life-cycle. Data recorded by information systems can be used to provide a better view on the actual processes, i.e., deviations can be analyzed and the quality of models can be improved.

Process mining is a relative young research discipline that sits between machine learning and data mining on the one hand and process modeling and analysis on the other hand. The idea of process mining is to discover, monitor and improve real processes (i.e., not assumed processes) by extracting knowledge from event logs readily available in today's systems.

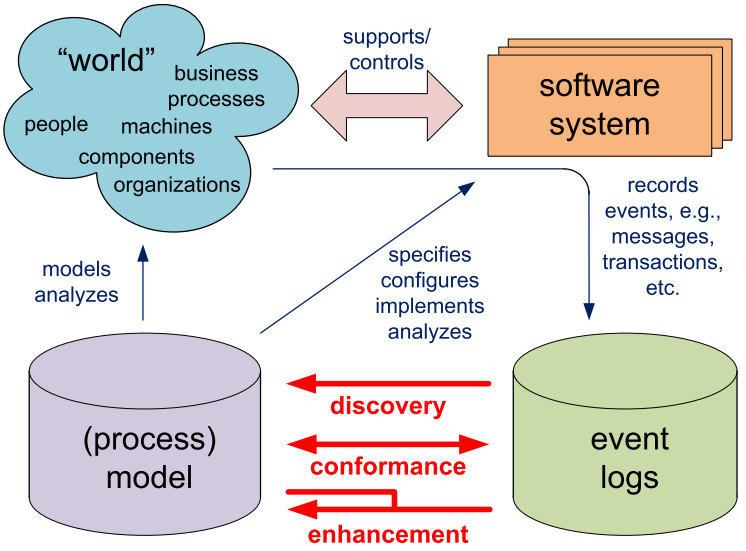

Fig. 2.5 Positioning of the three main types of process mining: *discovery*, *conformance*, and *enhancement*

Figure 2.5 shows that process mining establishes links between the actual processes and their data on the one hand and process models on the other hand. As explained in the previous chapter, the digital universe and the physical universe become more and more aligned. Today's information systems log enormous amounts of events. Classical WFM systems (e.g., Staffware and COSA), BPM systems (e.g., BPM|one by Pallas Athena, SmartBPM by Pegasystems, FileNet, Global 360, and Teamwork by Lombardi Software), ERP systems (e.g., SAP Business Suite, Oracle E-Business Suite, and Microsoft Dynamics NAV), PDM systems (e.g., Windchill), CRM systems (e.g., Microsoft Dynamics CRM and SalesForce), middleware (e.g., IBM's WebSphere and Cordys Business Operations Platform), and hospital information systems (e.g., Chipsoft and Siemens Soarian) provide detailed information about the activities that have been executed. Figure 2.5 refers to such data as *event logs*. All of the PAISs just mentioned directly provide such event logs. However, most information systems store such information in unstructured form, e.g., event data is scattered over many tables or needs to be tapped off from subsystems exchanging messages. In such cases, event data exist but some efforts are needed to extract them. Data extraction is an integral part of any process mining effort.

Let us 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). Consider, for example, the handling of requests for compensation modeled in Fig. 2.2. The cases are individual requests and per case a *trace* of events can be recorded. An example of a possible trace is $\langle$*register request*, *examine casually*, *check ticket*, *decide*, *reinitiate request*, *check ticket*, *examine thoroughly*, *decide*, *pay compensation*$\rangle$. Here activity names are used

to identify events. However, there are two *decide* events that occurred at different times (the fourth and eighth event of the trace), produced different results, and may have been conducted by different people. Obviously, it is important to distinguish these two decisions. Therefore, most event logs store additional information about events. In fact, whenever possible, process mining techniques use extra information such as the *resource* (i.e., person or device) executing or initiating the activity, the *timestamp* of the event, or *data elements* recorded with the event (e.g., the size of an order).

Event logs can be used to conduct three types of process mining as shown in Fig. 2.5.

The first type of process mining is *discovery*. A discovery technique takes an event log and produces a model without using any a-priori information. An example is the α-algorithm [157] that will be described in Chap. 6. This algorithm takes an event log and produces a Petri net explaining the behavior recorded in the log. For example, given sufficient example executions of the process shown in Fig. 2.2, the α-algorithm is able to automatically construct the Petri net without using any additional knowledge. If the event log contains information about resources, one can also discover resource-related models, e.g., a social network showing how people work together in an organization.

The second type of process mining is *conformance*. Here, an existing process model is compared with an event log of the same process. Conformance checking can be used to check if reality, as recorded in the log, conforms to the model and vice versa. For instance, there may be a process model indicating that purchase orders of more than one million Euro require two checks. Analysis of the event log will show whether this rule is followed or not. Another example is the checking of the so-called "four-eyes" principle stating that particular activities should not be executed by one and the same person. By scanning the event log using a model specifying these requirements, one can discover potential cases of fraud. Hence, conformance checking may be used to detect, locate and explain deviations, and to measure the severity of these deviations. An example is the conformance checking algorithm described in [121]. Given the model shown in Fig. 2.2 and a corresponding event log, this algorithm can quantify and diagnose deviations.

The third type of process mining is *enhancement*. Here, the idea is to extend or improve an existing process model using information about the actual process recorded in some event log. Whereas conformance checking measures the alignment between model and reality, this third type of process mining aims at changing or extending the a-priori model. One type of enhancement is *repair*, i.e., modifying the model to better reflect reality. For example, if two activities are modeled sequentially but in reality can happen in any order, then the model may be corrected to reflect this. Another type of enhancement is *extension*, i.e., adding a new perspective to the process model by cross-correlating it with the log. An example is the extension of a process model with performance data. For instance, by using timestamps in the event log of the "request for compensation" process, one can extend Fig. 2.2 to show bottlenecks, service levels, throughput times, and frequencies. Similarly, Fig. 2.2 can be extended with information about resources, decision rules, quality metrics, etc.

As indicated earlier, process models such as depicted in Figs. 2.2 and 2.3 show only the control-flow. However, when extending process models, additional perspectives are added. Moreover, discovery and conformance techniques are not limited to control-flow. For example, one can discover a social network and check the validity of some organizational model using an event log. Hence, orthogonal to the three types of mining (discovery, conformance, and enhancement), different perspectives can be identified.

In the remainder, we consider the following *perspectives*.

- The *control-flow 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 or some other notation (e.g., EPCs, BPMN, and UML ADs).
- The *organizational perspective* focuses on information about resources hidden in the log, i.e., which actors (e.g., people, systems, roles, and departments) 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 the social network.
- The *case perspective* focuses on properties of cases. Obviously, a case can be characterized by its path in the process or by the originators working on it. 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.
- The *time perspective* is concerned with the timing and frequency of events. When events bear timestamps it is possible to discover bottlenecks, measure service levels, monitor the utilization of resources, and predict the remaining processing time of running cases.

Note that the different perspectives are partially overlapping and non-exhaustive. Nevertheless, they provide a good characterization of the aspects that process mining aims to analyze.

In most examples given thus far it is assumed that process mining is done *off-line*, i.e., processes are analyzed afterward to see how they can be improved or better understood. However, more and more process mining techniques can also be used in an *online* setting. We refer to this as *operational support*. An example is the detection of non-conformance at the moment the deviation actually takes place. Another example is time prediction for running cases, i.e., given a partially executed case the remaining processing time is estimated based on historic information of similar cases. This illustrates that the "process mining spectrum" is broad and not limited to process discovery. In fact, today's process mining techniques are indeed able to support the whole BPM life-cycle shown in Fig. 2.4. Process mining is not only relevant for the design and diagnosis/requirements phases, but also for the enactment/monitoring and adjustment phases.

2.3 Analyzing an Example Log

After providing an overview of process mining and positioning it in the broader BPM discipline, we use the event log shown in Table 2.1 to clarify some of the foundational concepts. The table shows just a fragment of a possible log corresponding to the handling of requests for compensation. Each line presents one event. Note that events are already grouped per case. Case 1 has five associated events. The first event of Case 1 is the execution of activity *register request* by Pete on December 30th 2010. Table 2.1 also shows a unique id for this event: 35654423. This is merely used for the identification of the event, e.g., to distinguish it from event 35654483 that also corresponds to the execution of activity *register request* (first event of second case). Table 2.1 shows a date and a timestamp for each event. In some event logs this information is more coarse-grained and only a date or partial ordering of events is given. In other logs there may be more elaborate timing information also showing when the activity was started, when it was completed, and sometimes even when it was offered to the resource. The times shown in Table 2.1 should be interpreted as completion times. In this particular event log, activities are considered to be atomic and the table does not reveal the duration of activities. In the table, each event is associated to a resource. In some event logs this information will be missing. In other logs more detailed information about resources may be stored, e.g., the role a resource has or elaborate authorization data. The table also shows the costs associated to events. This is an example of a data attribute. There may be many other data attributes. For example, in this particular example it would be interesting to record the outcome of the different types of examinations and checks. Another data element that could be useful for analysis is the amount of compensation requested. This could be an attribute of the whole case or stored as an attribute of the *register request* event.

Table 2.1 illustrates the typical information present in an event log. Depending on the process mining technique used and the questions at hand, only part of this information is used. The minimal requirements for process mining are that any event can be related to both a case and an activity and that events within a case are ordered. Hence, the "case id" and "activity" columns in Table 2.1 represent the bare minimum for process mining. By projecting the information in these two columns we obtain the more compact representation shown in Table 2.2. In this table, each case is represented by a sequence of activities also referred to as *trace*. For clarity the activity names have been transformed into single-letter labels, e.g., a denotes activity *register request*.

Process mining algorithms for process discovery can transform the information shown in Table 2.2 into process models. For instance, the basic α-algorithm [157] discovers the Petri net described earlier when providing it with the input data in Table 2.2. Figure 2.6 shows the resulting model with the compact labels just introduced. It is easy to check that all six traces in Table 2.2 are possible in the model. Let us replay the trace of the first case—$\langle a, b, d, e, h \rangle$—to show that the trace "fits" (i.e., conforms to) the model. In the initial marking shown in Fig. 2.6, a is indeed enabled because of the token in *start*. After firing a places *c1* and *c2* are marked,

Table 2.1 A fragment of some event log: each line corresponds to an event

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
1	35654423	30-12-2010:11.02	register request	Pete	50	...
	35654424	31-12-2010:10.06	examine thoroughly	Sue	400	...
	35654425	05-01-2011:15.12	check ticket	Mike	100	...
	35654426	06-01-2011:11.18	decide	Sara	200	...
	35654427	07-01-2011:14.24	reject request	Pete	200	...
2	35654483	30-12-2010:11.32	register request	Mike	50	...
	35654485	30-12-2010:12.12	check ticket	Mike	100	...
	35654487	30-12-2010:14.16	examine casually	Pete	400	...
	35654488	05-01-2011:11.22	decide	Sara	200	...
	35654489	08-01-2011:12.05	pay compensation	Ellen	200	...
3	35654521	30-12-2010:14.32	register request	Pete	50	...
	35654522	30-12-2010:15.06	examine casually	Mike	400	...
	35654524	30-12-2010:16.34	check ticket	Ellen	100	...
	35654525	06-01-2011:09.18	decide	Sara	200	...
	35654526	06-01-2011:12.18	reinitiate request	Sara	200	...
	35654527	06-01-2011:13.06	examine thoroughly	Sean	400	...
	35654530	08-01-2011:11.43	check ticket	Pete	100	...
	35654531	09-01-2011:09.55	decide	Sara	200	...
	35654533	15-01-2011:10.45	pay compensation	Ellen	200	...
4	35654641	06-01-2011:15.02	register request	Pete	50	...
	35654643	07-01-2011:12.06	check ticket	Mike	100	...
	35654644	08-01-2011:14.43	examine thoroughly	Sean	400	...
	35654645	09-01-2011:12.02	decide	Sara	200	...
	35654647	12-01-2011:15.44	reject request	Ellen	200	...
5	35654711	06-01-2011:09.02	register request	Ellen	50	...
	35654712	07-01-2011:10.16	examine casually	Mike	400	...
	35654714	08-01-2011:11.22	check ticket	Pete	100	...
	35654715	10-01-2011:13.28	decide	Sara	200	...
	35654716	11-01-2011:16.18	reinitiate request	Sara	200	...
	35654718	14-01-2011:14.33	check ticket	Ellen	100	...
	35654719	16-01-2011:15.50	examine casually	Mike	400	...
	35654720	19-01-2011:11.18	decide	Sara	200	...
	35654721	20-01-2011:12.48	reinitiate request	Sara	200	...
	35654722	21-01-2011:09.06	examine casually	Sue	400	...
	35654724	21-01-2011:11.34	check ticket	Pete	100	...
	35654725	23-01-2011:13.12	decide	Sara	200	...
	35654726	24-01-2011:14.56	reject request	Mike	200	...

Table 2.1 (Continued)

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
6	35654871	06-01-2011:15.02	register request	Mike	50	...
	35654873	06-01-2011:16.06	examine casually	Ellen	400	...
	35654874	07-01-2011:16.22	check ticket	Mike	100	...
	35654875	07-01-2011:16.52	decide	Sara	200	...
	35654877	16-01-2011:11.47	pay compensation	Mike	200	...
...	...	...	...	...	...	...

Table 2.2 A more compact representation of log shown in Table 2.1: $a =$ *register request*, $b =$ *examine thoroughly*, $c =$ *examine casually*, $d =$ *check ticket*, $e =$ *decide*, $f =$ *reinitiate request*, $g =$ *pay compensation*, and $h =$ *reject request*

Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

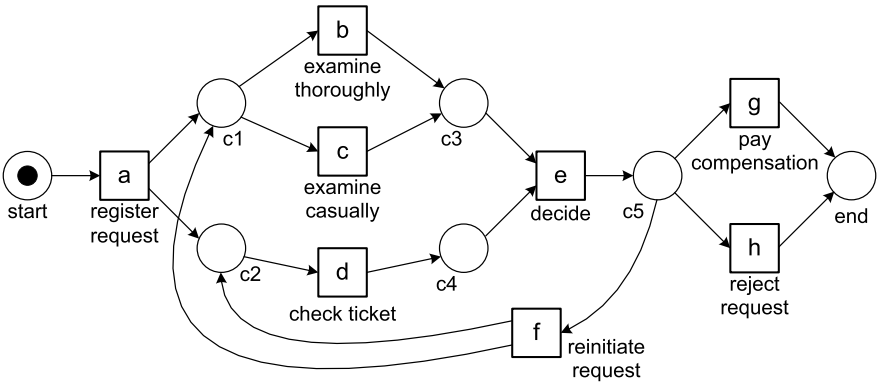

Fig. 2.6 The process model discovered by the α-algorithm [157] based on the set of traces $\{\langle a, b, d, e, h \rangle, \langle a, d, c, e, g \rangle, \langle a, c, d, e, f, b, d, e, g \rangle, \langle a, d, b, e, h \rangle, \langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle, \langle a, c, d, e, g \rangle\}$

i.e., both places contain a token. b is enabled at this marking and its execution results in the marking with tokens in *c2* and *c3*. Now we have executed $\langle a, b \rangle$ and the sequence $\langle d, e, h \rangle$ remains. The next event d is indeed enabled and its execution results in the marking enabling e (tokens in places *c3* and *c4*). Firing e results in the marking with one token in *c5*. This marking enables the final event h in the trace. After executing h, the case ends in the desired final marking with just a token in

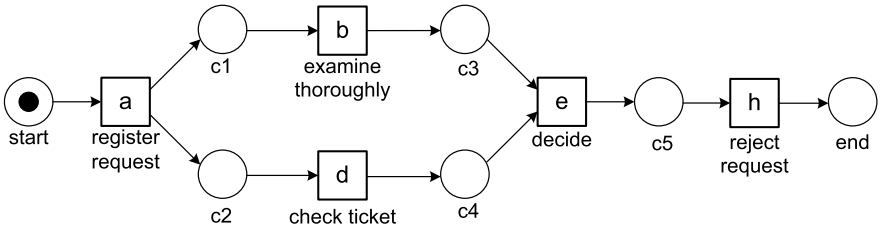

Fig. 2.7 The process model discovered by the α-algorithm based on cases 1 and 4, i.e., the set of traces $\{\langle a, b, d, e, h\rangle, \langle a, d, b, e, h\rangle\}$

place *end*. Similarly, it can be checked that the other five traces shown in Table 2.2 are also possible in the model and that all of these traces result in the marking with just a token in place *end*.

The Petri net shown in Fig. 2.6 also allows for traces not present in Table 2.2. For example, the traces $\langle a, d, c, e, f, b, d, e, g\rangle$ and $\langle a, c, d, e, f, c, d, e, f, c, d, e, f, c, d, e, f, b, d, e, g\rangle$ are also possible. This is a desired phenomenon as the goal is *not* to represent just the *particular set of example traces* in the event log. Process mining algorithms need to generalize the behavior contained in the log to show the most likely underlying model that is not invalidated by the next set of observations. One of the challenges of process mining is to balance between "overfitting" (the model is too specific and only allows for the "accidental behavior" observed) and "underfitting" (the model is too general and allows for behavior unrelated to the behavior observed).

When comparing the event log and the model, there seems to be a good balance between "overfitting" and "underfitting". All cases start with a and end with either g or h. Every e is preceded by d and one of the examination activities (b or c). Moreover, e is followed by f, g, or h. The repeated execution of b or c, d, and e suggests the presence of a loop. These characteristics are adequately captured by the net of Fig. 2.6.

Let us now consider an event log consisting of only two traces $\langle a, b, d, e, h\rangle$ and $\langle a, d, b, e, h\rangle$, i.e., cases 1 and 4 of the original log. For this log, the α-algorithm constructs the Petri net shown in Fig. 2.7. This model only allows for two traces and these are exactly the ones in the small event log. b and d are modeled as being concurrent because they can be executed in any order. For larger and more complex models it is important to discover concurrency. Not modeling concurrency typically results in large "Spaghetti-like" models in which the same activity needs to be duplicated.[1]

The α-algorithm is just one of many possible process discovery algorithms. For real-life logs more advanced algorithms are needed to better balance between "overfitting" and "underfitting" and to deal with "incompleteness" (i.e., logs containing

[1] See, for example, Figs. 14.1 and 14.10 to understand why we use the term "Spaghetti" to refer to models that are difficult to comprehend.

Table 2.3 Another event log: cases 7, 8, and 10 are not possible according to Fig. 2.6

Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
7	$\langle \mathbf{a}, \mathbf{b}, \mathbf{e}, \mathbf{g} \rangle$
8	$\langle \mathbf{a}, \mathbf{b}, \mathbf{d}, \mathbf{e} \rangle$
9	$\langle a, d, c, e, f, d, c, e, f, b, d, e, h \rangle$
10	$\langle \mathbf{a}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{b}, \mathbf{d}, \mathbf{g} \rangle$

only a small fraction of the possible behavior due to the large number of alternatives) and "noise" (i.e., logs containing exceptional/infrequent behavior that should not automatically be incorporated in the model). This book will describe several of such algorithms and guide the reader in selecting one. In this section, we used Petri nets to represent the discovered process models, because Petri nets are a succinct way of representing processes and have unambiguous and simple semantics. However, most mining techniques are independent of the desired representation. For instance, the discovered Petri net model shown in Fig. 2.6 can be (automatically) transformed into the BPMN model shown in Fig. 2.3.

As explained in Sect. 2.2, process mining is not limited to process discovery. Event logs can be used to check conformance and enhance existing models. Moreover, different perspectives may be taken into account. To illustrate this, let us first consider the event log shown in Table 2.3. The first six cases are as before. It is easy to see that Case 7 with trace $\langle a, b, e, g \rangle$ is not possible according to the model in Fig. 2.6. The model requires the execution of d before e, but d did not occur. This means that the ticket was not checked at all before making a decision and paying compensation. Conformance checking techniques aim at discovering such discrepancies [121]. When checking the conformance of the remainder of the event log it can also be noted that cases 8 and 10 do not conform either. Case 9 conforms although it is not identical to one of the earlier traces. Trace $\langle a, b, d, e \rangle$ (i.e., Case 8) has the problem that no concluding action was taken (rejection or payment). Trace $\langle a, c, d, e, f, b, d, g \rangle$ (Case 10) has the problem that the airline paid compensation without making a final decision. Note that conformance can be viewed from two angles: (a) the model does not capture the real behavior ("the model is wrong") and (b) reality deviates from the desired model ("the event log is wrong"). The first viewpoint is taken when the model is supposed to be *descriptive*, i.e., capture or predict reality. The second viewpoint is taken when the model is *normative*, i.e., used to influence or control reality.

The original event log shown in Table 2.1 also contains information about resources, timestamps and costs. Such information can be used to discover other perspectives, check the conformance of models that are not pure control-flow models, and to extend models with additional information. For example, one could derive

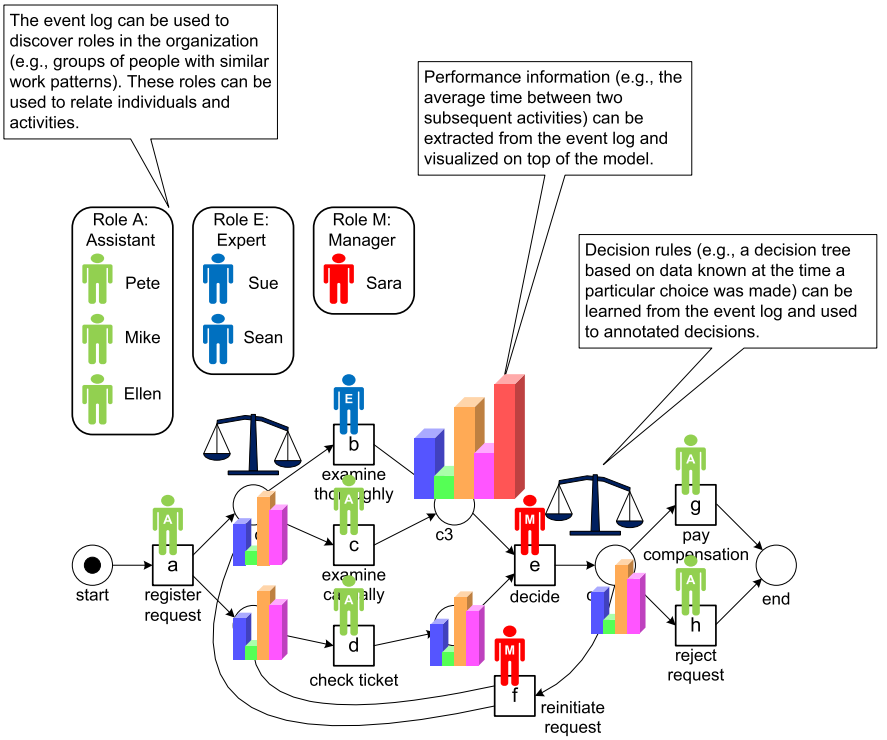

Fig. 2.8 The process model extended with additional perspectives: the organizational perspective ("What are the organizational roles and which resources are performing particular activities?"), the case perspective ("Which characteristics of a case influence a particular decision?"), and the time perspective ("Where are the bottlenecks in my process?")

a social network based on the interaction patterns between individuals. The social network can be based on the "handover of work" metric, i.e., the more frequent individual x performed an activity that is causally followed by an activity performed by individual y, the stronger the relation between x and y is [159].

Figure 2.8 illustrates the way in which a control-flow oriented model can be extended with the three other main perspectives mentioned in Sect. 2.2. Analysis of the event log shown in Table 2.1 may reveal that Sara is the only one performing the activities *decide* and *reinitiate* request. This suggests that there is a "manager role" and that Sara is the only one having this role. Activity *examine thoroughly* is performed only by Sue and Sean. This suggests some "expert role" associated to this activity. The remaining activities are performed by Pete, Mike and Ellen. This suggests some "assistant role" as shown in Fig. 2.8. Techniques for organizational process mining [130] will discover such organizational structures and relate activities to resources through roles. By exploiting resource information in the log, the organizational perspective can be added to the process model. Similarly, information on timestamps and frequencies can be used to add performance related information

to the model. Figure 2.8 sketches that it is possible to measure the time that passes between an examination (activities *b* or *c*) and the actual decision (activity *e*). If this time is remarkably long, process mining can be used to identify the problem and discover possible causes. If the event log contains case-related information, this can be used to further analyze the decision points in the process. For instance, through decision point analysis it may be learned that requests for compensation of more than € 800 tend to be rejected.

Using process mining, the different perspectives can be cross-correlated to find surprising insights. Examples of such findings could be: "requests examined by Sean tend to be rejected more frequently", "requests for which the ticket is checked after examination tend to take much longer", "requests of less than € 500 tend to be completed without any additional iterations". Moreover, these perspectives can also be linked to conformance questions. For example, it may be shown that Pete is involved in relatively many incorrectly handled requests. These examples show that privacy issues need to be considered when analyzing event logs with information about individuals (see Sect. 9.3.3).

2.4 Play-In, Play-Out, and Replay

One of the key elements of process mining is the emphasis on establishing a strong relation between a process model and "reality" captured in the form of an event log. Inspired by the terminology used by David Harel in the context of Live Sequence Charts [70], we use the terms *Play-In*, *Play-Out*, and *Replay* to reflect on this relation. Figure 2.9 illustrates these three notions.

Play-Out refers to the classical use of process models. Given a Petri net, it is possible to generate behavior. The traces in Table 2.2 could have been obtained by repeatedly "playing the token game" using the Petri net of Figure 2.6. Play-Out can be used both for the analysis and the enactment of business processes. A workflow engine can be seen as a "Play-Out engine" that controls cases by only allowing the "moves" allowed according to the model. Hence, Play-Out can be used to enact operational processes using some executable model. Simulation tools also use a Play-Out engine to conduct experiments. The main idea of simulation is to repeatedly run a model and thus collect statistics and confidence intervals. Note that a simulation engine is similar to a workflow engine. The main difference is that the simulation engine interacts with a modeled environment whereas the workflow engine interacts with the real environment (workers, customers, etc.). Also classical verification approaches using exhaustive state-space analysis—often referred to as model checking [30]—can be seen as Play-Out methods.

Play-In is the opposite of Play-Out, i.e., example behavior is taken as input and the goal is to construct a model. Play-In is often referred to as *inference*. The α-algorithm and other process discovery approaches are examples of Play-In techniques. Note that the Petri net of Fig. 2.6 can be derived automatically given an event log like the one in Table 2.2. Most data mining techniques use Play-In, i.e.,

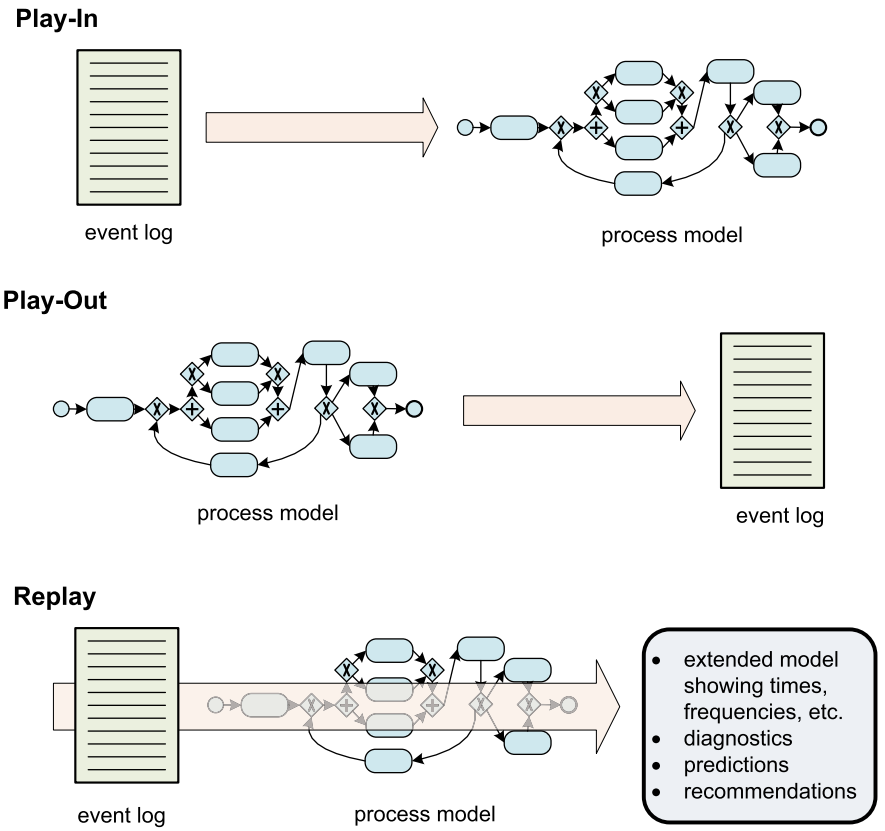

Fig. 2.9 Three ways of relating event logs (or other sources of information containing example behavior) and process models: *Play-In*, *Play-Out*, and *Replay*

a model is learned on the basis of examples. However, traditionally, data mining has not been concerned with process models. Typical examples of models are decision trees ("people that drink more than five glasses of alcohol and smoke more than 56 cigarettes tend to die young") and association rules ("people that buy diapers also buy beer"). Unfortunately, it is not possible to use conventional data mining techniques to Play-In process models. Only recently, process mining techniques have become readily available to discover process models based on event logs.

Replay uses an event log *and* a process model as input. The event log is "replayed" on top of the process model. As shown earlier it is possible to replay trace $\langle a, b, d, e, h \rangle$ on the Petri net in Fig. 2.6; simply "play the token game" by forcing the transitions to fire (if possible) in the order indicated. An event log may be replayed for different purposes:

- *Conformance checking*: discrepancies between the log and the model can be detected and quantified by replaying the log. For instance, replaying trace

$\langle a, b, e, h \rangle$ on the Petri net in Fig. 2.6 will show that d should have happened but did not.

- *Extending the model with frequencies and temporal information.* By replaying the log one can see which parts of the model are visited frequently. Replay can also be used to detect bottlenecks. Consider, for example, the trace $\langle a^8, b^9, d^{20}, e^{21}, h^{21} \rangle$ in which the superscripts denote timestamps. By replaying the trace on top of Fig. 2.6 one can see that e was enabled at time 20 and occurred at time 21. The enabling of e was delayed by the time it took to complete d; although d was enabled already at time 8, it occurred only at time 20.
- *Constructing predictive models.* By replaying event logs one can build predictive models, i.e., for the different states of the model particular predictions can be made. For example, a predictive model learned by replaying many cases could show that the expected time until completion after enabling e is eight hours.
- *Operational support.* Replay is not limited to historic event data. One can also replay partial traces of cases still running. This can be used for detecting deviations at run-time, e.g., the partial trace $\langle a^8, e^{11} \rangle$ of a case that is still running will never fit into Fig. 2.6. Hence, an alert can be generated before the case completes. Similarly, it is possible to predict the remaining processing time or the likelihood of being rejected of a case having a partial trace, e.g., a partial executed case $\langle a^8, b^9 \rangle$ has an expected remaining processing time of 3.5 days and a 40 percent probability of being rejected. Such predictions can also be used to recommend suitable next steps to progress the case.

Desire lines in process models

A desire line—also known as the social trail—is a path that emerges through erosion caused by footsteps of humans (or animals). The width and amount of erosion of the path indicates how frequently the path is used. Typically, the desire line follows the shortest or most convenient path between two points. Moreover, as the path emerges more people are encouraged to use it, thus stimulating further erosion. Dwight Eisenhower is often mentioned as one of the persons using this emerging group behavior. Before becoming the 34th president of the United States, he was the president of Columbia University. When he was asked how the university should arrange the sidewalks to best interconnect the campus buildings, he suggested letting the grass grow between buildings and delay the creation of sidewalks. After some time the desire lines revealed themselves. The places where the grass was most worn by people's footsteps were turned into sidewalks. In the same vein, replay can be used to show the *desire lines in processes*. The paths in the process model traveled most can be highlighted by using brighter colors or thicker arcs (cf. ProM's Fuzzy Miner [66]).

An interesting question is how desire lines can be used to better manage business processes. Operational support, e.g., predictions and recommendations derived from historic information, can be used to reinforce successful behavior and thus create suitable "sidewalks" in processes.

2.5 Positioning Process Mining

The process mining spectrum is quite broad and extends far beyond process discovery and conformance checking. Process mining also connects data science and process science (see Fig. 1.7). As a result, it is inevitable that process mining objectives are overlapping with those of other approaches, methodologies, principles, methods, tools, and paradigms. For example, some will argue that "process mining is part of data mining", but discussions on such inclusion relations are seldom useful and are often politically motivated. Most data mining tools do *not* provide process mining capabilities, most data mining books do *not* describe process mining techniques, and it seems that process mining techniques like conformance checking do *not* fit in any of the common definitions of data mining. It is comparable to claiming that "data mining is part of statistics". Taking the transitive closure of both statements, we would even be able to conclude that process mining is part of statistics. Obviously, this does not make any sense. Making definitions all-encompassing does not help to provide actual analysis capabilities. Nevertheless, it is important to position process mining in the context of existing technologies and management approaches.

2.5.1 How Process Mining Compares to BPM

Business Process Management (BPM) is the discipline that combines approaches for the design, execution, control, measurement and optimization of business processes. Process mining can be best related to BPM by looking at the so-called BPM life-cycle in Fig. 2.4. Initially, the main focus of BPM was on process design and implementation [143]. Process modeling plays a key role in the (re)design phase and directly contributes to the configuration/implementation phase. Originally, BPM approaches had a tendency to be model-driven without considering the "evidence" hidden in the data.

There is now a clear trend in the BPM community to focus more on the enactment/monitoring, adjustment, and diagnosis/requirements phases. These phases are more data-driven and process mining techniques are frequently used in this part of the BPM life-cycle. Hence, process mining can easily be positioned in Fig. 2.4. However, process mining is *not* limited to BPM. Any process for which events can be recorded, is a candidate for process mining.

Learning more about Business Process Management (BPM)
Although process mining is not limited to BPM, both are clearly related. Hence, the interested reader may want to read more on BPM. Developments in BPM have resulted in a well-established set of principles, methods and tools that combine knowledge from information technology, management sciences and industrial engineering for the purpose of improving business processes. BPM can be viewed as a continuation of the Workflow Management (WFM) wave in the 1990s. The survey paper [143] structures the BPM field using 20 *BPM Use Cases* and describes the development of the field since the late 1970s. For more details, we refer to the following BPM/WFM books that served as milestones in the evolution of the field:

- *Workflow Management: Modeling Concepts, Architecture, and Implementation* [76]: first comprehensive WFM book focusing on the different workflow perspectives and the MOBILE language,
- *Production Workflow: Concepts and Techniques* [92]: book on production WFM systems closely related to IBM's workflow products,
- *Business Process Management: Models, Techniques, and Empirical Studies* [152]: edited book that served as the basis for the BPM conference series,
- *Workflow Management: Models, Methods, and Systems* [151]: most cited WFM book using a Petri net-based approach to model, analyze and enact workflow processes,
- *Workflow-based Process Controlling: Foundation, Design and Application of workflow-driven Process Information Systems* [192]: book relating WFM systems to operational performance,
- *Process-Aware Information Systems: Bridging People and Software through Process Technology* [49]: edited book on process-aware information systems,
- *Business Process Management: The Third Wave* [127]: visionary book linking management perspectives to the π-calculus,
- *Business Process Management: Concepts, Languages, Architectures* [187]: book presenting the foundations of BPM, including different languages and architectures,
- *Modern Business Process Automation: YAWL and its Support Environment* [132]: book based on YAWL and the workflow patterns,
- *Handbooks on Business Process Management* [180, 181]: edited handbooks covering the broader BPM discipline,
- *Process Management: A Guide for the Design of Business Processes* [18]: book on the design of process-oriented organizations,
- *Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, Technologies* [115]: book on supporting flexibility in process-aware information systems, and
- *Fundamentals of Business Process Management* [50]: tutorial-style book covering the whole BPM life-cycle.

2.5.2 How Process Mining Compares to Data Mining

Data mining techniques aim to analyze (often large) data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner [69]. Like process mining, data mining is *data-driven*. However, unlike process mining, mainstream data mining techniques are typically *not process-centric*. Process models expressed in terms of Petri nets or BPMN diagrams cannot be discovered or analyzed in any way by the main data mining tools.

There are a few data mining techniques that come close to process mining. Examples are sequence and episode mining. However, these techniques do not consider end-to-end processes. Through process mining, it becomes easier to apply data mining techniques to event data. For example, decision rules can be learned using standard data mining tools after the control-flow backbone (e.g., a Petri net) has been learned using a process mining tool. *RapidProM*, available through the RapidMiner Marketplace, shows that process mining and data mining can be combined in various ways. Chapter 4 discusses the relation in more detail.

2.5.3 How Process Mining Compares to Lean Six Sigma

Lean Six Sigma is a methodology that combines ideas from *lean manufacturing* and *Six Sigma*. The idea is to improve performance by systematically removing waste. Lean principles originate from the Japanese manufacturing industry. The *Toyota Production System* (TPS) is a well-known example of a lean manufacturing approach developed by Taiichi Ohno and Eiji Toyoda between 1948 and 1975. The main objectives of the TPS are to eliminate "muri" (overburdening of people and equipment), "mura" (unevenness in operations), and "muda" (waste). The emphasis is on waste ("muda") reduction. Typically, seven types of waste are mentioned in this context [109]:

- *Transportation waste*: Each time a product is moved, it encounters the risk of being damaged, lost, delayed, etc. Transportation does not make any transformation to the product that the consumer is willing to pay for (except for the final delivery).
- *Inventory waste*: Inventory may exist in the form of raw materials, work-in-progress, or finished goods. Inventory that is not being actively processed can be considered as waste because it consumes capital and space.
- *Motion waste*: Resources (equipment and people) that are used in the production processes suffer from "wear and tear". Unnecessary activities (e.g., transformation and double work) result in additional degradation of resources and increase the risk of incidents (e.g., accidents).
- *Unnecessary waiting*: Whenever goods are not in transport or being processed, they are waiting. In traditional processes, a large part of an individual product's life is spent waiting to be worked on. The total flow time of a case is often orders of magnitude larger than the sum of all service times.

- *Over-processing waste*: All additional efforts done for a product not directly required by the customer are considered as waste. This includes using components that are more precise, complex, of higher quality, and thus more expensive than absolutely required.
- *Overproduction waste*: Producing more than what is required by the customers at a particular time is a potential form of waste. Overproduction may lead to excess inventory and the customer's preferences may change over time making products outdated or less valuable.
- *Defects*: Rework, scrap, missing parts, poor work instructions, and correction activities are defects that can increase the costs of a product drastically.

Various additional types of waste have been identified. Although the terminology is oriented towards production processes and physical products, the same principles can be used for information/financial services, administrative work, and other BPM-like processes. The above examples illustrate that the focus of lean manufacturing is on *eliminating all non-value added activities*. Six Sigma focuses on *improving the quality of value added activities*. Both complement each other and are combined in Lean Six Sigma.

What does "Six Sigma" mean?
Today the term "Six Sigma" refers to a broad set of tools, techniques and methods to improve the quality of processes [113]. Six Sigma was originally developed by Motorola in the early 1980s and extended by many others. The term "Six Sigma" refers to the initial goal set by Motorola to minimize defects. In fact, the σ in "Six Sigma" refers to the standard deviation of a normal distribution. Given a normal distribution, 68.3% of the values lie within 1 standard deviation of the mean, i.e., a random draw from normal distribution with a mean value of μ and a standard deviation of σ has a probability of 0.683 to be in the interval $[\mu - \sigma, \mu + \sigma]$. Given the same normal distribution, 95.45% of randomly sampled values lie within two standard deviations of the mean, i.e., $[\mu - 2\sigma, \mu + 2\sigma]$, and 99.73% of the values lie within three standard deviations of the mean, i.e., $[\mu - 3\sigma, \mu + 3\sigma]$. The traditional quality paradigm in manufacturing defines a process as "capable" if the process's natural spread, plus and minus three σ, was less than the engineering tolerance. So, if deviations of up to three times the standard deviations are allowed, then on average 2700 out of one million cases will have a defect (i.e., samples outside the $[\mu - 3\sigma, \mu + 3\sigma]$ interval). Six Sigma aims to create processes were the standard deviation is so small that any value within 6 standard deviations of the mean can be considered as non-defective. In the literature, often a 1.5 sigma shift (to accommodate for long term variations and decreasing quality) is taken into account [113]. This results in the following table:

Quality level	Defective Parts per Million Opportunities (DPMO)	Percentage passed
One Sigma	690,000 DPMO	31%
Two Sigma	308,000 DPMO	69.2%
Three Sigma	66,800 DPMO	93.32%
Four Sigma	6,210 DPMO	99.379%
Five Sigma	230 DPMO	99.977%
Six Sigma	3.4 DPMO	99.9997%

A process that "runs at One Sigma" has less than 690,000 defective cases per million cases, i.e., at least 31% of the cases are handled properly. A process that "runs at Six Sigma" has only 3.4 defective cases per million cases, i.e., on average 99.9997% of the cases are handled properly.

A typical Lean Six Sigma project follows the so-called *DMAIC* approach consisting of five steps:

- *Define* the problem and set targets,
- *Measure* key performance indicators and collect data,
- *Analyze* the data to investigate and verify cause-and-effect relationships,
- *Improve* the current process based on this analysis, and
- *Control* the process to minimize deviations from the target.

Numerous organizations heavily invested in (Lean) Six Sigma training over the past decade. Based on Karate-like skill levels (green belt, black belt, etc.), certification programs were implemented. Unfortunately, the actual techniques are typically very basic (from a data science point of view). As a result, many consider Lean Six Sigma training as a management fad. Fortunately, process mining can be used as a tool to add more substance to the methodology. For example, process discovery can be used to eliminate all non-value added activities and reduce waste. If the relevant events are being recorded, we can visualize unnecessary waiting and rework. Conformance checking can also improve the quality of value added activities. Deviations can be found and diagnosed easily, provided that the event data and normative process models are present.

Related to Lean Six Sigma are management approaches such as: *Continuous Process Improvement* (CPI), *Total Quality Management* (TQM), *5S* (workplace organization method characterized by the terms Sort, Straighten, Shine, Standardize, and Sustain), *Kaizen* (another continuous improvement method), and *Theory of Constraints* (management paradigm by Eliyahu Goldratt based in the idea that "a chain is no stronger than its weakest link", thus focusing on the constraints limiting performance). What these approaches have in common is that processes are "put under a microscope" to see whether further improvements are possible. Clearly, process mining can help to analyze deviations and inefficiencies.

2.5.4 How Process Mining Compares to BPR

Business Process Reengineering (BPR) is a management approach developed by people like Michael Hammer [68]. BPR is characterized by four key words: *fundamental*, *radical*, *dramatic* and *process* [151]. The keyword *fundamental* indicates that, when revitalizing a business process, it is of great importance always to ask the basic question: Why are we doing this, and why are we doing it in this way? *Radical* means that the reengineered process must represent a complete break from the current way of working. BPR does not advocate to gradually improve existing processes: It aims at finding by completely new ones. The third keyword also refers to the fact that BPR does not aim at marginal or superficial changes. Changes must be *dramatic* in terms of costs, service and quality. In order to achieve dramatic improvements, it is necessary to focus on the *processes* and not start from data or systems.

BPR is *process-centric*, i.e., the focus is on the process just like in process mining. However, BPR is *not data-driven*. It promotes "thinking outside the box" rather than analyzing data in great detail. Process mining helps to identify the problems and assists shareholders in defining improvement actions. However, process mining cannot come up with completely different ways of working (unless event data is enriched with domain knowledge).

2.5.5 How Process Mining Compares to Business Intelligence

Process mining can be positioned under the umbrella of *Business Intelligence* (BI). There is no clear definition for BI. On the one hand, it is a very broad term that includes anything that aims at providing actionable information that can be used to support decision making. On the other hand, vendors and consultants tend to conveniently skew the definition towards a particular tool or methodology. Process mining provides innovations highly relevant for the next generation of BI techniques. However, it is important to note that current BI tools are not really "intelligent" and do not provide any process mining capabilities. The focus is on querying and reporting combined with simple visualization techniques showing dashboards and scorecards. Some systems provide data mining capabilities or support *Online Analytical Processing* (OLAP). OLAP tools are used to view multidimensional data from different angles. On the one hand, it is possible to aggregate and consolidate data to create high-level reports. On the other hand, OLAP tools can drill down into the data to find detailed information. There are approaches that combine process mining with OLAP to create and analyze so-called *process cubes* filled with event data (see Sect. 12.4).

Under the BI umbrella, many fancy terms have been introduced to refer to rather simple reporting and dashboard tools. *Business Activity Monitoring* (BAM) refers to the real-time monitoring of business processes. *Corporate Performance Management* (CPM) is another buzzword for measuring the performance of a process or

organization. Typically, CPM focuses on financial aspects. Recently, more and more software vendors started to use the term "analytics" to refer to advanced BI capabilities. *Visual analytics* focuses on the analysis of large amounts of data while exploiting the remarkable capabilities of humans to visually identify patterns and trends. *Predictive analytics* uses historic data to make forecasts. Clearly, process mining also aims at providing advanced analytics and some process mining techniques also heavily rely on advanced visualization and human interpretation. Moreover, as will be demonstrated in Chapt. 10, process mining is not restricted to analyzing historic data and also includes operational support, i.e., providing predictions and recommendations in an online setting.

2.5.6 How Process Mining Compares to CEP

Process mining complements *Complex Event Processing* (CEP). CEP combines data from multiple sources to infer events or patterns that suggest higher-level events. The goal of CEP is to identify meaningful events (such as opportunities or threats) and respond to them as quickly as possible, e.g., immediately generate an alert when a combination of events occurs. CEP can be used as a preprocessing step for process mining, i.e., low level event data with many (seemingly) meaningless events can be converted into higher-level event streams used by process mining techniques (online or offline). CEP is particularly useful if there are many (low-level) events. By reducing torrents of event data to manageable streams or logs, analysis becomes easier.

2.5.7 How Process Mining Compares to GRC

Whereas management approaches such as Lean Six Sigma and BPR mainly aim at improving operational performance, e.g., reducing flow time and defects, organizations are also putting increased emphasis on *corporate governance*, *risk*, and *compliance*. The frequently used acronym *GRC* is composed of the pillars *Governance*, *Risk management* and *Compliance*, and refers to an organization's capability to reliably achieve its objectives while addressing uncertainty and acting with integrity. *Governance* is the combination of culture, policies, processes, laws, and institutions that define the structure by which the organization is directed and managed. *Risk management* is the process of identifying, assessing, and prioritizing risks, as well as creating a plan for minimizing or eliminating the impact of negative events. *Compliance* is the act of adhering to, and demonstrating adherence to, external laws and regulations as well as corporate policies and procedures.

Major corporate and accounting scandals including those affecting Enron, Tyco, Adelphia, Peregrine, and WorldCom have fueled interest in more rigorous auditing practices. Legislation such as the *Sarbanes–Oxley Act* (SOX) of 2002 and the different *Basel Accords* was enacted in response to such scandals. The financial crisis

a few years ago also underscores the importance of verifying that organizations operate "within their boundaries". Process mining techniques offer a means to a more rigorous compliance checking and ascertaining the validity and reliability of information about an organization's core processes. Since conformance checking can be used to reveal deviations, defects, and near incidents, it is a valuable tool to check compliance and manage risks.

2.5.8 How Process Mining Compares to ABPD, BPI, WM, ...

As described in the *Process Mining Manifesto* [75] by the IEEE Task Force on Process Mining, there are several alternative terms for process mining. Sometimes these terms are synonyms and at other times they refer to particular process mining tasks.

Automated Business Process Discovery (ABPD) is an example of such a term. ABPD was introduced by Gartner introduced in 2008 [84]. Often ABPD is used to refer to just process discovery (i.e., discovering process models from event data). This does not include bottleneck analysis, conformance checking, prediction, social network analysis, etc. Sometimes ABPD is used as a synonym for process mining. However, Fig. 2.5 clearly shows that process mining includes, for example, conformance checking. Moreover, later other forms of process mining will be described (e.g., prediction). Vendors that claim to support ABPD typically only uphold a fraction of the process mining spectrum covered in this book.

The term *Business Process Intelligence* (BPI) is used in different ways. It is often used as a synonym for process mining (perhaps with less emphasis on process models). See, for example, the annual International Workshop on Business Process Intelligence that has been running since 2005. Almost all papers presented at these BPI workshops use or propose process mining techniques.

BPI can also be understood as BI with a focus on analyzing operational processes. Some of the products positioned as BPI tools do not support discovery, i.e., performance data are mapped onto hand-made models. These tools assume a stable and known process model. Terms comparable to BPI are used by a range of vendors. For example, IBM's Business Process Manager refers to the latter BPI-like functionality (i.e., without process discovery) as *Business Process Analytics* (BPA).

The term *Workflow Mining* (WM) was a precursor for process mining. It dates from a time where the main aim of process mining was the automatic configuration of a WFM system. Ideally, one would like to observe an existing process and automatically generate the corresponding executable workflow model. This view turned out to be too narrow and often unrealistic. Process mining has a much wider applicability, also in areas unrelated to WFM/BPM systems. Moreover, through discovery one can indeed find a skeleton of the workflow model. However, the model needs to be enriched with technical details to obtain an executable workflow. This makes the automated generation of workflow models less practical. Today, process mining is predominantly an approach for performance and conformance analysis (see Fig. 2.1). Therefore, the term WM is no longer actively used.

2.5.9 How Process Mining Compares to Big Data

In Chap. 1, we listed the "four V's of Big Data": Volume, Velocity, Variety, and Veracity (Fig. 1.4). These reflect the typical characteristics of some of the exciting new data sources interesting for analysis. Big Data does not focus on a particular type of analysis and is not limited to process-related data. However, Big Data infrastructures enable us to collect, store, and process huge event logs. Process mining tools can exploit such infrastructures to distribute large analysis tasks over multiple computing resources. For example, the MapReduce programming model can be used for discovery algorithms and the Hadoop Distributed File System (HDFS) can be used to store event data in a distributed fashion. In principle, one can use thousands of compute nodes to perform process mining analyses. Chapter 12 will elaborate on this.

The many acronyms in this section—BPM, BI, OLAP, TPS, BAM, CEP, CPM, CPI, TQM, SOX, etc.—are just a subset of the jargon used by business consultants and vendors. Some are just variations on the same theme, others emphasize a particular aspect. What can be distilled from the above is that there is a clear trend towards actually using the data available in today's systems. The data is used to *reason about the process* and for *decision making within the process*. Moreover, the acronyms express a clear desire to get more *insight* into the actual processes, to *improve* them, and to make sure that they are *compliant*. Unfortunately, buzzwords are often used when the actual analysis capabilities are weak. When listening to a product presentation of conference talk, one is often tempted to play "buzzword bingo" (also known as bullshit bingo) illustrating that the foundational issues are not addressed. This book aims to provide a clear and refreshing view on the matter. Using recent breakthroughs in process mining, we will show that it is possible to simplify and unify the analysis of business processes based on facts. Moreover, the techniques and insights presented are directly applicable and are supported by process mining tools such as *ProM* (www.processmining.org).