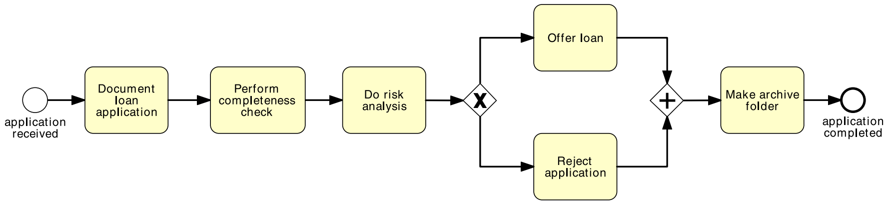


Fig. 5.10 A loan application process

structured model. Finally, several activities do not follow the naming conventions of G6.The model can be reworked and simplified to the one shown in Fig. 5.9.

5.7 Further Exercises

Exercise 5.11 Imagine you are responsible for the human resources department of a leading consultancy. Which characteristics would you check when hiring new process analysts?

Exercise 5.12 As responsible for human resources department of a consultancy, how would you develop the skills of your junior process analysts?

Exercise 5.13 How would you as a process analyst prepare for an interview with a domain expert?

Exercise 5.14 Analyze the loan application process model of Fig. 5.10 for soundness-related problems.

Exercise 5.15 Look up tools on the internet that offer a soundness check for process models.

Exercise 5.16 Consider again the loan application process model of Fig. 5.10. What are indications that it would not be complete?

Exercise 5.17 Have a look at the activity labels of the loan application model of Fig. 5.10 and propose improved labels where appropriate.

Exercise 5.18 Have a look at the process model of Fig. 5.10 showing a sales campaign process for one of our industry partners. Describe which 7PMG guidelines can be used to improve this model. Have a look at the process model of Fig. 5.11 showing a sales campaign process for one of our industry partners

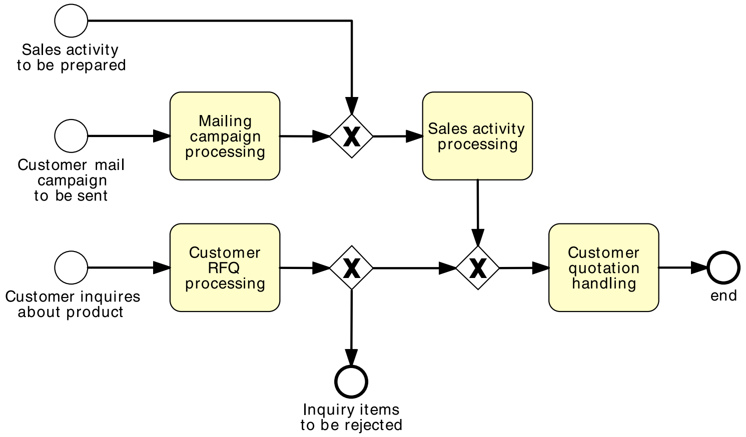


Fig. 5.11 A sales campaign process

5.8 Further Reading

The general topic of process discovery is well covered in the book on workflow modeling by Sharp and McDermott [86]. This book gives detailed advice on all phases of process discovery, specifically data gathering and workshop organization. Other practical advice is summarized by Verner [101] and by Stirna, Persson, and Sandkuhl [88]. Interview techniques are widely discussed as a social science research method for instance in the book by Berg and Lune [7] or the book by Seidman [85].

Frederiks and van der Weide [18] discuss the skills required from process analysts, particularly when engaging in process discovery efforts. In a similar vein, Schenk, Vitalari and Davis [83] and Petre [66] discuss the capabilities that expert process analysts (as opposed to novice ones) generally display when engaging in process discovery.

In this chapter, we emphasized "manual" process discovery techniques, wherein process models are manually constructed based on data collected from various process stakeholders by means of interviews, workshops and related techniques. As mentioned in Sect. 5.2.1, there is also a whole range of complementary techniques for automatic discovery process models from event logs. These automatic process discovery techniques are part of a broader set of techniques for analyzing event logs, collectively known as process mining [94]. We will discuss several process mining techniques later in Chap. 10.

The modeling method introduced in Sect. 5.3 revolves around the discovery of activities and control-flow relations between activities. This family of approaches is usually called *activity-based modeling* [68]. An alternative approach to process modeling is known as *artifact-centric modeling* [59] or *object-centric modeling* [68]. In artifact-centric modeling the emphasis is not on identifying activities, but rather *artifacts* (physical or electronic objects or documents) that are manipulated within a given process. For example, in an order-to-cash process, typical artifacts are the purchase order, the shipment notice and the invoice. Once these artifacts have been identified, they are analyzed in terms of the data that they hold and in terms of the phases they go through during the process. For example, a purchase order typically goes through the phases *received*, *accepted*, *manufactured*, *shipped* and *invoiced*. These phases and the transitions between these phases are called the artifact lifecycle. The main emphasis in artifact-centric process modeling is put on identifying these artifact lifecycles. Several industrial applications of artifact-centric process modeling have shown that it is quite suitable when discovering processes that exhibit significant amounts of variation, for example variation between business units, geographical regions or types of customer as discussed for example by Caswell et al. [59] and Redding et al. [68].

The quality of conceptual models in general, and of process models specifically, has received extensive attention in the research literature. The Sequal framework introduced by Lindland, Sindre, and Sølvberg adapts semiotic theory, namely the three perspectives of syntax, semantics and pragmatics, to the evaluation of conceptual model quality [45]. An extended version of this framework is presented by Krogstie, Sindre, and Jørgensen [42].

Validation and verification of process models has also received extensive attention in the literature. Mendling [51] for example provides numerous pointers to related research. The verification of Workflow nets specifically is investigated by van der Aalst [93] who connects soundness analysis of process models with classical Petri nets notions of liveness and boundedness.

The 7PMG guidelines discussed in Sect. 5.4.4 are by Mendling, Reijers, and van der Aalst in [53]. These guidelines build on empirical work on the relation between process model metrics on the one hand and error probability and understandability on the other hand [50, 54, 55, 63, 73, 74]. Specifically, the impact of activity label quality on process model understanding is investigated by Mendling, Reijers, and Recker [52]. Another set of modeling guidelines are the Guidelines of Process Modeling by Becker et al. [5].

As a complement to process modeling guidelines and conventions, it is useful to also keep in mind potential pitfalls to be avoided in process modeling projects. For example, Rosemann [78, 79] draws a list of 22 pitfalls of process modeling, including a potential lack of strategic connection, l'art pour l'art, to name but a few. His bottom line is that modeling success does not directly equate with process success.

Chapter 6
Qualitative Process Analysis

Quality is free, but only to those who are willing to pay heavily for it.
Tom DeMarco (1940–)

Analyzing business processes is both an art and a science. In this respect, qualitative analysis is the artistic side of process analysis. Like fine arts, such as painting, there is not a single way of producing a good process analysis, but rather a range of principles and techniques that tell us what practices typically lead to a "good" process analysis. When learning to paint, you learn how to hold the brush, how to produce different types of brushstroke, how to mix colors, etc. The rest of the art of painting is up to you to acquire by means of practice, discernment and critical self-assessment.

In this chapter, we introduce a few basic principles and techniques for qualitative process analysis. First, we present principles aimed at making the process leaner by identifying unnecessary parts of the process in view of their elimination. Next, we present techniques to identify and analyze the weak parts of the process, meaning the parts that create issues that negatively affect the performance of the process. In particular, we discuss how to analyze the impact of issues in order to prioritize redesign efforts.

6.1 Value-Added Analysis

Value-added analysis typically consists of two stages: value classification and waste elimination. Below we discuss each of these stages in turn.

6.1.1 Value Classification

Value-added analysis is a technique aimed at identifying unnecessary steps in a process in view of eliminating them. In this context, a *step* may be a task in the process, or part of a task, or a handover between two tasks. For example, if a task "Check

M. Dumas et al., *Fundamentals of Business Process Management*,
DOI 10.1007/978-3-642-33143-5_6, © Springer-Verlag Berlin Heidelberg 2013

purchase order" ends with the Purchase Order (PO) being sent by internal mail to a supervisor, and the next task "Approve purchase order" starts when the supervisor receives and checks the PO, then we can say that the transportation of the PO via internal mail is a step—a potentially unnecessary (non-value-adding) step in this context. It is often the case that one task involves several steps. For example, a task "Check invoice" may involve the following steps:

1. Retrieve the PO(s) that corresponds to the invoice.
2. Check that the amounts in the invoice and those in the PO coincide.
3. Check that the products or services referenced in the PO have been delivered.
4. Check that the supplier's name and banking details in the invoice coincide with those recorded in the Supplier Management System.

In some cases, steps within a task are documented in the form of checklists. The checklists tell the process participants what things need to be in place before a task is considered to be complete. If detailed checklists are available, the process analyst can use them to decompose tasks into steps. Unfortunately, such checklists are not always available. In many cases, process participants have an implicit understanding of the steps in a task because they perform the task day in and day out. But this implicit understanding is not documented anywhere. In the absence of such documentation, the process analyst needs to decompose each task into steps by means of observation and interviewing.

Having decomposed the process into steps, a second prerequisite for value-added analysis is to identify who is the customer of the process and what are the positive outcomes that the customer seeks from the process (cf. Sect. 1.2). These outcomes are said to add value to the customer, in the sense that fulfilling these outcomes is in the interest or for the benefit of the customers.

Having decomposed the process into steps and having clearly identified the positive outcomes of a process, we can then analyze each step in terms of the value it adds. Steps that directly contribute to positive outcomes are called *value-adding steps*. For example, consider a process for repairing a washing machine or other domestic appliance. The steps in this process where the technician diagnoses the problem with the machine are clearly value-adding, as it directly contributes to the outcome the customer wishes to see, that is, that the machine is repaired. Also, the steps related to repairing the machine are value-adding.

Some steps do not directly add value to the customer but they are necessary for the business. Consider again the example of a process for repairing a washing machine. Imagine that this process includes a step "Record defect" in which the technician enters data about the washing machine and an explanation of the defect found in a washing machine into an information system. This step per se is not value-adding for the customer. The customer wishes the machine to be fixed and does not get value by the fact that the defect in their machine was recorded in an information system of the repairing company. However, recording defects and their resolution helps the company to build up a knowledge base of typical defects and their resolution. This knowledge base is extremely valuable when new technicians are recruited in the company, since they can learn from knowledge that more experienced technicians have recorded. Also, such information allows the company to detect frequent defects and to report such defects to the manufacturer or distributor of the washing machine. Steps such as "Record defect" are termed *business value-adding steps*, in the sense that the customer is not willing to pay for the performance of these steps nor do they gain satisfaction from these steps being performed (so the steps are not value-adding) but the step is necessary or useful to the company that performs the process.

In summary, value-added analysis is a technique whereby an analyst decorticates a process model, extracts every step in the process and classifies these steps into one of three categories, namely:

- Value-adding (VA): This is a step that produces value or satisfaction vis-à-vis of the customer. When determining whether or not a step is value-adding, it may help to ask the following question: Would the customer be willing to pay for this activity?
- Business value-adding (BVA): The step is necessary or useful for the business to run smoothly, or it is required due to the regulatory environment of the business.
- Non-value adding (NVA): The step does not fall into any of the other two categories.

Example 6.1 We consider the process for equipment rental described in Example 1.1 (p. 2). As discussed in Sect. 1.2, the customer of this process is the site engineer who submits an equipment rental request. From the perspective of the site engineer, the positive outcome of the process is that the required piece of equipment is available in the construction site when needed. Let us analyze the fragment of this process described in Fig. 1.6. To identify the relevant steps, we will decorticate the model task by task. While we do this, we will also classify the steps into VA, BVA and NVA.

- The first task in the process model is the one where the engineer lodges the request. From the description in Example 1.1, we observe there are three steps in this task:

  1. Site engineer fills in the request.
  2. Site engineer sends the request to the clerk via e-mail (handover step).
  3. Clerk opens and reads the request (handover step).

  Arguably, filling the request is value-adding insofar as the site engineer cannot expect the equipment to be rented if they do not ask for it. In one way or another, the site engineer has to request the equipment in order to obtain it. On the other hand, the site engineer does not get value out of sending the request to the clerk by e-mail nor do they get value out of the clerk having to open and read the request. More generally, handover steps between process participants, such as sending and receiving internal messages, are not value-adding.
- The second task is the one where the clerk selects a suitable equipment from the supplier's catalog. We can treat this task as a single step. This step is value-adding insofar as it contributes to identifying a suitable equipment to fulfill the needs of the site engineer.

- In the third task, the clerk calls the supplier to check the availability of the selected equipment. Again, we can treat this task as a single step. This step is value-adding insofar as it contributes to identifying a suitable and available equipment. If the equipment is available, the clerk will recommend that this equipment be rented. To this end, the clerk adds the details of the recommended equipment and supplier to the rental request form and forwards the form to the works engineer for approval. Thus we have two more steps: (i) adding the details to the rental request and (ii) forwarding the rental request to the works engineer. The first of these steps is business value-adding since it helps the company to keep track of the equipment they rent and the suppliers they rent from. Maintaining this information is valuable when it comes to negotiating or re-negotiating bulk agreements with suppliers. The handover between the clerk and the works engineer is not value-adding.
- Next, the works engineer examines the rental request in view of approving it or rejecting it. We can treat this examination as one step. This step is a *control step*, that is, a step where a process participant or a software application checks that something has been done correctly. In this case, this control step helps the company to ensure that equipment is only rented when it is needed and that the expenditure for equipment rental in a given construction project stays within the project's budget. Control steps are generally business value-adding, although an analyst may ask the question of how many control steps are needed and how often they should be performed.
- If the works engineer has an issue with the rental request, the works engineer communicates it to the clerk or the site engineer. This communication is another step and it is business value-adding since it contributes to identifying and avoiding misunderstandings within the company. If approved, the request is sent back to the clerk; this is a handover step and it is thus non-value-adding.
- Finally, assuming the request is approved, the clerk produces and sends the PO. Here we can identify two more steps: produce the PO and send the PO to the corresponding supplier. Producing the PO is business value-adding. It is necessary in order to ensure that the rental request cost is correctly accounted for and eventually paid for. Sending the PO is value-adding: It is this act that makes the supplier know when the equipment has to be delivered on a given date. If the supplier did not get this information, the equipment would not be delivered. Note, however, that what is value-adding is the fact that the supplier is explicitly requested by the construction company to deliver the equipment on a given date. The fact that this request is made by sending a PO is secondary in terms of adding value to the site engineer.

The identified steps and their classification are summarized in Table 6.1.

Classifying steps into VA, BVA and NVA is to some extent subjective and depends on the context. For example, one may question whether producing the PO is a VA or a BVA step. Arguably, in order for the equipment to be available, the supplier needs to have an assurance that the equipment rental fee will be paid. So one could say that the production of the PO contributes to the rental of the equipment

Table 6.1 Classification of steps in the equipment rental process

Step	Performer	Classification
Fill request	Site engineer	VA
Send request to clerk	Site engineer	NVA
Open and read request	Clerk	NVA
Select suitable equipment	Clerk	VA
Check equipment availability	Clerk	VA
Record recommended equipment & supplier	Clerk	VA
Forward request to works engineer	Clerk	VA
Open and examine request	Works engineer	BVA
Communicate issues	Works engineer	BVA
Forward request back to clerk	Works engineer	NVA
Produce PO	Clerk	BVA
Send PO to supplier	Clerk	BVA

since the PO serves to assure the supplier that the payment for the rental equipment will be made. However, as mentioned above, what adds value to the site engineer is the fact that the supplier is notified that the equipment should be delivered at the required date. Whether this notification is done by means of a PO or by means of a simple electronic message sent to the supplier is irrelevant, so long as the equipment is delivered. Thus, producing a formal document (a formal PO) is arguably not value-adding. It is rather a mechanism to ensure that the construction company's financial processes run smoothly and to avoid disputes with suppliers, e.g. avoiding the situation where a supplier delivers a piece of equipment that is not needed and then asks for payment of the rental fee. More generally, we will take the convention that steps imposed by accounting or legal requirements are BVA, even though one could argue differently in some cases.

Exercise 6.1 Consider the process for university admission described in Exercise 1.1 (p. 4). What steps can you extract from this process? Classify these steps into VA, BVA and NVA.

6.1.2 Waste Elimination

Having identified and classified the steps of the process as discussed above, one can then proceed to determining how to eliminate waste. A general rule is that one should strive to minimize or eliminate NVA steps. Some NVA steps can be eliminated by means of automation. This is the case of handovers for example, which can be eliminated by putting in place an information system that allows all stakeholders to know what they need to do in order to move forward the rental requests. When the site engineer submits a rental request via this information system, the request would automatically appear in the to-do list of the clerk. Similarly, when the clerk records the recommended supplier and equipment, the works engineer would be notified and directed to the request. This form of automation makes these NVA steps transparent to the performers of the steps. The topic of process automation will be discussed in further detail in Chap. 9.

A more radical approach to eliminating NVA steps in the working example is to eliminate the clerk altogether from the process. This means moving some of the work to the site engineer so that there are less handovers in the process. Of course, the consequences of this change in terms of added workload to the site engineer need to be carefully considered. Yet another approach to eliminate NVA (and BVA) steps would be to eliminate the need for approval of rental requests in cases where the estimated cost is below a certain threshold. Again, this option should be weighted against the possible consequences of having less control steps in place. In particular, if the site engineers were given full discretion to rent equipment at their own will, there would need to be a mechanism in place to make them accountable in case they rent unnecessary equipment or they rent equipment for excessively and unnecessarily long periods. Such process redesign questions will be further discussed in Chap. 8.

While elimination of NVA steps is generally considered a desirable goal, elimination of BVA steps should be considered as a trade-off given that BVA steps play a role in the business. Prior to eliminating BVA steps, one should first map BVA steps to business goals and business requirements, such as regulations that the company must comply to and risks that the company seeks to minimize. Given a mapping between BVA steps on the one hand and business goals and requirements on the other, the question then becomes the following: What is the minimum amount of work required in order to perform the process to the satisfaction of the customer, while fulfilling the goals and requirements associated to the BVA steps in the process? The answer to this question is a starting point for process redesign.

6.2 Root Cause Analysis

When analyzing a business process, it is worth keeping in mind that "even a good process can be made better" [28]. Experience shows that any non-trivial business process, no matter how much improvement it has undergone, suffers from a number of issues. There are always errors, misunderstandings, incidents, unnecessary steps and other forms of waste when a business process is performed on a day-to-day basis.

Part of the job of a process analyst is to identify and to document the *issues* that plague a process. To this end, an analyst will typically gather data from multiple sources and will interview several stakeholders, chiefly the process participants but also the process owner and managers of organizational units involved in the process. Each stakeholder has a different view on the process and will naturally have a tendency to raise issues from their own perspective. The same issue may be perceived differently by two stakeholders. For example, an executive manager or a process owner will typically see issues in terms of performance objectives not being met or in terms of constraints imposed for example by external pressures (e.g. regulatory or compliance issues). Meanwhile, process participants might complain about insufficient resources, hectic timelines as well as errors or exceptions perceived to be caused by other process participants or by customers.

Root cause analysis is a family of techniques to help analysts identify and understand the root cause(s) of problems or undesirable events. Root cause analysis is not confined to business process analysis. In fact, root cause analysis is commonly used in the context of accident or incident analysis as well as in manufacturing processes where it is used to understand the root cause of defects in a product. In the context of business process analysis, root cause analysis is helpful to identify and to understand the issues that prevent a process from having a better performance.

Root cause analysis encompasses a variety of techniques. In general, these methods include guidelines for interviewing and conducting workshops with relevant stakeholders, as well as techniques to organize and to document the ideas generated during these interviews or workshops. Below, we will discuss two of these techniques, namely *cause-and-effect diagrams* and *why–why diagrams*.

6.2.1 Cause–Effect Diagrams

Cause–effect diagrams depict the relationship between a given *negative effect* and its causes. In the context of process analysis, a negative effect is usually either a recurrent issue or an undesirable level of process performance. Causes can be divided into causal and contributing factors (hereby called *factors*) as explained in the box below.

CAUSAL VERSUS CONTRIBUTING FACTORS

Two broad types of cause are generally distinguished in the area of root cause analysis, namely *causal factors* and *contributing factors*. Causal factors are those factors that, if corrected, eliminated or avoided would prevent the issue from occurring in future. For example, in the context of an insurance claims handling process, errors in the estimation of damages lead to incorrect claim assessments. If the damage estimation errors were eliminated, a number of occurrences of the issue "Incorrect claim assessment" would definitely be prevented. Contributing factors are those that set the stage for, or that increase the chances of a given issue occurring. For example, consider the case where the user interface for lodging the insurance claims requires the claimant to enter a few dates (e.g. the date when the claim incident occurred), but the interface does not provide a calendar widget so that the user can easily select the date. This deficiency in the user interface may increase the chances that the user enters the wrong date. In other words, this deficiency contributes to the issue "Incorrect claim data entry".

While the distinction between causal and contributing factor is generally useful when investigating specific incidents (for example investigating the causes of a given road accident), the distinction is often not relevant or not sufficiently sharp in the context of business process analysis. Accordingly, in this chapter we will use the term *factor* to refer to causal and contributing factors collectively.

In a cause–effect diagram, factors are grouped into categories and possibly also sub-categories. These categories are useful in order to guide the search for causes. For example, when organizing a brainstorming session for root cause analysis, one way to structure the session is to first go around the table asking each participant to give their opinion on possible causes of the issue at hand. The causes are first written down in any order. Next, the identified causes are classified according to certain categories and the discussion continues in a more structured way using these categories as a framework.

A well-known categorization for cause–effect analysis are the so-called 6M's, which are described below together with possible sub-categorizations.

1. **Machine** (technology)—factors pertaining to the technology used, like for example software failures, network failures or system crashes that may occur in the information systems that support a business process. A useful sub-categorization of Machine factors is the following:
   a. Lack of functionality in application systems.
   b. Redundant storage of data across systems, leading for example to double data entry (same data entered twice in different systems) and data inconsistencies across systems.
   c. Low performance of IT of network systems, leading for example to low response times for customers and process participants.
   d. Poor user interface design, leading for example to customers or process participants not realizing that some data are missing or that some data are provided but not easily visible.
   e. Lack of integration between multiple systems within the enterprise or with external systems such as a supplier's information system or a customer's information system.
2. **Method** (process)—factors stemming from the way the process is defined or understood or in the way it is performed. An example of this is when a given process participant A thinks that another participant B will send an e-mail to a customer, but participant B does not send it because they are not aware they have to send it. Possible sub-categories of Method factors include:
   a. Unclear, unsuitable or inconsistent assignment of decision-making and processing responsibilities to process participants.

   b. Lack of empowerment of process participants, leading to process participants not being able to make necessary decisions without consulting several levels above in their organizational hierarchy. Conversely, excessive empowerment may lead to process participants having too much discretion and causing losses to the business through their actions.
   c. Lack of timely communication between process participants or between process participants and the customer.
3. **Material**—factors stemming from the raw materials, consumables or data required as input by the activities in the process, like for example incorrect data leading to a wrong decision being made during the execution of a process. The distinction between raw materials, consumables and data provides a possible subcategorization of these factors.
4. **Man**—factors related to a wrong assessment or an incorrectly performed step, like for example a claims handler accepting a claim even though the data in the claim and the rules used for assessing the claim require that the claim be rejected. Possible sub-categories of Man factors include:
   a. Lack of training and clear instructions for process participants.
   b. Lack of incentive system to motivate process participants sufficiently.
   c. Expecting too much from process participants (e.g. overly hectic schedules).
   d. Inadequate recruitment of process participants.
5. **Measurement**—factors related to measurements or calculations made during the process. In the context of an insurance claim, an example of such a factor is one where the amount to be paid to the customer is miscalculated due to an inaccurate estimation of the damages being claimed.
6. **Milieu**—factors stemming from the environment in which the process is executed, like for example factors originating from the customer, suppliers or other external actors. Here, the originating actor is a possible sub-categorization. Generally, milieu factors are outside the control of the process participants, the process owner, and other company managers. For example, consider a process for handling insurance claims for car accidents. This process depends partly on data extracted from police reports (e.g. police reports produced when a major accident occurs). It may happen in this context that some errors during the claims handling process originate from inaccuracies or missing details in the police reports. These factors are to some extent outside the control of the insurance company. This example illustrates that milieu factors may need to be treated differently from other (internal) factors.

These categories are meant as guidelines for brainstorming during root cause analysis rather than gospel that should be followed to the letter. Other ways of categorizing factors may be equally useful. For example, one alternative categorization is known as the 4P's (Policies, Procedures, People, and Plant/Equipment). Also, it is sometimes useful to classify factors according to the activities in the process where they originate (i.e. one category per major activity in the process). This approach allows us to easily trace the relation between factors and activities in the process.

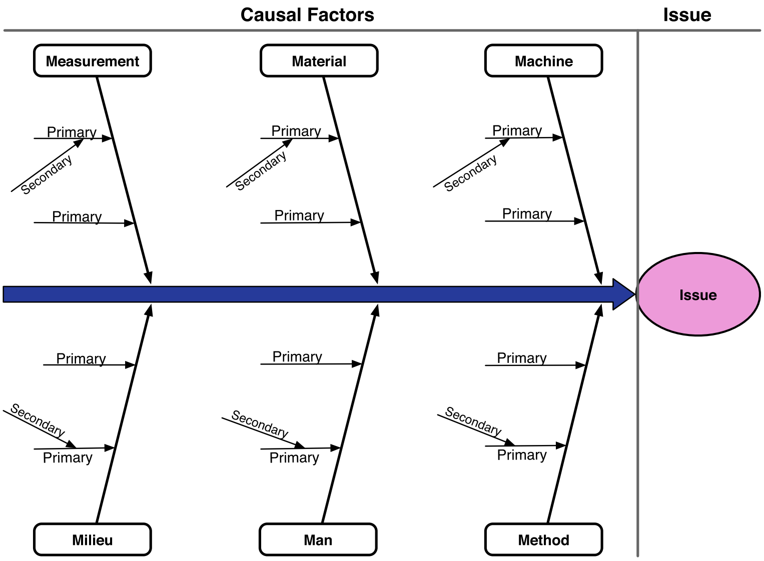


Fig. 6.1 Template of a cause–effect diagram based on the 6M's

The above categories are useful not only as a guide for brainstorming during root cause analysis, but also as a basis for documenting the root causes in the form of a cause–effect diagram. Concretely, a cause–effect diagram consists of a main horizontal line (the *trunk*) from which a number of branches stem (cf. Fig. 6.1). At one end of the trunk is a box containing the negative effect that is being analyzed (in our case the *issue* being analyzed). The trunk has a number of main branches corresponding to the categories of factors (e.g. the 6M's above). The root causes are written in the sub-branches. Sometimes, it is relevant to distinguish between *primary factors*, meaning factors that have a direct impact on the issue at hand, from *secondary factors*, which are factors that have an impact on the primary factors. For example, in the context of an insurance claims handling process, an inaccurate estimation of the damages leads to a miscalculation of the amount to be paid for a given claim. This inaccurate estimation of the damages may itself stem from a lack of incentive from the repairer to accurately calculate the cost of repairs. Thus, "Inaccurate damage estimation" can be seen as a primary factor for "Liability miscalculation", while "Lack of incentive to calculate repair costs accurately" is a secondary factor behind the "Inaccurate damage estimation". The distinction between primary and secondary factors is a first step towards identifying chains of factors behind an issue. We will see later in this chapter that why–why diagrams allow us to dig deeper into such chains of factors.

Because of their visual appearance, cause–effect diagrams are also known as *Fishbone diagrams*. Another common name for such diagrams is *Ishikawa diagrams* in allusion to one of its proponents—Kaoru Ishikawa—one of the pioneers of the field of quality management.

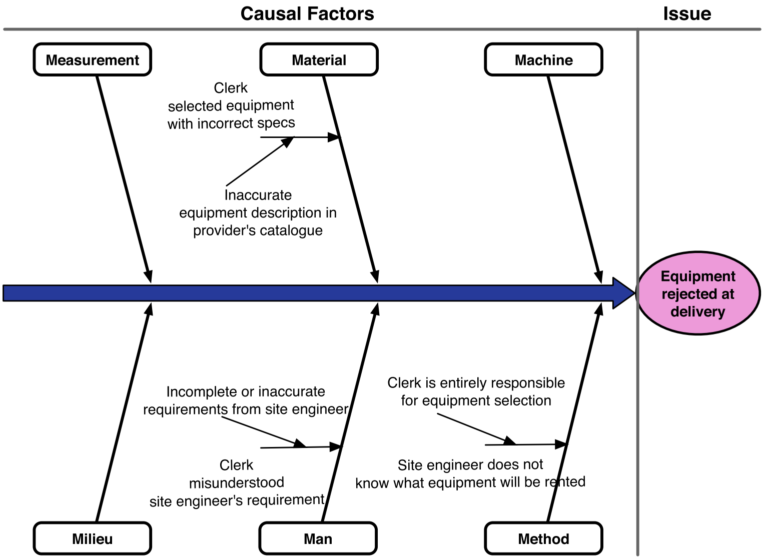


Fig. 6.2 Cause–effect diagram for issue "Equipment rejected at delivery"

Example 6.2 We consider again the equipment rental process described in Example 1.1 (p. 2). During an audit of this process, several issues were identified. It turns out that oftentimes the site engineer finds that the equipment delivered at the construction site is not suitable because it is either too small or not powerful enough for the job. Hence it has to be rejected. One clerk claims that the site engineers generally do not specify their requirements in sufficient detail. Other clerks blame the suppliers for giving inaccurate or incomplete descriptions of their equipment in their catalogs. On the other hand, site engineers complain that they are not consulted when there are doubts regarding the choice of equipment.

This scenario basically describes one issue, namely that the equipment is being rejected upon delivery. We can see three primary causes from the issue, which are summarized in the cause–effect diagram in Fig. 6.2. The diagram also shows secondary causes underpinning each of the primary causes. Note that the factor "clerk selected equipment with incorrect specs" has been classified under the Material category because this factor stems from incorrect input data. A defect in input data used by a process falls under the Material category.

Exercise 6.2 Consider the university admission process described in Exercise 1.1 (p. 4). One of the issues faced by the university is that students have to wait too long to know the outcome of the application (especially for successful outcomes). It often happens that by the time a student is admitted, the student has decided to go to another university instead (students send multiple applications in parallel to many universities). Analyze the causes of this issue using a cause–effect diagram.

6.2.2 Why–Why Diagrams

Why–why diagrams (also known as *tree diagrams*) constitute another technique to analyze the cause of negative effects, such as issues in a business process. The emphasis of root cause analysis is to capture the series of cause-to-effect relations that lead to a given effect. The basic idea is to recursively ask the question: Why has something happened? This question is asked multiple times until a factor that stakeholders perceive to be a *root cause* is found. A common belief in the field of quality management—known as the five Why's principle—has it that answering the "why" question five times recursively allows one to pin down the root causes of a given negative effect. Of course, this should not be treated as gospel, but as a guideline of how far one should go during root cause analysis.

Why–why diagrams are a technique for structuring brainstorming sessions (e.g. workshops) for root cause analysis. Such a session would start with an issue. The first step is to give a name to the issue that stakeholders agree on. Sometimes it is found that there is not one issue, but multiple issues, in which case they should be analyzed separately. Once the issue has been identified and a name has been agreed upon, this becomes the root of the tree. Then at each level the following questions are asked: "Why does this happen?" and "What are the main sub-issues that may lead to this issue?". Possible factors are then identified. Each of these factors is then analyzed using the same questions. When getting down in the tree (e.g. to levels 3 or 4) it is recommended to start focusing on factors that can be resolved, meaning that something can be done to change them. The leaves of the tree should correspond to factors that are fundamental in nature, meaning that they cannot be explained in terms of other factors. Ideally, these factors, called root causes, should be such that they can be eliminated or mitigated, but this is not necessarily the case. For example, in the context of an insurance claims handling process, a certain type of errors in a police report may be due to lack of time and hectic schedules on the side of police agents involved in filling these reports. There is relatively little the insurance agency can do in this case to eliminate the error, other than raising the issue with the relevant authorities. Yet, the impact of this factor could be mitigated by putting in place checks to detect such errors as early as possible in the process.

A simple template for why–why diagrams is given in Fig. 6.3. An alternative way of presenting the information in such diagrams is by means of nested bullet-point lists. In the rest of this chapter we will opt for the latter representation.

Example 6.3 We consider again the equipment rental process described in Example 1.1 (p. 2). In Example 6.2 above, we noted that one of the issues with this process is that the site engineer sometimes rejected the equipment upon delivery because it was not suitable for the job at hand. Another issue is that BuildIT spends more in equipment rental than what it budgeted for. An auditor pointed out that one of the reasons for excessive expenditure was that site engineers were keeping the rented equipment longer than initially planned by using deadline extensions. Site engineers knew that it was easy to get a deadline extension. They also knew that it took quite some time to get equipment rental requests approved, and the larger the cost and

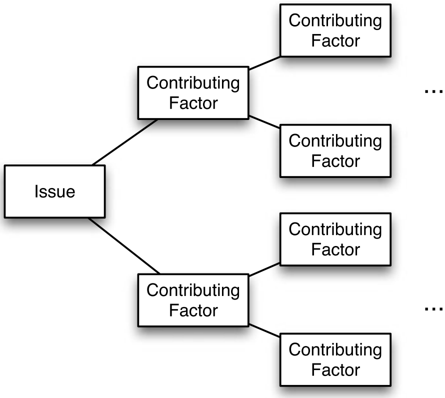


Fig. 6.3 Template of a why–why diagram

the duration of the rental, the slower it was to get it approved. So in many cases, site engineers were renting equipment several days before the date when they actually needed it. Also, they were specifying short periods in their equipment rental requests in order to get them approved quicker. When the deadline for returning an equipment approached, they just called the supplier to keep the equipment for a longer period.

Another issue spotted by the auditor is that a significant amount of late-payment penalty fees were paid to the suppliers because invoices for equipment rental were not paid by their due date. The clerks blamed the site engineers for being slow in approving the invoices.

In summary, we can distinguish at least three issues. First, the wrong equipment is being delivered on some occasions. Secondly site engineers are frequently asking for deadline extensions. Thirdly, BuildIT is often paying late payment fees to suppliers. A possible analysis root cause analysis of these issues leads to the following why–why diagrams (represented as nested bullet-point lists).

Issue 1 Site engineers sometimes reject delivered equipment, why?

Wrong equipment is delivered, why?

– miscommunication between site engineer and clerk, why?
  - site engineer provides only brief/inaccurate description of what they want
  - site engineer does not (always) see the supplier catalogs when making a request and does not communicate with the supplier, why?
    · site engineer generally does not have Internet connectivity
  - site engineer does not check the choice of equipment made by the clerk

– equipment descriptions in supplier's catalog not accurate

Issue 2 Site engineers keep equipment longer than needed via deadline extensions, why?

Site engineer fears that equipment will not be available later when needed, why?

– time between request and delivery too long, why?
  - excessive time spent in finding a suitable equipment and approving the request, why?
    · time spent by clerk contacting possibly multiple suppliers sequentially
    · time spent waiting for works engineer to check the requests

Issue 3 BuildIT often has to pay late payment fees to suppliers, why?

Time between invoice received by clerk and confirmation is too long, why?

– clerk needs confirmation from site engineer, why?
  - clerk cannot assert when was the equipment delivered and picked-up, why?
    · delivery and pick-up of equipments are not recorded in a shared information system
    · site engineer can extend the equipment rental period without informing the clerk
  - site engineer takes too long to confirm the invoice, why?
    confirming invoices is not a priority for site engineer

Exercise 6.3 Consider again the process for university admission described in Exercise 1.1 (p. 4) and the issue described in Exercise 6.2 above. Analyze this issue using a why–why diagram.

6.3 Issue Documentation and Impact Assessment

Root cause analysis techniques allow us to understand the factors behind a given issue. A natural next step is to understand the impact of these issues. Building up this understanding is critical in order to prioritize the issues so that the attention of the process owner, participants and analysts can be focused on the issues that most matter to the organization. Below we discuss two complementary techniques for impact assessment.

6.3.1 Issue Register

The *issue register* complements the output of root cause analysis by providing a more detailed analysis of individual issues and their impact. The purpose of the issue register is to determine how and to what extent each issue is impacting on the performance of the process. The impact of an issue can be described quantitatively, for example in terms of time or money lost, or qualitatively, in terms of perceived nuisance to the customer or perceived risks that the issue entails. For example, nuisances caused to the customer because of misunderstandings during the execution of the process can be classified as qualitative impact, since it is difficult to translate this nuisance into a monetary measure.

Concretely, an issue register is a listing that provides a detailed analysis of each issue and its impact in the form of a table with a pre-defined set of fields. The following fields are typically described for each issue:

- *Name of the issue*. This name should be kept short, typically two–five words, and should be understandable by all stakeholders in the process.
- *Description*. A short description of the issue, typically one–three sentences, focused on the issue itself as opposed to its consequences or impact, which are described separately.
- *Priority*. A number $(1, 2, 3, \ldots)$ stating how important this issue is relative to other issues. Note that multiple issues can have the same priority number.
- *Assumptions* (or *input data*). Any data used or assumptions made in the estimation of the impact of the issue, such as for example number of times a given negative outcome occurs, or estimated loss per occurrence of a negative outcome. In the early phases of the development of the issue register, the numbers in this column will be mainly assumptions or ballpark estimates. Over time, these assumptions and rough estimates will be replaced with more reliable numbers derived from actual data about the execution of the process.
- *Qualitative impact*. A description of the impact of the issue in qualitative terms, such as impact of the issue on customer satisfaction, employee satisfaction, long-term supplier relationships, company's reputation or other intangible impact that is difficult to quantify.
- *Quantitative impact*. An estimate of the impact of the issue in quantitative terms, such as time loss, revenue loss or avoidable costs.

Other fields may be added to an issue register. For example, in view of process redesign, it may be useful to include an attribute *possible resolution* that describes possible mechanisms for addressing the issue.

Example 6.4 We consider again the equipment rental process described in Example 1.1 (p. 2) and the issues described above in Examples 6.2 and 6.3. The issue register given in Table 6.2 provides a more detailed analysis of these issues and their impact.[1]

Question Issue or factor?

An issue register is likely to contain a mixture of issues that have a direct impact on business performance, and others that are essentially causal or contributing factors of other issues that then impact on business performance. In other words, the issue register contains both issues and factors. For example, in the issue register of the equipment rental process, one could find the following entries:

- Clerk misunderstood the site engineer's requirements for an equipment.

[1] In this issue register we do not use multiple columns. This is a pragmatic choice to better fit the issue register within the width of the page.

Table 6.2 Issue register of equipment rental process

Issue 1: Equipment kept longer than needed
Priority: 1
Description: Site engineers keep the equipment longer than needed by means of deadline extensions
Assumptions: BuildIT rents 3000 pieces of equipment per year. In 10 % of cases, site engineers keep the equipment two days longer than needed to avoid disruptions due to delays in equipment rentals. On average, rented equipment costs € 100 per day
Qualitative impact: Not applicable
Quantitative impact: $0.1 \times 3000 \times 2 \times$ € $100 =$ € 60,000 in additional rental expenses per year
Issue 2: Rejected equipment
Priority: 2
Description: Site engineers sometimes reject the delivered equipment due to non-conformance to their specifications
Assumptions: BuildIT rents 3000 pieces of equipment per year. Each time an equipment is rejected due to a mistake on BuildIT's side, BuildIT is billed the cost of one day of rental, that is € 100. 5 % of them are rejected due to an internal mistake within BuildIT (as opposed to a supplier mistake)
Qualitative impact: These events disrupt the construction schedules and create frustration and internal conflicts
Quantitative impact: $3000 \times 0.05 \times$ € $100 =$ € 15,000 per year
Issue 3: Late payment fees
Priority: 3
Description: BuildIT pays late payment fees because invoices are not paid by the due date
Assumptions: BuildIT rents 3000 pieces of equipment per year. Each equipment is rented on average for 4 days at a rate of € 100 per day. Each rental leads to one invoice. About 10 % of invoices are paid late. On average, the penalty for late payment is 2 % of the amount of the invoice
Qualitative impact: Suppliers are annoyed and later unwilling to negotiate more favorable terms for equipment rental
Quantitative impact: $0.1 \times 3000 \times 4 \times$ € $100 \times 0.02 =$ € 2400 per year

- Clerk did not select the correct equipment from the supplier's catalog due to inattention.
- Clerk indicated an incorrect delivery date in the PO and the supplier used this wrong date.
- Supplier did not deliver the exact equipment that had been ordered.
- Delivered equipment is faulty or is not ready-for-use.
- Supplier delivered the equipment to the wrong construction site or at the wrong time.

All of the above issues are possible causal or contributing factors of a top-level issue, namely "Equipment is rejected by the site engineer". The fact that the site engineer rejects the equipment creates a direct impact for BuildIT, for example in terms of delays in the construction schedule. Meanwhile, the issues listed above have an indirect business impact, in the sense that they lead to the equipment being rejected and the needed equipment not being available on time, which in turn leads to delays in the construction schedule.

When an issue register contains a combination of issues and factors, it may be useful to add two fields to the register, namely "caused by" and "is cause of", that indicate for a given issue, which other issues in the register are related to it via a cause–effect relation. This way it becomes easier to identify which issues are related between them so that related issues can be analyzed together. Also, when an issue X is a factor of an issue Y, instead of analyzing both the impact of X and Y, we can analyze the impact of Y and in the qualitative and quantitative impact fields of X we can simply refer to the impact of Y. For example, in the impact field of issue "Clerk misunderstood the site engineer's requirements" we can simply refer to the impact of "Equipment is rejected by the site engineer".

Alternatively, we can adopt the convention of including in the issue register only top-level issues, meaning issues that have a direct business impact, and separately, we can use why–why diagrams and cause–effect diagrams to document the factors underpinning these top-level issues. This convention is followed in the rest of this chapter, meaning that the issue registers shown below only contain top-level issues rather than factors.

Exercise 6.4 Write an issue register for the university admission process and the issue described in Exercise 6.2.

6.3.2 Pareto Analysis and PICK Charts

The impact assessment conducted while building the issue register can serve as input for *Pareto analysis*. The aim of Pareto analysis is to identify which issues or which causal factors of an issue should be given priority. Pareto analysis rests on the principle that a small number of factors are responsible for the largest share of a given effect. In other words:

- A small subset of issues in the issue register are likely responsible for the largest share of impact.
- For a given issue, a small subset of factors behind this issue are likely responsible for the largest share of occurrences of this issue.

Sometimes this principle is also called the 80–20 principle, meaning that 20 % of issues are responsible for 80 % of the effect. One should keep in mind, however, that the specific proportions are only indicative. It may be for example that 30 % of issues are responsible for 70 % of the effect.

A typical approach to conduct Pareto analysis is as follows:

1. Define the effect to be analyzed and the measure via which this effect will be quantified. The measure might be for example:

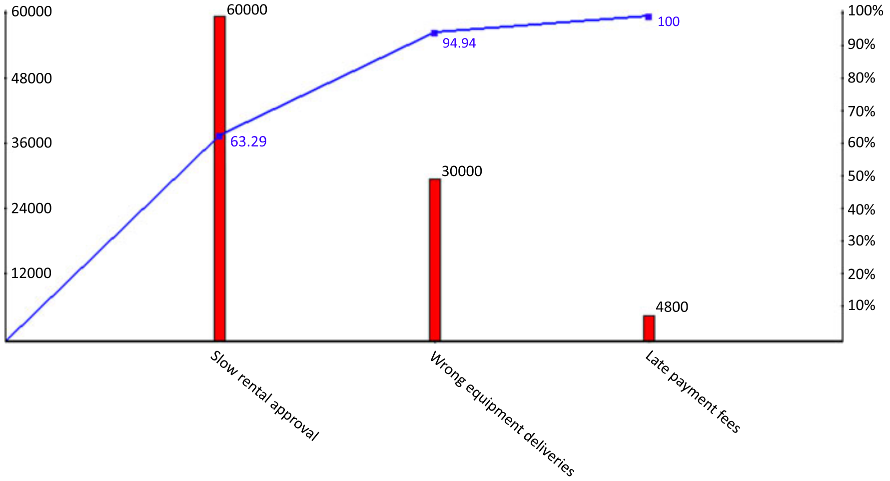


Fig. 6.4 Pareto chart for excessive equipment rental expenditure

- Financial loss for the customer or for the business.
- Time loss by the customer or by the process participants.
- Number of occurrences of a negative outcome, such as number of unsatisfied customers due to errors made when handling their case.

2. Identify all relevant issues that contribute to the effect to be analyzed.
3. Quantify each issue according to the chosen measure. This step can be done on the basis of the issue register, in particular, the quantitative impact column of the register.
4. Sort the issues according to the chosen measure (from highest to lowest impact) and draw a so-called *Pareto chart*. A Pareto chart consists of two components:
   a. A bar chart where each bar corresponds to an issue and the height of the bar is proportional to the impact of the issue or factor.
   b. A curve that plots the cumulative percentage impact of the issues. For example, if the issue with the highest impact is responsible for 40 % of the impact, this curve will have a point with a y-coordinate of 0.4 and an x-coordinate positioned so as to coincide with the first bar in the bar chart.

Example 6.5 Consider again the equipment rental process described in Example 1.1 (p. 2) and the issue register in Example 6.4. All three issues in this register share in common that they are responsible for unnecessary rental expenditure, which is a form of financial loss. From the data in the impact column of the register, we can plot the Pareto chart in Fig. 6.4.

This Pareto chart shows that issue "Slow rental approval" is responsible already for 63 % of unnecessary rental expenditure. Given that in this example there are only three issues, one could have come to this conclusion without conducting Pareto analysis. In practice though, an issue register may contain dozens or hundreds of issues, making Pareto analysis a useful tool to summarize the data in the issue register.

Exercise 6.5 Let us consider again the equipment rental process. This time we take the perspective of the site engineer, whose goal is to have the required equipment available on site when needed. From this perspective, the main issue is that in about 10 % of cases, the requested equipment is not available on site the day when it is required. When this happens, the site engineer contacts the suppliers directly to resolve the issue, but still, resolving the issue may take several days. It is estimated that each such delay costs € 400 per day to BuildIT. By inspecting a random sample of delayed equipment deliveries during a one-year period and investigating the cause of each occurrence, an analyst found that:

1. five occurrences were due to the site engineer not having ordered the equipment with sufficient advance notice: The site engineers ordered the equipment the day before it was needed, when at least two days are needed. These cases cause delays of one day on average.
2. nine occurrences were due to the fact that none of BuiltIT's suppliers had the required type of equipment available on the requested day. These cases cause delays of one to four days (three days on average).
3. 13 occurrences were due to the approval process taking too long (more than a day) due to mistakes or misunderstandings. For these cases, the delay was one day on average.
4. 27 occurrences were due to the equipment having been delivered on time, but the equipment was not suitable and the site engineer rejected it. These cases cause delays of two days on average.
5. four occurrences were due to mistakes or delays attributable entirely to the supplier. These cases lead to delays of one day. However, in these cases, the supplier compensated BuildIT by providing the equipment two days for free (the remaining days are still charged). Recall that the average cost of an equipment rental per day is € 100.
6. For two occurrences, the analyst did not manage to determine the cause of the delay (the process participants could not recall the details). The delays in these cases where two days per occurrence.

The sample of analyzed occurrences represents around 20 % of all occurrences of the issue during a one-year period.

Draw a Pareto chart corresponding to the above data.

It is worth highlighting that Pareto analysis focuses on a single dimension. In the example above, the dimension under analysis is the impact in monetary terms. In other words, we focus on the estimated payoff of addressing an issue. In addition to payoff, there is another dimension that should be taken into account when deciding which issues should be given higher priority, namely the level of difficulty of addressing an issue. This level of difficulty can be quantified in terms of the amount of investment required to change the process in order to address the issue in question.

A type of chart that can be used as a complement to Pareto charts in order to take into account the difficulty dimension is the *PICK chart*. A PICK chart (see Fig. 6.5) is a four-quadrant chart where each issue appears as a point. The horizontal axis

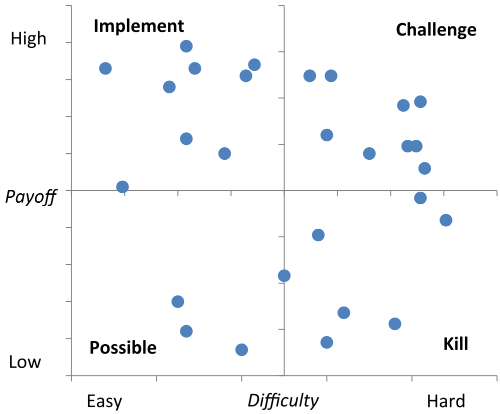


Fig. 6.5 PICK chart

captures the difficulty of addressing the issue (or more specifically the difficulty of implementing a given improvement idea that addresses the issue) while the vertical axis captures the payoff. The horizontal axis (difficulty) is split into two sections (easy and hard) while the vertical axis (payoff) is split into low and high. These splits lead to four quadrants that allow analysts to classify issues according to the trade-off between payoff and difficulty:

- *Possible* (low payoff, easy to do): issues that can be addressed if there are sufficient resources for doing so.
- *Implement* (high payoff, easy to do): issues that should definitely be implemented as a matter of priority.
- *Challenge* (high payoff, hard to do): issues that should be addressed but require significant amount of effort. In general one would pick one of these challenges and focus on it rather than addressing all or multiple challenges at once.
- *Kill* (low payoff, hard to do): issues that are probably not worth addressing or at least not to their full extent.

6.4 Recap

In this chapter, we presented a selection of techniques for qualitative analysis of business processes. The first presented technique, namely value-added analysis, aims at identifying waste, specifically time wasted in activities that do not give value to the customer or to the business. Next, we presented two techniques to uncover the causes of issues that affect the performance of a process, namely cause–effect analysis and why–why analysis. Whereas cause–effect analysis focuses on classifying the factors underpinning the occurrences of an issue, why–why analysis focuses on identifying the recursive cause–effect relations between these factors.

Finally, we presented an approach to systematically document issues in a process, namely the issue register. The purpose of an issue register is to document issues in a semi-structured way and to analyze their impact on the business both from a qualitative and a quantitative angle. In particular, the issue register provides a starting point to build Pareto charts and PICK charts—two visualization techniques that provide a bird's-eye view of a set of issues. These charts help analysts to focus their attention on issues that offer the best payoff (in the case of Pareto charts) or the best trade-off between payoff and difficulty (in the case of PICK charts).

6.5 Solutions to Exercises

Solution 6.1

- VA: receive online application, evaluate academic admissibility, send notification to student.
- BVA: check completeness, academic recognition agency check, English test check.
- NVA: receive physical documents from students, forward documents to committee, notify students service of outcomes of academic admissibility.

Note In this solution we treat the entire agency check as BVA. Part of this agency check consists of the admissions office sending the documents to the agency and the agency sending back the documents and their assessment to the admissions office. These two sub-steps could be treated as NVA. However, if we assume that the agency requires the documents to be sent by post to them, these sub-steps cannot be easily separated from the agency check itself. In other words, it would not be possible to eliminate these handover steps without eliminating the entire agency check. Thus the entire agency check should arguably be treated as a single step.

Solution 6.2 The cause–effect diagram corresponding to this exercise should include at least the name of the issue (e.g. "Student waiting time too long") and the following factors:

- Process stalls due to agency check. This is a "Method" issue, since the issue stems from the fact that the process essentially stalls until a response is received from the agency. One could argue that to some extent this is a "Milieu" issue. But while the slowness of the agency check is a "Milieu" issue, the fact that the process stalls until a response is received from the agency is a "Method" issue.
- Agency check takes too long. This is a "Milieu" issue since the agency is a separate entity that imposes its own limitations.
- Academic committee assessment takes too long. This is a "Method" issue since the process imposes that the academic committee only assesses applications at certain times (when it meets), rather than when applications are ready to be evaluated.
- Physical documents take too long to be received. This is a "Milieu" issue for two reasons. First, the physical documents are needed for the purpose of the agency

check and the delays in the arrival of physical documents are caused by the applicants themselves and postal service delays.
- Admission office delays the notification after academic assessment. This seems to be a "Method" issue, but the description of the process does not give us sufficient information to state this conclusively. Here, a process analyst would need to gather more information in order to understand this issue in further detail.

Solution 6.3
Admission process takes too long, why?

– Process stalls until physical documents arrive, why?
  - Agency check requires physical documents.
  - Other tasks are performed only after agency check, why?
    · Traditionally this is how the process is set-up but there is no strong reason for it.
– Agency check takes too long, why?
  - Exchanges with the agency are via post, why?
    · Agency requires original (or certified) documents due to regulatory requirements.

Academic committee takes too long, why?

– Documents are exchanged by internal mail between admissions office and committee.
– Academic committee only meets at specified times.

Admission office delays the notification after academic assessment, why?

– Not enough information available to analyze this issue (probably due to batching —admissions office sends notifications in batches).

The above analysis already suggests one obvious improvement idea: perform the academic committee assessment in parallel to the agency check. Another improvement opportunity is to replace internal mail communication between admissions office and academic committee with electronic communication (e.g. documents made available to committee members via a Web application).

Note that we could have done the analysis starting from the issue "Admitted students reject their admission offer". This might be useful since there might be several reasons why students reject their offer, some related to the admission process, but also some unrelated to the process.

Solution 6.4 In the following issue register, we only analyze the issue described in this chapter, namely that the admission process takes too long. In practice, the issue register would include multiple issues.

Issue 1: Students reject offer due to long waiting times
Priority: 1
Description: The time between online submission of an application to notification of acceptance takes too long, resulting in some students rejecting their admission offer
Assumptions: Circa 20 students per admission round reject their offer because of the delays. Assessment of each application costs € 100 per student to the university in time spent by admissions office and academic committee, plus an additional € 50 for the agency check. University spends € 100 in marketing for each application it attracts
Qualitative impact: Students who would contribute to the institution in a positive way are lost. Delays in the admission process affect the image of the university vis-a-vis of future students, and generate additional effort to handle enquiries from students while they wait for the admission decisions
Quantitative impact: 20 × € 250 = € 5000 per admission round

In the above issue analysis, the effort required to deal with enquiries during the pre-admission period is listed in the qualitative impact field. If it was possible (with a reasonable amount of effort) to estimate how many such enquiries arrive and how much time they consume, it would be possible to turn this qualitative impact into a quantitative one.

Solution 6.5 First, we analyze the cost incurred by each type of occurrence (i.e. each causal factor) in the sample:

1. Last-minute request: one day delay (because normally two days advance notice are needed), thus € 400 cost × 5 = € 2000.
2. Equipment out-of-stock: three days delay = € 1200 × 9 = € 10,800.
3. Approval delay: one day delay = € 400 × 13 = € 5200.
4. Rejected equipment: two days delay = € 800 × 27 = € 21,600. Note that in Example 6.4 we mentioned that when an equipment is rejected, a fee of € 100 (on average) has to be paid to the supplier for taking back the equipment. However, we do not include this fee here because we are interested in analyzing the costs stemming from equipment not being available on the required day, as opposed to other costs incurred by rejecting equipments.
5. Supplier mistake: one day delay = € 400 minus € 200 in rental cost saving = € 200 × 4 = € 800.
6. Undetermined: two days delay = € 800 × 2 = € 1600.

Since the sample represents 20 % of occurrences of the issue over a year, we multiply the above numbers by five in order to estimate the total yearly loss attributable to each causal factor. The resulting Pareto chart is given in Fig. 6.6.

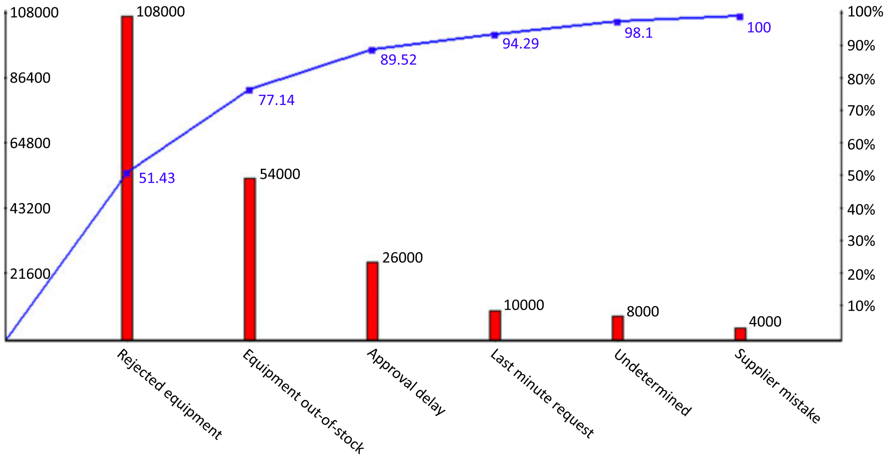


Fig. 6.6 Pareto chart of causal factors of issue "Equipment not available when needed"

6.6 Further Exercises

Exercise 6.6 Consider the following summary of issues reported in a travel agency.

> A travel agency has recently lost several medium-sized and large corporate customers due to complaints about poor customer service. The management team of the travel agency decided to appoint a team of analysts to address this problem. The team gathered data by conducting interviews and surveys with current and past corporate customers and also by gathering customer feedback data that the travel agency has recorded over time. About 2 % of customers complained about errors that had been made in their bookings. In one occasion, a customer had requested a change to a flight booking. The travel agent wrote an e-mail to the customer suggesting that the change had been made and attached a modified travel itinerary. However, it later turned out that the modified booking had not been confirmed in the flight reservation system. As a result, the customer was not allowed to board the flight and this led to a series of severe inconveniences for the customer. Similar problems had occurred when booking a flight initially: the customer had asked for certain dates, but the flight tickets had been issued for different dates. Additionally, customers complained of the long times it took to get responses to their requests for quotes and itineraries. In most cases, employees of the travel agency replied to requests for quotes within 2–4 working hours, but in the case of some complicated itinerary requests (about 10 % of the requests), it took them up to 2 days. Finally, about 5 % of customers also complained that the travel agents did not find the best flight connections and prices for them. These customers essentially stated that they had found better itineraries and prices on the Web by searching by themselves.

1. Analyze the issues described above using root cause analysis techniques.
2. Document the issues in the form of an issue register. To this end, you may assume that the travel agency receives around 100 itinerary requests per day and that the agency makes 50 bookings per day. Each booking brings a gross profit of € 100 to the agency.

Exercise 6.7 Consider the pharmacy prescription fulfillment process described in Exercise 1.6 (p. 28). Identify the steps in this process and classify them into value-adding, business value-adding and non-value-adding.

Exercise 6.8 Consider the procure-to-pay process described in Exercise 1.7 (p. 29). Identify the steps in this process and classify them into value-adding, business value-adding and non-value-adding.

Exercise 6.9 Write an issue register for the pharmacy prescription fulfillment process described in Exercise 1.6 (p. 28). Analyze at least the following issues:

- Sometimes, a prescription cannot be filled because one or more drugs in the prescription are not in stock. The customer only learns this when they come to pick up their prescription.
- Oftentimes, when the customer arrives to pick up the drugs, they find out that they have to pay more than what they expected because their insurance policy does not cover the drugs in the prescription, or because the insurance company covers only a small percentage of the cost of the drugs.
- In a very small number of cases, the prescription cannot be filled because there is a potentially dangerous interaction between one of the drugs in the prescription and other drugs that the customer has been given in the past. The customer only finds out about this issue when they arrive to pick up the prescription.
- Some prescriptions can be filled multiple times. This is call a "refill". Every prescription explicitly states whether a refill is allowed and if so how many refills are allowed. Sometimes, a prescription cannot be filled because the number of allowed refills has been reached. The pharmacist then tries to call the doctor who issued the prescription to check if the doctor would allow an additional refill. Sometimes, however, the doctor is unreachable or the doctor does not authorize the refill. The prescription is then left unfilled and the customer only finds it out when they arrive to pick-up the prescription.
- Oftentimes, especially during peak time, customers have to wait for more than 10 minutes to pick-up their prescription due to queues. Customers find this annoying because they find that having to come twice to the pharmacy (once for drop-off and once for pick-up) should allow the pharmacy ample time to avoid such queues at pick-up.
- Sometimes, the customer arrives at the scheduled time, but the prescription is not yet filled due to delays in the prescription fulfillment process.

When making assumptions to analyze these issues, you may choose to equate "oftentimes" with "20 % of prescriptions", "sometimes" with "5 % of prescriptions" and "very small number of cases" with "1 % of prescriptions". You may also assume that the entire chain of pharmacies consists of 200 pharmacies that serve 4 million prescriptions a year and that the annual revenue of the pharmacy chain attributable to prescriptions is € 200 million. You may also assume that every time a customer is dissatisfied when picking up a prescription, the probability that this customer will not come back after this experience is 20 %. You may also assume that on average a customer requires five prescriptions per year.

Taking the issue register as a basis, apply Pareto Analysis to determine a subset of issues that should be addressed to reduce the customer churn due to dissatisfaction by at least 70 %. Customer churn is the number of customers who stop consuming services offered by a company at a given point in time. In this context, this means the number of customers who stop coming to the pharmacy due to a bad customer experience.

Exercise 6.10 Write an issue register for the procure-to-pay process described in Exercise 1.7 (p. 29).

6.7 Further Reading

Value-added analysis, cause–effect analysis, why–why analysis and Pareto analysis are just a handful of a much wider range of techniques used in the field of Six Sigma (cf. "Related Fields" box in Chap. 1). Conger [8] shows how these and other Six Sigma techniques can be applied for business process analysis. The list of analysis techniques encompassed by Six Sigma is very extensive. A comprehensive listing of Six Sigma techniques is maintained in the iSixSigma portal (http://www.isixsigma.com/tools-templates/). A given business process improvement project will generally only make use of a subset of these techniques. In this respect, Johannsen et al. [38] provide guidelines for selecting analysis techniques for a given BPM project.

Straker's Quality Encyclopedia (see http://www.syque.com/improvement/a_encyclopedia.htm) provides a comprehensive compendium of concepts used in Six Sigma and other quality management disciplines. In particular, it provides definitions and illustrations of the 6M's and the 4P's used in cause–effect diagrams and other concepts related to root cause analysis. A related resource—also by Straker—is the Quality Toolbook, which summarizes a number of quality management techniques. Originally the Quality Toolbook was published as a hardcopy book [89], but it is nowadays also available freely in hyperlinked form at: http://www.syque.com/quality_tools/toolbook/toolbook.htm.

Why–why diagrams allow us to document sequences of cause–effect relations that link factors to a given issue. A related technique to capture cause–effect paths is the *causal factor chart* [77]. Causal factor charts are similar to why–why diagrams. A key difference is that in addition to capturing factors, causal factor charts also capture conditions surrounding the factors. For example, in addition to stating that "the clerk made a data entry mistake when creating the PO", a casual factor chart might also include a condition corresponding to the question "in which part of the PO the clerk made a mistake?" These additional conditions allow analysts to more clearly define each factor.

The issue register has been proposed as a process analysis tool by Schwegmann and Laske [84][2] who use the longer term "list of weaknesses and potential improvements" to refer to an issue register. Schwegmann and Laske argue that the issue register should be built up in parallel with the as-is model, meaning that the discovery of the as-is process and the documentation of issues should go hand in hand. The rationale is that during the workshops organized for the purpose of process discovery (cf. Chap. 5), workshop participants will often feel compelled to voice out issues related to different parts of the process. Therefore, process discovery is an occasion to start listing issues. Naturally, during process discovery, the documentation of issues is left incomplete because the focus is more on understanding the as-is process. Additional analysis after the process discovery phase is required in order to document the issues and their impact in detail.

Another framework commonly used for qualitative process analysis is the Theory of Constraints (TOC) [23]. TOC is especially useful when the goal is to trace weaknesses in the process to specific bottlenecks. In addition to providing a framework for process analysis, TOC also provides guidance to identify, plan and implement changes in order to address the identified bottlenecks. The application of TOC to business process analysis and redesign is discussed at length by Laguna and Marklund [43, Chap. 5] and by Rhee et al. [76].

Finally, a useful framework when it comes to analyzing individual tasks (or activities) in a business process—as opposed to the entire process—is provided by Harmon [31, Chap. 10].[3] This so-called Task Analysis or Activity Analysis framework includes a comprehensive collection of questions and checklists that an analyst should answer and complete in order to identify opportunities to improve the performance of a given activity.

[2]The sub-categorization of the 6M's given in Sect. 6.2.1 also comes from Schwegmann and Laske [84].

[3]An alternate reference describing this framework is [32].

Chapter 7
Quantitative Process Analysis

It is better to be approximately right than precisely wrong.
Warren Buffett (1930–)

Qualitative analysis is a valuable tool to gain systematic insights into a process. However, the results obtained from qualitative analysis are sometimes not detailed enough to provide a solid basis for decision making. Think of the process owner of BuildIT's equipment rental process preparing to make a case to the company's COO that every site engineer should be given a tablet computer with wireless access in order to query suppliers' catalogs and to make rental requests from any construction site. The process owner will be asked to substantiate this investment in quantitative terms and specifically to estimate how much time and money would be saved by doing this investment. To make such estimates, we need to go beyond qualitative analysis.

This chapter introduces a range of techniques for analyzing business processes quantitatively, in terms of performance measures such as cycle time, total waiting time and cost. Specifically, the chapter focuses on three techniques: flow analysis, queueing analysis and simulation. All these techniques have in common that they allow us to calculate performance measures of a process, given data about the performance of individual activities and resources in the process.

7.1 Performance Measures

7.1.1 Process Performance Dimensions

Any company would ideally like to make its processes faster, cheaper, and better. This simple observation leads us already to identifying three *process performance dimensions*: time, cost and quality. A fourth dimension gets involved in the equation once we consider the issue of change. A process might perform extremely well under normal circumstances, but then perform poorly in other circumstances which are perhaps equally or more important. For example, van der Aalst et al. [98] report the story of a business process for handling claims at an Australian insurance

M. Dumas et al., *Fundamentals of Business Process Management*,
DOI 10.1007/978-3-642-33143-5_7,

company. Under normal, everyday conditions, the process was performing to the entire satisfaction of all managers concerned (including the process owner). However, Australia is prone to storms and some of these storms cause serial damages to different types of properties (e.g. houses and cars), leading to numerous claims being lodged in a short period of time. The call center agents and backoffice workers involved in the process were over-flooded with claims and the performance of the process degraded—precisely at the time when the customers were most sensitive to this performance. What was needed was not to make the process faster, cheaper or better during normal periods. Rather, it was needed to make the process more flexible to sudden changes in the amount of claims. This observation leads us to identify a fourth dimension of process performance, namely flexibility.

Each of the four performance dimensions mentioned above (time, cost, quality, and flexibility) can be refined into a number of *process performance measures* (also called *key performance indicators* or *KPIs*). A process performance measure is a quantity that can be unambiguously determined for a given business process—assuming of course that the data to calculate this performance measure is available.

For example, there are several types of cost such as cost of production, cost of delivery or cost of human resources. Each of these types of cost can be further refined into specific performance measures. To do so, one needs to select an aggregation function, such as count, average, variance, percentile, minimum, maximum, or ratios of these aggregation functions. A specific example of a cost performance measure is the average delivery cost per item.

Below, we briefly discuss each of the four dimensions and how they are typically refined into specific performance measures.

Time Often the first performance dimension that comes to mind when analyzing processes is time. Specifically, a very common performance measure for processes is *cycle time* (also called *throughput time*). Cycle time is the time that it takes to handle one case from start to end. Although it is usually the aim of a redesign effort to reduce cycle time, there are many different ways of further specifying this aim. For example, one can aim at a reduction of the average cycle time or the maximal cycle time. It is also possible to focus on the ability to meet cycle times that are agreed upon with a client at run time. Yet another way of looking at cycle time is to focus on its variation, which is notably behind approaches like Six Sigma (cf. Chap. 1). Other aspects of the time dimension come into view when we consider the constituents of cycle time, namely:

1. *Processing time* (also called *service time*): the time that resources (e.g. process participants or software applications invoked by the process) spend on actually handling the case.
2. *Waiting time*: the time that a case spends in idle mode. Waiting time includes *queueing time*—waiting time due to the fact that no resources available to handle the case—and other waiting time, for example because synchronization must take place with another process or because input is expected from a customer or from another external actor.

Cost Another common performance dimension when analyzing and redesigning a business process has a financial nature. While we refer to cost here, it would also have been possible to put the emphasis on turnover, yield, or revenue. Obviously, a yield increase may have the same effect on an organization's profit as a decrease of cost. However, process redesign is more often associated with reducing cost. There are different perspectives on cost. In the first place, it is possible to distinguish between fixed and variable cost. Fixed costs are overhead costs which are (nearly) not affected by the intensity of processing. Typical fixed costs follow from the use of infrastructure and the maintenance of information systems. Variable cost is positively correlated with some variable quantity, such as the level of sales, the number of purchased goods, the number of new hires, etc. A cost notion which is closely related to productivity is *operational cost*. Operational costs can be directly related to the outputs of a business process. A substantial part of operational cost is usually labor cost, the cost related to human resources in producing a good or delivering a service. Within process redesign efforts, it is very common to focus on reducing operation cost, particularly labor cost. The automation of tasks is often seen as an alternative for labor. Obviously, although automation may reduce labor cost, it may cause incidental cost involved with developing the respective application and fixed maintenance cost for the life time of the application.

Quality The quality of a business process can be viewed from at least two different angles: from the client's side and from the process participant's side. This is also known as the distinction between external quality and internal quality. The *external quality* can be measured as the client's satisfaction with either the product or the process. Satisfaction with the product can be expressed as the extent to which a client feels that the specifications or expectations are met by the delivered product. On the other hand, a client's satisfaction with the process concerns the way how it is executed. A typical issue is the amount, relevance, quality, and timeliness of the information that a client receives during execution on the progress being made. On the other hand, the *internal quality* of a business process related to the process participants' viewpoint. Typical internal quality concerns are: the level that a process participants feels in control of the work performed, the level of variation experienced, and whether working within the context of the business process is felt as challenging. It is interesting to note that there are various direct relations between the quality and other dimensions. For example, the external process quality is often measured in terms of time, e.g., the average cycle time or the percentage of cases where deadlines are missed. In this book, we make the choice that whenever a performance measure refers to time, it is classified under the time dimension even if the measure is also related to quality.

Flexibility The criterion that is least noted to measure the effect of a redesign measure is the flexibility of a business process. Flexibility can be defined in general terms as the ability to react to changes. These changes may concern various parts of the business process, for example:

- The ability of resources to execute different tasks within a business process setting.
- The ability of a business process as a whole to handle various cases and changing workloads.
- The ability of the management in charge to change the used structure and allocation rules.
- The organization's ability to change the structure and responsiveness of the business process to wishes of the market and business partners.

Another way of approaching the performance dimension of flexibility is to distinguish between run time and build time flexibility. *Run time flexibility* concerns the opportunities to handle changes and variations while executing a specific business process. *Build time flexibility* concerns the possibility to change the business process structure. It is increasingly important to distinguish the flexibility of a business process from the other dimensions.

Example 7.1 Let us consider the following scenario.

> A restaurant has recently lost many customers due to poor customer service. The management team has decided to address this issue first of all by focusing on the delivery of meals. The team gathered data by asking customers about how quickly they liked to receive their meals and what they considered as an acceptable wait. The data suggested that half of the customers would prefer their meals to be served in 15 minutes or less. All customers agreed that a waiting time of 30 minutes or more is unacceptable.

In this scenario, it appears that the most relevant performance dimension is time, specifically serving time. One objective that distills from the scenario is to completely avoid waiting times above 30 minutes. In other words, the percentage of customers served in less than 30 minutes should be as close as possible to 100 %. Thus, "percentage of customers served in less than 30 minutes" is a relevant performance measure. Another threshold mentioned in the scenario is 15 minutes. There is a choice between aiming to have an average meal serving time below 15 minutes or again, minimizing the number of meals served above 15 minutes. In other words, there is a choice between two performance measures: "average meal delivery time" or "percentage of customers served in 15 minutes".

This example illustrates that the definition of performance measures is tightly connected to the definition of *performance objectives*. In this respect, one possible method for deriving performance measures for a given process is the following:

1. Formulate performance objectives of the process at a high level, in the form of a desirable state that the process should ideally reach, e.g. "customers should be served in less than 30 minutes".
2. For each performance objective, identify the relevant performance dimension(s) and aggregation function(s), and from there, define one or more performance measures for the objective in question, e.g. "percentage of customers served in less than 30 minutes". Let us call this measure ST_{30}.
3. Define a more refined objective based on this performance measure, such as $ST_{30} \geq 99\ \%$.

During the redesign and implementation phases, a possible additional step is to attach a timeframe to the refined performance objective. For example, one can state that the above performance objective should be achieved in 12 months time. A performance objective with a timeframe associated to it is usually called a *performance target*. At the end of the chosen timeframe, one can assess to what extent the redesigned process has attained its targets.

Exercise 7.1 Consider the travel agency scenario described in Exercise 6.6 (p. 208).

1. Which business processes should the travel agency improve?
2. For each of the business processes you identified above, indicate which performance measure should the travel agency improve.

7.1.2 Balanced Scorecard

Another way of classifying and defining performance measures is given by the concept of *Balanced Scorecard*. The Balanced Scorecard is essentially an approach to align the goals and measures that are used to evaluate the work of managers. The main argument behind the Balanced Scorecard is that it is not sufficient to use financial metrics, such as Return-On-Investment (ROI) or operating margin, when evaluating managers. An extreme focus on these measures is in the long-term detriment to the company as it neglects fundamental sources of value, namely the customer, the company's internal structure and the company's employees. Accordingly, the Balanced Scorecard is based on four performance dimensions—each one covering a fundamental concern of a company:

- Financial Measures, e.g. cash flow, to ensure survival, operating margin to ensure shareholder satisfaction.
- Internal Business Measures, e.g. cycle time, to ensure efficiency and low levels of inventory in the case of manufacturing organizations.
- Innovation and Learning Measures, e.g. technology leadership, to ensure competitive advantage and to attract and retain talent.
- Customer Measures, e.g. on-time delivery, to ensure customer satisfaction and loyalty.

The classical way to implement the Balanced Scorecard follows a top-down procedure. It begins with a corporate scorecard, followed by departmental ones with an emphasis on goals and metrics directly affected by the specific department. Process-related measures tend to appear only at the level of heads of units or their subordinates. This classical implementation of the Balanced Scorecard overemphasizes the functional division of organizations not paying enough attention to processes view. Companies implementing the Balanced Scorecard in conjunction with BPM need to carefully consider the relation between the measures in the Balanced Scorecard—both at the corporate level, departmental level and lower levels—and the performance measures associated with their business processes. One way to ensure this alignment is to implement a Balanced Scorecard structured according to the company's process architecture (cf. Chap. 2). This process-oriented Balanced Scorecard may co-exist with a Balanced Scorecard that is traditionally associated to the company's functional architecture.

In any case, we observe that the Balanced Scorecard is a useful tool for identifying process performance measures across an entire organization. This is in contrast with the method based on performance dimensions outlined in Sect. 7.1.1 is geared towards identifying performance measures for one given process. Thus, this latter method and the Balanced Scorecard are complementary.

7.1.3 Reference Models and Industry Benchmarks

Reference process models—previously mentioned in Chap. 2—provide another basis to identify process performance measures. For instance, within the Supply Chain Operations Reference Model (SCOR), processes and activities in the process hierarchy are linked to performance measures. An example of a performance measure in SCOR is the "Purchase Order Cycle Time", defined as the "average amount of time (e.g. days) between the moment an intention to purchase is declared and the moment the corresponding purchase order is received by the relevant vendor". This is basically the average cycle time of a fragment of a procure-to-pay process. Other measures in SCOR deal with inventory management or out-of-stock events. In addition to defining performance measures, SCOR also provides threshold values for each measure that allow a company to compare itself against peers within its industry and to determine whether they are in the top-10 %, top-50 % or bottom-50 % with respect to other companies in their industry sector.

Another relevant framework mentioned in Chap. 2 is APQC's Process Classification Framework (PCF). The primary aim of this framework is to provide a standardized decomposition of processes in an organization together with standardized names and definitions for these processes. As a complement to PCF, APQC has also developed a set of performance measures for the processes included in PCF. This is also a potentially useful tool for performance measure identification.

Yet another example of a reference model that provides a catalog of process performance measures is the *IT Infrastructure Library*—ITIL. ITIL's performance measures include, for example, "Incidents due to Capacity Shortages" defined as the "number of incidents occurring because of insufficient service or component capacity". This performance measure is linked to ITIL's Capacity Management process area, which includes a number of inter-related processes to manage the capacity of IT processes or components of an IT system.

Other reference models that provide catalogs of process performance measures include DCOR (Design Chain Operations Reference model) and eTOM (Enhanced Telecom Operations Map).

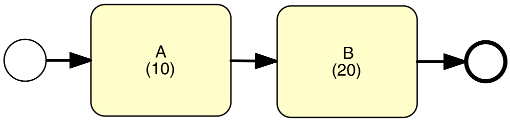


Fig. 7.1 Fully sequential process model

7.2 Flow Analysis

Flow analysis is a family of techniques that allow us to estimate the overall performance of a process given some knowledge about the performance of its activities. For example, using flow analysis we can calculate the average cycle time of an entire process if we know the average cycle time of each activity. We can also use flow analysis to calculate the average cost of a process instance knowing the cost-per-execution of each activity, or calculate the error rate of a process given the error rate of each activity.

In order to understand the scope and applicability of flow analysis, we start by showing how flow analysis can be used to calculate the average cycle time of a process. As a shorthand, we will use the term *cycle time* to refer to *average cycle time* in the rest of this chapter.

7.2.1 Calculating Cycle Time Using Flow Analysis

We recall that the cycle time of a process is the average time it takes between the moment the process starts and the moment it completes. By extension, we say that the cycle time of an activity is the average time it takes between the moment the activity is ready to be executed and the moment it completes.

To understand how flow analysis works, it is useful to start with an example of a purely sequential process as in Fig. 7.1. The cycle time of each activity is indicated between brackets. Since the two activities in this process are performed one after the other, we can intuitively conclude that the cycle time of this process is $20 + 10 = 30$. More generally, it is quite intuitive that the cycle time of a purely sequential fragment of a process is the sum of the cycle times of the activities in the fragment.

When a process model or a fragment of a model contains gateways, the cycle time is no longer the sum of the activity cycle times. Let us consider the example shown in Fig. 7.2. Here, it is clear that the cycle time of the process is not 40 (the sum of the activity cycle times). Indeed, in a given instance of this process, either activity B or activity C is performed. If B is performed, the cycle time is 30, while if C is performed, the cycle time is 20.

Whether the cycle time of this process is closer to 20 or closer to 30 depends on how frequently each branch of the XOR-split is taken. For instance, if in 50 % of instances the upper branch is taken and the remaining 50 % of instances the lower branch is taken, the overall cycle time of the process is 25. However, if the

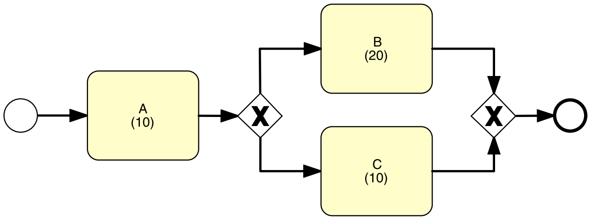


Fig. 7.2 Process model with XOR-block

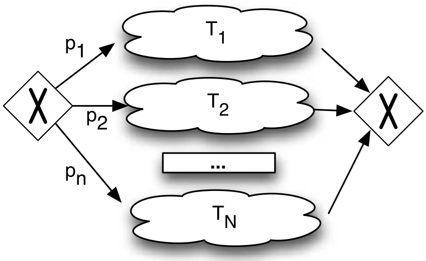


Fig. 7.3 XOR-block pattern

upper branch is taken only 10 % of the times and the lower branch is taken 90 % of the times, the cycle time should be intuitively closer to 30. Generally speaking, the cycle time of the fragment of the process between the XOR-split and the XOR-join is the weighted average of the cycle times of the branches in-between. Thus, if the lower branch has a frequency of 10 % and the upper branch has a frequency of 90 %, the cycle time of the fragment between the XOR-split and the XOR-join is: $0.1 \times 10 + 0.9 \times 20 = 19$. We then need to add the cycle time of activity A (which is always executed) in order to obtain the total cycle time, that is, $10 + 19 = 29$. In the rest of this chapter, we will use the term *branching probability* to denote the frequency with which a given branch of a decision gateway is taken.

In more general terms, the cycle time of a fragment of a process model with the structure shown in Fig. 7.3 is

$$CT = \sum_{i=1}^{n} p_i \times T_i \tag{7.1}$$

In Fig. 7.3, p_1, p_2, etc. are the branching probabilities. Each "cloud" represents a fragment that has a single entry flow and a single exit flow. The cycle times of these nested fragments are T_1, T_2, etc. Hereon, this type of fragment is called a *XOR-block*.

Let us now consider the case where parallel gateways are involved as illustrated in Fig. 7.4.

Again, we can observe that the cycle time of this process cannot be 40 (the sum of the activity cycle times). Instead, since tasks B and C are executed in parallel, their combined cycle time is determined by the slowest of the two activities, that is,

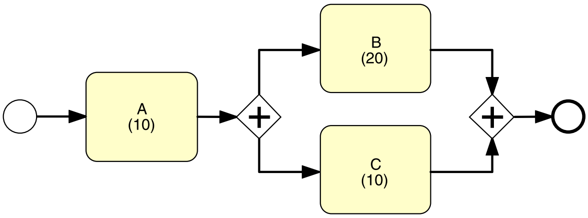


Fig. 7.4 Process model with AND-block

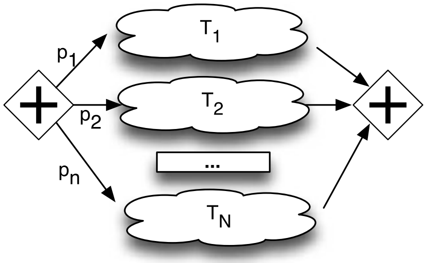


Fig. 7.5 AND-block pattern

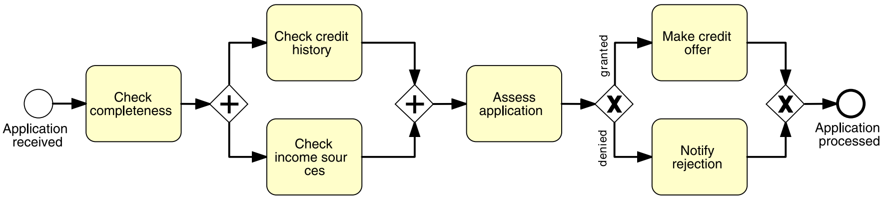


Fig. 7.6 Credit application process

by C. Thus, the cycle time of the process shown in Fig. 7.4 is $10 + 20 = 30$. More generally, the cycle time of an *AND-block* such as the one shown in Fig. 7.5 is

$$CT = Max(T_1, T_2, \ldots, T_n) \tag{7.2}$$

Example 7.2 Let us consider the credit application process model in Fig. 7.6 and the activity cycle times given in Table 7.1. Let us also assume that in 60 % of cases, the credit is granted.

To calculate the cycle time of this process, we first note that the cycle time of the AND-block is 3 days (slowest activity). Next, we calculate the cycle time of the fragment between the XOR-block using (7.1), that is, $0.6 \times 1 + 0.4 \times 2 = 1.4$ days. The total cycle time is then $1 + 3 + 3 + 1.4 = 8.4$ days.

In this example the cycle time is in great part determined by task "Check income sources", which is the one that determines the cycle time of the fragment between the AND-split and the AND-join. In this case, we say that this task is part of the *critical path* of the process. The critical path of a process is the sequence of tasks that determines the cycle time of the process, meaning that the cycle time of any instance of the process is never lower than the sum of the cycle times of this sequence of

Table 7.1 Cycle times for credit application process

Activity	Cycle time
Check completeness	1 day
Check credit history	1 day
Check income sources	3 days
Assess application	3 days
Make credit offer	1 day
Notify rejection	2 days

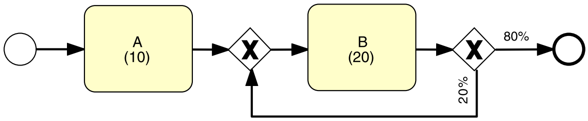


Fig. 7.7 Example of a rework loop

tasks. When optimizing a process with respect to cycle time, one should focus the attention on tasks that belong to the critical path.

Exercise 7.2 Consider the process model given in Fig. 3.8 (p. 73). Calculate the cycle time under the following assumptions:

- Each task in the process takes 1 hour on average.
- In 40 % of the cases the order contains only Amsterdam products.
- In 40 % of the cases it contains only Hamburg products.
- In 20 % of the cases it contains products from both warehouses.

Compare the process model in Fig. 3.8 (p. 73) with the one in Fig. 3.10 (p. 74). Does this comparison give you an idea of how to calculate cycle times for process models with OR gateways?

Another recurrent case worth considering is the case where a fragment of a process may be repeated multiple times. This situation is called *rework* and is illustrated in Fig. 7.7. Here the decimal numbers attached to the arcs denote the probability that the corresponding arc will be taken. For sure, we can say that activity B will be executed once. Next, we can say that activity B may be executed twice with a probability of 20 % (i.e. 0.2), which is the probability of going back from the XOR-split gateway to the XOR-join gateway. If we continue this reasoning, we find out that the probability that task B is executed three times is $0.2 \times 0.2 = 0.04$, and more generally, the probability that task B is executed N times is 0.2^N.

If we sum up the cycle times in the cases where B is executed once, twice, three times, etc., we get the following summation $\sum_{i=0}^{\infty} 0.2^i$. In essence, this is the number of times that B is expected to be executed. If we replace 0.2 with a variable r, this summation is a well-known series, known as the *geometric series* and it can be shown that this series is equivalent to $1/(1-r)$. Hence, the average number of times that B is expected to be executed is $1/(1-0.2) = 1.25$. Now, if we multiply

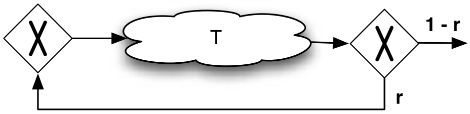


Fig. 7.8 Rework pattern

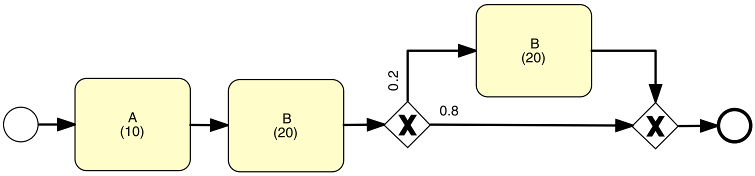


Fig. 7.9 Activity that is reworked at most once

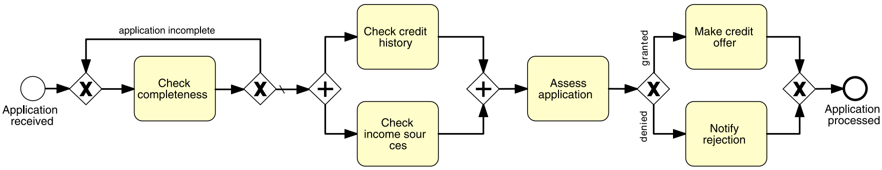


Fig. 7.10 Credit application process with rework

this expected number of instances of B times the cycle time of activity B, we get $1.25 \times 20 = 25$. Thus the total cycle time of the process in Fig. 7.7 is $10 + 25 = 35$.

More generally, the cycle time of the fragment with the structure shown in Fig. 7.8 is

$$CT = \frac{T}{1-r}. \tag{7.3}$$

In this formula, parameter r is called the *rework probability*, that is, the probability that the fragment inside the cycle will need to be reworked. From here on, this type of block will be called a *rework block*, or more generally a *repetition block*.

In some scenarios, an activity is reworked at most once. This situation would be modeled as shown in Fig. 7.9. Using what we have seen, we can already calculate the cycle time of this example. First, we observe that the cycle time of the fragment between the XOR-split and the XOR-join is $0.2 \times 20 + 0.8 \times 0 = 4$. Here, the zero comes from the fact that one of the branches between the XOR-split and the XOR-join is empty and therefore does not contribute to the cycle time. To complete this, we have to add the cycle time of the preceding activities, giving us a total cycle time of 34.

Example 7.3 Let us consider the credit application process model in Fig. 7.10 and the cycle times previously given in Table 7.1. Let us also assume that in 20 % of the cases, the application is incomplete and in 60 % of cases the credit is granted.

The cycle time of the rework block is $10/(1 - 0.2) = 1.25$ days. The cycle time of the AND-block is 3 days and that of the XOR-block is 1.4 days as discussed in Example 7.2. Thus the total cycle time is $1.25 + 3 + 3 + 1.4 = 8.65$ days.

7.2.2 Cycle Time Efficiency

As previously mentioned, the cycle time of an activity or of a process can be divided into *waiting time* and *processing time*. Waiting time is the portion of the cycle time where no work is being done to advance the process. This includes time spent in transferring information about the case between process participants, like for example when documents are exchanged by post, as well as time when the case is waiting for an actor to process it. Processing time on the other hand refers to the time that actors spend doing actual work. In many if not most processes, the waiting time makes up a considerable proportion of the overall cycle time. There are a variety of reasons for this phenomenon. For example, this situation may happen because work is performed in batches. In a process related to the approval of purchase requisitions at a company, the supervisor responsible for such approvals in a business unit might choose to batch all applications and check them only once at the start or the end of a working day. Also, sometimes waiting time is spent waiting for an external actor to provide some input for a task. For example, in the context of fulfilling a medical prescription, a pharmacist may require a clarification from the doctor. To do so, the pharmacist would try to call the doctor. But the doctor might be unavailable and so the pharmacist needs to put the prescription aside and wait until the doctor returns the call.

When analyzing a process with the aim of addressing issues related to cycle time, it may be useful to start by evaluating the ratio of overall processing time relative to the overall cycle time. This ratio is called *cycle time efficiency*. A cycle time efficiency close to 1 indicates that there is little room for improving the cycle time unless relatively radical changes are introduced in the process. A ratio close to zero indicates that there is a significant amount of room for improving cycle time by reducing the waiting time, for example due to handovers between participants.

The cycle time efficiency of a process can be calculated as follows. First, we need to determine the cycle time and the processing time of each activity. Given this information, we can then calculate the overall cycle time of the process using the same formulas we saw above. Let us call this amount CT. Next, using the same formulas, we can calculate the overall amount of time that is spent doing actual work. This is called the *theoretical cycle time* of the process. Essentially, this is the amount of time that an instance of the process would take if there was no waiting time at all. To calculate the theoretical cycle time, we apply the same method as for calculating cycle time, but instead of using the cycle time of each activity, we use the processing time of each activity. Let us call the theoretical cycle time TCT. The cycle time efficiency (CTE) is then calculated as follows:

$$CTE = \frac{TCT}{CT}$$

Example 7.4 Let us consider the credit application process model in Fig. 7.10 and the processing times given in Table 7.2. The activity cycle times (including both waiting and processing time) are those previously given in Table 7.1. Let us assume

Table 7.2 Processing times for credit application process

Activity	Cycle time
Check completeness	2 hours
Check credit history	30 minutes
Check income sources	3 hours
Assess application	2 hours
Make credit offer	2 hours
Notify rejection	30 minutes

Table 7.3 Activity cycle times and processing times for ministerial enquiry process

Activity	Cycle time	Processing time
Register ministerial enquiry	2 days	30 mins
Investigate ministerial enquiry	8 days	12 hours
Prepare ministerial response	4 days	4 hours
Review ministerial response	4 days	2 hour

that in 20 % of cases, the application is incomplete and in 60 % of cases the credit is "granted". Let us additionally assume that one day is equal to 8 working hours.

We have seen in Example 7.3 that the total cycle time of this process is 8.65 days, which translates to 69.2 working hours. We now calculate the theoretical cycle time in the same way as the total cycle time but using the processing times given in Table 7.2. This gives us: $2/(1-0.2)+3+2+0.6\times 2+0.4\times 0.5=9.9$ working hours. The cycle time efficiency is thus $8.9/69.2=12.9$ %.

Exercise 7.3 Calculate the overall cycle time, theoretical cycle time and cycle time efficiency of the ministerial enquiry process introduced in Exercise 3.7 (p. 77). Assume that the rework probability is 0.2 and that the waiting times and processing times are those given in Table 7.3.

7.2.3 Cycle Time and Work-In-Process

Cycle time is directly related to two measures that play an important role when analyzing a process, namely *arrival rate* and *Work-In-Process* (WIP).

The arrival rate of a process is the average number of new instances of the process that are created per time unit. For example, in a credit application process, the arrival rate is the number of credit applications received per day (or any other time unit we choose). Similarly, in an order-to-cash process, the arrival rate is the average number of new orders that arrive per day. Traditionally, the symbol λ is used to refer to the arrival rate.

Meanwhile, WIP is the average number of instances of a process that are active at a given point in time, meaning the average number of instances that have not yet completed. For example, in a credit application process, the WIP is the average number of credit applications that have been submitted and not yet granted or rejected. Similarly, in an order-to-cash process, the WIP is the average number of orders that have been received but not yet delivered and paid.

Cycle time (CT), arrival rate (λ) and WIP are related by a fundamental law known as Little's law, which states that:

$$WIP = \lambda \times CT$$

Basically what this law tells us is that:

- WIP increases if the cycle time increases or if the arrival rate increases. In other words, if the process slows down—meaning that its cycle time increases—there will be more instances of the process active concurrently. Also, the faster new instances are created, the higher will be the number of instances in an active state.
- If the arrival rate increases and we want to keep the WIP at current levels, the cycle time must decrease.

Little's law holds for any stable process. By stable, we mean that the number of active instances is not increasing infinitely. In other words, in a stable process, the amount of work waiting to be performed is not growing beyond control.

Although simple, Little's law can be an interesting tool for what-if analysis. We can also use Little's law as an alternative way of calculating total cycle time of a process if we know the arrival rate and WIP. This can be useful sometimes because determining the arrival rate and WIP is sometimes easier than determining the cycle time. For example, in the case of the credit application process, the arrival rate can be easily calculated if we know the total number of applications processed over a period of time. For example, if we assume there are 250 business days per year and we know the total number of credit applications over the last year is 2500, we can infer that the average number of applications per business day is 10. WIP on the other hand can be calculated by means of sampling. We can ask how many applications are active at a given point in time, then ask this question again one week later and again two weeks later. Let us assume that on average we observe that 200 applications are active concurrently. The cycle time is then $WIP/\lambda = 200/10 = 20$ business days.

Exercise 7.4 A restaurant receives on average 1,200 customers per day (between 10am and 10pm). During peak times (12pm to 3pm and 6pm to 9pm), the restaurant receives around 900 customers in total and, on average, 90 customers can be found in the restaurant at a given point in time. At non-peak times, the restaurant receives 300 customers in total and, on average, 30 customers can be found in the restaurant at a given point in time.

- What is the average time that a customer spends in the restaurant during peak times?
- What is the average time that a customer spends in the restaurant during non-peak times?

- The restaurant's premises have a maximum capacity of 110 customers. This maximum capacity is sometimes reached during peak times. The restaurant manager expects that the number of customers during peak times will increase slightly in the coming months. What can the restaurant do to address this issue without investing in extending its building?

7.2.4 Other Applications and Limitations of Flow Analysis

As mentioned earlier, flow analysis can also be used to calculate other performance measures besides cycle time. For example, assuming we know the average cost of each activity, we can calculate the cost of a process more or less in the same way as we calculate cycle time. In particular, the cost of a sequence of activities is the sum of the costs of these activities. Similarly the cost of an XOR-block is the weighted average of the cost of the branches of the XOR-block, and the cost of a rework pattern such as the one shown in Fig. 7.8 is the cost of the body of the loop divided by $1 - r$. The only difference between calculating cycle time and calculating cost relates to the treatment of AND-blocks. The cost of an AND-block such as the one shown in Fig. 7.5 is not the maximum of the cost of the branches of the AND-block. Instead, the cost of such a block is the sum of the costs of the branches. This is because after the AND-split is traversed, every branch in the AND join is executed and therefore the costs of these branches add up to one another.

Example 7.5 Let us consider again the credit application process model in Fig. 7.10 and the processing times given in Table 7.2. As previously, we assume that in 20 % of cases, the application is incomplete and in 60 % of cases the credit is granted. We further assume that activities "Check completeness", "Check credit history" and "Check income sources" are performed by a clerk, while activity "Assess application", "Make credit offer" and "Notify rejection" are performed by a credit officer. The hourly cost of a clerk is 25 while the hourly cost of a credit officer is 50. Performing a credit history requires that the bank submits a query to an external system. The bank is charged € 1 per query by the provider of this external system.

From this scenario, we can see that the cost of each task can be split into two components: the *human resource cost* and *other costs*. The human resource cost is the cost of the human resource(s) that performs the task. This can be calculated as the product of the hourly cost of the resource and the processing time (in hours) of the task. Other costs correspond to costs that are incurred by an execution of a task, but are not related to the time spent by human resources in the task. In this example, the cost per query to the external system would be classified as "other costs" for task "Check credit history". The remaining tasks do not have an "other cost" component. For the example at hand, the breakdown of resource cost, other cost and total cost per task is given in Table 7.4. Given this input, we can calculate the total cost-per-execution of the process as follows: $50/(1 - 0.2) + 13.5 + 75 + 100 + 0.6 \times 100 + 0.4 \times 25 = 321$.

Table 7.4 Cost calculation table for credit application process

Activity	Resource cost	Other cost	Total cost
Check completeness	$2 \times 25 = 50$	0	50
Check credit history	$0.5 \times 25 = 12.5$	1	13.5
Check income sources	$3 \times 25 = 75$	0	75
Assess application	$2 \times 50 = 100$	0	50
Make credit offer	$2 \times 50 = 100$	0	100
Notify rejection	$0.5 \times 50 = 25$	0	25

Exercise 7.5 Calculate the cost-per-execution of the ministerial enquiry process introduced in Exercise 3.7 (p. 77). Assume that the rework probability is 0.2 and that the times are those given in Table 7.3. Activity "Register ministerial enquiry" is performed by a clerk, activity "Investigate ministerial enquiry" is performed by an adviser, "Prepare ministerial response" is performed by a senior adviser, and "Review ministerial response" is performed by a minister counselor. The hourly resource cost of a clerk, adviser, senior adviser and minister counselor are 25, 50, 75, and 100, respectively. There are no other costs attached to these activities besides the resource costs.

Before closing the discussion on flow analysis, it is important to highlight some of its pitfalls and limitations. First of all, we should note that the equations presented in Sect. 7.2.1 do not allow us to calculate the cycle time of any process model. In fact, these equations only work in the case of block-structured process models. In particular, we cannot use these equations to calculate the cycle time of an unstructured process model such as the one shown in Exercise 3.9 (p. 93). Indeed, this example does not fit into any of the patterns we have seen above. Calculating the cycle time in this case is trickier. Also, if the model contains other modeling constructs besides AND and XOR gateways, the method for calculating cycle time becomes more complicated.

Fortunately, this is not a fundamental limitation of flow analysis, but only a limitation of the specific set of equations discussed in Sect. 7.2.1. There are other more sophisticated flow analysis techniques that allow us to calculate the cycle time of virtually any process model. The maths can get a bit more complex and in practice, one would not do such calculations by hand. But this is generally not a problem given that several modern process modeling tools include functionality for calculating cycle time, cost, and other performance measures of a process model using flow analysis.

A more fundamental roadblock faced by analysts when applying flow analysis is the fact that they first need to estimate the average cycle time of each activity in the process model. In fact, this obstacle is typical when applying any quantitative process analysis technique. There are at least two approaches to address this obstacle. The first one is based on interviews or observation. In this approach, analysts interview the stakeholders involved in each task or they observe how the stakeholders work during a given day or period of time. This allows analysts to at least make an "informed guess" regarding the average time a case spends in each activity, both in terms of waiting time and processing time. A second approach is to collect logs from the information systems used in the process. For example, if a given activity such as approving a purchase requisition is performed by means of an internal Web portal (an Intranet), the administrators of the portal should be able to extract logs from this portal that would allow the analyst to estimate the average amount of time that a requisition form spends in "waiting for approval" mode and also the average time between the moment the supervisor opens a purchase requisition for approval and the time they approve it. With careful analysis, these logs can provide a wealth of information that can be combined via flow analysis to get an overall picture of which parts of the process consume the most time.

A major limitation of flow analysis is that it does not take into account the fact that a process behaves differently depending on the load, that is, depending on the amount of instances of the process that are running concurrently. Intuitively, the cycle time of a process for handling insurance claims would be much slower if the insurance company is handling thousands of claims at once, due for example to a recent natural disaster such as a storm, versus the case where the load is low and the insurance company is only handling a hundred claims at once. When the load goes up and the number of resources (e.g. claim handlers) remains relatively constant, it is clear that the waiting times are going to be longer. This is due to a phenomenon known as *resource contention*. Resource contention occurs when there is more work to be done than resources available to perform the work, like for example more claims than insurance claim handlers. In such scenarios, some tasks will be in waiting mode until one of the necessary resources are freed up. Flow analysis does not take into account the effects of increased resource contention. Instead, the estimates obtained from flow analysis are only applicable if the level of resource contention remains relatively stable over the long-run.

7.3 Queues

Queueing theory is a collection of mathematical techniques to analyze systems that have resource contention. Resource contention inevitably leads to queues as we all probably have experienced in supermarket check-out counters, at a bank's office, post office or government agency. Queueing theory gives us techniques to analyze important parameters of a queue such as the expected length of the queue or the expected waiting time of an individual case in a queue.

7.3.1 Basics of Queueing Theory

In basic queueing theory, a *queueing system* is seen as consisting of one or multiple *queues* and a *service* that is provided by one or multiple *servers*. The elements inside a queue are called *jobs* or *customers*, depending on the specific context. For example, in the case of a supermarket, the service is that of checking out. This service is provided by multiple cashiers (the servers). Meanwhile, in the case of a bank office, the service is to perform a banking transaction, the servers are tellers, and there is generally a single queue that leads to multiple servers (the tellers). These two examples illustrate an important distinction between multi-line (i.e. multi-queue) queueing systems (like the supermarket) and single-line queueing systems (like the bank office).

Queueing theory provides a very broad set of techniques. It would be unrealistic to introduce all these techniques in this chapter. So instead of trying to present everything that queueing theory has to offer, we will present two queueing theory models that are relatively simple, yet useful when analyzing business processes or activities within a process.

In the two models we will be presenting, there is a single queue (single-line queueing system). Customers come at a given mean arrival rate that we will call λ. This is the same concept of arrival rate that we discussed above when presenting Little's law. For example, we can say that customers arrive at the bank office at a mean rate of 20 per hour. This implies that, on average, one customer arrives every 5 minutes ($\frac{1}{20}$ hour). This latter number is called the mean *inter-arrival time*. We observe that if λ is the arrival rate per time unit, then $1/\lambda$ is the mean inter-arrival time.

It would be illusory to think that the time between the arrival of two customers at the post office is always 5 minutes. This is just the mean value. In practice, customers arrive independently from one another, so the time between the arrival of one customer and the arrival of the next customer is completely random. Moreover, let us say that the time between the arrival of the first customer and the arrival of the second customer is 1 minute. This observation does not tell us absolutely anything about the time between the arrival of the second customer and the arrival of the third customer. It might be that the third customer arrives 1 minute after the second, or 5 minutes or 10 minutes. We will not know until the third customer arrives.

Such an arrival process is called a *Poisson process*. In this case, the distribution of arrivals between any two consecutive customers follows a so-called *exponential distribution* (specifically a *negative exponential distribution*) with a mean of $1/\lambda$. In a nutshell, this means that the probability that the inter-arrival time is exactly equal to t (where t is a positive number) decreases in an exponential manner when t increases. For instance, the probability of the time of inter-arrival time being 10 is considerably smaller than the probability of the inter-arrival time being 1. Hence, shorter inter-arrival times are much more probable than longer ones, but there is always a probability (perhaps a very small one) that the inter-arrival time will be a large number.

In practice, the Poisson process and the exponential distribution describe a large class of arrival processes that can be found in business processes, so we will be using them to capture the arrival of jobs or customers into a business process or an activity in a business process. The Poisson process can also be observed for example when we examine how often cars enter a given segment of a highway, or how often calls go through a telephone exchange.

Having said this, one must always cross-check that cases arrive to a given process or activity in an exponentially distributed manner. This cross-check can be done by recording the inter-arrival times for a given period of time, and then feeding these numbers into a statistical tool such as for example R, Mathworks's Statistical Toolbox or EasyFit. These tools allow one to input a set of observed inter-arrival times and check if it follows a negative exponential distribution.

Exponential distributions are not only useful when modeling the inter-arrival time. They are also in some cases useful when describing the processing time of an activity.[1] In the case of activities that require a diagnosis, a non-trivial verification or some non-trivial decision making, it is often the case that the activity's processing time is exponentially distributed. Take for example the amount of time it takes for a mechanic to make a repair on a car. Most repairs are fairly standard, and the mechanics might take for example one hour to do them. However, some repairs are very complex, and in such cases, it can take the mechanic several hours to complete. A similar remark can be made of a doctor receiving patients in an emergency room. A large number of emergencies are quite standard and can be dispatched in less than an hour, but some emergencies are extremely complicated and can take hours to deal with. So it is likely that such activities will follow an exponential distribution. As mentioned above, when making such a hypothesis, it is important that you first check it by taking a random sample of processing times and feeding them to a statistical tool.

In the queueing theory field, a single-queue system is called an *M/M/1 queue* if the inter-arrival times of customers follow an exponential distribution, the processing times follow an exponential distribution, there is one single server and jobs are served on a First-In-First-Out (FIFO) basis. In the case of M/M/1 queue, we also assume that when a job arrives, it enters the queue and it stays there until it is taken on by the server.

If the above conditions are satisfied, but there are multiple servers instead of a single server, the queueing system is said to be *M/M/c*, where *c* is number of servers. For example, a queue is M/M/5 if the inter-arrival times of customers follow an exponential distribution, the processing times follow an exponential distribution and there are five servers at the end of the queue. The "M" in this denomination stands for "Markovian", which is the name given to the assumptions that inter-arrival times and processing times follow an exponential distribution. Other queueing models exist that make different assumptions. Each such model is different, so the results we will obtain for an M/M/1 or M/M/c queue are quite different from those we would obtain from other distributions.

[1] In queueing theory, the term service time is used instead of processing time. For uniformity purposes, here we use the term processing time.

7.3.2 M/M/1 and M/M/c Models

To summarize the previous discussion, an M/M/1 queue or M/M/c queue can be defined by means of the following parameters:

- λ—the mean arrival rate per time unit. The mean inter-arrival time is then $1/\lambda$. For example, $\lambda = 5$ means that there are 5 arrivals per hour and this entails that the mean inter-arrival time between two consecutive jobs is $1/5$ hours, that is 12 minutes.
- μ—the mean number of customers that can be served per time unit. The mean processing time per job is then $1/\mu$. For example, $\mu = 6$ means six jobs are served per hour, that is, one job is served in 10 minutes (on average).
- In the case of M/M/c, the number of servers (c).

Given parameters λ and μ, we can already state how busy a server is. This is called the occupation rate ρ and is equal to λ/μ. In the above example, the occupation rate is $5/6 = 83.34$ %. It should be noted that this is a relatively high occupation rate. A system with an occupation rate of more than 100 % is unstable, meaning that the queue will become longer and longer forever because the server cannot cope with all the demand. In fact, even a system at close to 100 % of occupation rate is unstable because of the randomness at which new jobs arrive and the variability in the processing times per job. To understand why this is the case, just imagine if you were a doctor receiving patients at a rate of 6 per hour for 8 hours, knowing that every patient takes 10 minutes on average to be treated (sometimes less but sometimes more). Without any slack, most likely you will end up with a tremendous backlog at the end of the day.

In the case of an M/M/c system, the occupancy rate is $\frac{\lambda}{c\mu}$ since the system can handle jobs at a rate of $c\mu$. For example, if the system has two servers and each server can handle two jobs per hour, the system can handle four jobs per hour. If jobs arrive at a mean rate of 3 per hour, the occupancy rate of the system is $3/4 = 75$ %.

Given an M/M/1 or M/M/c system, queueing theory allows us to calculate the following parameters:

- L_q—The average number of jobs (e.g. customers) in the queue.
- W_q—The average time one job spends in the queue.
- W—The average time one job spends in the system. This includes both the time the customer spends in the queue but also the time the customer spends being serviced.
- L—The average number of jobs in the system (i.e. the Work-in-Progress referenced in Little's law).

To summarize, the general structure of a single-queue system—consisting of one queue and one or many servers—is depicted in Fig. 7.11. The parameters of the queue (λ, c and μ) are shown at the top. The parameters that can be computed from these three input parameters are shown under the queue and the server. The average time a job waits in the queue is W_q, while the average length of the queue is L_q. Eventually, a job goes into the server and in there it spends on average $1/\mu$

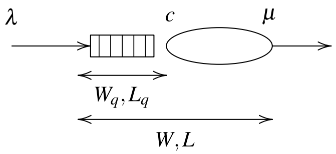


Fig. 7.11 Structure of an M/M/1 or M/M/c system, input parameters and computable parameters

time units. The average time between the moment a job enters the system and the moment it exits is W, while the average number of jobs inside the system (in the queue or in a server) is L.

Queueing theory gives us the following formulas for calculating the above parameters for M/M/1 models:

$$L_q = \rho^2/(1-\rho) \tag{7.4}$$

$$W_q = \frac{L_q}{\lambda} \tag{7.5}$$

$$W = W_q + \frac{1}{\mu} \tag{7.6}$$

$$L = \lambda W \tag{7.7}$$

Formulas (7.5), (7.6), and (7.7) can be applied to M/M/c models as well. The only parameter that needs to be calculated differently in the case of M/M/c models is L_q. For M/M/c models, L_q is given by the following formula:

$$L_q = \frac{(\lambda/\mu)^c \rho}{c!(1-\rho)^2\left(\frac{(\lambda/\mu)^c}{c!(1-\rho)} + \sum_{n=0}^{c-1} \frac{(\lambda/\mu)^n}{n!}\right)} \tag{7.8}$$

The formula for computing L_q in the case of M/M/c models is particularly complicated because of the summations and factorials. Fortunately, there are tools that can do this for us. For example, the Queueing Toolpack[2] supports calculations for M/M/c systems (called *M/M/s* in the Queueing Toolpack) as well as M/M/c/k systems, where k is the maximum number of jobs allowed in the queue. Jobs that arrive when the length of the queue is k are rejected (and may come back later). Other tools for analyzing queueing systems include QSim[3] and PDQ.[4]

Example 7.6 A company designs customized electronic hardware for a range of customers in the high-tech electronics industry. The company receives orders for designing a new circuit every 20 working days on average. It takes a team of engineers on average 10 working days to design a hardware.

This problem can be mapped to an M/M/1 model assuming that the arrival of designs follows a Poisson process, that the distribution of times for designing a circuit follows an exponential distribution and that new design requests are handled

[2] http://apps.business.ualberta.ca/aingolfsson/qtp/.

[3] http://www.stat.auckland.ac.nz/~stats255/qsim/qsim.html.

[4] http://www.perfdynamics.com/Tools/PDQ.html.

on a FIFO manner. Note that even though the team includes several people, they act as a monolithic entity and therefore it should be treated as a single server.

We will hereby take the working day as a time unit. On average, 0.05 orders are received per day ($\lambda = 0.05$), and 0.2 orders are fulfilled per day ($\mu = 0.1$). Thus, the occupation rate of this system $\rho = 0.05/0.1 = 0.5$. Using the formulas for M/M/1 models, we can deduce that the average length of the queue L_q is: $0.5^2/(1-0.5) = 0.5$ orders. From there we can conclude that the average time an order spends on the queue is $W_q = 0.5/0.05 = 10$ days. Thus, it takes on average order $W = 10 + 1/0.1 = 20$ working days for an order to be fulfilled.

Exercise 7.6 Consider now the case where the engineering team in the previous example takes 16 working days to design a hardware. What is then the average amount of time an order takes to be fulfilled?

Exercise 7.7 An insurance company receives 220 calls per day from customers who want to lodge an insurance claim. The call center is open from 8am to 5pm. The arrival of calls follows a Poisson process. Looking at the intensity of arrival of calls, we can distinguish three periods during the day: the period 8am to 11am, the period 11am to 2pm and the period 2pm to 5pm. During the first period, around 60 calls are received. During the 11am–2pm period, 120 calls are received, and during the 2pm–5pm period, 40 calls are received. A customer survey has shown that customers tend to call between 11am and 2pm because during this time they have a break at work and they take advantage of their break to make their personal calls.

Statistical analysis shows that the durations of calls follow an exponential distribution.

According to the company's customer service charter, customers should wait no more than one minute on average for their call to be answered.

- Assume that the call center can handle 70 calls per hour using seven call center agents. Is this enough to meet the 1-minute constraint set in the customer service charter? Please explain your answer by showing how you calculate the average length of the queue and the average waiting time.
- What happens if the call center's capacity is increased so that it can handle 80 calls per hour (using eight call center agents)?
- The call center manager has been given a mandate to cut costs by at least 20 %. Give at least two ideas to achieve this cut without reducing the salaries of the call center agents and while keeping an average waiting time below or close to one minute.

7.3.3 Limitations of Basic Queueing Theory

The basic queueing analysis techniques presented above allow us to estimate waiting times and queue length based on the assumptions that inter-arrival times and processing times follow an exponential distribution. When these parameters follow different distributions, one needs to use very different queueing models. Fortunately, queueing theory tools nowadays support a broad range of queueing models and of course they can do the calculations for us. The discussion above was intended as an overview of single-queue models, with the aim of providing a starting point from where you can learn more about this family of techniques.

A more fundamental limitation of the techniques introduced in this section is that they only deal with one activity at a time. When we have to analyze an entire process that involves several activities, events, and resources, these basic techniques are not sufficient. There are many other queueing analysis techniques that could be used for this purpose, like for example queueing networks. Essentially, queueing networks are systems consisting of multiple inter-connected queues. However, the maths behind queueing networks can become quite complex, especially when the process includes concurrent activities. A more popular approach for quantitative analysis of process models under varying levels of resource contention is process simulation, as discussed below.

7.4 Simulation

Process simulation is arguably the most popular and most widely supported technique for quantitative analysis of process models. The basic idea underpinning process simulation is quite simple. In essence, a process simulator generates a large number of hypothetical instances of a process, executes these instances step-by-step, and records each step in this execution. The output of a simulator typically includes the logs of the simulation as well as some statistics related to cycle times, average waiting times and average resource utilization.

7.4.1 Anatomy of a Process Simulation

During a process simulation, the tasks in the process are not actually executed. Instead, the simulation of a task proceeds as follows. When a task is ready to be executed, a so-called *work item* is created and the simulator first tries to find a resource to which it can assign this work item. If no resource able to perform the work item is found, the simulator puts the work item in waiting mode until a suitable resource is freed up. Once a resource is assigned to a work item, the simulator determines the duration of the work item by drawing a random number according to the probability distribution of the task's processing time. This probability distribution and the corresponding parameters need to be defined in the simulation model.

Once the simulator has determined the duration of a work item, it puts the work item in sleeping mode for that duration. This sleeping mode simulates the fact that the task is being executed. Once the time interval has passed (according to the simulation's clock), the work item is declared to be completed, and the resource that was assigned to it becomes available.

In reality, the simulator does not effectively wait for tasks to come back from their sleeping mode. For example, if the simulator determines that the duration of a work item is 2 days and 2 hours, it will not wait for this amount of time to pass by. You can imagine how long a simulation would take if that was the case. Luckily, simulators use smart algorithms to complete the simulation as fast as possible. Modern business process simulators can effectively simulate thousands of process instances and tens of thousands of work items in a matter of seconds.

For each work item created during a simulation, the simulator records the identifier of the resource that was assigned to this instance as well as three time stamps:

- The time when the task was ready to be executed.
- The time when the task was started, meaning that it was assigned to a resource.
- The time when the task completed.

Using the collected data, the simulator can compute the average waiting time for each task. These measures are quite important when we try to identify bottlenecks in the process. Indeed, if a task has a very high average waiting time, it means that there is a bottleneck at the level of this task. The analyst can then consider several options for addressing this bottleneck.

Additionally, since the simulator records which resources perform which work items and it knows how long each work item takes, the simulator can find out the total amount of time during which a given resource is busy handling work items. By dividing the amount of time that a resource was busy during a simulation by the total duration of the simulation, we obtain the *resource utilization*, that is, the percentage of time that the resource is busy on average.[5]

7.4.2 Input for Process Simulation

From the above description of how a simulation works, we can see that the following information needs to be specified for each task in the process model in order to simulate it:

- Probability distribution for the processing time of each task.
- Other performance attributes for the task such as cost and added-value produced by the task.
- The set of resources that are able to perform the task. This set is usually called a *resource pool*. For example, a possible resource pool could be the "Claim Handlers" or "Clerks" or "Managers". Separately, the analyst needs to specify for each resource pool the number of resources in this pool (e.g. the number of claim handlers or the number of clerks) and other attributes of these resources such as the hourly cost (e.g. the hourly cost of a claims handler).

[5] Note that when discussing queueing theory above, we used the term occupation rate instead of resource utilization. These two terms are synonyms.

Common probability distributions for task durations in the context of process simulation include:

- *Fixed.* This is the case where the processing time of the task is the same for all executions of this task. It is rare to find such tasks because most tasks, especially those involving human resources, would exhibit some variability in their processing time. Examples of tasks with fixed processing time can be found among automated tasks such as for example a task that generates a report from a database. Such a task would take a relatively constant amount of time, say for example 5 seconds.
- *Exponential distribution*. As discussed in Sect. 7.3, the exponential distribution may be applicable when the processing time of the task is most often around a given mean value, but sometimes it is considerably longer. For example, consider a task "Assess insurance claims" in an insurance claims handling process. You can imagine that in most cases, the insurance claims fall within very standard cases. In such cases, the claim is assessed in an hour, or perhaps less. However, some insurance claims require special treatment, for example because the assessor considers that there is a risk that the claim is fraudulent. In this case, the assessor might spend several hours or even an entire day assessing a single claim. A similar observation can be made of diagnostics tasks, such as diagnosing a problem in an IT infrastructure, or diagnosing a problem during a car repair process.
- *Normal distribution*. This distribution is used when the processing time of the task is around a given average, and the "deviation" around this value is symmetric, meaning that the actual processing time can be above or below the mean with the same probability. Simple checks, such as for example checking whether or not a paper form has been fully completed might follow this distribution. Indeed, it generally takes about 3 minutes to make such a check. In some cases, this time can be lower because for example the form is clearly incomplete or clearly complete, and in other cases it can take a bit longer because a couple of fields have been left empty and it is unclear if these fields are relevant or not for the specific customer who submitted the form.

When assigning an exponential distribution to a task duration, the analyst has to specify the mean value. Meanwhile, when assigning a normal distribution, the analyst has to specify two parameters: mean value and standard deviation. These values are derived through an informed guess (based on interviews with the relevant stakeholders), but preferably by means of sampling (the analyst collects data for a sample of tasks executions) or by analyzing logs of relevant information systems. Some simulation tools allow the analyst to import logs into the simulation tool and assist the analyst in selecting the right probability distribution for task durations based on these logs. This functionality is called *simulation input analysis*.

In addition to the above per-task simulation data, a branching probability needs to be specified for every arc stemming from a decision gateway. These probabilities are determined by interviewing relevant stakeholders, observing executions of the process during a certain period of time, or collecting logs from relevant information systems.

Finally, in order to run a simulation, the analyst additionally needs to specify at least the following:

- The inter-arrival times and the mean arrival rate. As explained above, a very frequent distribution of inter-arrival times is the exponential distribution and this is usually the default distribution supported by business process simulators. It may happen, however, that the inter-arrival times follow a different distribution such as for example a *normal distribution*. By feeding a sample of inter-arrival times during a certain period of time to a statistical tool, we can find out which distribution best matches the data. Some simulators provide a module for selecting a distribution for the inter-arrival times and for computing the mean inter-arrival time from a data sample.
- The starting date and time of the simulation (e.g. "11 Nov. 2012 at 8:00").
- One of the following:
  - The end date and time of the simulation. If this option is taken, the simulation will stop producing more process instances once the simulation clock reaches the end time.
  - The real-time duration of the simulation (e.g. 7 days, 14 days). In this way, the end time of the simulation can be derived by adding this duration to the starting time.
  - The required number of process instances to be simulated (e.g. 1,000). If this option is taken, the simulator generates process instances according to the arrival rate until it reaches the required number of process instances. At this point, the simulation stops. Some simulators will not stop immediately, but will allow the active process instances to complete before stopping the simulation.

Example 7.7 We consider the process for loan application approval modeled in Fig. 4.6 (p. 104). We simulate this model using the BIMP simulator available at: http://bimp.cs.ut.ee. This simulator takes as input BPMN process models in XML format produced by other process modeling tools such as Signavio Process Editor or OpenText Provision. We provide the following inputs for the simulation.

- Two loan applications per hour, meaning an inter-arrival time of 30.
- Tasks "Check credit history" and "Check income sources" are performed by clerks.
- Tasks "Notify rejection", "Make credit offer" and "Assess application" are performed by credit officers.
- Task "Receive customer feedback" is in fact an event. It takes zero time and it only involves the credit information system (no human actors involved). To capture this, the task is assigned to a special "System" role.
- There are three clerks and three credit officers. The hourly cost of a clerk is € 25 while that of a credit officer is € 50.
- Clerks and credit officers work from 9am to 5pm during weekdays.
- The cycle time of task "Assess application" follows an exponential distribution with a mean of 20 minutes.

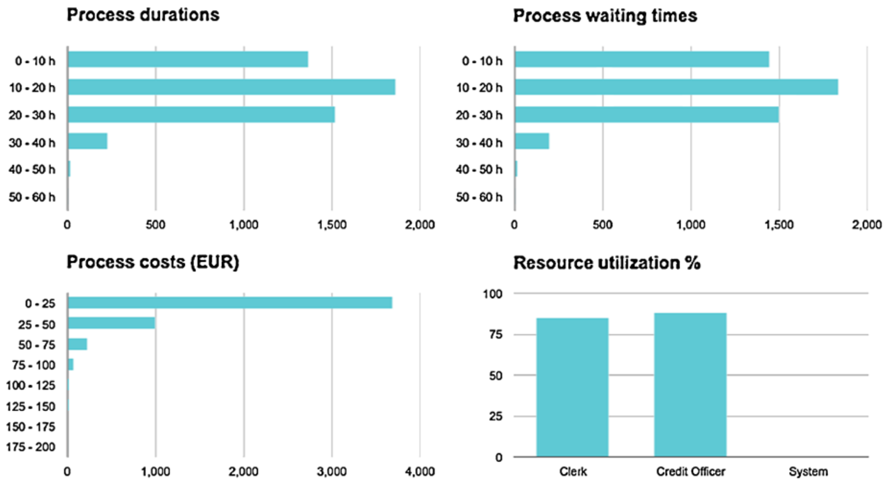


Fig. 7.12 Histograms produced by simulation of the credit application process

- Cycle times of all other tasks follow a normal distribution. Tasks "Check credit history", "Notify rejection" and "Make credit offer" have a mean cycle time of 10 minutes with a 20 % standard deviation, while "Check income sources" has a cycle time of 20 minutes with a 20 % standard deviation as well.
- The probability that an application is accepted is 80 %.
- The probability that a customer whose application was rejected, asks that the application be re-assessed is 20 %.

We run a simulation with 5,000 instances, which means around 104 days of loan applications arrivals assuming that applications arrive 24 hours a day, 7 days a week.[6] The simulation gives an average cycle time of around 17 hours. A variance of ± 2 hours can be observed when running the simulation multiple times. This variance is expected due to the stochastic nature of the simulation. Accordingly, it is recommended to run the simulation multiple times and to take averages of the simulation results. Figure 7.12 shows the histograms for process cycle time (called process duration in BIMP), waiting time (time a case spends waiting for resources to become available), cost (of resources), and resource utilization. It can be seen that applications spend most of the time in waiting mode, waiting for resources to become available. Resource utilization of clerks and credit officers is at 85 % and 88.5 %, respectively, meaning that there is some overload. As a rule of thumb, a resource utilization above 80 % means that one can expect long queues and high waiting times. If we add two clerks and two credit officers to the simulation we obtain an average cycle time of around 8 hours (compared to 17 hours) and an utilization rate of around 80 % for clerks and 50 % for credit officers.

[6] Some simulators additionally allow one to specify that new cases are only created during certain times of the day and certain days of the week, or according to a given calendar.

Exercise 7.8 An insurance company, namely Cetera, is facing the following problem: Whenever there is a major event (e.g. a storm), their claim-to-resolution process is unable to cope with the ensuing spike in demand. During normal times, the insurance company receives about 9,000 calls per week, but during a storm scenario, the number of calls per week doubles.

The claim-to-resolution process model of Cetera is presented in Fig. 7.13. The process starts when a call related to lodging a claim is received. The call is routed to one of two call centers depending on the location of the caller. Each call center receives approximately the same amount of calls (50–50) and has the same number of operators (40 per call center). The process for handling calls is identical across both call centers. When a call is received at a call center, the call is picked up by a call center operator. The call center operator starts by asking a standard set of questions to the customer to determine if the customer has the minimum information required to lodge a claim (e.g. insurance policy number). If the customer has enough information, the operator then goes through a questionnaire with the customer, enters all relevant details, checks the completeness of the claim and registers the claim.

Once a claim has been registered, it is routed by the claims handling office, where all remaining steps are performed. There is one single claims handling office, so regardless of the call center agent where the claim is registered, the claim is routed to the same office. In this office, the claim goes through a two-stage evaluation process. First of all, the liability of the customer is determined. Secondly, the claim is assessed in order to determine if the insurance company has to cover this liability and to what extent. If the claim is accepted, payment is initiated and the customer is advised of the amount to be paid. The activities of the claims handling department are performed by *claims handlers*. There are 150 claims handlers in total.

The mean cycle time of each task (in seconds) is indicated in Fig. 7.13. For every task, the cycle time follows an exponential distribution. The hourly cost of a call center agent is 30, while hourly cost of a claims handler is 50.

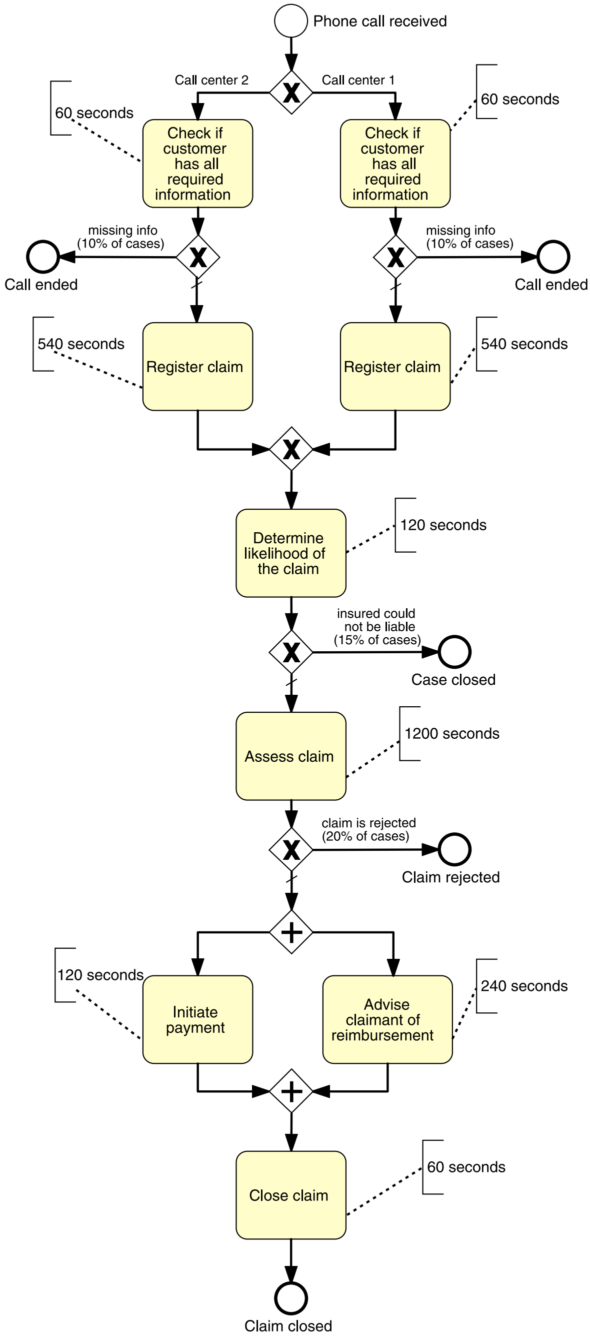
Describe the input that should be given to a simulator in order to simulate this process in the normal scenario and in the storm scenario. Using a simulation tool, encode the normal and the storm scenarios and run a simulation in order to compare these two scenarios.

7.4.3 Simulation Tools

Nowadays, most business process modeling tools provide simulation capabilities. Examples of tools with simulation support include: ADONIS, ARIS Business Designer, IBM Websphere Business Modeler, OpenText ProVision, Oracle Business Process Analysis (BPA) Suite, Savvion Process Modeler, Signavio Process Editor and TIBCO Business Studio. The landscape of tools evolves continuously, and thus it is very useful to understand the fundamental concepts of process simulation before trying to grasp the specific features of a given tool.

In general, the provided functionality varies visibly from one tool to another. For example, some tools allow one to capture the fact that resources do not work

Fig. 7.13 Cetera's claim-to-resolution process



continuously, but only during specific periods of time. This is specified by attaching a calendar to each resource pool. Some tools additionally allow one to specify that new process instances are created only during certain periods of time, for example only during business hours. Again, this is specified by attaching a calendar to the process model.

Some of the more sophisticated tools allow one to specify not only branching conditions, but also actual boolean expressions that make use of attributes attached to data objects in the process model. In this way, we can specify for example that a branch coming out of an XOR-split should be taken when the attribute "loanAmount" of a data object called "loan application" is greater than 10,000, whereas another branch should be taken when this amount is lower than 10,000. In this case, the probabilistic distribution of values for the attribute "loanAmount" needs to be specified. When the simulator generates objects of type loan, it will give them a value according to the probability distribution attached to that attribute.

There are also small nuances between tools. For example some tools require one to specify the mean arrival rate, that is the number of cases that start during one time unit (e.g. 50 cases per day), while other tools require one to specify the mean inter-arrival time between cases (e.g. one case every 2 minutes). Recall that the distinction between mean arrival rate (written λ in queueing theory) and mean inter-arrival time ($1/\lambda$) was discussed in Sect. 7.3.1. Other tools go further by allowing one to specify not only the inter-arrival time, but how many cases are created every time. By default, cases arrive one by one, but in some business processes, cases may arrive in batches as illustrated by the following scenario extracted from a description of an archival process at the Macau Historical Archives:

> At the beginning of each year, transfer lists are sent to the Historical Archives by various organizations. Each transfer list contains approximately 225 historical records. On average two transfer lists are received each year. Each record in a transfer list needs to go through a process that includes appraisal, classification, annotation, backup, and re-binding among other tasks.
>
> If we consider that each record is a case of this archival process, then we can say that cases arrive in batches of $225 \times 2 = 450$ cases. Moreover, these batches arrive at a fixed inter-arrival time of one year.

Finally, process simulation tools typically differ in terms of how resource pools and resource costs are specified. Some tools would only allow one to define a resource pool and define the number of resources in the pool. A single cost per time unit is then attached to the entire resource pool. Other tools would allow one to create the resources of a pool one by one and to assign a cost to each created resource (e.g. create 10 clerks one by one, each with its name and hourly cost).

The above discussion illustrates some of the nuances found across simulation tools. In order to avoid diving straight away into the numerous details of a tool, it may be useful for beginners to take their first steps using the BIMP simulator referred to in Example 7.7. BIMP is a rather simple BPMN process model simulator that provides the core functionality found in commercial business process simulation tools.

7.4.4 A Word of Caution

One should keep in mind that the quantitative analysis techniques we have seen in this chapter, and simulation in particular, are based on models and on simplifying assumptions. The reliability of the output produced by these techniques largely depends on the accuracy of the numbers that are given as input. Additionally, simulation assumes that process participants work continuously on the process being simulated. In practice though, process participants are not robots. They get distracted due to interruptions, they display varying performance depending on various factors, and they may adapt differently to new ways of working.

In this respect, it is good practice whenever possible to derive the input parameters of a simulation from actual observations, meaning from historical process execution data. This is possible when simulating an as-is process that is being executed in the company, but not necessarily when simulating a to-be process. In a similar spirit, it is recommended to cross-check simulation outputs against expert advice. This can be achieved by presenting the simulation results to process stakeholders (including process participants). The process stakeholders are usually able to provide feedback on the credibility of the resource utilization levels calculated via simulation and the actual manifestation of the bottlenecks shown in the simulation. For instance, if the simulation points to a bottleneck in a given task, while the stakeholders and participants perceive this task to be uncritical, there is a clear indication that incorrect assumptions have been made. Feedback from stakeholders and participants helps to reconfigure the parameters such that the results come closer to matching the actual behavior. In other words, process simulation is an iterative analysis technique with potentially multiple validation loops.

Finally, it is advisable to perform sensitivity analysis of the simulation. Concretely, this means observing how the output of the simulation changes when adding one resource to or removing one resource from a resource pool, or when changing the processing times by $\pm 10\ \%$ for example. If such small changes in the simulation input parameters significantly affect the conclusions drawn from the simulation outputs, one can put a question mark on these conclusions.

7.5 Recap

In this chapter we saw three quantitative process analysis techniques, namely flow analysis, queueing theory and simulation. These techniques allow us to derive process performance measures, such as cycle time or cost, and to understand how different activities and resource pools contribute to the overall performance of a process.

Flow analysis allows us to calculate performance measures from a process model and performance data pertaining to each activity in the model. However, flow analysis does not take into account the level of busyness of the resources involved in the process, i.e. their level of resource utilization. Yet, waiting times are highly dependent on resource utilization—the busier the resources are, the longer the waiting times.

Basic queueing theory models, such as the M/M/1 model, allow us to calculate waiting times for individual activities given data about the number of resources and their processing times. Other queueing theory models such as queueing networks allow us to perform fine-grained analysis at the level of entire processes. However, in practice it is convenient to use process simulation for fine-grained analysis. Process simulation allows us to derive process performance measures (e.g. cycle time or cost) given data about the activities (e.g. processing times) and data about the resources involved in the process. Process simulation is a versatile technique supported by a range of process modeling and analysis tools.

7.6 Solutions to Exercises

Solution 7.1

1. There are at least two business processes that need improvement: the quote-to-booking process—which starts from the moment a quote is received to the moment that a booking is made—and the process for modifying bookings.
2. The quote-to-book process needs to be improved with respect to cycle time, and with respect to error rate. The booking modification process needs improvement with respect to error rate.

Solution 7.2 First we observe that the cycle time of the AND-block is 1. Next, we calculate the cycle time of the XOR-block as follows: $0.4 \times 1 + 0.4 \times 1 + 0.2 \times 1$ hour. The total cycle time is thus: $1 + 1 + 1 = 3$ hours.

Solution 7.3 The cycle time of the process is $2 + 8 + \frac{4+4}{1-0.2} = 20$ days. Assuming 8 working hours per day, this translates to 160 working hours. The theoretical cycle time is $0.5 + 12 + \frac{4+2}{1-0.2} = 20$ hours. Hence, cycle time efficiency is 12.5 %.

Solution 7.4 Little's law tells us that: $CT = WIP/\lambda$. At peak time, there are 900 customers distributed across 6 hours, so the mean arrival rate $\lambda = 150$ customers per hour. On the other hand, $WIP = 90$ during peak time. Thus, $CT = 90/150 = 0.6$ hours (i.e. 36 minutes). During non-peak time, $\lambda = 300/6 = 50$ customer per hour while $WIP = 30$, thus $CT = 30/50 = 0.6$ hours (again 36 minutes). If the number of customers per hour during peak times is expected to go up but the WIP has to remain constant, we need to reduce the cycle time per customer. This may be achieved by shortening the serving time, the interval between the moment a customer enters the restaurant and the moment they place an order, or the time it takes for the customer to pay. In other words, the process for order taking and payment may need to be redesigned.

Solution 7.5 Given that there are no other costs, we calculate the cost of the process by aggregating the resource costs as follows: $0.5 \times 25 + 12 \times 50 + (4 \times 75 + 2 \times 100)/(1 - 0.2) = 1237.50$.

Solution 7.6 On average, 0.05 orders are received per day ($\lambda = 0.05$), and 0.0625 orders are fulfilled per day ($\mu = 0.0625$). Thus, the occupation rate of this system $\rho = 0.05/0.0625 = 0.8$. Using the formulas for M/M/1 models, we can deduce that the average length of the queue L_q is: $0.8^2/(1 - 0.8) = 3.2$ orders. From there we can conclude that the average time an order spends on the queue is $W_q = 3.2/0.05 =$ 64 days. Thus, it takes on average order $W = 64 + 16 = 80$ working days for an order to be fulfilled.

Solution 7.7 Strictly speaking, we should analyze this problem using an M/M/c queueing model. However, the formulas for M/M/c are quite complex to show the calculations in detail. Accordingly, we will assume in this solution that the entire call center behaves as a single monolithic team, so that we can use an M/M/1 queueing model to analyze the problem. Because of this assumption, the results will not be exact.

If we only had seven call center agents, then the occupation rate $\rho = 40/70 = 0.57$, $L_q = \rho^2/(1 - \rho) = 0.57^2/(1 - 0.57) = 0.76$, and $W_q = L_q/\lambda = 0.76/40 = 0.0189$ hours $= 1.13$ minutes. So we cannot meet the customer service charter.

If we can handle 80 calls per hour (eight call center agents), then the occupation rate $\rho = 40/80 = 0.5$, $L_q = \rho^2/(1 - \rho) = 0.5^2/(1 - 0.5) = 0.5$, and $W_q = L_q/\lambda = 0.5/40 = 0.0125$ hours $= 45$ seconds, so we meet the customer service charter.

Ways to reduce costs while staying as close as possible to the customer service charter:

- We could reduce the number of call center agents to 7 and still have an average waiting time of 1.13 minutes. That reduces costs by 12.5 % (one call center agent less).
- We could introduce a self-service system, whereby people lodge their application online (at least for simple claims).
- We could extend the call center working times (e.g. work until 6pm or 7pm instead of 5pm) so that people can call after work, therefore easing the call center load during its peak time.
- Reduce the time of each call by providing better training to call center agents.

Solution 7.8 For this problem, we will reason exclusively in terms of working hours as a unit of time, as opposed to calendar hours. We assume that a week consists of 40 working hours. Calls arrive only during these 40 working hours and call center operators and claims handlers work only during these 40 hours. By taking working hours as a time unit, we avoid the need to attach calendars resources.

In the normal scenario (no storm), the arrival rate is 9,000 cases per week, that is one case every 16 seconds (this is the inter-arrival time). In the storm scenario the inter-arrival time is 8 seconds. In both cases we use an exponential distribution for the inter-arrival time. We run simulations corresponding to 5 weeks of work, meaning 45,000 cases for the normal scenario and 90,000 cases for the storm scenario.

In order to distinguish between the two call centers, we define two separate resource pools, namely "Call Center Operator 1" and "Call Center Operator 2" each one with 40 resources at an hourly cost of 30, plus a resource pool "Claims Handler" with 150 resources. We assign tasks to resource pools as indicated in the scenario and we use the cycle times indicated in the process model as input for the simulation. Running the simulation using the BIMP simulator gives us the following outputs. Under the normal scenario, we obtain a resource utilization of around 53 % for claims handlers and 41 % for call center operators. The average cycle time is around 0.5 working hours and the maximum observed cycle time is around 4.4 working hours. In other words, the resources are under-utilized and thus the cycle time is low.

In the storm season, resource utilization of claims handlers is close to 100 % and that of claims handlers (in both claims handling centers) is around 79 %. The average cycle time is 12 working hours while the maximum cycle time is around 33 hours, that is approximately 4 working days. The high resource utilization indicates that, in practice, the claims handling office is over-flooded during storm season and additional staff are required. On the other hand, the call center has sufficient capacity for storm season. The average waiting time for the tasks in the call center is around 20 seconds.

7.7 Further Exercises

Exercise 7.9 Calculate the cycle time, cycle time efficiency and cost of the university admission process described in Exercise 1.1, assuming that:

- The process starts when an online application is submitted.
- It takes on average 2 weeks (after the online application is submitted) for the documents to arrive to the students service by post.
- The check for completeness of documents takes about 10 minutes. In 20 % of cases, the completeness check that some documents are missing. In this cases an e-mail is sent to the student automatically by the University admission management system based on the input provided by the international students officer during the completeness check.
- A student services officer spends on average 10 minutes to put the degrees and transcripts in an envelope and send them to the academic recognition agency. The time it takes to send the degrees/transcripts to the academic recognition agency and to receive back a response is 2 weeks on average.
- About 10 % of applications are rejected after the academic recognition assessment.
- The university pays a fee of € 5 each time it requests the academic recognition agency to accept an application.
- Checking the English language test results takes 1 day on average, but in reality the officer who performs the check only spends 10 minutes on average per check. This language test check free.
- About 10 % of applications are rejected after the English language test.

- It takes on average 2 weeks between the time students service sends the copy of an application to the committee members and the moment the committee makes a decision (accept or reject). On average, the committee spends 1 hour examining each application.
- It takes on average 2 days (after the decision is made by the academic committee) for the students service to record the academic committee's decision in the University admission management system. Recording a decision takes on average 2 minutes. Once a decision is recorded, a notification is automatically sent to the student.
- The hourly cost of the officers at the international students office is € 50.
- The hourly cost of the academic committee (as a whole) is € 200.

Exercise 7.10 Let us consider the following process performed by an IT helpdesk that handles requests from clients. The clients are employees of a company. There are about 500 employees in total. A request may be an IT-related problem that a client has, or an access request (e.g. requesting rights to access a system). Requests need to be handled according to their type and their priority. There are three priority levels: "critical", "urgent" or "normal". The current process works as follows.

> A client calls the help desk or sends an e-mail in order to make a request. The help desk is staffed with five "Level-1" support staff who, typically, are junior people with less than 12 months experience, but are capable of resolving known problems and simple requests. The hourly cost of a Level-1 staff member is € 40.
> When the Level-1 employee does not know the resolution to a request, the request is forwarded to a more experienced "Level-2" support staff. There are three Level-2 staff members and their hourly cost is € 60. When a Level-2 employee receives a new request, they evaluate it in order to assign a priority level. The job tracking system will later assign the request to the same or to another Level-2 staff depending on the assigned priority level and the backlog of requests.
> Once the request is assigned to a Level-2 staff member, the request is researched by the Level-2 employee and a resolution is developed and sent back to the Level-1 employee. Eventually, the Level-1 employee forwards the resolution to the client who tests the resolution. The client notifies the outcome of the test to the Level-1 employee via e-mail. If the client states that the request is fixed, it is marked as complete and the process ends. If the request is not fixed, it is resent to Level-2 support for further action and goes through the process again.
> Requests are registered in a job tracking system. The job tracking system allows help desk employees to record the details of the request, the priority level and the name of the client who generated the request. When a request is registered, it is marked as "open". When it is moved to level 2, it is marked as "forwarded to level 2" and when the resolution is sent back to "Level 1" the request is marked as "returned to level 1". Finally, when a request is resolved, it is marked as "closed". Every request has a unique identifier. When a request is registered, the job tracking system sends an e-mail to the client. The e-mail includes a "request reference number" that the client needs to quote when asking questions about the request.

Calculate the cycle time efficiency and the cost-per-execution of the as-is process assuming that:

- Submitting and registering a new request takes 5 minutes on average.

- Requests spend on average 1 hour waiting for a Level-1 staff to check them. This applies both to new requests and to re-submitted requests.
- Checking if a new request is "known" takes on average 10 minutes. In 20 % of cases the request is known. In this case, it takes between 2 and 10 minutes (average 5 minutes) for the Level-1 staff to communicate the resolution to the client. Once this is done, the request is marked as "closed". On the other hand, if the request is not "known", the request is automatically forwarded to Level 2.
- New requests spend on average 2 hours waiting for a Level-2 staff to evaluate them. Level-2 staff take on average 20 minutes to evaluate a new request.
- Level-2 staff take 5 minutes to prioritize a request.
- The time between the moment a request has been prioritized, and the moment the request is picked up by a Level-2 staff member is 20 hours.
- The time required to research and resolve a request is on average 2 hours.
- The time to write the resolution to a request is on average 20 minutes.
- Once a Level-2 staff has written the resolution of a request, it takes on average 20 hours before a the request is fetched from the job tracking system by a Level-1 staff.
- It takes on average 20 minutes for a Level-1 staff to send to the client a problem resolution previously written by a Level-2 staff.
- It takes on average 20 hours between the moment a resolution is sent by the Level-1 staff, and the moment the resolution is tested by the client.
- It takes the client around 10 minutes to e-mail the test results to the Level-1 staff.
- In 20 % of cases the request is not resolved, and it needs to be forwarded to Level-2 again. In this latter case, it takes about 2 minutes for the Level-1 to forward the request to the Level-2 staff. Unresolved requests that are forwarded in this way are automatically marked as prioritized, since they have already been prioritized in the previous iteration.
- There are no other costs besides the resource costs.

Hint To calculate theoretical cycle time and cost, only take into consideration time spent doing actual work, excluding waiting times and handovers.

Acknowledgement This exercise is inspired by an example developed by Sue Conger [8].

Exercise 7.11 Consider the scenario described in Exercise 7.6. The company in question is being pressed by several of its customers to fulfill their orders faster. The company's management estimates that the company stands to lose € 250,000 in revenue if they do not reduce their order fulfillment time below 40 working days. Adding one engineer to the existing team would reduce the time to design a hardware down to 14 working days (from 16 days). An additional engineer would cost the company € 50,000. On the other hand, hiring a second engineering team would cost € 250,000. Analyze these two scenarios and recommend an option to the company.

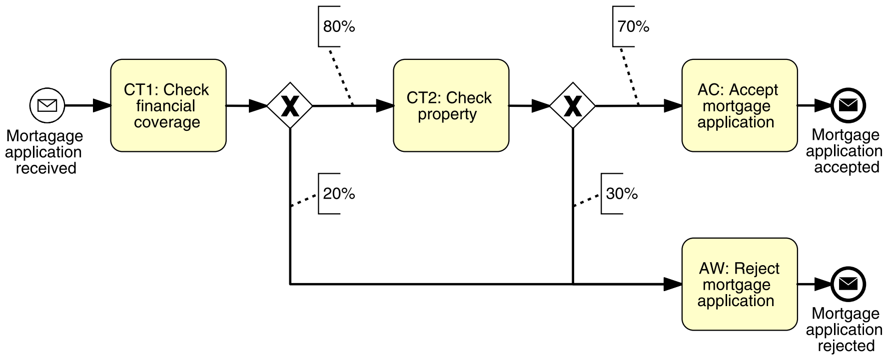


Fig. 7.14 Mortgage process

Exercise 7.12 We consider a Level-2 IT service desk with two staff members. Each staff member can handle one service request in 4 working hours on average. Service times are exponentially distributed. Requests arrive at a mean rate of one request every 3 hours according to a Poisson process. What is the average time between the moment a service request arrives at this desk and the moment it is fulfilled?

Exercise 7.13 Consider again the IT helpdesk process described in Exercise 7.10. Model and simulate it assuming that cases arrive at a rate of 50 per day according to an exponential distribution. Assume that all the activity cycle times follow an exponential distribution with the mean (average) given in Exercise 7.10.

Note When modeling the process, do not model the waiting times between activities, only the activities themselves.

Exercise 7.14 Consider the process model in Fig. 7.14. This model captures a simplified process for handling applications for mortgages. There are two checks involved. CT1 deals with a check of the financial coverage of the mortgage application. The second check, CT2, concerns the verification of the property that is to be mortgaged. If the result of both checks is positive, the application is accepted (task AC). On average, after the execution of task CT1, 20 % of all applications are rejected. Meanwhile, task CT2 leads to 30 % of further rejections. If either of the checks has an unsatisfactory result, the application is rejected (task AW). The arrival process is Poisson, with an average arrival of five cases per hour during business hours. For each task, exactly one dedicated resource is available. The processing time of every task follows an exponential distribution. The mean processing times for tasks CT1, CT2, AC, and AW are, respectively, 5, 4, 3, and 3 minutes. The wage of each resource is € 20 per hour. Business hours are from Monday to Friday from 9am to 5pm. Resources are only available during these hours.

a. Determine the resource utilization of each resource.
b. Determine the average cycle time of the process.

c. Determine the cycle time efficiency of the process.
d. Determine the average number of mortgage applications that are being handled at any given point in time.

Hint For this exercise, it might be convenient to use a combination of process simulation, Little's law and flow analysis.

7.8 Further Reading

The Balanced Scorecard concept alluded to in Sect. 7.1.2 was proposed by Kaplan and Norton in 1992 [39] and quickly gained popularity thereafter as a tool to define organizational strategy and performance measures. Harmon [31] argues that the traditional approach to apply the Balanced Scorecard leads to a bias towards functional units (i.e. performance measures are defined for company departments). To address this bias, he elaborates an approach to apply the Balanced Scorecard along the process architecture rather than the functional architecture. Fürstenau [21] gives a more detailed overview of approaches to process performance measurement all the way from the identification of performance measures using the Balanced Scorecard, to their implementation in the context of IT-enabled processes.

In Sect. 7.2, we showed how flow analysis techniques can be used to calculate cycle time and cost. Laguna and Marklund [43] additionally show how to use flow analysis to calculate *process capacity*. Process capacity is the maximum number of cases per time unit (e.g. cases per hour) that can be theoretically handled. Another possible application of flow analysis is to estimate the error rate of the process, meaning the number of cases that will end up in a negative outcome. This latter application of flow analysis is discussed for example by Yang et al. [109]. Yang et al. also present a technique for flow analysis that is applicable not only to block-structured process models but to a much broader class of process models.

As mentioned in Sect. 7.3, the formula for determining the average queue length in the context of the M/M/c model is particularly complicated. Laguna and Marklund [43, Chap. 6] analyze the M/M/c model (including the formula for average queue length) and its application to process analysis. They also analyze the M/M/c/K model, where an upper-bound to the length of the queue is imposed (this is parameter K in the model). The M/M/c/K model is suitable for example when there is a maximum length of queue beyond which customers are rejected from the queue. Adan and Resing [1] give detailed introductions to M/M/1, M/M/c, M/M/c/K and other queueing theory models.

As stated in Sect. 7.4, business process simulation is a versatile approach for quantitative process analysis. Numerous case studies illustrating the use of process simulation in various domains can be found in the literature. For example, Greasley [24] illustrates the use of business process simulation for redesigning a process for road traffic accident reporting. In a similar vein, Van der Aalst et al. [98] discuss the use of business process simulation to evaluate different strategies to avoid or to mitigate deadline violations in the context of an insurance claims handling process in an insurance company. Exercise 7.8 is based on this latter paper.

Current tools for business process simulation have various limitations. Several of these limitations are discussed at length by van der Aalst et al. [99]. One such limitation has to do with batching of cases. For example, consider the University admissions process described in Exercise 1.1 (p. 4). In this process, each individual application progresses independently of other applications, up the point where the admissions committee has to assess the application. The committee does not meet to handle each individual application, but rather meets at certain times and examines a batch of applications. This batching is necessary because it is not practical for the committee to meet too often since the committee involves several members with busy diaries. Also, the committee needs to compare applications with respect to one another, so it only makes sense to meet when either all applications have arrived, or a sufficient number of applications have arrived to make comparisons between candidates. Let us imagine specifically that the committee meets every 2 weeks but only if there are at least 10 applications to be examined. If there are less than 10 applications, the meeting is skipped and the assessment of pending applications is postponed until the next scheduled meeting. Many business process simulation tools simply cannot capture this batching behavior.

To address this and other limitations, Van der Aalst et al. [99] propose to use more sophisticated tools for process simulation, namely Discrete-Event Simulation (DES) tools. They specifically put forward *CPN Tools* as a possible DES that can be used for business process simulation. CPN Tools is based on *Colored Petri Nets*—a language that extends Petri nets. Other DES tools that can be used for business process simulation include ExtendSim [43] and Arena [40]. For example, Arena is used in the aforementioned case study of a road traffic reporting process [24]. DES tools are clearly more powerful than specialized business process simulation tools. However, the choice of a DES tool means that one cannot directly use a BPMN model for simulation. Instead the model has to be re-encoded in another notation. Moreover, the use of DES tools requires more technical background from the analyst. These trade-offs should be considered when choosing between DES tools and specialized business process simulation tools based for example on BPMN.

We saw throughout the chapter that quantitative analysis techniques allow us to identify critical paths and bottlenecks. These are essentially paths and activities in the process that require special attention if the goal is to reduce cycle time. Anupindi et al. [4] offers detailed advice on how to deal with critical paths and bottlenecks in business processes as well as how to reduce waste and repetition. The following chapter will discuss some of these insights.

Chapter 8
Process Redesign

We know what we are, but not what we may be.
William Shakespeare (1564–1616)

The thorough analysis of a business process typically sparks various ideas and directions for redesign. The problem is, however, that redesign is often not approached in a systematic way, but rather considered as a purely creative activity. The critical point with creative techniques is that parts of the spectrum of potential redesign options could be missed. As an alternative, suitable methods can be utilized to yield more and, hopefully, better redesign options.

This chapter deals with rethinking and re-organizing business processes with the specific purpose of making them perform better. We clarify the motivation and the trade-offs of redesign. Then, we present two methods for systematically redesigning processes. First, we introduce Heuristic Process Redesign as a method that builds upon an extensive set of redesign options. The method is illustrated by the help of a case of a health care institute. Second, we present Product-based Design. This method derives a process design based on the composition of a product.

8.1 The Essence of Process Redesign

In this section, we describe the motivations and the trade-offs of redesign. We introduce the Devil's Quadrangle and discuss options of how redesign can be approached.

8.1.1 Why Redesign?

As stated, the focus of this chapter is on how to redesign business processes. Before explaining this, it is good to reflect on why again it is beneficial at all to focus on business processes. Recall that a business process creates and delivers a certain product or service that customers are after. If someone would like to improve the quality of such a product or service from the perspective of a customer, arguably

M. Dumas et al., *Fundamentals of Business Process Management*,
DOI 10.1007/978-3-642-33143-5_8,

the best way to do that is to improve the related business process. For example, if at some point clients like to make use of a service they purchase from a company much earlier than it is able to deliver, it makes sense to think of streamlining the business process in question. In that way, a *customer-oriented* organization is in fact a *process-centered* organization. Business process redesign is all about improving the quality of products and services by rethinking and re-organizing business processes.

Question Why would anyone like to *redesign* business processes?

One could argue that if a business process has been designed well in the first place, then the products and services are already produced in a satisfactory way. Indeed, if you can step into a bank or a governmental agency, you can see processes in action in ways very similar to how they were introduced there some 50 years ago. This is not necessarily a good thing, though. There are at least two reasons why it makes sense to consider the redesign of an existing business process, even when it was perfectly designed in the first place. The first of these relates to the *organic nature* of organizations. All business processes tend to evolve organically over time. As a result, they grow more complex and their performance gradually deteriorates. Some common examples of such situations are as follows:

- At some point, a clerk forgets to carry out a particular quality check. The product was delivered to a client who became really upset because of the unnoticed flaws of the product. In response, an extra check is incorporated which involves a second clerk to check out whether the quality check is performed at all. This works quite well, but after some time the initial quality check becomes automated through the introduction of a new production system. The check-on-the-check becomes superfluous, but is still part of the process, in this way consuming unnecessary resources and time.
- The marketing department of an organization introduces a special offer for a particular type of customers. Each time such a customer engages with this organization, their account managers ask for extra information beyond what is normally asked. In this way, the marketing campaign can make a perfectly targeted offer to these customers. Yet, the information is not really necessary for the services the clients contact this organization in the first place. After some time, the marketing campaign has come to and end, but the account managers will still ask for the extra information whenever they interact with the particular kind of customer: an unnecessary and time-consuming step.
- An internal auditing department demands at some point that the monetary value of certain financial activities are always reported to it, whenever such activities are carried out. This cause an extra calculation and an extra reporting step in each of the business processes that are affected. Over time, the management of the auditing department changes its priorities and starts looking into other, non-financial information. The reports, nonetheless, keep coming in.

None of the above examples seem so difficult that they cannot be overcome, of course. The point is that people who are busy with carrying out day-to-day operations are usually neither inclined nor equipped to start rethinking the overall structure of operations within an organization. Specifically, it is very common for people to have a limited insight into why a business process is organized in the way it is: People know how to perform their own activities, perhaps some of the activities up- and downstream from their position in the process, but certainly not much more. Even managers, of whom it can be expected that they take a "helicopter view", are usually more concerned with day-to-day execution than structural improvement. People, it seems, are creatures of habit. A business process perspective helps to overcome the inhibition to improve. So, to fight the troubles that go hand in hand with the organical development of a process, redesign is a good idea.

Another reason why it is worthwhile to redesign a business process, even a process that was perfect when it was designed in the first place, is that the *world evolves* as well. New competitors enter the market place that can deliver the same product or service that you can, but against lower cost or tailored to a customer's specific needs. The preferences of customers may change too: People may have been willing to pay a premium for your high-quality product for a long time, but they may now prefer a similar product of lower quality that is offered against a considerably lower price. Whether it is sensible for an organization to keep on producing a certain product or service is not within the scope of this book; that is much more of a strategic decision. What we care about are the operations of an organization. So, assuming there is a strategic incentive to keep on offering a product or service, business process redesign is the approach to create and deliver it in a more performative way.

Exercise 8.1 Can you identify business processes from your own experience that may perhaps have been competitive at some stage, but which seem at this point to be unnecessarily complex or outdated given what competitors offer?

While the two reasons we discussed are important to consider redesigning an existing process, the principles behind redesign approaches can also be helpful to develop business processes from scratch. For example, in 2002 the Netherlands Authority for the Financial Markets was set up to regulate behavior on the Dutch financial markets. Many of the business processes it had to start executing had to be newly developed. Process redesign principles were applied to find the right balance between effectiveness and efficiency. We will still refer to such occasions as process redesign, even though it is technically a misnomer—we would be more precise when referring to this situation as process *design*. We will return to this issue of developing processes from scratch when we will be discussing the various types of redesign approach.

8.1.2 What Is Redesign?

Let us now take a closer look at what redesign is. If you would follow a broad interpretation, any change to an existing process, minor or major, qualifies. Since business processes are rather encompassing—they concern among other the steps in a process, the workforce that is committed to carrying out the process, the information that is being exchanged, and the information systems employed—this actually seems to cover quite a lot. When we talk about process redesign in the context of this book, we will not refer to casual or minor updates, or to changes of parts peripheral to a process or that have no relation to the business process concept. For example, let us suppose that a bank prints the conditions under which a mortgage is granted on ordinary paper, and is accustomed to sending the paper work to applicants when the conditions are completely settled and approved. If the paper type is changed into an eco-friendly, recycled alternative, then we would not consider this as an act of process redesign. If, on the other hand, the client would be provided at any time with an insight into an electronic file that shows the conditions as they are developed during the execution of the process, we would be much more confident in calling this process redesign, especially if the idea behind it is to improve the customer's experience.

Rather than trying to pinpoint business process redesign to an exact definition, we present a framework that helps to think and reason about the most important manifestations of this approach. In this framework, seven elements are identified:

1. the internal or external *customers* of the business process
2. the *business process operation* view, which relates to how a business process is implemented, specifically the number of activities that are identified in the process and the nature of each, and
3. the *business process behavior* view, which relates to the way a business process is executed, specifically the order in which activities are executed and how these are scheduled and assigned for execution
4. the *organization* and the participants in the business process, captured at two levels: the organization structure (elements: roles, users, groups, departments, etc.), and the organization population (individuals: agents which can have activities assigned for execution and the relationships between them)
5. the *information* that the business process uses or creates
6. the *technology* the business process uses, and
7. the *external environment* the process is situated in

Process redesign, then, is first of all concerned with changing the business process itself, covering both its operational and behaviorial view. Yet, process redesign extends to changes that are on the interplay between on the one hand process and on the other the organization or even the external environment that the process operates in, the information and technology it employs, as well as the products it delivers to its customers. This is a comprehensive way of looking at process redesign but it does exclude some activities. For example, the way to train people to optimally perform new activities they become responsible for is out of scope.

Exercise 8.2 Consider the following list and indicate which of these you would consider as process redesign initiatives. Motivate your answer and, if applicable, provide the links to the elements discussed.

1. An airline has seen its profits falling over the past year. It decides start a marketing campaign among its corporate clients in the hope that it can extend its profitable freight business.
2. A governmental agency notices that it is structurally late to respond to citizen's queries. It decides to assign a manager to oversee this particular process and mandates her to take appropriate counter actions.
3. A video rental company sees that its customer base is evaporating. It decides to switch to the business of promoting and selling electronic services through which clients can see movies on-line and on-demand.
4. A bank notices internal conflicts between two different departments over the way mortgage applications are dealt with. It decides to analyze the role of the various departments in the way applications are received and handled to come up with a new role structure.
5. A clinic wants to introduce the one-stop-shop concept to improve over the situation that its patients need to make separate appointments for the various diagnostic tests that are part of a procedure for skin cancer screening.

Not each business domain is equally suitable for the application of business process redesign. To appreciate this, consider the differences between the manufacturing and services domain. In the *manufacturing domain*, the emphasis is on transforming raw materials into tangible products, which often relies on the use of robots and sophisticated machinery. It is in the *services domain* where mostly knowledge is involved in the processing of information to deliver a particular service. Compare, for example, a car manufacturing company with an insurance company as two characteristic examples of the respective domains. In general, it is fair to say that for service organizations the following properties hold:

- Making a copy is easy and cheap. In contrast to making a copy of a product like a car, it is relatively easy to copy a piece of information, especially if the information is in electronic form.
- There are no real limitations with respect to the in-process inventory. Informational products do not require much space and are easy to access, especially if they are stored in a database.
- There are less requirements with respect to the order in which activities are executed: Human resources are flexible in comparison with machines; there are few technical constraints with respect to the lay-out of the service process.
- Quality is difficult to measure. Criteria to assess the quality of a service, an informational product, are usually less explicit than those in a manufacturing environment.
- Quality of end products may vary. A manufacturer of goods usually has a minimal number of components that any product should incorporate. However, in the services domain it might be attractive to skip certain checks in producing the informational product to reduce the workload.

- Transportation of electronic data is timeless. In a computer network, information travels almost at the speed of light; in a manufacturing environment, the transportation of parts is an essential share of the total lead-time, for example think of parts and sub-assemblies that have to be moved from one plant to the other.

From these differences, it is clear that there are more degrees of freedom for redesigning business process in the services domain, than is the case in the manufacturing domain. To optimize a manufacturing process, one has to look for opportunities while juggling many physical constraints. For example, parts that have to be assembled must be transported to the same physical location; by contrast, pieces of information can be put together while they are physically stored on different locations. Similarly, where logistics has evolved as a field to deal with the inventory of parts and half-products, the storage of (digital) information is usually a *no-brainer*. Business process redesign, therefore, is at this point mostly applicable in the services domain. Since there is a trend that manufacturing and high-tech organizations are increasingly making money with providing services along with their physical products, it can be expected that process redesign will become of greater importance here as well.

Exercise 8.3 Consider the following processes and decide whether they are suitable for being redesigned. Use the properties that distinguish the manufacturing and services domain as a mental checklist to support your choice.

1. Dealing with a customer complaint.
2. Carrying out cardiovascular surgery.
3. The production of a wafer stepping machine.
4. Transporting a package.
5. Providing financial advice on composing a portfolio.
6. Designing a train station.

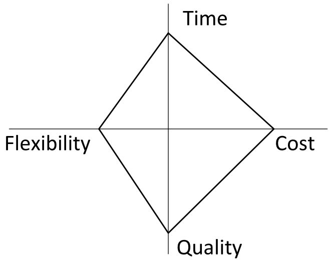
8.1.3 The Devil's Quadrangle

So far, we have not been overly specific about the goals behind redesign. Clearly, it is about making a business process perform better, but we have not discussed the available directions for improvement.

Question What do we want to achieve exactly when a process is redesigned?

A framework that helps answering this question is the *Devil's Quadrangle*, which is depicted in Fig. 8.1. This framework is based on the four performance dimensions discussed in Chap. 7, namely time, cost, quality and flexibility. Ideally, a business process redesign *decreases* the time required to handle a case, it lowers the required cost of executing the process, it *improves* the quality of the service delivered, and

Fig. 8.1 The Devil's Quadrangle



it *increases* the ability of the business process to deal with variation. The interesting property of this framework is how it expresses that improving a process in one dimension may have a weakening effect on another. For example, one may decide to add a reconciliation activity to a business process to improve the quality of the delivered service. Yet, this may backfire on the timeliness of the service delivery. The ominous name of the framework refers to the difficult trade-offs that sometimes have to be made. Awareness of these trade-offs is utterly important for effective process redesign.

Exercise 8.4 Consider the following redesign acts. Which performance measures are affected by these, either positively or negatively?

1. A new computer application is developed that speeds up the calculation of the maximum loan amount that a given client can be offered.
2. Whenever a quote is needed from a financial provider, a clerk must use a direct messaging system instead of e-mail.
3. By the end of the year, additional, temporary workers are hired and assigned to picking items for fulfilling Christmas orders.

While the performance dimensions of the Devil's Quadrangle are helpful to think of the desired effects of business process redesign in general and for a particular business process in particular, they are also useful to think about common approaches to improve business processes. We will devote more attention to this topic when dealing with different types of process redesign approach; we will then refer to them as *redesign heuristics*.

8.1.4 How to Redesign?

There is a great variety of books on process redesign. These deal, among other topics, with different methodologies, present case studies, advance success factors and management lessons. Since the supply may be a bit overwhelming, the following classification may help to see the forest for the trees.

There are three levels of abstractions for methods with respect to process redesign: methodologies, techniques, and tools.

A *methodology*, the highest level of abstraction, is defined as a collection of problem-solving methods governed by a set of principles and a common philosophy for solving targeted problems. This is primarily the field of consulting firms which developed proprietary methodologies, which stretch out from the early analysis phase of a redesign project until the implementation and after care.

At the next level of abstraction, a *technique* is defined as a set of precisely described procedures for achieving a standard task. Some oft-encountered techniques for process analysis—one of the phases in a redesign project—are e.g., fishbone diagramming, Pareto analysis, and cognitive mapping (see Chap. 6). To support the activity of redesigning, creativity techniques like out-of-box-thinking, affinity diagramming, and the Delphi method (brainstorm) are available. For the modeling and evaluation of business processes, techniques are in use as flowcharting, IDEF, speech act modeling, data modeling, activity-based costing, time motion studies, Petri nets, role-playing, and simulation.

At the lowest, most concrete level a *tool* is defined as a computer software package to support one or more techniques. The majority of what some would call process redesign tools are actually process modeling tools. A large number of tools is also available for the evaluation of business process models, in particular supporting the technique of simulation (see Chap. 7). Fewer tools are available to structurally capture knowledge about the redesign directions or to support existing creativity techniques. Tools are often presented as "intelligent" or "advanced", although hardly any of those actively design business processes.

Our concern in this chapter is foremost with redesign methodologies. Now, if you would take the effort to consider all the existing ones you will find that they are usually very specific about preliminary steps in a process redesign project, e.g. the assembly of the project team, and similarly specific towards the end, e.g. how to evaluate a new business process. They are not specific, however, on *how* to take an existing process and turn it into a better performing one. In other words, the *technical challenge* of process redesign is an underdeveloped area. An apt observation that we encountered on this phenomenon is provided by Alec Sharp and Patrick McDermott:

> How to get from the as-is to the to-be [in a process redesign project] isn't explained, so we conclude that during the break, the famous *ATAMO procedure* is invoked ("And Then, A Miracle occurs").

This part of the chapter provides concrete guidance for the technical challenge of process redesign. The two methodologies that we will describe are rather different. To see what distinguishes them it is important to understand the generic traits of process redesign methodologies. Generally speaking, such methodologies can differ with respect to their intensity and their starting point.

The *intensity* of a methodology refers to the pace that one aims with changing the process. Here, we can distinguish between what are commonly referred to as revolutionary and evolutionary levels. Process redesign was originally positioned as an approach that would aim for a radically different outcome, in other words: a revolution. However, methodologies that aim for a more incremental approach have become much more popular. The latter kind would, obviously, qualify as aiming for an evolutionary intensity.

The other distinguishing point is the *starting point* of the redesign effort. One can (a) start from scratch, (b) from the traits of the existing process that is to be redesigned, or (c) from a good, general design, also known as a *reference model*. We will deal with these one by one.

Option (a): Historically, process redesign would follow a *clean slate* approach: the existing process would be completely abandoned and new ways of producing a particular product or service would be developed. It is for good reason that Michael Hammer, one of the gurus behind process redesign famously quipped: “Obliterate, don’t automate.” There are certain advantages to such an approach: It is much easier to get rid off the inefficiencies that have crept into the process by organic growth (see our earlier discussion). Also, the potential to come up with some truly innovative process alternative is in this way better exploited.

Option (b): Over the course of time, however, it has become a far more popular approach to closely look at the existing process. The reason behind this is that it turned out to be extremely hard to develop a complete process from scratch, in particular to cover all exceptions, to not forget any steps, and to add the required level of detail.

Option (c): The newest development is to start work from a *blueprint* or *reference model*. Such standard solutions are typically developed by consultancy and IT companies as representing the state-of-the-art on how to do purchasing, hire someone, or deal with complaints. The *IT infrastructure library (ITIL)* is a good example of such a solution, as it incorporates practical guidelines on how to do problem and incident management within service organizations. The promise of starting from a blueprint is that it will give an up-to-date and standardized view on how to carry out a business process.

Overseeing the landscape of process redesign methodologies, it is fair to say that using the existing process as a starting point has become at this point the most popular approach, followed by the use of a reference model. Building on either the existing design or a reference design, local updates are then identified, each of which contribute a gradual improvement of performance in comparison with the starting situation. Radical approaches are still being applied. In general, clean sheet, revolutionary approaches tend to be more risky as they break away from existing, known procedures. Yet, they also tend to deliver higher benefits *if* they succeed. After all, inefficiencies can be completely rooted out.

In the remainder of this chapter, we will deal with two different methodologies, which represent two extreme variants of the spectrum. First of all, we will discuss a methodology that is based on so-called redesign heuristics, which starts from an existing process to achieve gradual performance improvement. The second methodology is called Product-Based Design; this approach starts from a blank sheet of paper to come up with a radically improved process design. The two methods will give a fairly good idea of mainstream process redesign and at the same demonstrate how redesign methodologies can fundamentally differ.

8.2 Heuristic Process Redesign

We will now discuss the main stages in the methodology of *Heuristic Process Redesign*. Since there is an overlap between the activities that have been described in other chapters, we will focus on the technical challenge of generating a new process design and provide pointers to other parts of the book here. We will first outline the stages and then turn to its most important ingredient in more detail, i.e. the redesign heuristics that we mentioned earlier.

1. *Initiate*: In the first stage, the redesign project is set up. There are various organizational measures that have to be taken, e.g. setting up the project team, but from a technical perspective the most important goals are: (a) to create an understanding of the existing situation (as-is) and (b) to set the performance goals for the redesign project. For (a), the modeling techniques that have been discussed in Chaps. 3 and 4 are useful, as well as the analysis techniques explained in Chaps. 6 and 7 to gain an understanding of performance issues, bottlenecks, and improvement opportunities. To come up with a clearer picture on (b), the Devil's Quadrangle that has been discussed in this chapter is a great asset.
2. *Design*: Given the outcomes of the initiate stage, the design stage makes use of a fixed list of redesign heuristics to determine potential improvement actions on the existing process. For each of the heuristics that is being considered, it needs to be determined whether it is at all applicable and, if so, what a desirable action is. A redesign heuristics is desirable to apply if it helps to attain the desired performance improvement of the process under consideration. After consideration of each of the redesign heuristics, it makes sense to see which clusters of applicable and desirable heuristics can be created. While for some of the heuristics it may make sense to be applied together, for others this is not the case. For example, if you decide to automate a certain activity, it makes no sense to empower the resource that initially carried out that activity. In this way, a set of scenarios can be generated, each of which describes which redesign heuristics are applied in this scenario and, very importantly, how this is done. For example, if the heuristic to automate an activity is applied it needs to be specified which activities are subjected to it. The scenarios, therefore, should be seen as *alternatives* for the process redesign.
3. *Evaluate*: This is the stage where the different redesign scenarios as developed in the previous stage need to be evaluated. This evaluation can be done in a qualitative way, e.g. employing the techniques from Chap. 6, or in a quantitative way, see Chap. 7. In many practical settings, a combination of the two is used where a panel of experts assesses the attractiveness of the various scenarios and where simulation studies are used to underpin the choice for one particular scenario to develop further, potentially all the way to implementing it. An outcome of the evaluation stage may also be that none of the scenarios seems attractive to pursue or even powerful enough to establish the desirable performance improvement. Depending on the exact outcome, the decision may be to adjust the performance goals, to step back to the design stage, or to drop the redesign project altogether.

The description of the stages are here described as separate ones, but in practice they will be executed in highly iterative and overlapping ways. We will now focus the discussion of the methodology to the *redesign heuristics*. A redesign heuristic can be seen as a rule of thumb for deriving a different process. Many of the heuristics we present suggest a particular action to take, while others merely indicate the dimension along which a business process can be changed. The heuristics we present here are all based on historic redesign projects, where they were applied successfully to generate redesign scenarios.

We explained that improving a process is related to the elements we described in Sect. 8.1.2. Thus, we classify the redesign heuristics in a similar way. We identify redesign heuristics that are oriented towards the seven elements we discussed above: customers, business process operation, business process behavior, organization, information, technology, and external environment. Note that this distinction is not mutually exclusive. Therefore, some redesign heuristics could actually be assigned to more than one of these classes.

8.2.1 Customer Heuristics

Heuristics in this category focus on improving the interaction with customers. They focus on control relocation, contact reduction, and integration.

Control relocation: "Move controls towards the customer". Different checks and reconciliation operations that are part of a business process may be moved towards the customer. Consider the example of Pacific Bell that moved its billing controls towards its customers, in this way eliminating the bulk of its billing errors. It also improved customer satisfaction. A disadvantage of moving a control towards a customer is higher probability of fraud, resulting in less yield.

Contact reduction: "Reduce the number of contacts with customers and third parties". The exchange of information with a customer or third party is always time-consuming. Especially when information exchanges take place by regular mail, substantial wait times may be involved. Also, each contact introduces the possibility of an error injection. Imagine a situation where the multitude of bills, invoices, and receipts creates a heavy reconciliation burden. Reducing the number of contacts may in such a case decrease throughput time. Note that it is not always necessary to skip certain information exchanges, but that it is possible to combine them with limited extra cost. A disadvantage of a smaller number of contacts might be the loss of essential information, which is a quality issue. Combining contacts may result in the delivery or receipt of too many data, which involves cost.

Integration: "Consider the integration with a business process of the customer or a supplier". This heuristic can be seen as exploiting the supply-chain concept known from production. The actual application of this heuristic may take on different forms. For example, when two parties have to agree upon a product they jointly produce, it may be more efficient to perform several intermediate

Table 8.1 Characteristics of the customer heuristics

	Time	Cost	Quality	Flexibility
Control relocation	·	−	+	·
Contact reduction	+	−	+	·
Integration	+	+	·	−

reviews than performing one large review after both parties have completed their parts. In general, integrated business processes should render a more efficient execution, both from a time and cost perspective. The drawback of integration is that mutual dependence grows and therefore, flexibility may decrease.

Using the dimensions of the Devil's Quadrangle introduced earlier, a summary of the general effects of the three customer heuristics is shown in Table 8.1. This table shows that a heuristic can be generally expected to have a positive effect (+), a negative effect (−) or a neutral effect (·) on any of the dimensions of the Devil's Quadrangle. Note that a *positive* effect on the cost dimension, as can be expected from the Integration heuristic, means that cost actually goes *down*.

Exercise 8.5 Explain the + sign for the contact reduction heuristic with respect to the time dimension.

8.2.2 Business Process Operation Heuristics

Business process operation puts the focus on the elements of a business process. There are five heuristics relating to case types: activity elimination, case-based work, triage, and activity composition.

Case types: "Determine whether activities are related to the same type of case and, if necessary, distinguish new business processes". One should be cautious of parts of business processes that are not specific for the business process they are part of. Ignoring this phenomenon may result in a less effective management of such a *subflow* and a lower efficiency. Applying this heuristic may result in faster processing times and less cost. Yet, it may also result in more coordination problems between the business process (quality) and less possibilities for rearranging the business process as a whole (flexibility).

Activity elimination: "Eliminate unnecessary activities from a business process". A common way of regarding an activity as unnecessary is when it adds no value from a customer's point of view. Typically, control activities in a business process do not do this; they are incorporated in the model to fix problems created (or not elevated) in earlier steps. Control activities can often be identified by iterations in a process. The redundancy of an activity can also trigger activity elimination. The aims of this heuristic is to increase the speed of processing and to reduce the cost of handling an order. An important drawback may be that the quality of the service deteriorates.

Table 8.2 Characteristics of the business process operation heuristics

	Time	Cost	Quality	Flexibility
Case types	+	+	−	−
Activity elimination	+	+	−	·
Case-based work	+	−	·	·
Triage	·	−	+	−
Activity composition	+	+	·	−

Case-based work: "Consider removing batch-processing and periodic activities from a business process". Some notable examples of disturbances in handling a single case are (a) that the case becomes piled up in a batch and (b) that the case is slowed down by periodic activities, e.g. because processing depends on a computer system that is only available at specific times. Getting rid of these constraints may significantly speed up the handling of individual cases. On the other hand, efficiencies of scale can be reached by batch processing which are reversed by this heuristic. Also, the cost of making information systems permanently available may be costly.

Triage: "Consider the division of a general activity into two or more alternative activities". Through this heuristic, it is possible to design activities that are better aligned with the capabilities of resources and the characteristics of the cases being processed, which improves quality. On the other hand, specialization in general makes a process less flexible and may decrease the efficiency of work. An alternative form of the triage heuristic is to divide an activity into similar instead of alternative activities for different subcategories of the cases being processed. For example, a special cash desk may be set up for customers with an expected low processing time. Note that the triage heuristic can be seen as a translation of the case types heuristic on an activity level.

Activity composition: "Combine small activities into composite activities and divide large activities into workable smaller activities". Composing larger activities should result in the reduction of setup times, i.e., the time that is spent by a resource to become familiar with the specifics of a case. By executing a large activity which used to consist of several smaller ones, a positive effect may also be expected on the quality of the delivered work. On the other hand, making activities too large may result in (a) smaller run-time flexibility and (b) lower quality as activities become unworkable. Both effects are exactly countered by dividing activities into smaller ones. Obviously, smaller activities may also result in longer set-up times. This heuristic is related to the triage heuristic in the sense that they both are concerned with the division and combination of activities.

The assessment of the heuristics that aim at the business process operation is summarized in Table 8.2. The meaning of the signs is the same as in the previous table. For example, case-based work can be expected to be highly beneficial in the time dimension; triage may boost the quality of a process. Note that only one manifestation of the activity composition heuristic is shown, i.e. the case that large activities are composed of smaller ones.

8.2.3 Business Process Behavior Heuristics

Business process behavior regulates the logic within the business process. There are four heuristics for this category, namely resequencing, parallelism, knock-out, and exception.

Resequencing: "Move activities to more appropriate places". In existing business processes, actual activity orderings often do not reveal the necessary dependencies between activities. Sometimes it is better to postpone an activity if it is not required for its immediate follow-up activities. The benefit would be that perhaps its execution may prove to become superfluous, which saves cost. Also, an activity may be moved into the proximity of a similar activity, in this way diminishing set-up times. This heuristic is also known as process order optimization.

Parallelism: "Consider whether activities may be executed in parallel". The obvious effect of placing activities in parallel is that throughput time may be considerably reduced. The applicability of this heuristic in business process redesign is large. In practical settings, activities are often ordered sequentially without the existence of hard logical restrictions prescribing such an order. A drawback of introducing more parallelism in a business process that incorporates possibilities of knock-outs is that the cost of business process execution may increase. Also, the management of business processes with concurrent behavior can become more complex (flexibility).

Knock-out: "Order knock-outs in an increasing order of effort and in a decreasing order of termination probability". A typical element of a business process is the subsequent checking of various conditions that must be satisfied to deliver a positive end result. Any condition that is not met may lead to a termination of that part of the business process: the *knock-out*. If there is freedom in choosing the order in which the various conditions are checked, the condition that has the most favorable ratio of expected knock-out probability versus the expected effort to check the condition should be pursued. Next, the second best condition, and so forth. This way of ordering checks yields on average the least costly business process execution. Implementing this heuristic may result in a (part of a) business process that takes a longer throughput time than a full parallel checking of all conditions. The knock-out heuristic is a specific form of the resequencing heuristic.

Exception: "Design business processes for typical cases and isolate exceptional cases from the normal flow". Exceptions may seriously disturb normal operations. An exception will require workers to get acquainted with the specifics of the exception even though they may not be able to handle it. Setup times are then wasted. Isolating exceptions may possibly increase the overall performance as specific expertise can be build up by workers working on the exceptions, even though this may come at a cost. A drawback is that the business process will become more complex, possibly decreasing its flexibility.

The assessment of the heuristics that target the behavior of the business process can be seen in Table 8.3. Again, the meaning of the signs is similar to those in

Table 8.3 Characteristics of the business process behavior heuristics

	Time	Cost	Quality	Flexibility
Resequencing	+	+	·	·
Parallelism	+	−	·	−
Knock-out	−	+	·	·
Exception	+	−	+	−

earlier tables. Note how the exception heuristic stands out with respect to improving the quality dimension.

8.2.4 Organization Heuristics

Organization refers to two categories of heuristics. The first set relates to the *structure* of the organization (mostly the allocation of resources). There are seven heuristics in this category, namely case assignment, flexible assignment, centralization, split responsibilities, customer teams, numerical involvement, and case manager.

Case assignment: "Let workers perform as many steps as possible for single cases". By using case assignment in the most extreme form, for each activity execution the participant is selected who has worked on the case before—if any. The obvious advantage of this heuristic is that this person will have become acquainted with the case and will need less set-up time in carrying out subsequent activities. An additional benefit may be that the quality of service is increased: the participant knows exactly what the specifics of the case are. On the negative side, the flexibility of work allocation is seriously reduced. The execution of a case may experience substantial queue time when the person to whom it is assigned is not available, nullifying expected time gains.

Flexible assignment: "Assign work in such a way that maximal flexibility is preserved for the near future". Suppose that an activity can be executed by either of two available participants, then the heuristic suggests to assign it to the most specialized person. In this way, the likelihood to commit the free, more general resource to another work package is maximal. The advantage of this heuristic is that an organization stays flexible with respect to assigning work and that overall queueing time is reduced: it is less probable that the execution of a case has to wait for the availability of a specific resource. Another advantage is that the workers with the highest specialization can be expected to take on most of the work, which may result in a higher quality. The disadvantages of applying this heuristic can be subtle. For example, work load may become unbalanced, resulting in less job satisfaction. Also, possibilities for specialists to evolve into generalists are reduced. These are both quality concerns. In certain situations, specialists may even be more expensive to carry out work.

Centralization: "Treat geographically dispersed resources as if they are centralized". This heuristic is explicitly aimed at exploiting the benefits of a Business

Process Management System or BPMS for short (see Chap. 9). After all, when a BPMS takes care of assigning work to resources it becomes less relevant where these resources are located geographically. In this sense, this heuristic can be seen as a special form of the integral technology heuristic (see later in this chapter). The specific advantage of this measure is that resources can be committed more flexibly, which gives a better utilization and possibly a better throughput time. The introduction of a BPMS and training of the workforce may, of course, be substantial.

Split responsibilities: "Avoid shared responsibilities for tasks by people from different functional units". The idea behind this heuristic is that activities for which different departments share the responsibility are more likely to be a source of neglect and conflict. Reducing the overlap in responsibilities should lead to a better quality of activity execution. Also, a higher responsiveness to available work may be developed, so that customers are served quicker. On the other hand, applying this heuristic may reduce the effective number of resources that is available for a work item. This may have a negative effect on its throughput time, as more queuing may occur, and the organization becomes less flexible.

Customer teams: "Consider to compose work teams of people from different departments that will take care of the complete handling of specific sorts of cases". Depending on its exact desired form, the customer team heuristic may be implemented by the case assignment heuristic. On the other hand, a customer team may involve more workers with the same qualifications, in this way relaxing the strict requirements of the case assignment heuristic. Advantages and disadvantages are similar to those of the case assignment heuristic. In addition, working as a team may improve the attractiveness of the work, which is a quality aspect.

Numerical involvement: "Minimize the number of departments, groups and persons involved in a business process". Applying this heuristic may lead to less coordination problems. Less time spent on coordination makes more time available for the processing of cases. Reducing the number of departments may lead to less split responsibilities, with similar pros (quality) and cons (flexibility) as the split responsibilities heuristic discussed before. Note that smaller numbers of specialized units may prohibit the build up of expertise (a quality issue) and routine (a cost issue).

Case manager: "Appoint one person to be responsible for the handling of each type of case, the case manager". The case manager is responsible for a specific order or customer. Note that a case manager is not necessarily someone who works on the actual case and even if the person does, not exclusively so. The difference with the case assignment practice is that the emphasis is on management of the process—not its execution. The most important aim of the heuristic is to improve upon the external quality of a business process. The business process will become more transparent from the viewpoint of a customer: the case manager provides a single point of contact. This, in general, positively influences customer satisfaction. It may also have a positive effect on the internal quality of the business process, as someone is accountable for and committed to correcting mistakes. Obviously, the assignment of a case manager has financial consequences as capacity must be devoted to this job.

Table 8.4 Characteristics of the organization structure heuristics

	Time	Cost	Quality	Flexibility
Case assignment	·	·	+	−
Flexible assignment	+	−	·	+
Centralization	+	−	·	+
Split responsibilities	·	·	+	−
Customer teams	·	·	+	−
Numerical involvement	+	−	·	−
Case manager	·	−	+	·

The assessment of the heuristics that target the side of the organizational structure involved in a business process can be seen in Table 8.4.

The second set relates to the organizational population and the resources being involved in terms of type and number. This category includes three heuristics: extra resources, specialist-generalist, and empower.

Extra resources "If capacity is insufficient, consider increasing the available number of resources". This is straightforward heuristic, which aims at extending capacity. extending the capacity to handle cases, in this way reducing queue time. It may also help to implement a more flexible assignment policy. Of course, hiring or buying extra resources has its cost. Note the contrast of this heuristic with the numerical involvement heuristic.

Specialist-generalist "Consider to deepen or broaden the skills of resources". Participants in a process may be turned from specialists into generalists or the other way around. A specialized resource can be trained to gain more qualifications. A generalist, on the other hand, may be assigned to the same type of work for a longer period of time, so that skills in this area deepen while other qualifications become obsolete. In the context of designing an entirely new business process, the application of this heuristic comes down to considering the specialist-generalist ratio of new hires. Clearly, specialists build up routine more quickly and may have more profound knowledge in an area than generalists have. As a result, they work more quickly and deliver higher quality. On the other hand, the availability of generalists adds more flexibility to the business process and can lead to a better utilization of resources. Depending on the degree of specialization or generalization, either type of resource may be more costly. Note that this heuristic differs from the triage concept in the sense that the focus is not on the division of activities.

Empower "Give workers most of the decision-making authority instead of relying on middle management". In traditional business processes, substantial time may be spent on authorizing the outcomes of activities that have been performed by others. If workers are empowered to take decisions autonomously, this may result in smoother operations with lower throughput times. The reduction of middle management from the business process also reduces the labor cost spent on the

Table 8.5 Characteristics of the organization population heuristics

	Time	Cost	Quality	Flexibility
Extra resources	+	−	·	+
Specialist-generalist	+	·	+	−
Empower	+	·	−	+

processing of cases. A drawback may be that the quality of the decisions is lower and that obvious errors are no longer identified. If bad decisions or errors result in rework, the cost of handling a case may actually increase compared to the original situation.

The assessment of the heuristics for the organization population can be seen in Table 8.5. Note that for the specialist-generalist heuristic, the general effects are included of investing into *more specialized* skills of the workforce. Clearly, investing in generalists gives the opposite effect.

8.2.5 Information Heuristics

The information category describes redesign heuristics related to the information the business process uses, creates, may use or may create. It includes control addition and buffering.

Control addition: "Check the completeness and correctness of incoming materials and check the output before it is sent to customers". This heuristic promotes the addition of controls to a business process. Such additions may lead to a higher quality of the business process execution. Obviously, an additional control will require time, which may be substantial, and absorbs resources. Note the contrast between the intent of this heuristic and that of the activity elimination heuristic discussed earlier.

Buffering: "Instead of requesting information from an external source, buffer it and subscribe to updates". Obtaining information from other parties is a time-consuming part in many business processes. By having information directly available when required, throughput times may be substantially reduced. This heuristic can be compared to the caching principle that microprocessors apply. Of course, the subscription fee for information updates may be costly. This is certainly so if we consider information sources that contain far more information than is ever used. Substantial cost may also be involved with storing all the information. Note that this heuristic is a weak form of the integration heuristic that is yet to be discussed.

A summary of the general effects of the two information heuristics are shown in Table 8.6.

Table 8.6 Characteristics of the information heuristics

	Time	Cost	Quality	Flexibility
Control addition	−	−	+	·
Buffering	+	−	·	·

Table 8.7 Characteristics of the technology heuristics

	Cost	Quality	Time	Flexibility
Activity automation	+	−	+	−
Integral technology	+	−	·	·

8.2.6 Technology Heuristics

This category describes redesign heuristics related to the technology the business process uses or may use. It includes activity automation and integral technology.

Activity automation: "Consider automating activities". A particularly positive result of automating activities may be that activities can be executed faster and with a more predictable result. An obvious disadvantage is that the development of a system that performs an activity may be very costly. Generally speaking, a system performing an activity is also less flexible in handling variations than a human resource. Instead of fully automating an activity, it may also be considered to provide automated support to a resource executing an activity.

Integral technology: "Try to elevate physical constraints in a business process by applying new technology". In general, new technology can offer all kinds of positive effect. For example, the application of a BPMS may result in less time that is spent on routing (electronic) work. A Document Management System, in its turn, will open up to all participants the information available on cases. This may result in a better quality of service. New technology can also change the traditional way of doing business by giving participants completely new opportunities. The purchase, development, implementation, training, and maintenance efforts related to technology obviously incur costs. In addition, new technology may instill workers with apprehension, which may decrease the quality of the business process.

The two technology heuristics can be characterized as is shown in Table 8.7.

8.2.7 External Environment Heuristics

The external environment category contains heuristics that try to improve upon the collaboration and communication with the third parties. These include trusted party, outsourcing, and interfacing.

Table 8.8 Characteristics of the external environment heuristics

	Cost	Quality	Time	Flexibility
Trusted party	+	+	·	−
Outsourcing	+	+	·	−
Interfacing	+	·	+	−

Trusted party: "Instead of determining information oneself, use the results of a trusted party". Some decisions or assessments that are made within a business process are not specific to that process. Other parties may have determined the same information in another context, which—if it were available—could replace the decision or assessment. An example is the creditworthiness of a customer that bank A wants to establish. If a customer can present a recent creditworthiness certificate of bank B, then bank A may be likely to accept it. Obviously, the trusted party heuristic reduces cost and may even cut back throughput time. On the other hand, the quality of the business process becomes dependent upon the quality of some other party's work. Some coordination effort with trusted parties is also likely to be required, which diminishes flexibility. This heuristic is different from the buffering heuristic, because the business process owner is not the one obtaining the information.

Outsourcing: "Consider outsourcing a business process completely or parts of it". Another party may be more efficient in performing the same work, so it might as well perform it for one's own business process. The obvious aim of outsourcing work is that it will generate less cost. A drawback may be that quality decreases. Outsourcing also requires more coordination efforts and will make managing the business process more complex. Note that this heuristic differs from the trusted party heuristic. In the case of outsourcing, an activity is executed at run time by another party. The trusted party heuristic allows for the use of a result in the (recent) past.

Interfacing: "Consider a standardized interface with customers and partners". The idea behind this heuristic is that a standardized interface diminishes the occurrence of mistakes, incomplete applications, or unintelligible information exchanges. So, a standardized interface may result in less errors (quality) and faster processing (time). However, by standardizing the interface it becomes impossible to deal with exceptional situations (flexibility). The interfacing heuristic can be seen as a specific interpretation of the integration heuristic, although it is not specifically aimed at customers.

The characteristics of the external environment heuristics are summarized in Table 8.8. This concludes the description of the various heuristics. In what follows, we will be looking at their application.

8.3 The Case of a Health Care Institution

At this point, we will consider a realistic case of a healthcare institute (see Fig. 8.2). The case concerns the Intake process for elderly patients with mental problems, which is styled after the way this is carried out in the Eindhoven region. The Intake process starts with a notice by telephone at the secretarial office of the healthcare institute. This notice is done by the family doctor of the person who is in need of mental treatment. The secretarial worker inquires after the name and residence of the patient. On basis of this information, the doctor is put through to the nursing officer responsible for the part of the region that the patient lives in.

The nursing officer makes a full inquiry into the mental, health, and social status of the patient in question. This information is recorded on a registration form. After this conversation had ended, this form is handed in at the secretarial office of the institute. Here, the information on the form is stored in the information system and subsequently printed. For new patients, a patient file is created. The registration form as well as the print from the information system are stored in the patient file. Patient files are kept at the secretarial office and may not leave the building. At the secretarial office, two registration cards are produced for, respectively, the future first and second intaker of the patient. The registration card contains a set of basic patient data. The new patient is added on the list of new notices.

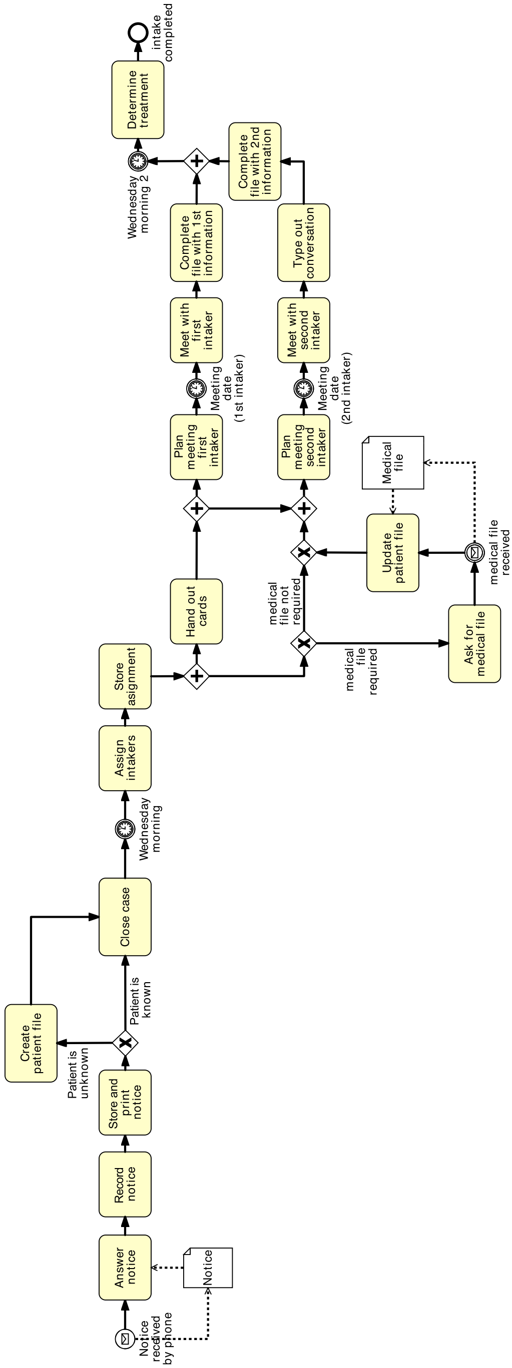
Halfway during each week, on Wednesday, a staff meeting of the entire medical team takes place. The medical team consists of social–medical workers, physicians, and a psychiatrist. During this meeting, the team leader assigns all new patients on the list of new notices to members of the team. Each patient will be assigned to a social–medical worker, who will act as the *first intaker* of the patient. One of the physicians will act as the *second intaker*. In assigning intakers, the team leader takes into account their expertise, the geographical region they are responsible for, earlier contacts they might have had with the patient, and their case load. The assignments are recorded on an assignment list which is handed to the secretarial office. For each new assignment, it is also determined whether the medical file of the patient is required. This information is added to the assignment list.

The secretarial office stores the assignment of each patient of the assignment list in the information system. It passes the produced registration cards to the first and second intaker of each newly assigned patient. An intaker keeps this registration at times when visiting the patient and being at the office. For each patient for which the medical file is required, the secretarial office prepares and sends a letter to the family doctor of the patient, requesting for a copy of the medical file. As soon as this copy is received, the secretarial office will inform the second intaker and add the copy to the patient file.

The first intaker plans a meeting with the patient as soon as this is possible. During the first meeting, the patient is examined using a standard checklist which is filled out. Additional observations are registered in a personal notebook. After a visit, the first intaker puts a copy of these notes in the file of a patient. The standard checklist is also added to the patient's file.

The second intaker plans the first meeting only after the medical information of the physician—if required—has been received. Physicians use dictaphones to record

Fig. 8.2 The intake process



their observations made during meetings with patients. The secretarial office types out these tapes, after which the information is added to the patient file.

As soon as the meetings of the first and second intaker with the patient have taken place, the secretarial office puts the patient on the list of patients that reach this status. For the staff meeting on Wednesday, they provide the team leader with a list of these patients. For each of these patients, the first and second intaker together with the team leader and the attending psychiatrist formulate a treatment plan. This treatment plan formally ends the intake procedure.

What we will now do is discuss three alternatives to the process, which all have been generated using the heuristics discussed so far. In other words, each design is derived from the existing process by the application of one or more redesign heuristics at appropriate places. To guide the derivation of these designs, we will assume that bringing back the cycle time of this process is the main objective.

8.3.1 Sending Medical Files by Post

A considerable part of the cycle time in the Intake process is consumed by the wait time for the medical file to arrive by post. On basis of the *integration* and *technology* heuristics we consider the alternative that medical files become available on-line to the mental health care institute. (In practice, this should presumably be restricted to read-only access for patients that are indeed reported to the mental health-care institute.) Note that this alternative presupposes a considerable usage of technology: doctors should store their patient information electronically and communication facilities should be installed as well.

By the direct availability of the medical file, the "Ask for medical file" activity in Fig. 8.2 is replaced by the "Access medical file" activity, which is performed by the secretarial office. Probably, roughly the same time that was spent on preparing and sending a request letter will be required for accessing and printing the patient file. The "Update client file" activity stays in place, but it is not triggered anymore by the "Medical file". The wait time for the medical file is now completely reduced, which favorably influences the cycle time.

8.3.2 Periodic Meetings

As part of the Intake process, the staff meeting is planned at regular weekly intervals, on Wednesdays. During a staff meeting two important things take place:

1. for new cases, the first and second intakers are assigned, and
2. for cases for which both intake interviews have taken place, treatment plans are determined

From a process perspective, periodic restrictions on activities seem odd. Let us assume that an additional analysis of the Intake process points out that the first activity does not really require a meeting context, provided that the team leader has sufficient information on the criteria used for new assignments. Admittedly, the second activity is indeed best performed in the context of a meeting. This is because of the limited availability of the psychiatrists, which prohibits more flexible measures.

On basis of the *case-based work* heuristic, we consider as an alternative to the existing process that the team leader will carry out new case assignments as soon as they are due; the weekly meeting is strictly used for determining treatment plans. The process in Fig. 8.2 then changes in the sense that the "Wednesday morning" event is removed. Because the information is available to the team leader for taking assignment decisions, it can be expected that the original duration of the activity decreases. This time includes the report of the assignment to the secretarial office. Both the social–medical worker and the physician will no longer spend this time on the case. The cycle time of an average case will drastically drop, on average by 2.5 working days—half a working week—as this is the expected time a new case has to wait before it is assigned (assuming a uniform distribution of cases over the week).

8.3.3 Requesting Medical Files

For each new case, a decision has to be made whether the medical file of the patient will be requested. This request is made to the family doctor. It should be noted that the family doctor is also the one who notifies the new case at the initiation of the process. This raises the question whether the *contact reduction* heuristic may be applicable. Closer inspection of the routing of individual cases shows that in 95 % of all new cases the medical file is requested for. This extremely high figure certainly justifies consideration of the *exception* heuristic. After all, not requiring the medical information seems to be the exception.

A combined application of the *contact reduction* heuristic, the *exception* heuristic and the *resequencing* heuristic leads to an alternative process design where the secretarial office directly asks for the medical file after the family doctor makes contact with the mental health care institute. Also, the routine is dropped to determine for each case at a staff meeting whether medical information is required. The new process design is shown in Fig. 8.3.

Note that in this case, the exception heuristic coincides with the secondary interpretation of the *triage* heuristic. What used to be an alternative activity, asking for medical information, has become a general part of the process. As a result, the cycle time is sharply reduced.

This ends the explanation of Heuristic Process Redesign. While this methodology still involves a fair dose of creativity and skills, the availability of redesign heuristics helps to more easily generate new designs. Each time the heuristics are used, it makes sense to consider which sets of heuristics are applicable by taking the redesign goals into account.

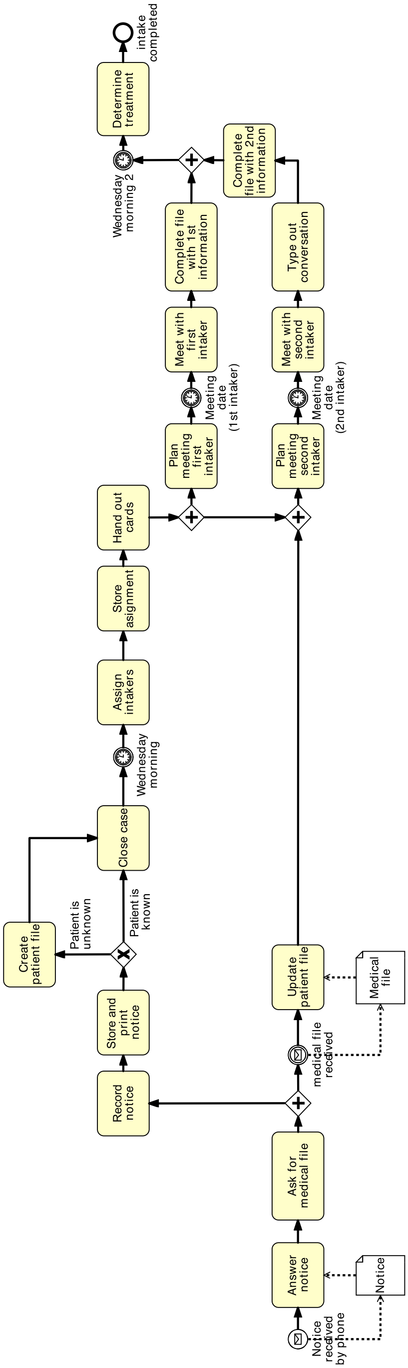


Fig. 8.3 The intake process after the medical file redesign

Exercise 8.6 Consider the three redesign scenarios that were discussed previously. As explained, these scenarios focus on the reduction of cycle time of the process in question. Can you explain how the other performance dimensions are affected by these scenarios?

8.4 Product-Based Design

The methodology of *Product-based Design* is very different from Heuristic Process Redesign. First of all, it aims at radically rethinking how a particular product or service can be created instead of using an incremental approach as we saw before.

Secondly, not so much the existing process is the starting point of the redesign. Rather, the characteristics of the particular product that the process-to-be is expected to deliver are used to, in fact, *reason back* what that process should look like. Think of it in this way: if you like to produce a red, electronic vehicle on four wheels, you are certain that the process to produce it at some stage must involve the production or purchase of a chassis, that there is a step needed to assemble four wheels to that chassis, that you will need to insert a battery at some point, and that you will need to paint the vehicle (if you cannot get your hands on red parts, that is). You are perhaps not sure in what order these things need to take place exactly, but you can at least identify some logical dependencies. For example, you are better off painting the vehicle *after* you acquired the chassis.

The idea behind Product-based Design is that by *ignoring* the existing process to create a particular product it becomes feasible to develop the leanest, most performative process possible. While Product-based Design is more ambitious than Heuristic Process Redesign, it is also more limited in its application scope: It has been specifically developed to design processes that produce informational products, e.g. a decision, a proposal, or a permit. It is this informational product that is analyzed and laid down in a *product data model*. There is a striking resemblance between this model and the *bill-of-material* (BOM) as used in the manufacturing domain. The product data model is subsequently used by the designer to determine the best process structure to create and deliver that product. Given that there are, in general, multiple ways to produce an informational product, Product-based Design discloses all of these.

After this brief introduction, we will now outline the steps of Product-based Design. The most important stages are:

1. Scoping: In this initial phase the business process is selected that will be subject to the redesign. The performance targets for this process are identified, as well as the limitations to be taken into consideration for the final design.
2. Analysis: A study of the product specification leads to its decomposition into information elements and their logical dependencies in the form of a *product data model*. The existing business process—if any—is diagnosed to retrieve data that are both significant for designing the new business process and for the sake of evaluation.

3. Design: Based on the redesign performance objectives, the product data model, and (estimated) performance figures, one or more process designs are derived that best match with the design goals.
4. Evaluation: The process designs are verified, validated with end-users, and their estimated performance is analyzed in more detail. The most promising designs can be presented to the commissioning management to assess the degree in which objectives can be realized and to select the most favorable design to be implemented.

These phases are presented in a sequential order, but in practice it is often desirable that iterations will take place. For example, the evaluation phase is explicitly aimed at identifying design errors, which may result in rework on the design. The focus of the remainder of this section will be on the analysis and design phases. The purpose is not to treat all the details of this method, but to give the reader an idea of the approach, its main artifacts, and how it is different from Heuristic Process Redesign.

8.4.1 Analysis: Creating a Product Data Model

In the analysis phase, all distinguished materials that may be relevant sources on the characteristics of the product to-be-delivered are analyzed. The purpose is to identify information elements, their dependencies, and the processing logic involved, i.e. how existing information can be combined to create new information. For a proper representation of this information, we will be using a tree-like structure that we will refer to as a *product data model*. This structure is different from the traditional BOM found in manufacturing, which is due to several differences between informational products and physical products. These differences lead to two important updates of the traditional BOM. First, the same piece of information may be used to manufacture various kinds of new information. Therefore, also non-tree-like structures are possible. For example, the age of an applicant for a life insurance may be used to estimate both (a) the involved health risks for that patient and (b) the risks of work related accidents. Secondly, there are no physical constraints to produce an informational product and therefore there are typically multiple ways to derive a piece of information. For example, health risks may be estimated using either a patient questionnaire or a full medical examination of that patient.

At this point, we present a graphical example of a product data model, as shown in Fig. 8.4.

All nodes in this figure correspond to information elements that may be used to decide whether some candidate is suitable to become a helicopter pilot in the Dutch Air force. We will refer to this model throughout the remainder of this section as the helicopter pilot product data model . Arcs are used to express the dependencies between the various pieces of information, i.e. the information elements. The meaning of the information elements is as follows:

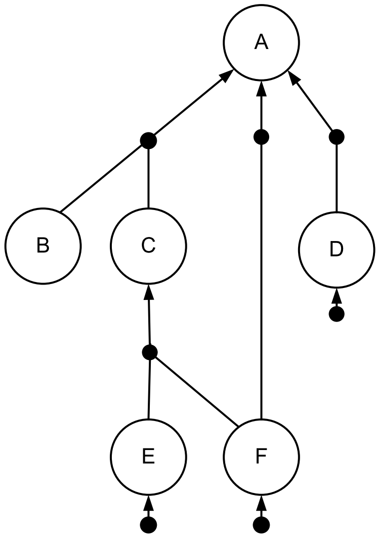


Fig. 8.4 The helicopter pilot product data model

- A: suitability to become a helicopter pilot.
- B: psychological fitness.
- C: physical fitness.
- D: latest result of suitability test in the previous two years.
- E: quality of reflexes.
- F: quality of eye-sight.

Each incoming arc of a node signifies an *alternative way* of determining a value for the corresponding information element for a specific case. If outgoing arcs of multiple nodes are *joined*, this means that values of all of the corresponding information elements are required to determine a value for the information element the arrow leads to. There are also information elements which have incoming arrows that do not origin from other information elements. These relate to those elements that do not rely on the values of other information elements, e.g. element B. We will refer to such information elements as *leaf elements*.

One of the things that is expressed in Fig. 8.4 is that there are three ways to determine a value for information element A. The suitability of a candidate (a) can be determined on the basis of:

1. the combined results of the psychological test (B) and the physical test (C)
2. the result of a previous suitability test (D), or
3. the candidate's eye-sight quality (F)

The way in which a new piece of information is determined on the basis of one or more pieces of other information is called a *production rule* or an *operation*. In reality, different production rules may be applicable under different conditions. It may be the case that a pilot's eye-sight is extremely bad (F), which directly gives as a result that the candidate is not suitable (A). However, in a more common case, the eye-sight quality is one of the many aspects that are incorporated in a physical test (B), which should be combined with the outcome of the psychological test (C) to determine the suitability result (A). Also, not for each candidate that applies to become a pilot any previous test result (D) will be available—quite the contrary. But if there is one of a recent date, it can be used directly.

From the helicopter pilot product data model it becomes clear how the dependencies between data may be used to derive a favorable design. For example, if the target is to minimize the cost it may be wise to check first whether there is a previous test result and next to check the eyes of the candidate. Only if these checks do not lead to rejecting the candidate, a full examination is additionally required. Obviously, the expected cost of all these activities really determine whether this is a good design.

In practice, when analyzing materials that cover the product specification, it is a good idea to distinguish the top information element first. Examples of typical top elements are:

- for a banking process: the decision whether a loan should be granted to a company and, if so, under which conditions and for which amount.
- for a claim process of a social security agency: the decision whether an applicant should receive an unemployment allowance and if so for what reasons, for which period, and for which amount.
- for an intake process of an insurance company: the decision whether a family can be accepted as the holders of a health insurance policy.

Using the top element as the desired end result, it is a logical exercise to identify the information that can be used to directly render a value for the top information element. Obviously, this approach can be repeated for the newly found information elements.

Instead of such a top-down analysis, it may at times be more attractive to start at the beginning of the existing process, for example by analyzing application forms, complaint forms, and request forms that are in use to start the process. This is certainly feasible, but it bears the risk of the inclusion of superfluous information elements in the product data model. In a practical application of Product-based Design for a Dutch bank, we compared a posteriori the amount of information that was originally obtained in the business process and the information that was obtained in the final design. This comparison showed that almost 30 % of the originally obtained information was superfluous.

Another issue is how to pick the right information elements in a product data model. The following aspects are relevant for this choice:

1. an information element is too large if different parts of it are used in different production rules; the information element should be broken up to enable the application of production rules without determining irrelevant information.
2. information elements should not be necessarily associated with their physical manifestation, nor is it necessary that physical information carriers have an information element counterpart (avoid information elements like “intake form” or “computer screen output”).

3. information elements may be atomic, for example a name or a credit score, or composite. Examples for the latter are: all members of a family, a listing of all the requested products with their characteristics, or an overview of all the payment conditions that are in effect. The type of a composite information element is composed type, e.g. a set of numerals, free text, or a Boolean value.

Exercise 8.7 The following is an excerpt of the stipulations of a Dutch bank concerning medium length business loans:

> The funds for a medium length loan that is made available to a client but which is not withdrawn by the client must be placed on the money market. If the funding cost of the loan is higher than the rewards of the temporary placing, this difference is the basis for the monthly disposal provision. The disposal provision amounts to half of this difference with a minimum of $1/12$ % per month. The disposal provision should be part of the loan proposal.

Develop a product data model. Consider the "loan proposal" as the top information element. You may leave out the production rules for this exercise.

The next step in completing the product data model is describing as accurately as possible the involved production rules. All the production rules that relate to a specific product data model are referred to as its *production logic*. The step to determine the production logic may either follow up on the complete identification of all production rules, or may take place as soon as a new production rule has been distinguished. The production logic specifies how the value of an output information element may be determined on the basis of the values of its inputs. Note that some of the inputs may not be used in every calculation, but required for specific cases or to test constraints. The description of production logic may be given in pseudo code or another rather precise specification language. For example, using the helicopter pilot product data model again: The production rule that relates to the use of a value for F to determine a value for A may be: "If a candidate's vision of either or both eyes as expressed in diopters is above $+0.5$ or below -0.5, then such a candidate is considered as *unsuitable* to become a helicopter pilot".

Exercise 8.8 In the helicopter pilot product data model, there are two additional ways to determine someone's suitability to become a helicopter pilot beyond the one that was just mentioned. Provide sample production rules for both of these. Indicate separately under which conditions they are applicable.

The most important criteria on any language for the purpose of specifying the production logic are expressiveness and clarity. A representation of the production logic for each production rule is valuable for at least four reasons:

1. Writing out the full specification is a direct validation on the distinguished inputs of the involved production rule: forgotten inputs or bad data types can be detected.
2. An analysis of the production logic is relevant for the estimation of performance characteristics when actually determining information with this production rule:

labor and computer cost, speed, accuracy, etc. These characteristics are useful—as will be shown—in designing the workflow.
3. A representation of production logic that is of an algorithmic nature can be used as a functional specification for the information system that can execute this production rule. This is an important stepping stone for system development activities that may follow up the workflow redesign.
4. If the production logic is not totally algorithmic, it is likely that a human operator must execute it in practice. Then, the production logic is of use to develop task instructions for these operators.

The most accurate descriptions of production logic can be given when it involves an exact algorithm. In such a case, we will speak of a *formal* production rule. However, the production of many information elements in office settings is often not or not completely formal. It may be relevant, required or even the only option that a human passes a judgment without following a totally formalized decision making process. A typical example of such a non-formal production rule would involve the question whether some one is responsible for one's own discharge. If there is a dispute, opposite explanations of different parties must be taken into account. A human expert may be called in to determine the plausibility of these explanations, as there are no known algorithms to do this. Another example is whether the purchase of some good is ethically admissible, which is a relevant piece of information in determining whether a loan should or should not be granted for this purpose. This decision may suppose a value system that is hard to describe formally.

If a production rule is not of a formal nature it is important to at least check if all the required inputs are identified. Also, as noted before, describing as precisely as possible how the output must be produced on the basis of its inputs is a valuable step in determining working instructions for non-formal production rules. These working instructions can be provided to the people who will actually be responsible for determining the involved information in practice, that is to say: when the designed process is put into production. These rules may very well signal where Knowledge Management Systems can be beneficial.

Exercise 8.9 Consider an issue-to-resolution process (see Chap. 1). Determine two information elements that are important to provide a value for in this process, one of which involves a formal production rule and the other which is not.

Although a complete univocal procedure may not exist for a production rule, it is often the case that *under specific circumstances* this decision is completely formal. For example, in determining whether someone qualifies for an unemployment allowance it is relevant to determine what the usual pattern of labor hours for this person was during a week. In special cases, someone's actual labor pattern may be whimsical, e.g. due to a combination of different jobs or seasonal labor. So, determining the usual pattern is best done in those cases by using human interpretation.

However, if the applicant has a steady pattern of working hours for a long period of time, e.g. eight hours per day within one job, from Monday to Friday over the last five years, determining the usual labor pattern is straightforward and can be described formally. Another example is the authorization function that must be performed to determine whether a loan proposal may be sent to a client. Generally, this function is a matter of human judgment, which must take a large number of factors into account. On the other hand, if the loan sum is small, the client is a known client with sufficient coverage, and the purchasing goal is standard, the proposal may be acceptable with no further inspection.

When all information elements, their inter-dependencies and the production logic have been described, a final analysis step follows. This last step is required to identify all the characteristics that are relevant to design a business process that is efficient in terms of cost, reliability, or speed. The final analysis step consists of three steps, which we will describe here only briefly:

Source analysis The source analysis is aimed at identifying the sources of all the leaf elements in the product data model, i.e. the ones that do not rely on other information elements. Typically, multiple sources are available to obtain the same piece of information. For example, a record of historical grants of unemployment allowances may be obtained directly from an applicant or from the agencies that have provided these allowances in the past. Another example is somebody's payable debt position. In Europe, a bank may obtain this information from different scoring agencies (e.g. Experian, Equifax, Schufa, BKR). Different ways of obtaining information may have different characteristics. A client may be very willing to self-report information about credit positions, but this information may not be very reliable. Similarly, local authorities may provide correct domestic information at a very low cost, but their response time may be considerable. Depending on the criteria that are identified in the scoping phase, it is wise to first identify the possible sources for each leaf element and subsequently score them on relevant points of comparison. Assuming general performance goals like improving efficiency, bringing back throughput time while maintaining (or improving) an existing quality level, relevant points of comparison for each leaf are: cost of obtaining it, delivery speed of the information, availability of the specific information and reliability of the provided information.

Production analysis The production analysis focuses on the identified production rules with the aim to estimate the involved cost, speed, and quality of producing the new information. As there may be different ways to obtain a piece of information, similarly different production rules typically exist for the same piece of information. Designing the process is for a large part concerned with selecting the right set and the right execution order of production rules given a set of performance targets. From these targets it becomes clear which optimization criteria are prevailing. For example, suppose that an important performance target aims at a reduction of the labor cost. If there are two rule for the same piece of information, the one that takes the least amount of time is the one may be preferred. After all, the formal production rule may be automated. Obviously, this efficiency gain should be set off against the cost and time, which is involved with developing the software. It should be noted here that the production analysis is a very time-consuming part of the analysis phase, even more so when there is a poor tradition of operations measurement within the company at hand. The time that should be invested in obtaining reliable information should be balanced against the desired reliability of the quality estimates of the process design.

Fraction analysis The fraction analysis involves a study of the distribution of information element values. As we already explained, an information element may carry specific values. For example, the value of the information element "travel insurance required" may be either yes or no. The figures on the likelihood of information elements taking on specific values may be very relevant to design an efficient process. In combination with the figures from the production analysis (cost, speed, etc.), favorable orderings of executing production rules may be determined. For example, suppose that there are two production rules for the same information element with different applicability domains, with very different input elements, but with a similar cost structure to obtain values for them. In such a case, it may be wise from a cost perspective to aim at executing the rule with the widest applicability first. Only if this production rule does not yield an outcome, applying the other rule may be tried.

As has become clear, each of the separate analyzes are useful for deciding on the actual design of the to-be business process. Note that the steps not necessarily need to be executed in the order they are described here. The heart of the matter is that the results of these analyzes together provide a good basis for the actual act of process design.

8.4.2 Design: Deriving a Process from a Product Data Model

In this part, we will consider how a product data model can be used to derive a process design. For that purpose we will refer again to the helicopter pilot product data model. When developing a process design, we will allow for the creation of an activity for each information element in the product data model. Such an activity is concerned with creating a value for that information element according to a particular production rule. If the need is there, more than one activity for each information element may appear. Such duplicates, for example, can help to improve the readability of a process model that describes the process design.

Activities may only be incorporated in a process model in a way such that the dependencies between the information elements in the product data model are respected. That means that an information element can only be created by invoking a corresponding activity if the process design ensures that the required values for its inputs elements may have become available. So, for example, if we like to create an activity that produces a value for element X on the basis of a production rule that requires input values for Y and Z, the process design should ensure that values for Y and Z are created before this new activity is initiated. A process design that is generated with Product-based Design in such a way is called *correct*. Note that activities for creating values for leaf elements may be inserted at any stage: They do not require inputs after all.

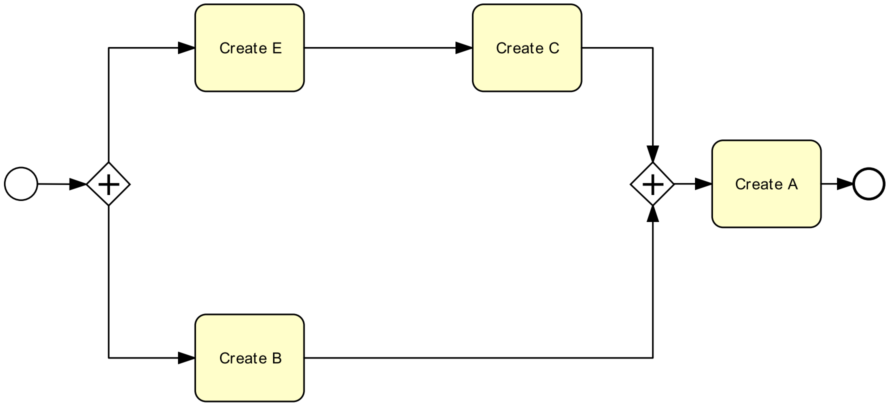


Fig. 8.5 An incorrect process design for the helicopter pilot product data model

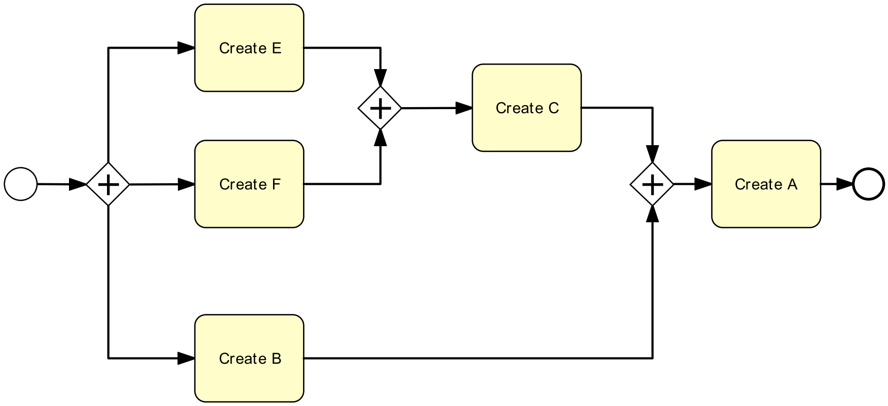


Fig. 8.6 A correct process design for the helicopter pilot product data model

Consider the process models that are depicted in Figs. 8.5 and 8.6. They represent alternative process designs. Both designs include an activity that is involved with the creation of information element A. In Fig. 8.5 it can be seen that the "Create A" activity will be invoked after the creation of information element B, which is created in parallel to the successive creation of values for E and C. However, the product data model in Fig. 8.4 shows no production rule for creating a value for C on the basis of E alone. Yet, there is a production rule that shows how the combined use of E and F can be used for that, but in this design no value for F is determined before the creation of a value for C. The design in Fig. 8.5 is, therefore, *not correct*.

Compare this design with the one in Fig. 8.6. Here, all creation activities are either producing values for leaf elements in the product data model (E, F, B) or create values for information elements on basis of production rules for which the inputs are created by preceding activities (C, A). The design in this figure is, therefore, *correct*.

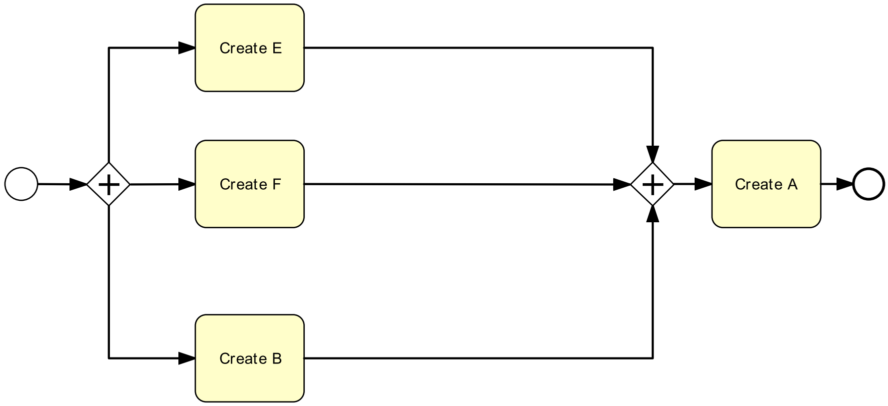


Fig. 8.7 An alternative process design for the helicopter pilot product data model

Exercise 8.10 Consider the process design that is visualized in Fig. 8.7. Is this a correct process design? Motivate your answer.

The second important criterion to be taken into consideration for a process design that is generated with Product-based Design is called *completeness*. A process design may or may not cover all the information elements that have been identified in a product data model. If a process design covers all information elements in a given product data model we will call it *complete*. Consider the designs in Figs. 8.5, 8.6, and 8.7. None of these designs cover the creation of a value for *D*, even though *D* is part of the product data model that these designs refer to. We therefore say that these designs are *not complete*.

Leaving out the creation of values for information elements from a process design may be a deliberate choice. In such a case, leaving out *D* inhibits the potential determination of a value for *A* using the production rule that has *D* as an input. It may very well be that on the basis of the analysis of data, the process designer deliberately decided not to include this option. This would make sense in a situation where very few applicants try to re-apply to become a helicopter pilot. After all, including this option may complicate the design, but not add much value in practical terms. On the other hand, the designer may have overseen this option overall, which explains why it is useful to check for completeness.

Exercise 8.11 Develop a *complete* process design on the basis of helicopter pilot product data model and capture that design as a process model.

Note that a process design that is complete with respect to information elements may still not exploit all available production rules. Without exact information on which production rules are used for which creation activities, though, this may be hard to determine. Clearly, it may be important for designers to check in practical applications of Product-based Design whether they have exploited all the opportunities that a product data model provides.

We can impose other quality criteria on process designs and many of these criteria can be extracted from more widely applicable ones. For example, we often like to capture a process design in the form of a *sound* process model, see Chap. 5.4.1.

At this point, we consider as a more important issue the *performance* of the process that is designed with Product-based Design. We mentioned that during the scoping phase of a project, the performance criteria are to be established for the process in question. One of the most important trade-offs is whether the design should result in a *fast* process versus an *efficient* process. A fast process can be designed by exploiting the opportunities to work in parallel. This may, however, not at all be an efficient process. Given that, in general, there may be different ways to establish a value for the same information element, a parallel process potentially induces too much work to be done. A more efficient way of carrying out a process would be to do as little work as possible, prioritize less costly yet effective activities, and only refer to alternatives if absolutely necessary. Note how these two different perspectives coincide with the parallelism and knock-out redesign heuristics we discussed earlier in this chapter.

Exercise 8.12 Develop a complete process design on the basis of helicopter pilot product data model and capture it as a process model. The design should involve a cost-efficient process. You may assume that the production rules to create a value for A on the basis of D or F are rather, just as the creation of values for the leaf elements. The use of the production rule for A that has B and C as inputs is, however, much more expensive.

While many other performance criteria can be taken into account when applying Product-based Design, we will not deal with them at this place. It is important to realize that a more sophisticated notion of performance will also assume more detailed information to be available. For example, if it is important to design a *secure* process it is important to understand the *risks* that are involved with obtaining or creating values for information elements.

8.5 Recap

In this chapter, we discussed the motivation for process redesign. The Devil's Quadrangle helped us to clarify that many redesign options have to be discussed from the perspective of a trade-off between time, cost, quality, and flexibility. Redesign can be approached as a purely creative activity or using a systematic technique. In this chapter, we focus on two of such systematic approaches, namely Heuristic Process Redesign and Product-based Design.

The methodology of Heuristic Process Redesign involves the phases of initiation, design, and evaluation. Various heuristics are available to support the design phase. They focus on the seven areas being related to processes, including customers, business process operations, business process behavior, organization, information, technology, and the external environment. We studied the application of some the heuristics in the case of a health care institution.

As an alternative method, we discussed a product-based design approach. The idea is to use a decomposition model of the product as a starting point, and infer options on what the process model for constructing the product could look like. Central to this method is the analysis and specification of the product data model. The actual design can then be tuned to the desirable performance characteristics of the process.

8.6 Solutions to Exercises

Solution 8.1 This is a hands-on exercise. A potential to approach this question might be to think of companies that offered services which are now provided by other companies via the internet.

Solution 8.2

1. "An airline has seen its profits falling over the past year. It decides to launch a marketing campaign among its corporate clients in the hope that it can extend its profitable freight business": Not a redesign initiative, no link to process.
2. "A governmental agency notices that it is structurally late to respond to a citizen's queries. It decides to assign a manager to oversee this particular process and to take appropriate counter actions": Redesign refers to *participants* and the *business process* itself.
3. "A video rental company sees that its customer base is evaporating. It decides to switch to the business of promoting and selling electronic services through which clients can see movies on-line and on-demand": Not so much a process redesign initiative; although there is certainly a link to process and products, this is much more a strategic initiative.
4. "A bank notices internal conflicts between two different departments over the way mortgage applications are dealt with. It decides to analyze the role of the various departments in the way applications are received and handled to come up with a new role structure": A redesign initiative touches on *process* and *participants*.
5. "A clinic wants to introduce the one-stop-shop concept to improve over the situation that its patients need to make separate appointments for the various diagnostic tests that are part of a procedure for skin cancer screening": A redesign initiative touches on *process* and *customers*.

Solution 8.3

1. Dealing with a customer complaint: Suitable.
2. Carrying out cardiovascular surgery: Mildly suitable, there are physical constraints involved here.
3. The production of a wafer stepping machine: Not very suitable, highly physical process.

4. Transporting a package: Mildly suitable, there are physical constraints involved here.
5. Providing financial advice on composing a portfolio: Suitable.
6. Designing a train station: Suitable.

Solution 8.4 Consider the following redesign acts. Which performance measures are affected by these, either positively or negatively?

1. "A new computer application is developed that speeds up the calculation of the maximum loan amount that a given client can be offered": Time is positively affected, development of the application may be costly.
2. "Whenever a clerk wants to have a quote from a financial provider, the clerk must use a direct messaging system instead of e-mail": Quality and time may be positively influenced since the feedback is obtained directly and may be more to the point. Quality may also be negatively affected, depending on the kind of feedback this interaction generates.
3. "By the end of the year, additional, temporary workers are hired and assigned to picking items for fulfilling Christmas orders": This provides more flexibility which may also be exploited to improve timeliness. It is clearly a costly affair and temporary workers may deliver lower quality since they are less familiar with the operations.

Solution 8.5 The + sign signifies a positive development with respect to the time dimension. Since process redesign is in general concerned with speeding up the process, a positive contribution necessarily relates to bringing back the time in terms of, for example, cycle time or service time. In the case of the contact reduction heuristic, the most likely scenario is that the time is reduced that is related to waiting for responses of customers or third parties. This explains the + sign.

Solution 8.6 This is a hands-on exercise. What is important is to identify under which assumptions positive or negative impacts can be expected.

Solution 8.7 The product data model is shown in Fig. 8.8. The meaning of the labels of the information elements is as follows:

1. LP: The loan proposal, the top element. Within the limits of the provided text, this is the ultimate piece of information that can be established.
2. DP: The disposal provision. This is the single element that is being mentioned in the given text as the information that must be included in the loan proposal.
3. FC: The funding cost of the loan. It is this information element that must be compared with rtp, the next element.
4. RTP: The rewards of the temporary placing. This piece of information is to be compared with fc. The disposal provision is based on this comparison.
5. MP: The minimum percentage. This is actually a constant, which is used under certain conditions.

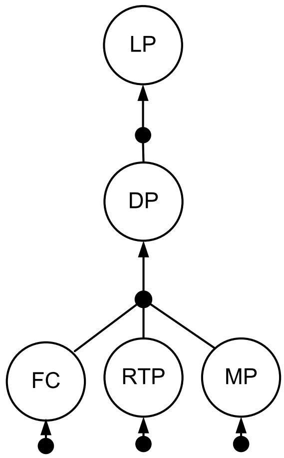


Fig. 8.8 Solution for the loan proposal

Solution 8.8 Production rule 1. To determine the value for A on the basis of B and C: "If both the psychological fitness and physical fitness of the candidate are considered excellent, then the candidate can be considered as suitable to become a helicopter pilot".

Production rule 2. To determine the value for A on the basis of D: "If the earlier test result indicated that a candidate was unsuitable to become a helicopter pilot, then the candidate is (still) considered as unsuitable to become a helicopter pilot".

Production rule 1 is always applicable. Production rule 2 is only applicable if the candidate has undergone a previous suitability test.

Solution 8.9 Information element 1: Issue class. This information element provides the severity of the issue that a client brings forth. It can be conceived as an information element that results from a formal production rule, where an employee must use a weighted evaluation formula to determine this severity.

Information element 2: Client agreement. This information element provides a client's opinion on whether an issue has been satisfactorily resolved. There is no formal production rule that relates it to it. The client provides her opinion, which is not governed by an algorithm.

Solution 8.10 The solution is shown in Fig. 8.9. Note that the design is complete since all information elements of the product data model are covered. Note that not all production rules are included, i.e. there is no way to establish a value for A on the basis of a value for B. This, however, does not affect its completeness. Also note that the design is correct (check for yourself).

Solution 8.11 The solution is shown in Fig. 8.10. Note that in this design as little work is done at each step, in search of opportunities to quickly terminate the process. First, the use of a value for D is pursued but, of course, an earlier result may not be available. Next, a value for F is created. Under certain circumstances, i.e. when

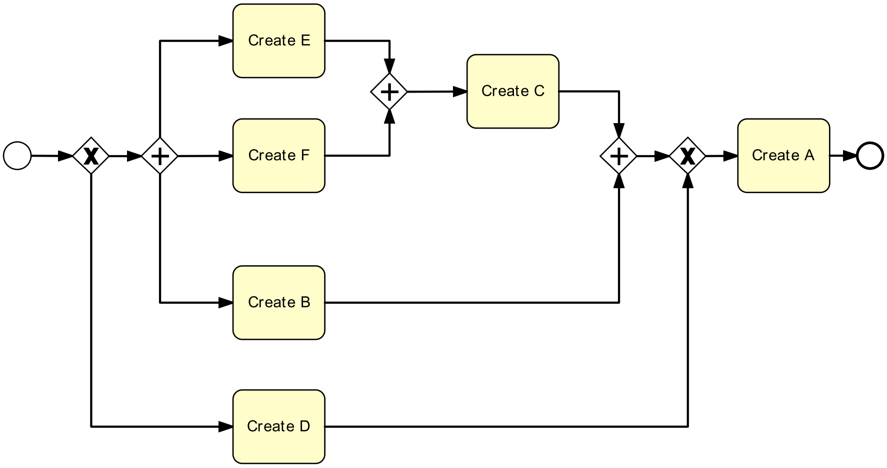


Fig. 8.9 A complete process design for the helicopter pilot product data model

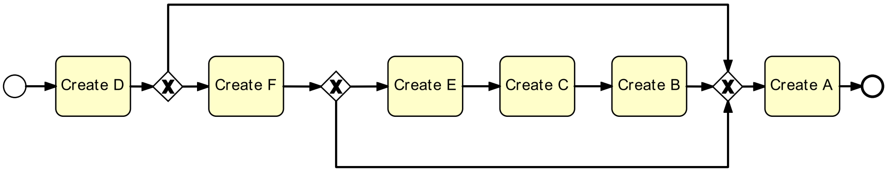


Fig. 8.10 A cost-efficient process design for the helicopter pilot product data model

eye-sight is very bad, this may lead to a determination of a value for *A* (not suitable). If this fails to stop the process, there is no other option than to produce values for all remaining information elements. Compare this design with the solution for the previous exercise.

8.7 Further Exercises

Exercise 8.13 Following is the literal description of a redesign case at IBM Credit Corporation, taken from the book "Reengineering the corporation" by Hammer and Champy [29]. It is split up into several parts. Please read these and answer the questions.

> Our first case concerns IBM Credit Corporation, a wholly owned subsidiary of IBM, which, if it were independent, would rank among the Fortune 100 service companies. IBM Credit is in the business of financing the computers, software, and services that the IBM Corporation sells. It is a business of which IBM is fond, since financing customers' purchases is an extremely profitable business. In its early years, IBM Credit's operation was positively Dickensian. When IBM field salespersons called in with a request for financing, they reached one of 14 people sitting around a conference room table in Old Greenwich, Connecticut. The person taking the call logged the request for a deal on a piece of paper. That

was step one. In step two, someone carted that piece of paper upstairs to the credit department, where a specialist entered the information into a computer system and checked the potential borrower's creditworthiness. The specialist wrote the results of the credit check on the piece of paper and dispatched it to the next link in the chain, which was the business practices department. The business practices department, step three, was in charge of modifying the standard loan covenant in response to customer request. Business practices had its own computer system. When done, a person in that department would attach the special terms to the request form. Next, the request went to a pricer, step four, who keyed the data into a personal computer spreadsheet to determine the appropriate interest rate to charge the customer. The pricer wrote the rate on a piece of paper, which, with the other papers, was delivered to a clerical group, step five. There, an administrator turned all this information into a quote letter that could be delivered to the field sales representative by Federal Express.

(a) Model the described business process. Use pools and lanes where needed.

The entire process consumed six days on average, although it sometimes took as long as two weeks. From the sales reps' point of view, this turnaround was too long, since it gave the customer six days to find another source of financing, to be seduced by another computer vendor, or simply to call the whole deal off. So the rep would call time and again to ask, "Where is my deal, and when are you going to get it out?" Naturally, no one had a clue, since the request was lost somewhere in the chain.

(b) Which dimension of the Devil's Quadrangle would be dominant for a redesign? Give an exact definition of the performance criterion.

In their efforts to improve this process, IBM Credit tried several fixes. They decided, for instance, to install a control desk, so they could answer the rep's questions about the status of the deal. That is, instead of each department forwarding the credit request to the next step in the chain, it would return it to the control desk where the calls were originally taken. There, an administrator logged the completion of each step before sending the paper out again. This fix did indeed solve one problem: The control desk knew the location of each request in the labyrinth and could give the rep the information they wanted. Unfortunately, this information was purchased at the cost of adding more time to the turnaround.

(c) Model the adapted process. Use pools/lanes where needed. (d) Can you explain in terms of the performance dimensions of the Devil's Quadrangle what has happened?

Eventually, two senior managers at IBM Credit had a brainstorm. They took a financing request and walked it themselves through all five steps, asking personnel in each office to put aside whatever they were doing and to process this request as they normally would, only without the delay of having it sit in a pile on someone's desk. They learned from their experiments that performing the actual work took in total only 90 minutes—one and a half hours. The remainder—now more than seven days on average—was consumed by handing the form off from one department to the next. Management had begun to look at the heart of the issue, which was the overall credit issuance process. Indeed, if by the wave of some magic wand the company were able to double the personal productivity of each individual in the organization, total turnaround time would have been reduced by only 45 minutes. The problem did not lie in the activities and the people performing them, but in the structure of the process itself. In other words, it was the process that had to change, not the individual steps.

In the end, IBM Credit replaced its specialists—the credit checkers, pricers, and so on-with generalists. Now, instead of sending an application from office to office, one person called a deal structurer processes the entire application from beginning to end: No handoffs.

> How could one generalist replace four specialists? The old process design was, in fact, founded on a deeply held (but deeply hidden) assumption: that every bid request was unique and difficult to process, thereby requiring the intervention of four highly trained specialists. In fact, this assumption was false; most requests were simple and straightforward. The old process had been over-designed to handle the most difficult applications that management could imagine. When IBM Credit's senior managers closely examined the work the specialists did, they found that most of it was little more than clerical: finding a credit rating in a database, plugging numbers into a standard model, pulling boilerplate clauses from a file. These activities fall well within the capability of a single individual when this is supported by an easy-to-use computer system that provides access to all the data and tools the specialists would use.
> IBM Credit also developed a new, sophisticated computer system to support the deal structurer. In most situations, the system provides the deal structurer with the guidance needed to proceed. In really tough situations, the deal structurer can get help from a small pool of real specialists-experts in credit checking, pricing, and so forth. Even here handoffs have disappeared because the deal structurer and the specialists he or she calls in work together as a team.
> The performance improvement achieved by the redesign is extraordinary. IBM Credit slashed its seven-day turnaround to four hours. It did so without an increase in head count-in fact, it has achieved a small head-count reduction. At the same time, the number of deals that it handles has increased a hundredfold. Not 100 percent, but 100 times.

(e) Consider the list of heuristics dealt with in this paper. Which of these can you recognize in the new process redesign?

Exercise 8.14 Indicate in what respect the application of the *Outsourcing* heuristic and the composition of larger activities—as a specific case of the *Activity composition* heuristic—can lead to similar or different results. Use the performance dimensions of the Devil's Quadrangle and provide specific interpretations.

Exercise 8.15 Consider the equipment rental process described in Example 1.1 (p. 2) and the corresponding issues documented in Example 6.4 (p. 199).

a Apply the redesign heuristics in order to address the issues documented in Example 6.4.
b Capture the resulting to-be model in BPMN.
c Explain the impact of the changes you propose in terms of the performance dimensions of the Devil's Quadrangle.

Exercise 8.16 Consider the university admission process described in Exercise 1.1 (p. 4) and the corresponding issues documented in Exercise 6.4 (p. 201).

a Apply the redesign heuristics in order to address the issues documented in Exercise 6.4.
b Capture the resulting to-be model in BPMN.
c Explain the impact of the changes you propose in terms of the performance dimensions of the Devil's Quadrangle.

Exercise 8.17 Consider the pharmacy prescription fulfillment process described in Exercise 1.6 (p. 28) and the corresponding issues documented in Exercise 6.9 (p. 209).

a Apply the redesign heuristics in order to address the issues documented in Example 6.9.
b Capture the resulting to-be model in BPMN.
c Explain the impact of the changes you propose in terms of the performance dimensions of the Devil's Quadrangle.

Exercise 8.18 Consider the procure-to-pay process described in Exercise 1.7 (p. 29) and the corresponding issues documented in Exercise 6.10 (p. 210).

a Apply the redesign heuristics in order to address the issues documented in Example 6.10.
b Capture the resulting to-be model in BPMN.
c Explain the impact of the changes you propose in terms of the performance dimensions of the Devil's Quadrangle.

8.8 Further Reading

Michael Hammer has written many, highly worthwhile books with his co-authors on process redesign, for example [27, 29]. Other management books that deal with the topic are, for example, [10, 46, 86]. In contrast to the topic of process modeling, process redesign has not received so much attention from the scientific community. When BPR is studied, the focus is mostly on case studies or the diffusion of the concept in practice itself is studied, for example in which domains it is applied or in which countries it is most popular. One of the most interesting studies in this category is quite dated [62], but it clearly shows the problems of what was initially considered business process redesign and how it quickly evolved into a more incremental approach. A very interesting study into the characteristics of different redesign methodologies is provided in [41], which has inspired various concepts that were dealt with in this part of the book.

The redesign heuristics that were discussed in this chapter have been described in quite some detail. After their initial presentation as best practices in [72], they have been validated and further analyzed in follow-up studies [47, 48]. Current efforts by various researchers are aimed at supporting practitioners in making sensible selections of redesign heuristics in specific cases [30, 44]. Also, attempts have been made to extend the set of redesign heuristics to their application in other domains, for example in [58].

Product-based Design was developed at Eindhoven University of Technology in cooperation with a Dutch consultancy company. Various case studies are available, which give a better idea of the practical application of this method and its potential benefits [70, 71]. Recently, the emphasis of researchers working on this topic is moving towards the automatic generation of process designs and the automated support of the execution of such processes [100]. Another way of looking at Product-based Design is that it is an approach that blends data and process. This is currently a hot topic: In this sense, IBM's artifact-centeric process modeling approach [36] and the data-driven process structures developed by the University of Ulm [57] are comparable approaches.

One of the main open questions in the area of process redesign is to what extent it makes sense to follow industrial reference models or to try and develop company-specific designs. While industrial reference models are offered by many vendors, it is not so obvious that they represent the best possible way to carry out processes.

Chapter 9
Process Automation

Besides black art, there is only automation and mechanization.
Federico García Lorca (1898–1936)

This chapter deals with process automation. First, we will briefly explain what an automated business process is, after which we will focus on a specific kind of technology that is particularly suitable to achieve process automation, i.e. Business Process Management Systems (BPMSs). We will explain the features and advantages of these systems, present the different types of BPMS, and discuss some of the challenges that are involved with introducing a BPMS in an organization. Finally, we will discuss what changes are required to a *business-oriented* process model like the ones seen so far, to make it *executable* and run in a BPMS.

9.1 Automating Business Processes

Process automation is a subject that may be approached from different angles. In a broad sense, it may refer to the intent to automate *any* conceivable part of procedural work that is contained within a business process, from *simple* operations that are part of a *single* process activity up to the automated coordination of *entire*, complex processes.

Take, for example, the order fulfillment process that we modeled in Chap. 3. Automating such a process may imply that every time the seller receives a purchase order, this is automatically dispatched to the ERP systems of the warehouse and distribution department, where the availability of the product is checked against the warehouse database. If the product is not in stock, the relevant suppliers are automatically contacted, e.g. via a Web service interface, to manufacture the product. Otherwise, instructions are sent to a warehouse worker, e.g. using an electronic form, to manually retrieve the product from the warehouse. Subsequently, an order clerk from sales receives a notification that a new order needs to be confirmed, e.g. via email. That clerk would then log into the purchase order tracking system within sales, see the order electronically, and confirm it by pressing a button, and so on.

M. Dumas et al., *Fundamentals of Business Process Management*,
DOI 10.1007/978-3-642-33143-5_9,

In this example the dispatching of the purchase order, the automated check of the product's availability, or the automated web messages are all manifestations of process automation in its broadest sense: they automate a particular aspect of a process. In this context, we will refer to an *automated business process*, also known as *workflow*, as a process that is automated in whole or in part by a software system, which passes information from one participant to another for action, according to the temporal and logical dependencies set in the underlying process model. Let us now consider systems that work with automated business processes.

9.1.1 Business Process Management Systems

A specific kind of process automation, which we are particularly concerned with in this book, exploits knowledge about how different process activities *relate* to one another. In other words, the type of information systems that we consider are *process-aware*. The main category of process-aware information systems that we will discuss are the so-called *Business Process Management Systems* (BPMSs). While there are other types of process-aware system, such as Customer Relationship Management (CRM) systems and Enterprise Resource Planning (ERP) systems, the special feature of BPMSs is that they exploit an explicit description of a business process, in the form of a *process model*, to coordinate that process. In that sense, a BPMS can be tailored to specific processes of any kind.

The purpose of a BPMS is to coordinate an automated business process in such a way that all work is done at the right time by the right resource. To explain how a BPMS accomplishes that, it is useful to see that a BPMS is in some way similar to a *Database Management System* (DBMS). A DBMS is a standard, *off-the-shelf* software package offered by many vendors in many different flavors, such as Microsoft SQL Server, IBM DB2 or Oracle Database. With a DBMS it is possible to capture company-specific data in a structured way, without ever having to consider how the exact retrieval and storage of the involved data takes place. These tasks are taken care of by standard facilities of the system. Of course, at some point it is necessary to configure the DBMS, fill it with data, and it may also be necessary to periodically adapt the system and its content to actual demands.

In a similar manner, a BPMS is also a standard type of software system. Vendors offer different BPMSs with a varying set of features, spanning the whole process lifecycle: from simple systems only catering for the design and automation of business processes, to more complex systems also involving process intelligence functionality (e.g. advanced monitoring and process mining), complex event processing, SOA functionality, and integration with third-party applications and social networks. Despite the variety of functionality a BPMS can offer, the core feature of such a software system resides in the automation of business processes. With a BPMS it becomes feasible to support the execution of a specific business process using the standard facilities offered by the system. However, it is essential that a business process is captured in such a way that the BPMS can deal with it, i.e. that

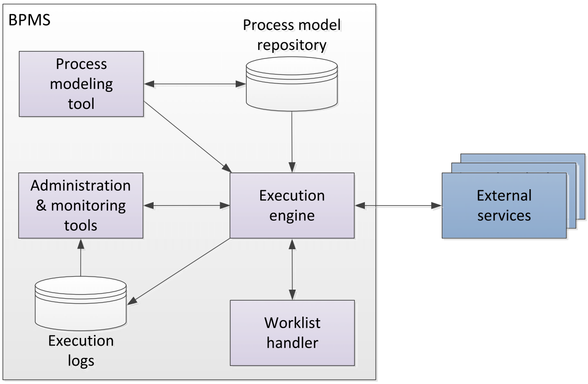


Fig. 9.1 The architecture of a BPMS

the BPMS can support its execution. From the moment a process is captured in a format that the BPMS can work with, it is important to keep that description of the business process up-to-date so that the process is supported properly over time.

In the past, mainly before the emergence of BPMSs, there existed a large number of tools focused on process automation, which did not encompass process intelligence functionality and which had relatively minimal support for process modeling. These tools were known under the name of *Workflow Management Systems* (WfMSs). Over time, many of these tools evolved towards BPMSs. Additionally, a plethora of stand-alone tools exist that cover a single feature of advanced BPMSs, such as stand-alone process modeling tools, process simulation tools and process analytics tools. All these tools provide value in supporting various parts of the BPM lifecycle, but do not generally support process automation. For this reason, they will not be the focus of this chapter.

9.1.2 Architecture of a BPMS

Figure 9.1 shows the main components of a BPMS, namely the execution engine, the process modeling tool, the worklist handler, and the administration and monitoring tools. The execution engine may interact with external services.

Execution Engine Central to the BPMS is the *execution engine*. The engine provides different functionalities including: (i) the ability to create executable process instances (also called cases); (ii) the ability to distribute work to process participants in order to execute a business process from start to end; (iii) the ability to automatically retrieve and store data required for the execution of the process and to delegate (automated) activities to software applications across the organization. Altogether, the engine is continuously monitoring the progress of different cases and coordinating which activities to work on next by generating *work items*, i.e. instances of process activities that need to be taken care of for

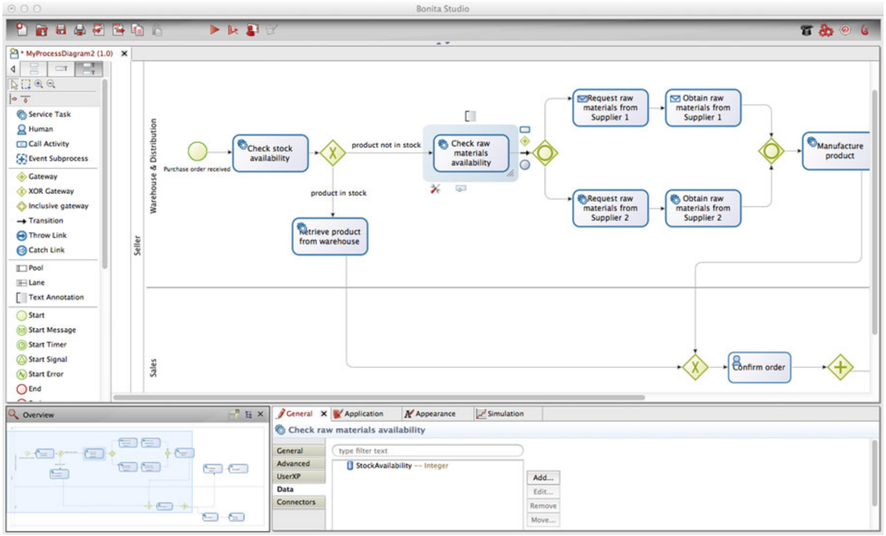


Fig. 9.2 The process modeling tool of Bonita Open Solution from Bonita Soft

specific cases. Work items are then allocated to resources which are both qualified and authorized to carry out these. More specifically, the execution engine interacts with the other components, as discussed next.

Process modeling tool The process modeling tool component offers functionality such as (i) the ability for users to create and modify process models; (ii) the ability to annotate process models with additional data, such as data input and output, participants, business rules associated with activities, or performance measures associated with a process or an activity; and (iii) the ability to store, share and retrieve process models from a *process model repository*. A process model can be *deployed* to the engine in order to be executed. This can either be done directly from the modeling tool or from the repository. The engine uses the process model to determine the temporal and logical order in which the activities of a process have to be executed. On that basis, it determines which work items should be generated and to whom they should be allocated or which external services should be called. Figure 9.2 shows the process modeling tool of Bonita Open Solution from Bonita Soft.

Worklist handler A worklist handler is the component of a BPMS through which process participants are (i) offered work items and (ii) commit to these. It is the execution engine that keeps track of which work items are due and makes them available through the worklist handlers of individual process participants. The standard worklist handler of a BPMS can best be imagined as an *inbox*, similar to that of an email client. Through an inbox, participants can see what work items are ready for them to be executed. The worklist handler might use electronic forms for an activity's input and output data. When a work item of this activity is selected and started by the participant from their worklist, the corre-

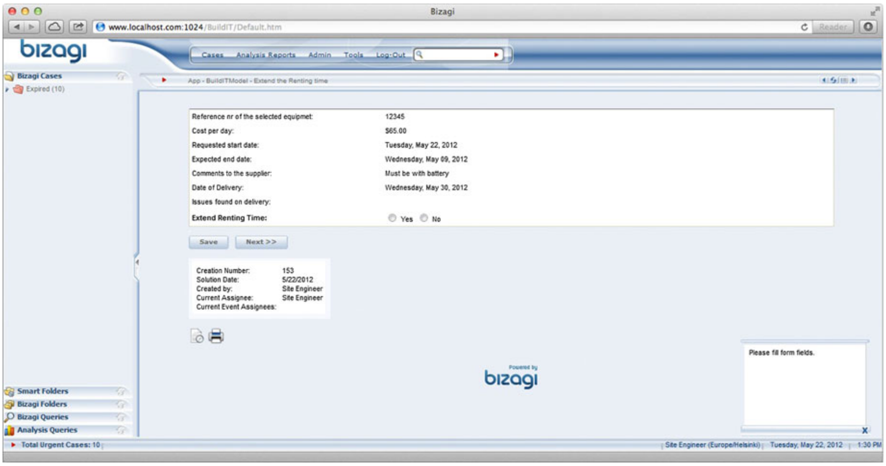


Fig. 9.3 The worklist handler of Bizagi's BPM Suite

sponding electronic form is rendered on screen. This step is called *check-out*. Participants can then enter data into the form, and signal completion to the engine. This step is called *check-in*. Afterwards, the engine determines the next work items that must be performed for the case in question. Often, participants can to some extent exert control over the work items in their worklist, e.g. with respect to the order in which they are displayed and the priority they assign to these work items. Also, a worklist handler will typically support a process participant in temporarily suspending work items or passing on control to someone else. Which exact features are available depends on the BPMS in question and its specific configuration. It is fairly common to customize worklist handlers, for example according to corporate design, to foster its efficient usage and acceptance within an organization. Figure 9.3 shows the worklist handler of Bizagi's BPM Suite.

External services It may be useful to involve external applications in the execution of a business process. In many business processes, there are activities which are not to be executed in a completely manual fashion. Some of these activities can be performed fully automatically, such that the execution engine can simply call an external application, for example to assess the creditworthiness of a client. The external application has to expose a service interface with which the engine can interact. Thus, we simply refer to such applications as *external services*. The execution engine provides the invoked service with the necessary data it will need for performing the activity for a specific case. On completion of the request, the service will return the outcome to the engine and signal that the work item is completed. This, again, is stored in the execution log. Some other activities in a business process are neither completely manual nor completely automated. Instead, such activities are to be performed by process participants with some form of automated support. For this category of activities, the execution engine will invoke the appropriate services with the right parameters, exactly at the moment

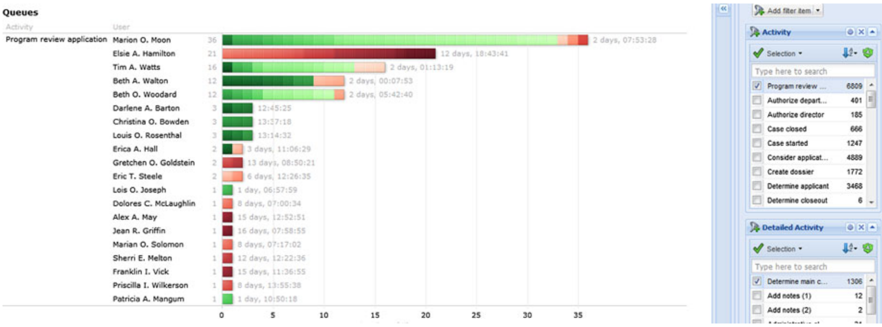


Fig. 9.4 The monitoring tool of Perceptive Software's BPMOne

that the employee selects a particular work item to work on. A typical example would be the invocation of a Document Management System (DMS) to display to the process participant a file that is important to carry out a specific work item. Think of the equipment rental request, where this would help a clerk to determine the appropriate piece of equipment to order. Sometimes too, a BPMS may need to transfer control over cases between different organizational units or organizations. One way of achieving this is by interacting with an external BPMS which exposes a service interface for this purpose. For example, consider a global insurance company that has offices in three different time zones: Japan, the UK and California. At the end of the working day in each of these time zones, all work items can be transferred to the execution engine in the next zone where the work day has just started. In this way, the execution of the business process never stops.

Administration and monitoring tools Administration and monitoring tools are the tools necessary for the administration of all operational matters of a BPMS. Consider, as a prime example, the actual availability of specific participants. If someone is unavailable to work because of illness or a vacation, the BPMS has to be made aware of this fact in order to avoid allocating work items to such a person. Administration tools are also required to deal with exceptional situations, for example to remove outdated work items from the system. Administration tools are also equipped with process monitoring functionality. One can use these tools to monitor the performance of the running business processes, in particular with respect to the progress of individual cases. These tools can aggregate data from different cases, such as average cycle times of cases, or the fraction of cases that are delivered too late. The BPMS records the execution of a process model step by step. The execution-related events recorded in this way are stored and can be exported in the form of *execution logs*. Some monitoring tools can analyze historical data extracted from the logs and compare it with live data (cf. Chap. 10). Figure 9.4 shows the monitoring tool of Perceptive Software's BPMOne.

This generic architecture, which underpins the functioning of any BPMS, is the evolution of a reference model for WfMSs which was proposed by the Workflow

Management Coalition (WfMC) in the nineties. A dedicated box "WfMC Reference Model" expands on this model.

To illustrate how a BPMS works, recall BuildIT's business process for renting equipment from Chap. 1. Let us suppose it is supported by a BPMS. The execution engine can track that for orders # 1,220 and #1,230 *site engineers* have already filled out the equipment rental requests. On the basis of a process model of the renting equipment process, the execution engine can detect that for both of these cases the proper piece of equipment must be determined. This needs to be done by any of the clerks at the depot. Therefore, the BPMS passes on the request to all worklist handlers of all clerks for further processing. For order #1,240 on the other hand, the equipment rental request is not available yet. So, the BPMS engine will not pass on a similar request for this order yet. Instead, it will await the completion of this work item.

Exercise 9.1 In which state is the process after all the actions of the rental process of BuildIT have been performed as described above? Which work items can you identify that are under control of the BPMS? Make sure to identify both the case and the activity for each work item.

WfMC REFERENCE MODEL

The Workflow Management Coalition (WfMC) is a standardization organization, founded in 1993, in which BPMS vendors, users and researchers have a seat. The purpose of the WfMC is to achieve generally accepted standards for terminology and interfaces for the components of a BPMS [35].

The WfMC has produced the so-called *WfMC reference model* which has become well-established in the world of process automation. The idea behind this reference model is that any supplier of a BPMS can explain the functioning of its specific system on the basis of it. The original reference model included six components, which resemble the components of the BPMS architecture in Fig. 9.1. They are: workflow engine, process modeling tools, administration and monitoring tools, worklist handler, external applications and external BPMSs. In the reference model, the interactions between its components take place through so-called *interfaces*, which are numbered from 1 to 5. Three of these interfaces are still recognizable in the BPMS architecture that is discussed in this chapter:

- *Interface 1* concerns the interaction between the engine and process modeling tools,
- *Interface 2* concerns the interaction between the engine and the worklist handler,
- *Interface 5* concerns the interaction between the engine and the administration and monitoring tools.

The other interfaces of the WfMC reference model are subsumed by recent developments. *Interface 3* governed the interaction between IT applications and the execution engine, but this has become outdated by the advent of standard service interfaces over the Web (e.g. Web services). Similarly, *interface 4*—which addressed the integration with external BPMSs—has also been subsumed since this can also be realized through standard service interfaces.

While most components of the WfMC reference model reappear in the BPMS architecture of Fig. 9.1, it should be noted that the WfMC architecture does not include the process model repository and does not explicitly represent execution logs. These elements, however, have become crucial assets in the area of process automation, analysis and redesign.

Exercise 9.2 Consider the following questions about a BPMS:

- Why would it not be sufficient to only create a business process model with the modeling tools, without any information on the types of resource that are available?
- In what situation will the execution engine generate multiple work items on the basis of the completion of a single work item?
- Can you provide examples of external services that may be useful to be invoked when a participant wishes to carry out a work item?
- What would be a minimal requirement to be able to pass on work items between the BPMSs of the insurance company that was provided as an example?
- If it is important that a BPMS hands out work items to available resources, can you imagine other, relevant types of information on resources that are useful to be captured by an administration tool (apart from whether they are ill or on vacation)?

9.1.3 The Case of ACNS

Building on the explanation of the BPMS architecture in the previous section it is now possible to sketch an example of an operational BPMS. We use a simplified view on a process in which claims are assessed within the ACNS company (*A Claim is No Shame*). The first activity in this process is an assessment of the claim, which is to be done by a senior acceptor or a regular acceptor. A regular acceptor is only qualified to make this assessment in the situation where the claim amount of a case is below €1,000. In case of a negative assessment, it is the responsibility of the account manager to convey the bad news to the customer. In case of a positive assessment, an electronic invoice is to be generated by a clerk of the finance department, who needs to dispatch that to the client in question. After these activities, the process is completed.

The above description shows that there are two dimensions that must be covered with the process modeling tool of the BPMS: (1) the procedure that specifies the various activities and (2) the various participants who are involved in carrying out these activities. The former part is recorded in a process model; the second is captured in, what is often referred to as, a *resource classification*. In addition, the relations between these models or specifications must be defined, i.e. who is able and qualified to perform which activity. Often, these relations are also specified as part of the process model. These relationships may not be static but be dependent on all kinds of business rule. For example, the distinction between the authorization levels of the senior and ordinary acceptor in assessing claims is an example of a dynamic rule, i.e. it is determined by the current value one of a piece of information.

Once these descriptions are defined, the execution engine of a BPMS would generally be able to support this process. Now let us assume that almost simultaneously two claims come in:

1. A car damage of €12,500, as claimed by Mr. Bouman.
2. A car damage of €500, as claimed by Mrs. Fillers.

Ms. Withagen has been with ACNS for a long time and, for the past years, functions on the level of a senior acceptor. This month, Mr. Trienekens has started his training and works as an acceptor. At the start of his contract, the system administrator, Mr. Verbeek, has used the administration tool of the BPMS to add Mr. Trienekens to the pool of available acceptors.

Based on the process model, the resource classification, and operational data on the availability of the various employees, the enactment service of the BPMS now takes care of forwarding both newly received claims to the worklist handlers of Ms. Withagen. After all, she may assess both claims based on her qualifications. Mr. Trienekens, in his turn, will only see in his worklist handler the damage claim of Mrs. Fillers.

On noting the work item in his worklist, Mr. Trienekens starts to work on it immediately. He selects the damage claim of Ms. Fillers to deal with. In response to that action, the execution engine ensures that the corresponding work item *disappears* from the worklist of Ms. Withagen. The reason, of course, is that this piece of work needs to be carried out only once. In any case, Ms. Withagen is still working on the handling of an earlier case, but shortly thereafter selects the claim of Mr. Bouman to deal with through her worklist handler.

In response to the selection of work items by both Mr. Trienekens and Mrs. Withagen, the execution engine will ensure that both will see the electronic claim file on their screen of the respective customers. The execution engine does so by using the appropriate parameters in invoking the DMS of ACNS at the workstations of the acceptors. The DMS also displays the scanned version of the claims, which were originally sent in on paper. In addition, the BPMS takes care of displaying to both the acceptors an electronic form that they can use to record their assessment, also through the invocation of a service.

Mr. Trienekens decides to reject the claim. The worklist handler notices this, because it monitors the specific field on the electronic form that receives a negative value. Based on the logic captured in the process model, the execution engine can determine that the case must be handed over to the account manager of Ms. Fillers and sends a work item to that participant, requesting to inform the client on the negative assessment.

Ms. Withagen arrives at a positive assessment of the claim under her watch and decides to approve it. The execution engine ensures that a service is invoked to determine the new monthly premium for Mr. Bouman, taking into account his no-claim history which is registered in a claim database. The retrieval of the information from this database is also realized through a service. Once this calculation is completed, a work item for the various available financial employees is created to pay the damage. The work item appears in the worklist of each of these financial officers, until one of them selects it for processing. After the payment has been carried out, the process is completed.

As can be seen in the ANCS example, all components of the BPMS architecture play a role in coordinating the work, specifically to ensure that the appropriate work items are created and carried out by the involved participants.

Exercise 9.3 Consider the following developments and indicate which components of the BPMS architecture are affected when they are to be taken into account:

1. A new decision support system is developed to support acceptors in making their assessment of claims.
2. Ms. Withagen retires.
3. A new distinction between claims becomes relevant: regular acceptors are now also qualified to deal with claims above €1,000 as long as they worked on previous claims by the same client.
4. Claims that are issued on cars which are over 10 years old need to be continuously monitored by management.

Up to this point, we have discussed BPMSs as if they are particular software packages that deliver more or less the same functionality. However, it is closer to the truth to state that there are distinct flavors of BPMSs and that different organizations or different processes within the same organization may be best served by different types of BPMS. This observation is further discussed in the box "Types of BPMS".

TYPES OF BPMS

There are several ways to distinguish between the available BPMSs. One classification is based on the use of two axes: one that captures the *degree of support* that the BPMS delivers and the other that expresses how that systems differ from each other with respect to their *orientation on process or data*. We describe and illustrate four different types of system: groupware systems, adhoc workflow systems, production workflow systems, and case handling systems. These systems can be positioned in the spectrum of BPMSs as shown in Fig. 9.5.

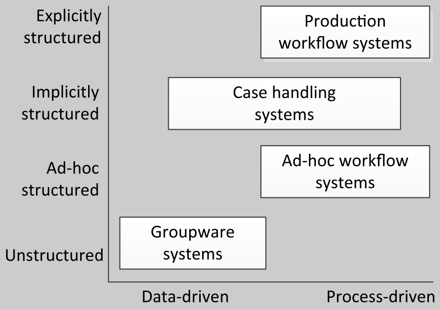


Fig. 9.5 The spectrum of BPMS types

Groupware systems: The two underlying principles of groupware systems are that the user is enabled to easily share documents and information on the one hand and to directly communicate with other users on the other. The best known example of a groupware system is IBM's Lotus Notes. Groupware systems are very widely used and particularly popular for their high operational flexibility. When it comes to supporting business processes, these systems cannot do such in a strict sense. Groupware systems traditionally do not support an explicit notion of the business processes; however, extensions like Lotus Domino Workflow exist as well.

Ad-hoc workflow systems: Ad-hoc BPMSs, like TIBCO's BusinessWorks or Comalatech's Ad-hoc Workflows, allow on-the-fly process definitions that can be created and modified. So, even though there is already a process definition in existence to deal with a case type, it is possible during the execution of that process to adapt the process, for example to include an additional step. For these kinds of system, it is often so that each case (an order, claim, complaint, etc.) has its own private process definition. This prevents many problems that are acclimated with updating a general process definition when it has running instances. Clearly, ad-hoc BPMSs allow much flexibility. The BPMS InConcert, for example, even allows users to trigger a business process instance on the basis of a completely empty process definition, which is extended as it becomes clearer what needs to happen and in what order. Another direction is that initially the BPMS works on the basis of a standard solution or template, which can be modified at will. Interestingly, such a modified procedure may be used as the template for starting the processing of a new case. In general, there are two major requirements to successfully apply ad-hoc BPMSs in an organization. The first requirement is that end users are very aware of the processes in which they operate. This means that only processes should be defined or modified by people with a very good overview of the process and the consequences of deviating from usual practice. The second requirement is that users have sophisticated tools at their disposal to model business processes and that they are capable of modeling. The combination of these requirements hinders the wide-scale application of ad-hoc BPMSs at this point.

Production workflow systems: The most familiar type of BPMS is the production workflow system. Typical representatives are IBM's Business Process Manager and Bizagi's BPM Suite. Much of what we described in the previous sections on workflow applies to this class of BPMSs. Work is routed strictly on the basis of explicitly defined process descriptions captured in process models. Managing data, such as documents or e-mail support, is generally not offered by these systems. In general, it is also impossible or hard to deviate from a process logic if that has not been explicitly captured in the process model. Sometimes a further distinction is made within production workflow into *administrative* and *transaction processing* BPMSs. This makes it possible to distinguish further shades with respect to the degree of automation of the work that is coordinated. Administrative BPMSs, are used in settings where a (large) portion of work is still performed by people; transaction processing BPMSs support business processes that are (almost) fully automated.

Case handling systems: The idea behind a case handling system (or Adaptive Case Management system) is that it does not strive for a tight and complete specification of a business process in a model. Rather, *implicit* process models are used, which capture a conventional flow from which—unless this is explicitly prohibited—a user can deviate. A case handling system is usually fully aware of the precise details of the data belonging to a case (including customer data, financial or medical data). On the basis of such awareness, the system is able to provide end users a highly precise insight into the status of a case, as well as the most obvious steps to continue the process. Contemporary examples are i-Sight's Case Management Software and BPMOne by Perceptive Software. The latter also, if desired, supports a production workflow approach and in that sense is a hybrid BPMS.

There are other types of system that share characteristics and functionality with BPMSs. *Document management systems* (DMSs) primarily take care of the storage and retrieval of documents, like document scans and PDFs, but they often offer workflow automation features as well. An example is Adobe's LiveCycle Enterprise Suite. *Process Orchestration Servers* focus on process automation but have a specific emphasis on automated processes that require the integration of multiple enterprise applications. An example is Oracle's SOA Suite.

9.2 Advantages of Introducing a BPMS

In this section we reflect on why it would be attractive for organizations to use a BPMS. There are four broad categories of advantages which we will discuss here: workload reduction, flexible system integration, execution transparency, and rule enforcement.

9.2.1 Workload Reduction

A BPMS automates part of the work that is done by people in settings where such a system is not in place. First of all, it will take care of *transporting work* itself. In a paper-based organization, work is usually transported by internal postal services, often delaying processing for one work day at each handover, or by the participants themselves at the expense of their working time. All such delays are completely eradicated when a BPMS can be used to dispatch work items electronically. In some situations, the BPMS can take care of the entire process by invoking fully automated applications. In such cases, we speak of *Straight-Through-Processing* (STP). Particularly in the financial services, many business processes that used to involve human operations are now in STP mode and coordinated by BPMSs. Also in other domains, think for instance of electronic visa, at least a portion of the cases can be handled in a completely automatic fashion.

The second type of work that is being taken over by the BPMS concerns *coordination*. The BPMS uses the process model for determining which activities need to be performed and in what order. So, every time the BPMS uses this knowledge to route a work item it potentially saves someone the time to even think about what should be done next. Another form of coordination time saved is the signaling of completed work. In a paper-based organization, work will be lying around for quite some time in case of work hand-overs. What often happens is that someone takes over work, suspends it for some reason, after which the work package gets stuck in another pile of work. The BPMS will at all times be able to signal the status of all work items and it can take actions to ensure that progress is being made.

The final type of workload reduction by using a BPMS is the *gathering of all relevant information* to carry out a particular task. In a situation without a BPMS, it is the employee who needs to do this collection. Finding the right file in particular—it is never there where you would expect it—can be a time-consuming affair. Note that this type of advantage rests on the assumption that along with the introduction of the BPMS the effort is taken to digitalize the stream of documents in an organization. The implementation of a DMS is actually what is often observed alongside a BPMS implementation. Certain vendors, such as IBM and Perceptive Software, offer integrated suites of BPMS and DMS functionality. Other BPMS vendors often have strategic cooperations with companies that offer a DMS, such that it is relatively easy to integrate their joint systems.

9.2.2 Flexible System Integration

Originally, the most mentioned argument to start with a BPMS is the increased flexibility that organizations achieve with this technology. To explain this best, a short reflection on the history of computer applications is due. There is an interesting trend, as identified by Van der Aalst and Van Hee [95] that generic functionality is split off from applications at some point.

Roughly throughout the 1965–1975 period, computer applications were run directly on the operating systems (OS) of a computer. Each application would take care of its own data management and would be using proprietary techniques to do this efficiently. As a result, it turned out to be difficult to share data among applications and to maintain consistency. Clearly, programmers of different applications would be involved with developing similar routines to solve similar problems. From 1975 onwards, DBMSs as a new type of standard software emerged that took on the generic task of managing data efficiently. As a result, data could be shared rather easily and programmers of new applications would not need to worry anymore about ways to store, query, or retrieve date. Some 10 years later, around 1985, User Interface Management Systems (UIMS) were introduced to provide a very generic interface component to many applications. Through the provision of facilities like drop-down boxes or radio buttons in accessible libraries, each computer programmer would be able to make use of these. By 1995, the first commercial BPMSs enter the market place (considerable time before, research prototypes have been available). Like DBMSs and UIMSs in their focus area, BPMSs would provide generic support for the area of business process logic.

The introduction of a BPMS is a logical sequel to the separation of generic functionality of what were one monolithic computer programs. Still in the 1990s, it was estimated that 40 % of all the lines of code running on the mainframe computers of banks would have to do with business process logic, not with the calculations or data processing themselves. The typical kind of information processing in the context relates to the identification of activities, their order of execution, and or the participants responsible for carrying them out. For example, it would be specified that after a mortgage offering was completed, this needed to be signaled to the manager of the department, triggering a signal on her monitor.

The obvious advantage related to this development is that it has become much easier with a BPMS to manage business process logic on its own. This is due to the fact that it is much more convenient to update the description of a business process without having to inspect the application code. Also, the reverse would become easier, i.e. modifying an application while not touching on the order of how things on the business process level would need to unfold. BPMSs, in short, would enable organizations to become more flexible in managing and updating their business processes as well as their applications.

BPMSs also provide the means to glue together separate systems. Large service organizations typically deploy myriads of IT systems, which all exist more or less independent of each other. Often, such a situation is referred to as *island automation*. A BPMS may be introduced in such a situation as a means of integrating such systems. It will safeguard that all the separate systems will play their due role in the business process they support.

A word of caution is due here. The BPMS itself will offer no direct solution to the problem that there is often a redundant storage of information across many different IT systems. In fact, a BPMS will in general have no knowledge of the actual data that end users will manipulate using the various IT systems. If the BPMS is to operate as an integrator between all the existing systems, this will require a thorough information analysis to map which data is used and available.

9.2.3 Execution Transparency

An advantage that is often overlooked is that a BPMS can operate as a treasure trove with respect to the way that business processes are really executed. It is likely that the developers of the first BPMSs did not clearly have any advantage in mind on providing any insights about the execution of a process to anyone else than people executing the process itself. Sure enough, to have a BPMS operate at all, it must keep an accurate administration on which work items are due, which can only be determined by actively monitoring and recording which work items have been completed by which resources and at what time new cases enter the process. Yet, for a BPMS to function properly, it is not necessary to keep all that data available once the associated cases are completed. The management overseeing such a process, however, may have an entirely different perspective on this issue. There are two types of information that may be useful to generate interesting insights on basis of the administration a BPMS keeps:

1. *Operational information*, which relates to recent, running cases, and
2. *Historic information*, which relates to completed cases

Operational information is relevant for the management of individual cases, participants, or specific parts of a business process. A characteristic example is the following. From analyses of various governmental agencies involved with granting permits in the Netherlands it has become clear that determining the exact status of a permit application was one of the most time-consuming activities for the civil servants involved. By the use of most commercially available BPMSs, retrieving that status is a futility. Such a status may be, for example, that the request of Mr. Benders to extend his house with an extra garden wing has been received, matched with the development plans for the area he is living in, and that further processing is now dependent on the receipt of the advice of an external expert. Another use of operational information would relate to the length of queue in terms of work items. For example, there are 29 applications for building permits that await the advice of an external expert. From these examples it may become clear that initiatives to improve customer service, in particular with respect to answering questions about their orders, often relies on the use of a BPMS.

Historic information, in contrast to operation information, is often of interest on a particular level of aggregation, for example covering more cases over an extended period of time. This kind of information is of the utmost importance to determine the performance of a particular process or its conformance to particular rules. With respect to the former, you may want to think of average cycle times, the number of completed cases over a particular period, or the utilization of resources. The latter category could cover issues like the kind of exceptions that have been generated or the number of cases that violated a particular deadline.

It makes sense to consider the kind of insights that need to be retrieved from a BPMS before it is actually implemented in an organization. Technical issues play a role here, like the period of time that the logs of a BPMS need to be kept and, therefore, how much storage space should be devoted for that. Consider that it becomes a bit problematic if historic information is important on the aggregate level of years if there is only space to save the events of at most a months. There are also conceptual issues. If it is important to monitor a certain milestone within a process it is essential that it makes part of the process model that is used for the execution of the related business process. To use the previous example: If it is important to be able to recognize the stage in which a case has to wait for the advice of an external expert, then that milestone must be part of the process model. In this way, process automation provides the foundation for process intelligence (cf. Chap. 10).

9.2.4 Rule Enforcement

Except for the obvious advantage that a business process could be executed more efficiently by using a BPMS, such a system will also take care of that the process is carried out in precisely the way that it has been designed. When rules are explicitly enforced, this can be considered as a quality benefit: one does, what one promises. In many settings, employees will have considerable freedom to carry out a business process in the way it looks best to them (or most convenient). This individual assessment does not necessarily coincide with the best possible way a business process is executed from the overall perspective of an organization. In this respect, a BPMS can be a means to safeguard that business processes are executed in a predefined way, without any concessions.

As an illustration, consider the *separation of duties* control that is well known in the financial services domain. It means that the registration and inspection of a financial transaction will need to be carried out by different individuals. This type of logic is both quite easily implemented in BPMSs and enforced by such systems. The BPMS registers which individuals have carried out which work items and can take this information into account when allocating new work items. Note that a BPMS is, in general, sufficiently sophisticated so that employees can alternatively fulfill the registering and inspecting role for different cases.

The capacity of a BPMS to enforce rules is currently of much interest to organizations. Until a decade ago, primarily governmental organizations were implementing such systems purely motivated by this concern. After all, there are various laws that such organizations have to conform with. Nowadays, financial and other professional service organizations have become similarly enamored by BPMSs. An important development in this is the rise of various governance frameworks, which started in 2002 with the *Sarbanes–Oxley Act* as a reaction to misconduct in Enron and Worldcom. The law places a high responsibility with company executives to install management controls and procedures within organizations and to see their proper execution. Obviously, this is were BPMSs can play an important role.

Exercise 9.4 To which categories would you classify the following incentives to introduce a BPMS in an organization?

- An auditing agency has found that the written procedures and actual execution of business processes are not aligned. The management of that organization wishes to enforce the written procedures and decides to introduce a BPMS.
- The clients of an organization complain that they can only get very shallow updates on the progress of the orders they make. The IT manager of that organization looks into the use of a BPMS to capture and provide status information of all these order.
- An insurance organization finds out that there is a high need to quickly adjust the way claims processing is carried out to the offerings that its competitors bring to the market. Using a BPMS is considered to address this demand.

9.3 Challenges of Introducing a BPMS

Despite the many advantages of using BPMSs, there are some notable obstacles with respect to introducing these in an organization. We will distinguish between technical and organizational challenges here.

9.3.1 Technical Challenges

What should be one of the strengths of a BPMS is also one of the pitfalls. A BPMS is capable of integrating the use of different types of information system in their support of a business process. The challenge is that many applications have never been developed with this coordinated use in mind. The mainframe applications that can still be found within banks and insurance companies today are notorious in this regard. In the most favorable case, such systems are technically documented but it happens often that there is no one of the original development team available anymore who knows exactly how these are structured. In such cases, it is very hard to determine how a BPMS can trigger such systems to make them support the execution of a particular work item, to exchange information between the BPMS and such a system (e.g. case data), and how to determine when an employee has used such a system to complete a particular work item.

A technique that has been used to make interaction with such legacy systems at all possible is *screen scraping*. The interaction between a BPMS and the mainframe application then takes place on the level of the user interface: the key strokes that an end user should make are emulated by the BPMS and the signals sent to the display are tracked to establish the progress of carrying out an activity. It will come as no surprise that such low-level integration solutions will incur much rigidity to the overall solution and will, in fact, undermine the flexibility advantages that are normally associated with using a BPMS.

A specific problem that occurs with respect to the integration of existing applications with BPMSs is the lack of process-awareness of traditional systems. In a process-aware system, separate cases will be handled separately. In other words, such a system works on a case-by-case basis. In many traditional systems, *batch processing* is the dominant paradigm. This means that a particular task is executed for a potentially large set of cases, which does not always go well with the philosophy of a BPMS. Note how the Case-based work heuristic mentioned in Chap. 8 explicitly targets this situation.

Fortunately, in the area of system integration much progress has been made in the past decade. Many old systems are being phased out and new, open systems with clearly defined interfaces take their places. Technologies that are referred to as *Middleware* and *Enterprise Application Integration* tools are now available that strongly facilitate the communication and management of data in distributed applications. Microsoft's BizTalk and IBM's WebSphere are well-known software suites that can be used in this respect and there are open source technologies available as well. The success of *Web services* is another driver behind improved, coordinated use of different types of information system, including BPMSs. A Web service is a piece of functionality, for example the identification of the best possible price for a particular good within a range of providers of that good, which can be invoked over the Internet. Most BPMSs provide good support for integrating specific Web services in executable business processes. This kind of set-up would fit within a popular software architecture paradigm that is commonly referred to as *Service-Oriented Architecture*.

With respect to technical integration capabilities it is fair to say that recent developments are favorable for the use of BPMSs and that technical challenges, at least with respect to this aspect, are likely to further decrease over the next years.

9.3.2 Organizational Challenges

The introduction of an operational BPMS often has an impact on extensive parts of an organization. This implies that the introduction of a BPMS can be challenging from an organizational perspective. The interests of different stakeholders have to be balanced, who usually have diverging performance objectives and vie for the same resources. Getting an insight into how existing processes unfold is an enormous challenge in itself, sometimes taking months of work. Here, not only political motives may play a role—not everyone will be happy to give away how work is done, especially if not much work is done at all—but psychological ones as well: people tend to focus on describing the worst possible exceptions when asked to describe what their role is in a process. One scholar has referred to this tendency as the reason "why modelers wreck workflow innovation".

A factor that adds to this complexity is that organizations are dynamic entities. It is fairly usual that during the introduction of a BPMS, which may span a couple of months, organizational rules change, departments are scrapped or combined, participants get other responsibilities, new products are introduced or taken off the market. These are all examples of events that may be important to consider when the aim is to make a BPMS function properly in an organizational setting. In practice, this accounts for the insight that the gradual introduction of a BPMS is usually more successful than a "big bang" strategy, in which a BPMS from one day on the other is expected to replace the way operations were managed.

The perspective of the users on the introduction of a BPMS should be considered carefully. Most subject experts will first need to experience *hands-on* what it is to use a BPMS before they can really appreciate what that means for their job. There may also be concerns and fears. First, there might be a "Big brother is watching you" sentiment. Indeed, a BPMS will record all the events that are involved with executing a process, including who carried out what piece of work and at what time. It makes no use—and from a change management perspective, it could actually be self-defeating—to ignore this concern. Rather it is up to organizations to clarify how this information will be used and that there are positive effects that can be expected of the usage of this information as well. Another fear that is common with end users of BPMSs is that their work will take on a mechanistic trait, almost as if they are working on a chain gang. This fear is in part genuine. It is true that the BPMS will take care of the allocation and routing of work. What can be argued, though, is that these are not the most exciting or valuable parts of the work that needs to be done (which is precisely the reason that they could be automated in the first place). If you would consider the situation where an employee needs to spend large parts of their time on finding the right information to do the job properly, the BPMS can be a favorable mechanism to give that time back to the employee. Another line of reasoning is that it highly depends on the configuration of the BPMS whether the mechanization effect will actually occur. Compare, for example, the situation where a BPMS pushes a single work item at a time to an employee to be carried out or rather a range of work items that someone could choose according to one's own preferences. These options, which result from a configuration decision, can make a huge difference in the perception of the value of the BPMS.

To sum up, the introduction of a BPMS is particularly complex, precisely because it supports entire business processes. It is not for nothing that out of many research projects into IT projects "strong management commitment" is always on top of the factors that explain successful implementations. The introduction of a BPMS is, perhaps even more so than for other types of technology, not for the faint of heart.

Exercise 9.5 Consider the following issues that come up when introducing a BPMS in a hospital to support preoperative care, i.e. the preparation and management of a patient prior to surgery. Classify these to the technical or organizational issues, or both.

1. On hearing about the plans to introduce a BPMS, the surgeons flatly reject to cooperate on this endeavor. Their claim is that each patient is an individual person that cannot be trusted to the care of a one-size-fits-all system.
2. The anesthetists in the hospital use a decision support system that monitors the proper dosage of anesthetics to patients. The system is developed as a stand-alone system that is difficult to synchronized with the BPMS, which has to feed the decision support system with patient data.
3. The nurses are provided with mobile devices, which they can use to access their worklist handlers. However, they find it difficult to follow up on the automatic notifications which are signaled to them as gentle vibrations of the device.

9.4 Turning Process Models Executable

In this section we will show how to automate a business-oriented process model in order to execute it on a BPMS. In particular, we will consider the case of production BPMSs.

Mapping processes for automation requires an approach to process modeling that is quite different from what is needed for communication or analytically focused process models. In fact, because of their intent, business-oriented process models are not necessarily precise and may thus contain ambiguities. Conversely, *executable process models* must be precise specifications in order to be interpreted by a BPMS.

We propose a five-step method to incrementally transform a business-oriented process model into an executable one, as follows:

1. Identify the automation boundaries
2. Review manual tasks
3. Complete the process model
4. Bring the process model to an adequate granularity level
5. Specify execution properties

Through these steps, the business-oriented model will become less abstract and more IT-oriented. These steps should be carried out on a process model that is both correct in terms of structure and behavior. For example, if the model contains behavioral errors like a deadlock, the BPMS may get stuck while executing an instance of this process model, with a potential impact on the operations of the organization. We have already discussed process model verification in Sect. 5.4. From now on we assume that the process model is correct.

9.4.1 Identify the Automation Boundaries

First, we need to identify what parts of our process can be coordinated by the BPMS, and what parts cannot. In a process there are *automated*, *manual* and *user* tasks. Automated tasks are performed by the BPMS itself or by an external service while manual tasks are performed by process participants without the aid of any software. A user task sits in-between an automated and a manual task. It is a task performed by a participant with the assistance of the worklist handler of the BPMS or of an external task list manager.

This difference between automated, manual and user tasks is relevant: automated and user tasks can easily be coordinated by a BPMS, while manual tasks cannot. Thus, in this first step we need to identify the type of each task. Then, in the next step, we will review the manual tasks and assess whether we can find a way to hook up these tasks to the BPMS. If this is not possible, we will have to consider whether or not it is convenient to automate the rest of the process without these manual tasks.

Let us consider again the order fulfillment process model that we created in Chap. 3, which is shown in Fig. 9.6 for convenience (for the moment, discard the activity markers). Let us assume we get this model from a business analyst and our task is to automate it from the seller viewpoint. This means we need to focus on the process in the Seller pool and discard the rest. The first activity, "Check stock availability", sits in the ERP lane. This means that already at the conceptual level it was identified as an automated task. ERP systems do provide modules to manage inventories which automatically check the stock levels of a product against a warehouse database. This activity is highly repetitive as it is performed for each purchase order received. Performing it manually would be very inefficient and expensive, since it would employ a process participant though it does not require any particular human skill. Similar considerations hold for "Check raw materials availability", which is also an automated task. Another example is activity "Manufacture product". This is performed by an equipment (the manufacturing plant) which exposes its functionality via a service interface. So from the perspective of a BPMS, this is also an automated activity.

Continuing with our example, there are other tasks such as "Request raw materials from Supplier 1(2)" and "Get shipping address" that are devoted to sending and receiving messages. These are also examples of automated tasks. They can be implemented via automatic e-mail exchange or Web service invocation (and BPMSs typically provide these capabilities), despite they are not explicitly modeled inside a system lane. Recall that we are looking at a business-oriented process model, where it may not be relevant to model via lanes all existing systems (in this case an e-mail service or a Web service).

Other tasks like "Retrieve product from warehouse", "Obtain raw materials from Supplier 1(2)" and "Ship product" are manual. For example, "Retrieve product from warehouse" requires a warehouse worker to physically pick up the product from the shelf for shipping. In the presence of a manual task we have two options: (i) we isolate the task and focus on the automation of the process before and after it, or (ii) we find a way for the BPMS to be notified when the manual task has started or

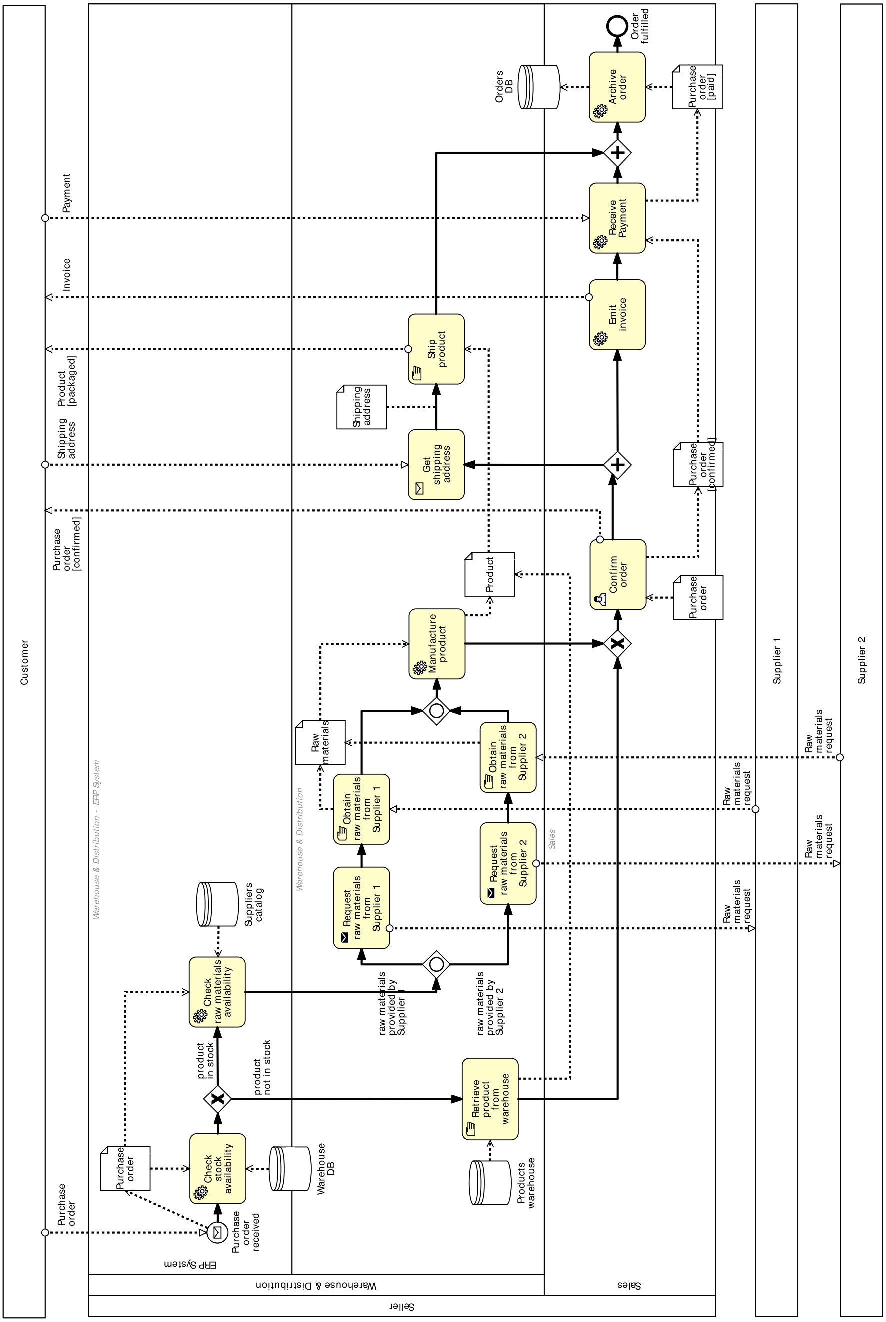


Fig. 9.6 The order fulfillment model that we want to automate

completed. We will get back to this point in the second step. For now, all we need to do is identifying these manual tasks.

"Confirm order" is an example of a user task: it requires somebody in sales (e.g. an order clerk) to verify the purchase order and then confirm that the order is correct. User tasks are typically scheduled through the worklist handler of the BPMS. In our example, an electronic form of the purchase order will be rendered on screen for the order clerk, who will verify that the order is in good state, confirm the order and thus submit the form back to the execution engine.

The distinction between automated, manual and user tasks is captured in BPMN via specific markers on the top-left corner of the task box. Manual tasks are marked with a hand while user tasks are marked with a user icon. Automated tasks are further classified into the following subtypes in BPMN:

- *script* (script marker), if the task executes some code (the script) internally to the BPMS. This task can be used when the functionality is simple and does not require access to an external application, e.g. opening a file or selecting the best quote from a number of suppliers.
- *service* (wheels marker), if the task is executed by an external application, which exposes its functionality via a service interface, e.g. "Check stock availability" in our example.
- *send* (filled envelope marker), if the task sends a message to an external service, e.g. "Request raw materials from Supplier 1".
- *receive* (empty envelope marker), if the task waits for a message from an external service, e.g. "Get shipping address".

These markers apply to tasks only. They cannot be used on sub-processes since a sub-process may contain tasks of different types. The relevant markers for our example are shown in Fig. 9.6.

Exercise 9.6 Assume you have to automate the loan assessment process model of Solution 3.7 for the loan provider. Start by classifying the tasks of this process into manual, automated and user ones and represent them with appropriate task markers.

9.4.2 Review Manual Tasks

Once we have identified the type of each task, we need to check whether we can link the manual tasks to the BPMS, so that we can maximize the value obtained by the BPMS. Alternatively, we need to isolate these tasks so that we can automate the rest of our process. There are two ways of linking a manual task to a BPMS: either we implement it via a user task or via an automated task.

If the participant involved in the manual task can notify the BPMS of the task completion using the worklist handler of the BPMS, the manual task can be turned into a user task. For example, the warehouse worker performing task "Retrieve product from warehouse" could check-out a work item of this task from their worklist to indicate that they are about to perform the job, manually retrieve the product from the shelf, and then check-in the work item back into the BPMS engine. Alternatively, check-out and check-in can be combined in a single step whereby the worker simply notifies the worklist handler that the job has been completed.

In some cases, a participant may use the aid of technology integrated with the BPMS to notify the engine of a work item completion. For example, the warehouse worker could use a barcode scanner to scan the barcode of the raw materials being obtained. The scanner would be connected to the BPMS so scanning the barcode would automatically signal the completion of "Obtain raw materials from Supplier 1(2)". In this case, the manual task can be implemented as a receive task awaiting the notification from the scanner, or by a user task handled by a worklist handler which in turn is connected to the scanner. If we use a receive task, the BPMS will only be aware of the work item's completion, in which case informing the warehouse worker that a new work item is available would be outside the scope of the BPMS. If we use a user task, the worker will be notified of the new work item by the BPMS, and will use the scanner to signal the work item's completion to the BPMS engine. Similar considerations hold for task "Ship order". Since each manual task of our example can be linked with a BPMS, this process can be automated as a whole.

Exercise 9.7 Consider the loan assessment model that you obtained in Exercise 9.6. Review the manual tasks of this model in order to link them to a BPMS.

There are cases in which it is not convenient to link manual tasks to a BPMS.

Example 9.1 Let us consider the university admission process described in Exercise 1.1, with the improvements discussed in Solution 1.5. The process until the point where the application is batched for the admissions committee (shown in Fig. 9.7a) can be automated. Once all the applications have been batched, the committee will meet and examine all of them at once. However, this part of the process (shown in Fig. 9.7b) is outside the scope of a BPMS. The tasks required for assessing applications cannot be automated because they involve various human participants who interact on an ad-hoc basis, and it would not be convenient to synchronize all these tasks with the BPMS. Eventually, the committee will draw a list of accepted candidates, transfer it to the admissions office, and a clerk at the admissions office will update the various student records, at which time the rest of the process can proceed within the scope of the BPMS (shown in Fig. 9.7c).

In this example we cannot automate the whole process. So we need to isolate activity "Assess application", an ad-hoc activity containing various manual tasks, and automate the process before and after this activity. An option is to split the model into three fragments as shown in Fig. 9.7 and only automate the first and the third fragment. Another option is to keep one model, and simply remove the ad-hoc activity. Some BPMSs are tolerant to the presence of manual tasks and ad-hoc activities in executable models, and will discard them at deployment time (like comments in a programming language). If this is the case, we can keep these elements in. Observe the use of the untyped event to start the third process model fragment in Fig. 9.7. In

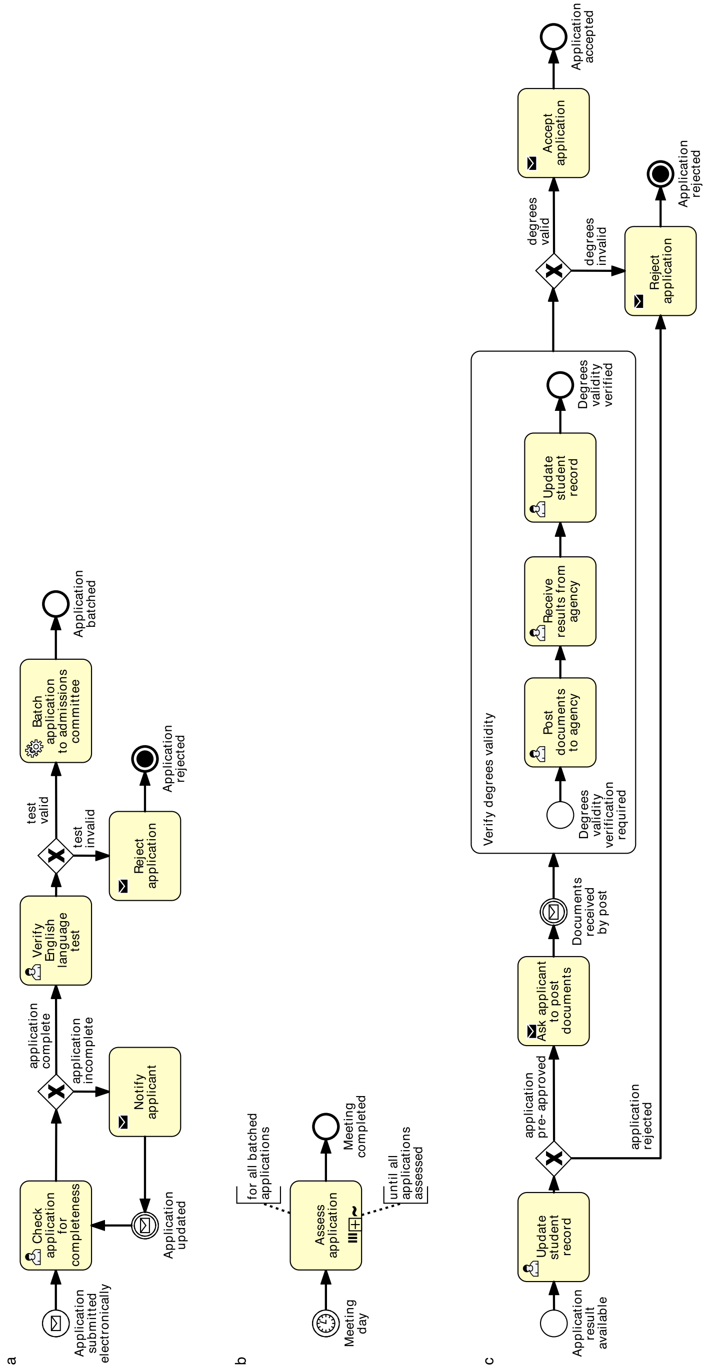


Fig. 9.7 Admission process: the initial (**a**) and final (**c**) assessments can be automated in a BPMS; the assessment by the committee (**b**) is a manual process outside the scope of the BPMS

BPMN, a process that starts with an untyped event indicates that instances of this process are explicitly started by a BPMS user, in our case a clerk at the admissions office. This process initiation is called *explicit instantiation* as opposed to *implicit instantiation* where process instances are triggered automatically by the event type indicated in the start event, e.g. an incoming message or a timer.

Finally, when the process is mostly or entirely made up of unordered manual tasks, automation does not provide any benefit, and is often even unfeasible. Take for example the post-production process in the screen industry. Activities like "Hold spotting session", where director and composer watch the picture to decide the music cues, "Conduct negative cutting", where the camera negative is manually cut and spliced to match the final edit specified by the film editor, and "Compose soundtrack", can hardly be coordinated by a BPMS, since there is no strict order for their execution and each of them may be repeated multiple times. As a rule of thumb, a process whose tasks are performed in an ad-hoc manner, without any predictable order, is not suitable for automation via a production BPMS. In this case, a case handling or ad-hoc workflow system would be more appropriate.

Exercise 9.8 Consider the final part of the prescription fulfillment process described in Exercise 1.6:

> Once the prescription passes the insurance check, it is assigned to a technician who collects the drugs from the shelves and puts them in a bag with the prescription stapled to it. After the technician has filled a given prescription, the bag is passed to the pharmacist who double-checks that the prescription has been filled correctly. After this quality check, the pharmacist seals the bag and puts it in the pick-up area. When a customer arrives to pick up their prescription, a technician retrieves the prescription and asks the customer for their co-payment or for the full payment in case the drugs in the prescription are not covered by the customer's insurance policy.

One way of modeling this fragment is by defining the following tasks: "Check insurance", "Collect drugs from shelves", "Check quality", "Collect payment" (triggered by the arrival of the customer), and finally "Retrieve prescription bag". Assume the pharmacy system automates the prescription fulfillment process. Identify the type of each task and if there are any manual tasks, specify how these can be linked to the pharmacy system.

There are other modeling elements, besides manual tasks, that are relevant at a conceptual level but cannot be interpreted by a BPMS. These are physical data objects and data stores, messages bearing physical objects and text annotations. Pools and lanes are also used at a conceptual level only. In fact, as we have seen, pools and lanes are often used to capture coarse-grained resource assignments, e.g. activity "Confirm order" is done within the sales department. When it comes to execution, we need to define resource assignments for each task and capturing this information via dedicated lanes (potentially one for each task) will just make the diagram too cluttered. Electronic data stores are also not directly interpreted by a BPMS, as the BPMS assumes the existence of dedicated services that can access these data stores, e.g. an inventory information service that can access the warehouse DB. So the BPMS will interface with these services rather than directly with the data stores. Also, the state of a data object indicated in the object's label, e.g. "Purchase order [confirmed]", cannot be interpreted as such by a BPMS. Later we will show how to explicitly represent object states so that they can be interpreted by a BPMS.

Some BPMSs tolerate the presence of these non-executable elements too. If this is the case, we suggest to leave these elements in. Especially pools, lanes, message flows bearing electronic objects, electronic data stores and annotations will guide us in the specification of some execution properties. For example, the Sales lane in the order fulfillment model tells us that the participant to be assigned task "Confirm order" has to be from the sales department. Other BPMS modeling tools do not support these elements, so it is not even possible to represent them in the diagram.

Exercise 9.9 Consider the loan assessment model that you obtained in Exercise 9.7. Identify the modeling elements that cannot be interpreted by a BPMS.

9.4.3 Complete the Process Model

Once we have established the automation boundaries of the process and reviewed manual tasks, we need to check that our process model is *complete*. Often business-oriented process models neglect certain information because modelers deem it is not relevant for the specific modeling purpose, they assume it is common knowledge, or simply, they are not aware of it. It may be fine to neglect this information in a business-oriented model, depending on the application scenario. However, information that is not relevant in a business-oriented model may be highly relevant for a process model to be executed.

A typical example is when the process model focuses on the "sunny-day" scenario and neglects all negative situations that may arise during the execution of the process, working under the assumption that everything will work well. As we saw in Chap. 4, the order fulfillment process is indeed subjected to a number of exceptions. For example, it may be aborted if the materials required to manufacture the product are not available at the suppliers or if the customer sends an order cancellation. So we need to make sure that all exceptions are handled using appropriate exception handlers. For example, if the order cancellation is received after the product has been shipped or after the payment has been received, we also have to compensate for these activities by returning the product and reimbursing the customer. Another exception that is commonly neglected is that representing an activity that cannot complete. What happens if the customer's address is never received? Or if the ERP module for checking the stock availability does not respond? We cannot assume that the other party will always respond or that a system will always be functional. Similarly, we cannot assume that activities always lead to a positive outcome. For example, an order may not always be confirmed.

You may be surprised how rarely exceptions are modeled in a business-oriented process in practice. Thus, in the majority of cases, such a model will require to be completed with these aspects before being executed.

In this step, we also need to specify all *electronic data objects* that are required as input and output by the tasks of our process. For instance, in Fig. 9.6 there is no input data object to task "Request raw materials from Supplier 1(2)", though this task does need the list of raw materials to be ordered. Another example is task "Check stock availability". This task uses the Purchase order as input (to obtain the code of the product to be looked up in the Warehouse DB) but does not produce any output data to store the results of the search. However, without this information, the subsequent XOR-split cannot determine which branch to take (we can now see why this is called *data-based XOR-split*). If you have not noticed the absence of these data objects so far, it is probably because you assumed their existence. This is fine in a business-oriented model where only aspects relevant to the specific modeling purpose are documented, but not in an executable model, where an engine has to run the model. So, make sure each activity has the required input and output electronic data objects. The principle is that every data object needed by the BPMS engine to pass control between activities and to take decisions must be modeled.

The completed order fulfillment example, including exception handlers and data objects that are relevant for execution, is shown in Fig. 9.8.[1]

Exercise 9.10 Take the loan assessment model that you obtained in Exercise 9.6 after incorporating the revisions from Exercise 9.7. Complete this model with control-flow and data-flow aspects relevant for automation. For simplicity, you may disregard the modeling elements that are not interpretable by a BPMS.

9.4.4 Bring the Process Model to an Adequate Granularity Level

There is not necessarily a one-to-one mapping between the tasks in a business-oriented model and those in the corresponding executable model. Indeed, we should keep in mind that a BPMS is intended to coordinate and manage handovers of work between multiple resources (human or non-human). Accordingly, two or more consecutive tasks assigned to the same resource are candidates for *aggregation*. If this was the case, the BPMS would not add value between these two tasks because it would not manage any handover. All it would do is to interfere in the work of a given resource. For example, a sequence of user tasks "Enter customer name", "Enter customer policy number" and "Enter damage details", such that all three tasks will be performed by the same claims handler, should be aggregated into a single user task "Enter claim".

[1]The content of the sub-processes and some of the elements that cannot be interpreted by a BPMS have been omitted for simplicity.

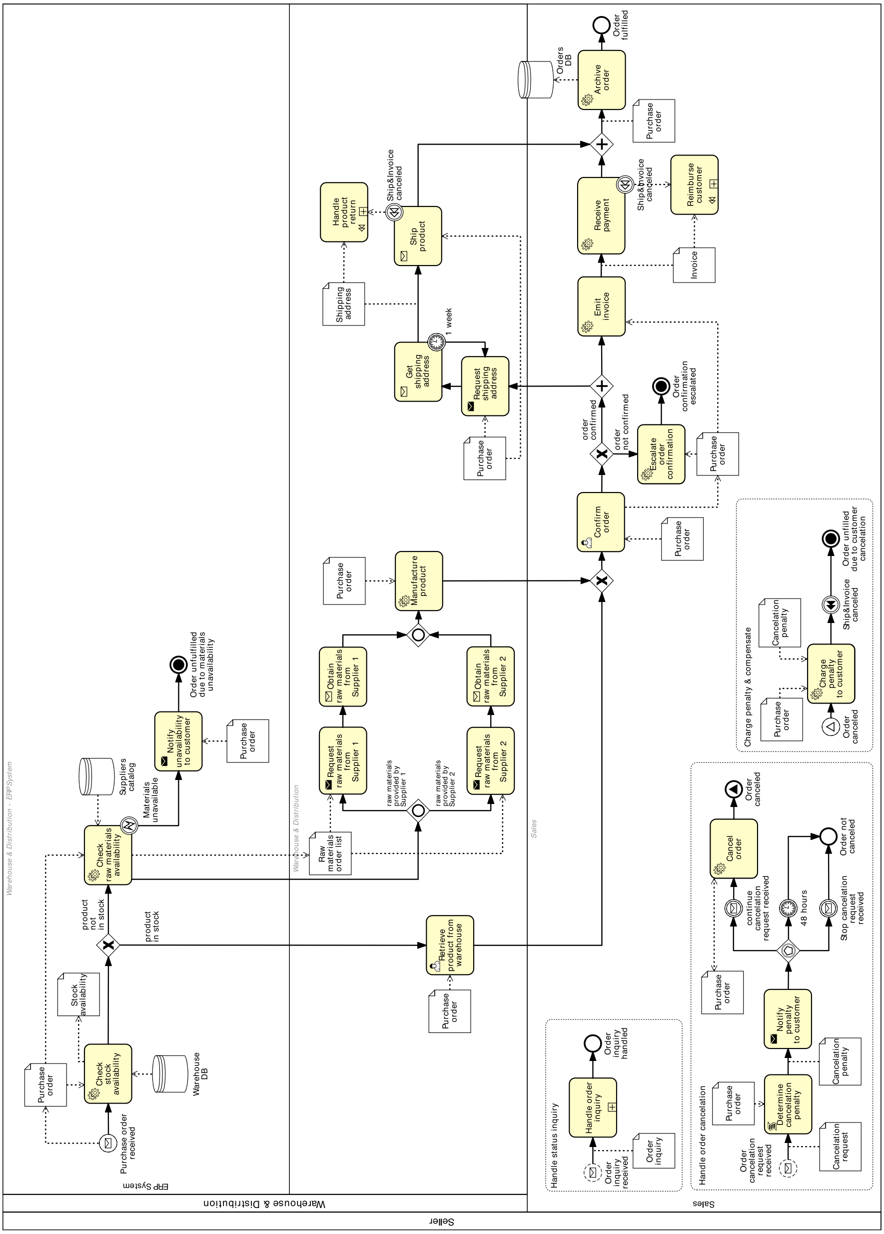


Fig. 9.8 The order fulfillment model of Fig. 9.6, completed with control-flow and data-flow aspects relevant for automation

There are some cases, though, where we may actually need to keep consecutive tasks separate, despite they are performed by the same resource. For example, in

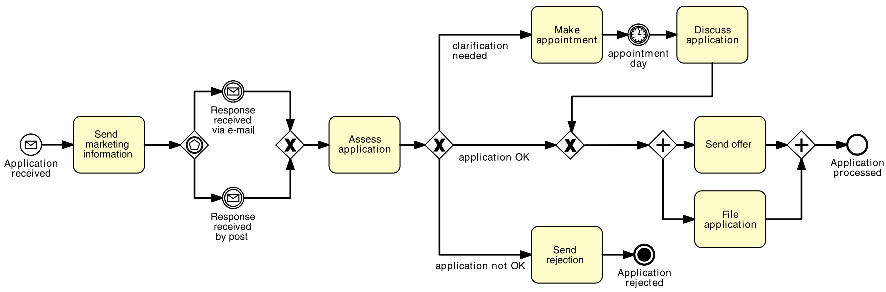


Fig. 9.9 The sales process of a B2B service provider

Fig. 9.7c we have three user tasks within sub-process "Verify degrees validity": "Post documents to agency", "Receive results from agency" and "Update student record". While these may be performed by the same admin clerk, we do want to keep track of when each task has been completed, for the sake of monitoring the progress of the application and manage potential exceptions. For example, if the results are not received within a given timeframe, we can handle this delay by adding an exception handler to task "Receive results from agency".

Exercise 9.11 Are there activities that can be aggregated in the model obtained in Exercise 9.10? Hint: candidate tasks for aggregation may not necessarily be consecutive due to a sub-optimal order of tasks in the business-oriented model. In this case, you need to resequence the tasks first (see Sect. 8.2.3).

In a similar vein, if an activity requires more than one resource to be performed, it is too *coarse-grained*, so we should disaggregate it into more fine-grained tasks such that these can be assigned to different resources. For example, an activity "Enter and approve money transfer" is likely to be performed by two different participants even if they have the same role, in order to enforce separation of duties: first a financial officer enters the order, then a different financial officer approves it.

Exercise 9.12 Figure 9.9 shows the model for the sales process of a business-to-business (B2B) service provider. The process starts when an application is received from a potential client. The client is then sent information about the available services and a response is awaited either via e-mail or postal mail. When the response is received, the next action is decided upon. Either an appointment can be made with the client to discuss the service options in person, or the application is accepted or rejected right away. If the application is accepted, an offer is sent to the client and at the same time the application is filed. If it is rejected, the client is sent a thank-you note and let go. If an appointment has to be made, this is done and at the time of the appointment, the application is discussed with the client. Then the process continues as if the application had been accepted right away.

1. Identify the type of each task and find ways of linking the manual tasks to a BPMS.

2. Remove elements that cannot be interpreted by a BPMS.
3. Complete the model with control-flow and data aspects required for execution.
4. Bring the resulting model to a granularity level that is adequate for execution.

Acknowledgement This exercise is adapted from a similar exercise developed by Remco Dijkman, Eindhoven University of Technology.

9.4.5 Specify Execution Properties

In the last step, we need to specify *how* each model element is effectively implemented by the BPMS. For example, take the first service task of our revised order fulfillment example: "Check stock availability". Saying this task requires the purchase order as input to contact the warehouse ERP system is not enough. We need to specify the service provided by the ERP system to check stock levels and its location in the network, the product information in the purchase order that is required by this service (i.e. the format of the input object) and the information produced by the service (i.e. the format of the output object). These implementation details are called *execution properties*. More, specifically, these are:

- Process variables, messages, signals and errors
- Task and event variables and their mappings to process variables
- Service details for service, send and receive tasks, and for message and signal events
- Code snippets for script tasks
- Participant assignment rules and user interface structure for user tasks
- Task, event and sequence flow expressions
- BPMS-specific properties

These properties do not have a graphical representation in the BPMN diagram, but are stored in the BPMN 2.0 interchange format. The BPMN interchange format is a textual representation of a BPMN model in XML format. It is intended to support the interchange of BPMN models between tools and also to serve as input to a BPMN execution engine. BPMN modeling tools provide a visual interface to edit most of these non-graphical properties, so most of the times you will not need to write the XML directly. Still, you will need to understand standard Web technology, especially XML, XML Schema (XSD), and be familiar with the notion of (Web) service, to be able to implement a process model. This section assumes that you have basic knowledge of these technologies. We provide pointers to further readings on these technologies in Sect. 9.8.

Figure 9.10 shows the structure of the BPMN format. It consists of a list of elements, where some are optional (those with a dashed border) and others are mandatory (those with solid borders). The process element is mandatory and stores information about the process model. This consists of electronic data objects, events, tasks and flows. The elements outside the process are reusable components needed

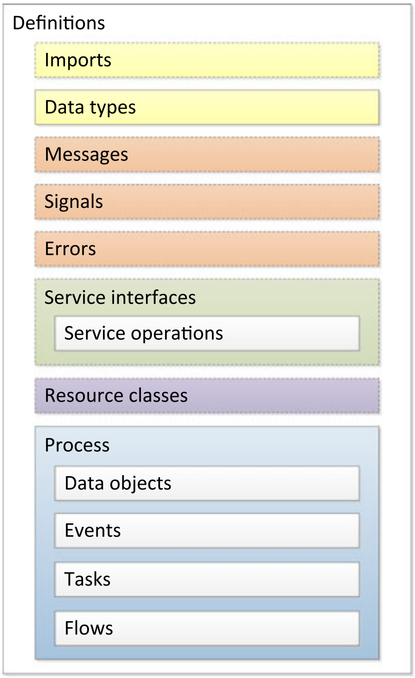


Fig. 9.10 Structure of the BPMN format

by the various process elements, like message definitions and service interfaces which are used by service, send and receive tasks, and by message and signal events. With reference to this structure, let us now go through each of the execution properties above.

Process Variables, Messages, Signals and Errors Process variables are managed by the BPMS engine to allow data exchange between process elements. Each electronic data object, e.g. the purchase order in the order fulfillment process, represents a process variable. The lifetime of a process variable is confined to the life of the process instance in which the variable is created, and is only visible to the process level in which it is defined and to all its sub-processes. This means that a variable defined in a sub-process is not visible in the parent process.

We need to assign a *data type* to each process variable in order for a BPMS to be able to interpret and manipulate these variables. In BPMN, the type of each process variable is specified as an XSD type. The type of a variable can be *simple* or *complex*. Simple types are strings, integers, doubles (numbers containing decimals), booleans, dates, times, etc., and are already defined in the XSD specification. For example, the object Stock availability can be represented as a process variable of type integer (representing the number of available units of a product). Complex types are hierarchical compositions of other types. A complex type can be used for example to represent a business document, such as a purchase order or an invoice. Figure 9.11a shows the complex type of the purchase order, called purchaseOrderType, while a)

```
<complexType name="purchaseOrderType">
  <sequence>
    <element name="order">
      <complexType>
        <sequence>
          <element name="orderNumber" type="integer"/>
          <element name="orderDate" type="date"/>
          <element name="status" type="string"/>
          <element name="currency" type="string"/>
          <element name="productCode" type="string"/>
          <element name="quantity" type="integer"/>
        </sequence>
      </complexType>
    </element>
    <element name="customer">
      <complexType>
        <element name="name" type="string"/>
        <element name="surname" type="string"/>
        <element name="address">
          <complexType>
            <sequence>
              <element name="street" type="string"/>
              <element name="city" type="string"/>
              <element name="state" type="string"/>
              <element name="postCode" type="string"/>
              <element name="country" type="string"/>
            </sequence>
          </complexType>
        </element>
        <element name="phone" type="string"/>
        <element name="fax" type="string"/>
      </complexType>
    </element>
  </sequence>
</complexType>
```

b)

```
<purchaseOrder>
  <order>
    <orderNumber>15664</orderNumber>
    <orderDate>2012-10-23</orderDate>
    <status>confirmed</status>
    <currency>EUR</currency>
    <productCode>345-EAR</productCode>
    <quantity>10</quantity>
  </order>
  <customer>
    <name>John</name>
    <surname>Brown</surname>
    <address>
      <street>8 George St</street>
      <city>Brisbane</city>
      <state>Queensland</state>
      <postCode>4000</postCode>
      <country>Australia</country>
    </address>
    <phone>+61 7 3240 0010</phone>
    <fax>+61 7 3221 0412</fax>
  </customer>
</purchaseOrder>
```

Fig. 9.11 The XSD describing the purchase order (**a**) and one of its instances (**b**)

Fig. 9.11b is the XML representation of a particular purchase order instance at run-time. From the type definition we can see that a purchase order contains a sequence of two elements:

- Order, to store the order information (order number, order date, status, currency, product code and quantity), and
- Customer, to store the customer information (name, surname, address, phone and fax)

The data fields order, customer and address are complex types so that they can contain sub-elements. Also, observe the field status within order: this is used to capture the state of the purchase order, e.g. "confirmed".

Similar to process variables, we also need to assign data types to each message, signal and error used in the process model. For the messages, we can look at the existing message flows in the diagram and define one data type for each uniquely labeled message flow. So for example if we have two message flows labeled purchase order, they will obviously take the same type purchaseOrderType. If message flows are not modeled, we can look at the send, receive and service tasks, and at the message events present in the diagram in order to understand what messages to define. For signals and errors we have to look at the signal and error events that we have defined in the diagram. While for a signal, the data type describes the content of the signal being broadcasted or listened to, for an error the data type defines what information is carried with the error. For example, if it is a system error, we can use this to specify the error message returned by the system. In addition, we need to assign an *error code* to each error. This code uniquely identifies an error within the process model, so that a catching error event can be related to a throwing error event.

Task and Event Variables Besides the above data elements, we need to define the internal variables of each task, called *data inputs* and *data outputs* in BPMN. Data inputs and outputs act as interfaces between a task and its input and output data objects. They also need to refer to an XSD type defining their structure, but different from process variables, they are only visible within the task (or sub-process) in which they are defined. Data inputs capture data that is required by the task to be executed; data outputs capture data that is produced by the task upon completion. Thus, data inputs are populated with the content of input data objects while data outputs are used to populate the content of output data objects. For example, we need a data input for task "Check stock availability" in order to store the content of the purchase order. Thus, the type of this data input must match that of the input object, i.e. purchaseOrderType. Similarly, the data output must be of type integer to store the number of items in stock, so that this information can be copied into stock availability upon task completion.

The mapping between data objects and task data inputs/outputs is defined via the task *data associations*. Data associations can also be used to define complex data assignments beyond one-to-one mappings. For example, consider task "Manufacture product". The service invoked by this task only requires the order product code and quantity in order to start the manufacturing of the product. Thus, we can use a data association to extract the product code and quantity from the input purchase order, and populate a data input containing two sub-elements of types string, respectively, integer. In most cases, the BPMS will automatically create all the tedious data mappings between data objects and tasks. For example, for the case above all we need to do is to select the sub-elements of the purchase order we want to use as input to "Manufacture product", and the BPMS will create the required data inputs and their mappings for this task. BPMN relies on XPATH 1.0 as the default language for expressing data assignments like the one above. However, other languages can be used like Java Universal Expression Language (UEL) or Groovy. The choice depends on the BPMS adopted. For example, Activiti supports UEL, Bonita Open Solution and Camunda Fox support Groovy while BizAgi's BPM Suite supports its own expression language.

Similar to tasks, events that transmit or receive data, i.e. message, signal and error events, also have internal variables. Specifically, the catching version of these events has one data output only, to store the content of the event being caught (e.g. an incoming message), whereas the throwing version has one data input only, to store the content of the event being thrown (e.g. an error). Thus, we also need to assign these data inputs and outputs a type that has to match that of the message, signal or error associated with the event. For example, the start catching message event "Purchase order received" in the order fulfillment example uses a data output to store the purchase order message once this has been received. Thus, this data output must match the type of the incoming message, which is precisely purchaseOrderType. In turn, the output object must have the same type as the output data, to contain the purchase order.

The complex types for all data elements of the process can be defined directly in the BPMN model or imported from an external document (see Fig. 9.10).

Service Tasks Once we have defined the types of all data elements, and mapped task and event data inputs and outputs to these types, we have to specify how tasks and events have to be implemented. For service tasks we need to specify how to communicate with the external application that will execute the task. Be it a complex system or a simple application, from the perspective of the BPMS all that is required is that the external application provides a *service interface* that the service task can use. A service interface contains one or more *service operations*, each describing a particular way of interacting with a given service. For example, a service for retrieving inventory information provides two operations: one to check the current stock levels and one to check the stock forecast for a given product (based on product code or name). An operation can either be *in-out* or *in-only*. In an in-out operation (also called *synchronous* operation), the service expects a request message and replies with a response message once the operation has been completed, or optionally with an error message if something goes wrong. For example, the service invoked by task "Check raw materials availability" receives stock availability information as input message and replies with a list of raw materials to be ordered as output message. Alternatively, if the service experiences an exception (e.g. the suppliers catalog is unreachable), it replies with an error message which triggers the boundary error event of this task so that the relative exception handler can be performed.[2] Conversely, in an in-only operation (also called *asynchronous* operation), the service expects a request message but will not reply with a response message. For example, task "Archive order" notifies an archival service of the purchase order to be archived, however, the process does not wait for an archival confirmation.

Each message of a service operation needs to reference a message in the BPMN model, so that it can be assigned a data type. For instance, the request and the response messages to interact with the inventory service have data type purchaseOrderType, respectively, XSD integer. For each interface, we also need to specify how this is concretely implemented, i.e. what communication protocols are used by

[2] Note that there is no throwing end error event inside "Check raw materials availability" since the catching error event is triggered by the receipt of an error message by the service task. The ability to link error messages with error events is a common feature of BPMSs.

the service and where the service is located in the network. By default, BPMN uses Web service technology to implement service interfaces, and relies on WSDL 2.0 to specify this information. In practice, this corresponds to defining one or more external WSDL documents and importing them into our BPMN model. Once again, other implementations are possible, e.g. one could implement a service interface via Java remote procedure call or plain XML over HTTP.

After defining the service interfaces for our process, we need to associate each service task with a service operation defined in a service interface. Based on the type of the operation (in-out or in-only), we then need to define a single data input that must match the type of the request message in the referenced service operation, and optionally a single data output that must match the type of the response message in the operation. The BPMS engine will copy the task data input to the request message and send it out to the service, and once the response message has been received, will copy the content of this message to the task data output.

Send and Receive Tasks, Message and Signal Events Send and receive tasks work similarly. A send task is a special case of the service task: it sends a message to an external service using its data input, but there is no response. An example is task "Notify unavailability to customer". A receive task waits for an incoming message and uses its data output to store the message content. Task "Get shipping address" is an example of this. Both task types need to reference an in-only service operation where the message is defined. However, for the receive, the message being received is seen as a request coming from an external service requester. Thus, in this case the process itself acts as the service provider.

A receive task can also be used to receive the response of an asynchronous service which has previously been invoked with a send task. This is the case of tasks "Request shipping address" and "Get shipping address". The asynchronous service is provided by the customer. Accordingly, in the send task the seller's process acts as the service requester sending a request message to the customer. In the receive task the roles get swapped: the seller acts as the service provider to receive the response message from the customer. This pattern is used for long-running interactions, where the response may arrive after a while. The drawback of using a synchronous service task in place of a send-receive is that this task would block the process to wait for the response message. This is not the case in Fig. 9.8, where the send and receive tasks are in parallel to "Emit invoice" which may thus be performed in-between.

Message and signal events work exactly like send and receive tasks. For signal events, it is assumed that the service being contacted has publish-subscribe capabilities, e.g. a Web service for subscribing to RSS feeds.

Script Tasks For script tasks, we need to provide the snippet of code that will be executed by the BPMS. This code can be written in a programming language such as JavaScript or Groovy. BPMN does not prescribe the use of a specific programming language so the choice depends on the BPMS used. The task data inputs store the parameters for invoking the script while the data outputs store the results of executing the script. For example, for task "Determine cancellation penalty" we can define a script that extracts the order date and the cancellation request date from two data inputs mapped to the input objects purchase order and cancellation request, uses this information to compute a penalty of €15 for each day past the order date, and copies this value to the data output.

User Tasks For each user task we need to specify the rules for assigning work items of this task to process participants at runtime, the technology to communicate with participants and the details of the user interface to use. Moreover, like for any other task, we need to define data inputs to pass information to the participant, and data outputs to receive the results.

Process participants that can be assigned user tasks are called *potential owners* in BPMN. A potential owner is a member of a resource class. In the context of user tasks, a resource class identifies a static list of *participants* sharing certain characteristics, e.g. holding the same role or belonging to the same department or unit. An example of resource class for the order fulfillment process is order clerk, which groups all participants holding this role within the sales department of the seller organization. Note that these resource classes are unrelated to pools and lanes, which are only notational elements in a business-oriented process model. A resource class can be further characterized by one or more *resource parameters*, where a parameter has a name and a data type. For example, we can define two parameters product and region of type string to indicate the particular products an order clerk works with, and the region they work in.

Once we have defined all required resource classes and optionally their parameters, we can assign each user task to one or more resource classes based on an expression. For example, we can express that work items of task "Confirm order" have to be assigned to all participants of type Order clerk who deal with the particular product being ordered and work in the same region as the customer. For this, we can define an XPATH expression that selects all members of Order clerk whose properties product and region are equal to the product code, respectively, country contained in the purchase order.

We also need to specify the implementation technology used to offer the work item to the selected participant(s). This entails aspects such as how to reach the participant (e.g. via email or worklist notification), how to render the content of the task data inputs on screen (e.g. via one or more web forms organized through a particular screenflow), and the strategy to assign the work item to a single participant out of those satisfying the assignment expression (e.g. assign it to the order clerk with the shortest queue or randomly). The configuration of these aspects, as well as the association of participants to resource classes is dependent on the specific BPMS being used.

Task, Event and Sequence Flow Expressions Finally, we need to write expressions for the various attributes of tasks and events, and for the sequence flows bearing conditions. For instance, in a loop task we need to write a boolean expression that implements the textual annotation indicating the loop condition (e.g. "until response approved"). This boolean expression will determine when will the loop task be repeated. This expression can be defined over data elements, e.g. it can be an XPATH expression that extracts the value of the boolean element "approved" from a Response object. We can also use *instance attributes* inside these expressions. These are variables that vary by instance at execution. An example is *loop count*, which counts the number of iterations for a loop task. For the timer event we need to specify an expression to capture the temporal event informally expressed by its label (e.g. "Friday afternoon"). Here we have three options: we can either provide a temporal expression in the form of a precise date or time, a relative duration, or a repeating interval. Once again, these expressions can be linked to data elements and instance properties so as to be resolved dynamically at execution. For example, we can set an order confirmation timeout based on the number of line items in an order. Finally, we need to write a boolean expression to capture the condition attached to each sequence flow following an (X)OR-split. For example, condition "product in stock" after the first XOR-split in the order fulfillment example can be implemented as an XPATH expression that checks whether the value of variable stock availability is at least equal to the product quantity contained in the purchase order. There is no need to assign an expression to a default sequence flow, since this arc will be taken by the BPMS engine if the expressions assigned to all other arcs emanating from the same (X)OR-split evaluate to false.

BPMS-Specific Properties Strictly speaking, the only BPMS-specific properties that we have to configure in order to make a process model executable are those of user tasks. In practice, however, we will likely need to link our executable process with the enterprise system of our organization. This is called *system binding*. Luckily, BPMSs offer a range of predefined service task extensions, called *service adapters*service adapter (or *service connectors*), to implement common system binding functions in a convenient way. Examples of such binding functions include: performing a database lookup, sending an email notification, posting a message to Twitter or setting an event in Google Calendar, reading or writing a file and adding a customer in a CRM system. Each adapter comes with a list of parameters that we need to configure. However, BPMSs provide wizards with capabilities to auto-discover some of the parameter values. For instance, to use a database lookup we need to provide the type of the database server (e.g. MySQL, Oracle DB) and the URL where the server can be reached, the schema to be accessed, the SQL query to run and the credentials of the user authorized to run the query.

Coming back to our example, instead of implementing task "Check stock availability" as a service task, which assumes the existence of an inventory information service at the seller, we could implement this task with a generic database lookup adapter, provided we know what to search for and where. Similarly, we could implement the tasks for communicating with the customer like "Notify unavailability to customer" and "Request shipping address" as email adapters, so that we do not need to implement a dedicated email service in our organization. The number and variety of adapters that a BPMS provides largely contribute to increasing the value of the product over competing solutions.

Exercise 9.13 Consider the loan assessment process model that you obtained in Exercise 9.11. The loan application contains these data fields:

- Applicant information:
  - Identity information (name, surname, ...)
  - Contact information (home phone, cell phone, ...)
  - Current address (street name and number, city, ...)
  - Previous address (as above plus duration of stay)
  - Financial information (job details, bank details)
- Reference information (identity, contact, address, relation to applicant)
- Property information (property type, address, purchasing price)
- Loan information (amount, number of years, start date, interest type: variable/fixed)
- Application identifier
- Submission date and time
- Revision date and time
- Administration information (a section to be compiled by the loan provider):
  - Status (a string to keep track of the state of the application, with predefined values: "incomplete", "complete", "assessed", "rejected", "canceled", "approved")
  - Comments on status (optional, e.g. used to explain the reasons for rejection)
  - Eligibility (a boolean to store whether or not the applicant is eligible for a loan)
  - Loan officer identifier
  - Insurance quote required (a boolean to store whether or not a home insurance quote is sought)

The credit history report contains these data fields:

- Report identifier
- Financial officer identifier
- Reference to a loan application
- Applicant's credit information:
  - Loan applications made in the last five years (loan type: household/personal/domestic, amount, duration, interest rate)
  - Overdue credit accounts (credit type, default amount, duration, interest rate)
  - Current credit card information (provider: Visa, Mastercard, ..., start date, end date, interest rate)
  - Public record information (optional, if any):
    - Court judgments information
    - Bankruptcy information
- Credit assessment (a string with predefined values: AAA, AA, A, BBB, BB, B unrated).

The risk assessment contains the following data fields:

- Assessment identifier
- Reference to a loan application
- Reference to a credit history report

- Risk weight (an integer from 0 to 100)

The property appraisal contains the following data fields:

- Appraisal identifier
- Reference to a loan application
- Property appraiser identifier
- Property information (property type, address)
- Value of three surrounding properties with similar characteristics
- Estimated property market value
- Comments on property (optional, to note serious flaws the property may have)

The agreement summary contains the following data fields:

- Reference to a loan application
- Conditions agreed (a boolean indicating if the applicant agreed with the loan conditions)
- Repayment agreed (a boolean indicating if the applicant agreed with the repayment schedule)
- Link to digitized copy of the repayment agreement

The loan provider offers a website where applicants can submit and revise loan applications online, track the progress of their applications and if required, cancel applications in progress. This website implements an underlying Web service with which the loan assessment process interacts. In practice, this service acts as the applicant from the perspective of the loan assessment process. For example, if the applicant submits a new loan application through the website, this service wraps this application into a message and sends it to the BPMS engine of the loan provider, which in turn starts a new instance of the loan assessment process. If the loan assessment process sends an application for review to this service, the service presents this information to the applicant via the loan provider's website.

Further, the loan assessment process interacts with an internal service for assessing loan risks. This service determines a risk weight which is proportional to the credit assessment contained in the credit history report, on the basis of the applicable risk rules read from a database (the interaction between the service and the database is transparent to the BPMS). The risk assessment service returns a risk assessment containing an identifier (freshly generated), a reference to the loan application and one to the credit history report (both extracted from the credit history report), and the risk weight.

Based on the above information, specify the execution properties for the elements of this process model. You are not required to define the actual XSD type of each data element, nor to specify the actual Groovy scripts or XPATH expressions. All you need to do is to identify what properties have to be specified, i.e. what data inputs and outputs, service interfaces, operations, messages and errors are required, and determine their data type in relation to that of process variables. For example, a data input may map to a process variable or to a data field within this. For scripts, you need to define via task data inputs and outputs what data is required by the script, what data is produced and how the input data is transformed into the output one. For example, based on the value of a data field in a process variable, a script may write a particular value in the data field of another process variable. Similarly, for each user task, you need to identify what information is presented to the task performer, and how the data output is obtained. Finally, you need to explain how each expression can be evaluated on the basis of data fields within process variables (e.g. to implement the condition of a sequence flow), or constant values like a date (e.g. to implement a timer event).

9.4.6 The Last Mile

Now that you have become familiar with what is required to turn a process model executable, the last step for you is to take a process model and implement it using a BPMS of your choice (e.g. Activiti, Bonita Open Solution, Bizagi's BPM Suite, YAWL). The landscape of BPMSs and their specificities evolves continuously. We can identify three categories of BPMSs with respect to their support for BPMN:

1. **Pure BPMN** These tools have been designed from the ground up to support BPMN natively. They follow the specification "to the letter" though they might not fully support it. Examples are Activiti and Camunda Fox.
2. **Adapted BPMN** These tools use a BPMN skin but rely on an internal representation to execute the process model. They can import and sometimes also export in the BPMN 2.0 format. They typically predate BPMN and evolved from previous versions to support the specification. Examples are Bizagi's BPM Suite and Bonita Open Solution.
3. **Non BPMN** There is finally a general category of BPMSs which use their own proprietary language and semantics. These tools do not support BPMN. Examples are BPMOne from Perceptive Software and YAWL.

At the time of writing, most of the BPMSs supporting BPMN in one way or another, still do not cover all aspects of the specification that are relevant for execution. For example, elements like message boundary events, compensation events and non-interrupting events are hardly supported. So concretely we have to give up on one or more of these elements depending of the BPMS that we adopt. One expects though that support for executable BPMN will increase over time.

This section illustrated how to design executable BPMN models in a vendor-independent manner. The book's website (http://fundamentals-of-bpm.org) provides tutorial notes showing how to configure an executable process model for concrete BPMSs.

Exercise 9.14 Based on the execution properties that you specified in Exercise 9.13, implement the loan assessment process using a BPMS of your choice.

9.5 Recap

In this chapter we focused on a specific type of process-aware information system, namely Business Process Management Systems (BPMSs). We discussed the architecture of a BPMS and its main components: the execution engine, the process modeling tool and the process model repository, the administration and monitoring tools and the execution logs, as well as the external services that can be invoked.

There are many reasons for considering process automation. First, it provides workload reduction in terms of coordination: work is assigned to process participants or software services as soon as it is available. Second, it offers integration flexibility. Processes can be changed with significantly less effort as compared to legacy systems, provided they are explicitly represented via process models. Third, the execution in a BPMS generates valuable data on how processes are executed, including performance-relevant data. Finally, BPMSs improve the quality of process execution as they directly enforce rules such as separation of duties.

Introducing BPMSs poses various challenges. Technical challenges arise from the fact that many applications that have to be integrated are typically not designed as open systems with transparent interfaces. Beyond that, organizational challenges are rooted in the fact that BPMSs directly interfere with how people do their job. This fact calls for sensitive change management.

Finally, we presented a method for transforming business-oriented process models into executable specifications, so that they can be interpreted by a BPMS. First, we need to identify the type of each process task (automated, manual or user) and review manual tasks to find, whenever it is possible, a way to link these to the BPMS. Next, we need to complete the process model by specifying all control-flow and data aspects that are relevant for execution and bridge the diverging level of granularity between a business-oriented process model and its executable counterpart. Finally, we need to specify a number of execution properties for each model element. Some of these properties, like for user tasks, are vendor-specific and so will vary depending on the specific BPMS that we decide to adopt.

9.6 Solutions to Exercises

Solution 9.1 There are three current work items:

1. Case #1,220: Determine Proper Piece of Equipment
2. Case #1,230: Determine Proper Piece of Equipment
3. Case #1,240: Complete Equipment Rental Request

Solution 9.2

- The execution engine would be unable to determine to allocate work items to resources on the basis of a process model alone, when it would only cover control-flow information.

- One common situation would be that the process model in question specifies that after a certain activity there is a parallel split, enabling the execution of various follow-up activities.
- Other examples of services that can be useful to be invoked: calculation services (e.g. to determine a mortgage rate or to estimate the total cost of a service), information storage and retrieval services (e.g. to register the outcome of a completed work item or to look up client information), scheduling services (e.g. to plan work that is to be done in follow-up or to estimate a delivery date), communication services (e.g. to get in touch with the client or a business partner), etc.
- The descriptions of the business processes and the available kinds of resource should exactly be the same to allow for passing on work items among different BPMSs. Otherwise, it may become impossible to map the availability of a work item within a process in one time zone to a particular state of that process in another time zone.
- Most notably, it would be important to specify on which working days particular resources are available and during which hours, e.g. Ms. Withagen only works on Mondays and Fridays from 9 AM to 4 PM. In this way, it becomes possible for the execution engine to allocate work items in an efficient manner.

Solution 9.3

1. It should become possible that the new decision support system can be invoked as an external service.
2. If Ms. Withagen retires, this must be recorded with the administration tool.
3. The new rule to allocate work items to resources must be captured in an updated process model.
4. The monitoring service must be implemented in a monitoring tool.

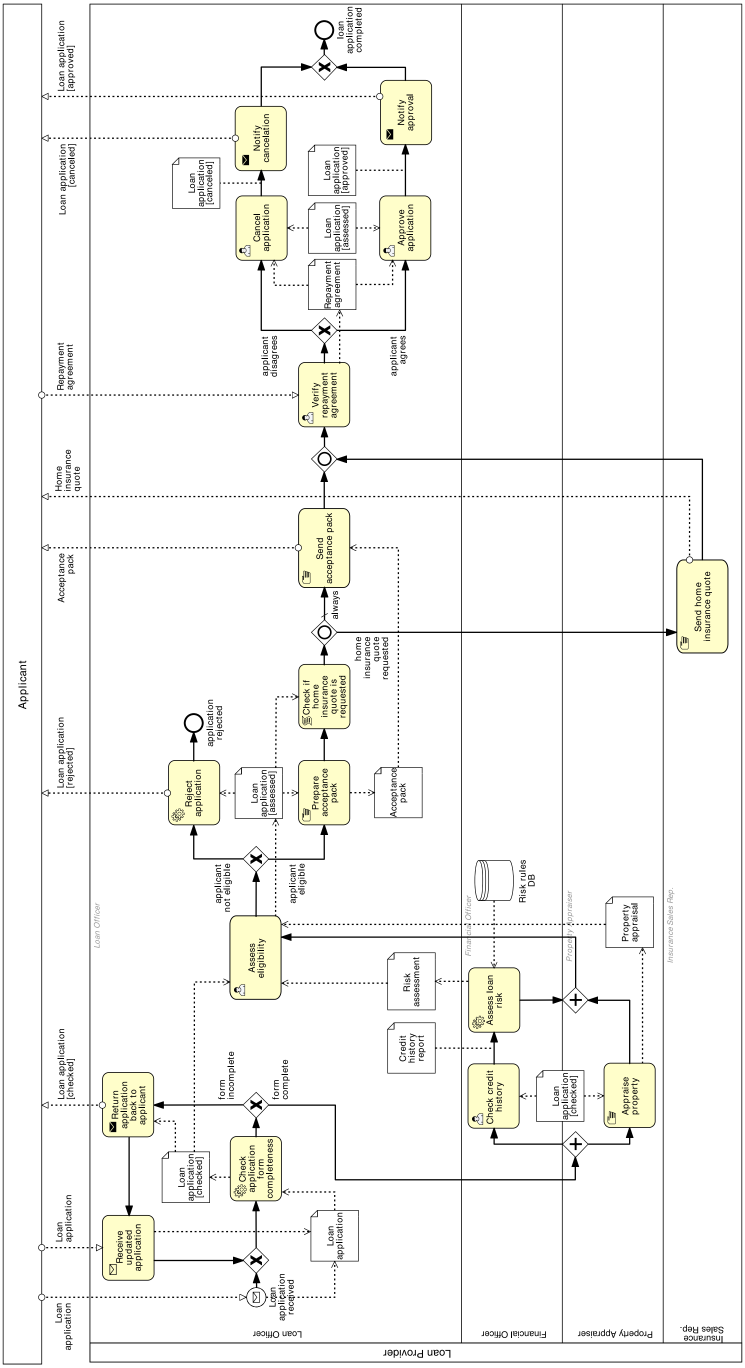
Solution 9.4

- Quality
- Transparency
- Flexibility

Solution 9.5

1. Organizational issue: BPMSs can be highly tailored to take patient-specific data into account.
2. Technical issue: The integration of the decision support system may require additional, customized software development.
3. Organizational/technical: The nurses may one the one hand get accustomed to using the BPMS in general and worklist handlers specifically. On the other hand, it may not be a good technical solution to use vibration signals—an alternative would be, for example, to use sound signals.

Solution 9.6



Applicant
Loan Provider
Loan Officer
Financial Officer
Property Appraiser
Insurance Sales Rep.
Loan application
Loan application [checked]
Loan application [rejected]
Acceptance pack
Home insurance quote
Repayment agreement
Loan application [canceled]
Loan application [approved]
Loan application received
Receive updated application
Check application form completeness
Return application back to applicant
form incomplete
form complete
Check credit history
Appraise property
Assess loan risk
Credit history report
Risk assessment
Risk rules DB
Property appraisal
Assess eligibility
applicant not eligible
applicant eligible
Reject application
application rejected
Loan application [assessed]
Prepare acceptance pack
Check if home insurance quote is requested
home insurance quote requested
always
Send acceptance pack
Send home insurance quote
Verify repayment agreement
applicant disagrees
applicant agrees
Cancel application
Approve application
Notify cancelation
Notify approval
loan application completed

Solution 9.7 All five manual tasks of this process, namely “Appraise property”, “Prepare acceptance pack”, “Send acceptance pack”, “Send home insurance quote” and “Verify repayment agreement” can be implemented as user tasks. In “Appraise property”, the property appraiser is notified through their worklist that they have to appraise a new property. The information on the property is carried by the work item of this task (e.g. property type and address). The property appraiser physically goes to the property address for an inspection and checks the value of surrounding properties. Once done, he prepares the appraisal on an electronic form and submits it to the BPMS engine via the worklist handler. “Prepare acceptance pack”, “Send acceptance pack”, “Send home insurance quote” can be implemented as user tasks in a similar way.

“Verify repayment agreement” appears in the loan officer’s worklist as soon as the acceptance pack, and optionally the insurance quote, have been sent to the applicant. The officer checks out this work item once they have received the repayment agreement from the applicant by post. They manually verify the agreement, digitize it and attach it as a file to the agreement summary—an electronic form associated with this work item and pre-populated with information extracted from the loan application. If the applicant accepted all loan conditions and agreed with the repayment schedule, the officer ticks the respective checkboxes in the agreement summary and submits this to the BPMS engine.

Solution 9.8 Task “Check insurance” can be automated through a service that determines the amount of the co-payment based on the details of the prescription and on the customer’s insurance policy.

Tasks “Collect drugs from shelves” and “Check quality” are manual tasks. These tasks can be implemented as user tasks in the automated process. To do so, the pharmacy technician who collects the drugs, and the pharmacist who quality-checks the prescription and seals the bag, should have a convenient mechanism to signal the completion of these activities to the BPMS. This could be achieved by putting in place a system based on barcode scans to track prescriptions. For example, the technician would see a list of prescriptions to be filled from their worklist. They would then pick up one of the prescriptions and the system would associate the prescription to a new barcode which is printed on an adhesive label. The technician would then attach the label to a bag, collect the drugs and put them in a bag, and when done, they would scan the barcode from the label to record that the prescription has been fulfilled. This signals the completion of task “Collect drugs from shelves” to the pharmacy system. In turn, it generates a new work item of task “Check quality” in the pharmacist’s worklist. The pharmacist can then quality-check the prescription and scan the barcode again.

Task “Collect payment” is also a manual task. This task could be implemented as a service task whereby the pharmacy system would push the task of collecting the payment for a prescription to a Point-of-Sale (POS) system and expect the POS system to indicate that the payment has been collected. The pharmacy technician would interact with the POS system once the customer arrives, but this interaction is outside the scope of the pharmacy system. The pharmacy system merely pushes work to the POS system and waits for completion.

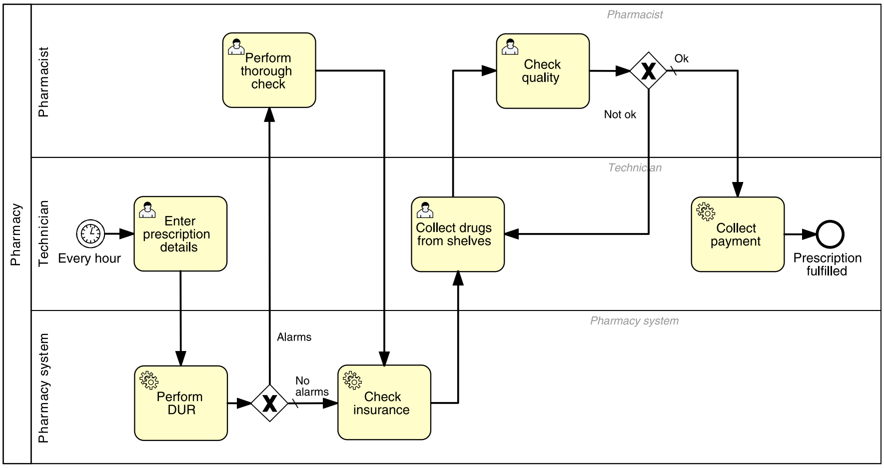


Fig. 9.12 The automated prescription fulfillment process

The description of the process implicitly refers to a manual task whereby the pharmacist seals the bag and puts it into the pick-up area. However, this “Seal bag” task is not included in the executable process model. Instead, this task is integrated into the “Check quality” task. In other words, at the end of the quality check, the pharmacist is expected to seal the bag if the prescription is ready and drop the bag in the pick-up area.

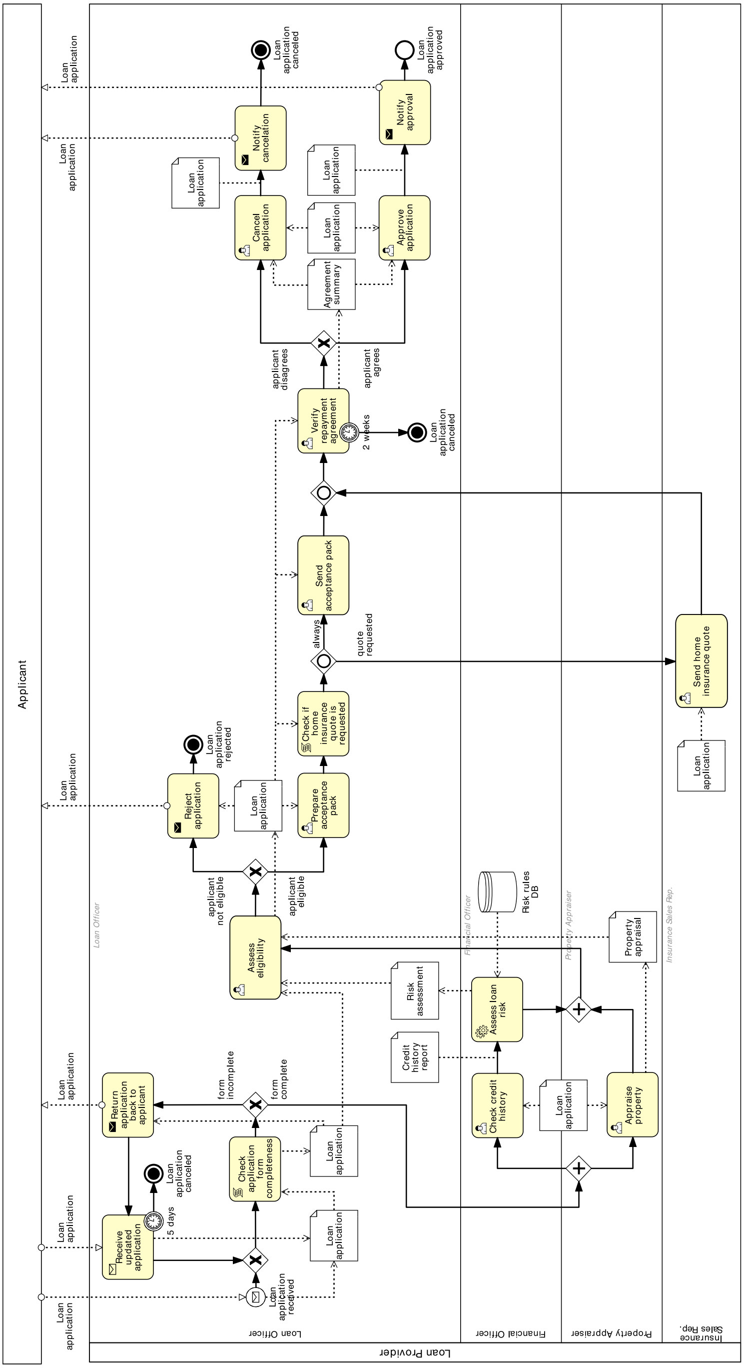
Task “Retrieve prescription bag” is also manual but there is no value in automating it in any way. So this task is left out of the executable process model, which completes once the payment has been made.

The executable model of the entire prescription fulfillment process is illustrated in Fig. 9.12.

Solution 9.9

- Physical data objects: Acceptance pack (this is the loan offer on paper), Repayment agreement (this is signed by the applicant on paper and has been replaced by the Agreement summary, an electronic document containing a link to a digitized copy of the repayment agreement plus a reference to the loan application). We assume all other communications between applicant and loan provider to occur via e-mail
- Messages carrying physical objects: Acceptance pack, Repayment agreement, Home insurance quote (the quote is sent on paper)
- Data stores: Risk Rules DB
- States of data objects
- Pools and lanes

Solution 9.10



Note that a work item of task “Verify repayment agreement” automatically disappears from the loan officer’s worklist if the officer does not start it within two weeks. This happens if the officer has not received the repayment agreement by post within that timeframe.

Solution 9.11 It makes sense for tasks “Prepare acceptance pack” and “Send acceptance pack” to be performed by the same loan officer. However, task “Check if home insurance quote is requested” is meant to be executed between these two tasks. Since there is no temporal dependency between “Check if home insurance is requested” and the other two tasks, we can postpone the former to after “Send acceptance pack” or parallelize it with the other two tasks. This way we can aggregate the two consecutive tasks into “Prepare and send acceptance pack”.

Solution 9.12 The solution is shown in Fig. 9.13.

1. Task types: the manual task of this process is “Discuss application”. This can be implemented as a user task that completes by producing a recommendation.
2. Non-executable elements: all elements can be interpreted by a BPMS. Note that the catching message event “Response received by post” assumes the existence of an internal service at the service provider that notifies the process when the response has been received by post.

3.1. Missing control flow: task “Create electronic response” is needed to convert the response received by post into an electronic version that can be consumed by a BPMS. Task “Assess response” may be interrupted by a request to cancel the application, for which the process is aborted. This request may also be received during the acceptance handling, in which case tasks “Send offer” and “File application” need be compensated. A one-week timeout is added to receive the response.

3.2. Missing data: all electronic data objects were missing in the conceptual model.

4. Granularity level: task “Make appointment” has been disaggregated to explicitly model the task of notifying the client of the result. Similarly, “Send offer” and “Send rejection” have been disaggregated to model the preparation of the offer, respectively, the rejection letter. Given that “Send offer” has been split into two activities (“Make offer” and “Send offer”) each needs to be compensated if a cancellation request is received.

Solution 9.13 We need two service interfaces to interact with the Web service behind the loan provider’s website. One interface where the loan provider acts as the service provider and the other where the website service acts as the service provider. The former interface contains one in-only operation for the loan provider to receive the initial loan application. The latter interface contains the following four operations for the website service:

- an in-out operation to receive the assessed loan application (containing change requests), and to respond with the revised loan application (where changes have been made)

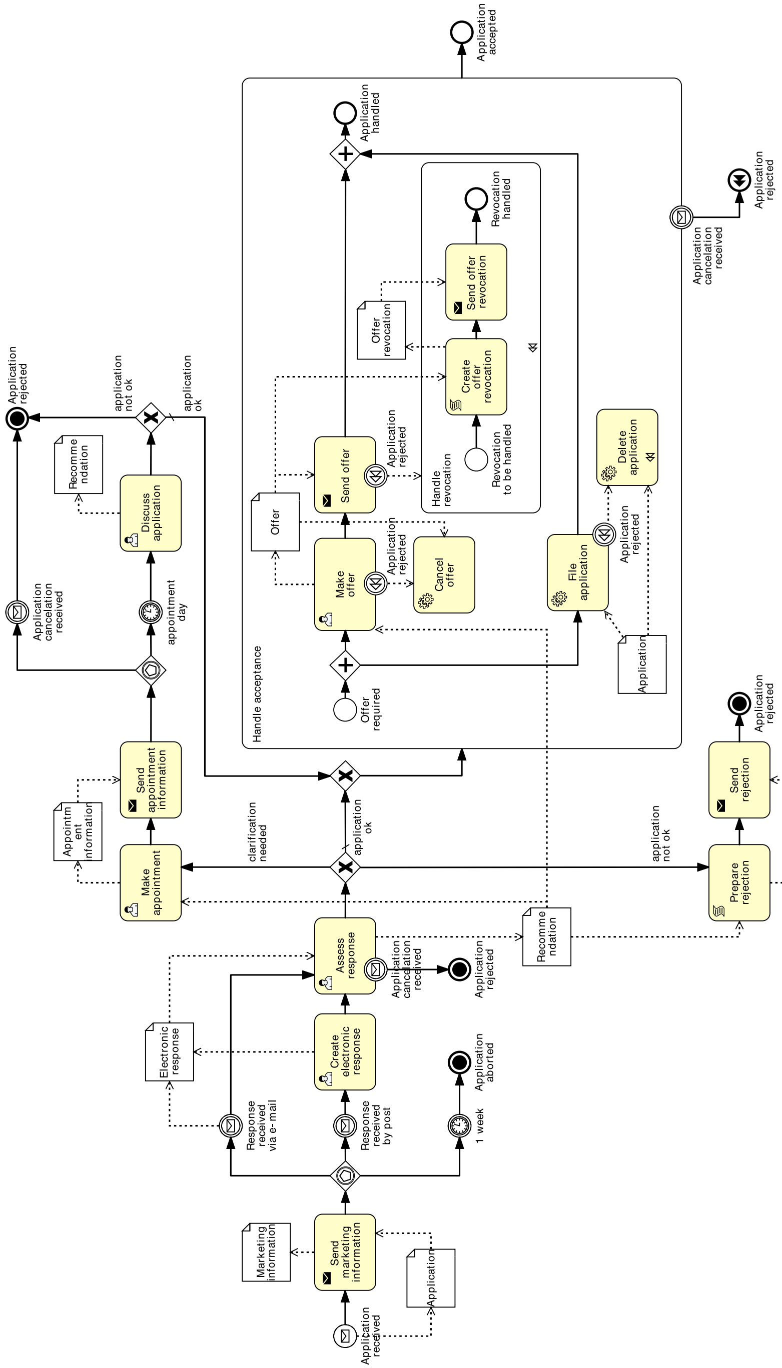


Fig. 9.13 The model for the sales process of a B2B service provider, completed with missing control flow and data relevant for execution

- an in-only operation to receive the rejected loan application
- an in-only operation to receive the approved or canceled loan application

The four operations above require five messages in total, all of the same data type as the loan application's. These operations are assigned to the start message event, the four send tasks and the receive task of the process, which need to have suitable data inputs and outputs to contain the loan application. The mapping of these data inputs and outputs to data objects is straightforward, except for the send task "Reject application", which needs to modify the status of the loan application to "rejected" while copying this input data object into the task data input.

A third service interface is required to interact with the service for assessing loan risks in task "Assess loan risk". This interface has an in-out operation with two messages: an input message to contain the credit history report and an output message for the risk assessment.

The script for task "Check application form completeness" takes a loan application as input and checks that all required information is present. Depending on the outcome of the check it changes the application status to either "complete" or "incomplete", assigns a fresh application identifier to the application if empty, writes the submission or revision date and time and, if applicable, fills out the status comments section with pointers to incomplete data fields. The script task "Check if home insurance quote is requested" is actually not needed. While in a business-oriented model it is important to explicitly capture each decision with an activity as we have illustrated in Chap. 3, in an executable model this can be directly embedded in the conditions of the outgoing arcs of an (X)OR-split if the outcome of the decision can easily be verified. In fact, our example just requires to check the value of a boolean field in the application, which can be achieved with an XPATH expression directly on the arc labeled "quote requested".

All user tasks of this process are implemented via the worklist handler of the loan provider and offered to participants having the required role (e.g. task "Assess eligibility" is offered to a participant with role loan officer). This implementation depends on the BPMS adopted. The mapping between data objects and data inputs and outputs for these tasks is straightforward. In the case of task "Assess eligibility", at runtime the loan officer will see an electronic form for the loan application (editable), and two more forms for the risk assessment and for the property appraisal (non editable). The officer is required to edit the loan application by entering their identifier, specifying whether or not the applicant is eligible for the loan and adding status comments in case of ineligibility. The other user tasks work similarly.

We have already discussed how to implement the condition of arc "quote requested". The conditions on the other sequence flows can be implemented with an expression that extracts data from a data object in a similar way. The expression for the arc labeled "always" is simply "true" as this arc is taken always. The temporal expression for the two timer events is simply a relative duration (5 days and 2 weeks).

Solution 9.14 This is a hands-on exercise, no solution is provided. Resources related to implementing processes on top of specific BPMSs can be found in the companion website http://fundamentals-of-bpm.org.

9.7 Further Exercises

Exercise 9.15 Draw the architecture of a BPMS and identify all its components.

Exercise 9.16 Explain the similarities and differences between production and ad-hoc workflow systems. Include in your explanation a reflection on the type of support they provide on the one hand and their orientation in the spectrum of data versus process on the other.

Exercise 9.17 Classify the following objectives of the various organizations described that use a BPMS and use the categories that were explained in Sect. 9.2.

- A legal company wishes to track all the participants it has involved in its formalized process for the preparation of litigation cases.
- A governmental agency wishes to reduce the penalties it must pay for late payments of invoices.
- A bank wishes to demonstrate to its external auditor that it strictly enforces the principle that each large loan is approved by two separate clerks.

Exercise 9.18 In a 2009 posting on LinkedIn, the director of Walgreens, an online pharmacy, asks what the common pitfalls are when implementing a workflow management system. A consultant at Microsoft answers as follows:

> It's really all about people buying in to the system. The general tendency of people is that they don't like change, even if they say they do. Even if their current processes are very inefficient, they know how it works. So when you introduce something that changes their world (with a workflow mgt system), they'll be very apprehensive. Also, the more the system changes how they do their job, the more resistance you'll get. Then it becomes an issue of how you gathered your business requirements. Chances are that due to misunderstandings the requirements will be different from expectations of how things should get done.

Explain whether you think this explanation relates to a technical or an organizational challenge.

Exercise 9.19 Identify the type of the tasks in Fig. 4.15, and represent them using appropriate BPMN markers.

Exercise 9.20 Consider the following business processes. Identify which of these models can be automated and justify your choice.

1. Recruiting a new soldier.
2. Organizing a court hearing.
3. Buying an item at an auction on eBay.

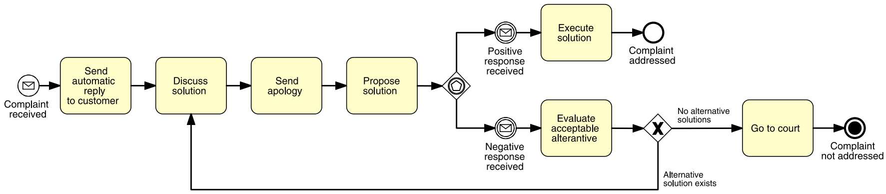


Fig. 9.14 FixComp's process model for handling complaints

4. Managing inventory assets disposition.
5. Booking a trip on-line.
6. Handling an IT-system maintenance job.
7. Servicing a used car at a mechanic.
8. Making online trade customs declarations.
9. Processing employee payrolls.
10. Synchronizing data servers in a distributed environment.

Exercise 9.21 Figure 9.14 shows the process model that FixComp follows when a client files a complaint. Upon receipt of a new complaint from a client, the process starts by sending an automatic reply to the client, in order to reassure them that FixComp is following up on their request. A complaints representative then takes the complaint for discussion with people in the department the complain refers about. Next, the complaints representative sends a personal letter of apology to the client and propose them a solution. The client can either accept or reject the solution. If the client accepts the solution, the solution is executed by the relevant department. If the client rejects the solution, the client is called on the phone to discuss possible alternatives by the complaints representative. If one of these alternatives is promising, it is discussed with the department and the process continues. If no agreement can be reached, the case is brought to court.

The company wants to automate this process to deal with complaints in a more efficient manner. Your task is to prepare this model for execution.

Acknowledgement This exercise is adapted from a similar exercise developed by Remco Dijkman, Eindhoven University of Technology.

Exercise 9.22 Consider the claims handling process modeled in Fig. 9.15. Implement this business process using a BPMS of your choice.

The process starts when a customer submits a new insurance claim. Each insurance claim goes through a two-stage evaluation process. First of all, the liability of the customer is determined. Secondly, the claim is assessed in order to determine if the insurance company has to cover this liability and to what extent. If the claim is accepted, payment is initiated and the customer is advised of the amount to be paid. All activities except "Initiate Payment" are performed by claims handlers. There are three claims handlers. Activity "Initiate Payment" is performed by a financial officer. There are two financial officers.

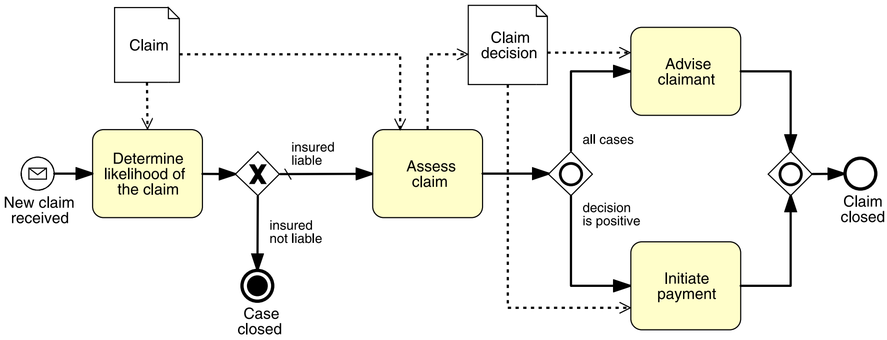


Fig. 9.15 Claims handling process model

As shown in the model, there are two data objects involved in this process: Claim and Claim decision. A claim includes the following data fields:

- Name of claimant
- Policy number (a string with alphanumeric characters)
- Description of the claim
- Amount claimed

A claim decision consists of the following data fields:

- Reference to a claim
- Decision (positive or negative)
- Explanation
- Amount to be reimbursed (greater than zero if the decision is positive)

You may add other data fields into the above objects if you deem it necessary.

Exercise 9.23 Consider the following equipment rental process, which is a variant of the one described in Example 1.1. Implement this business process using a BPMS of your choice.

The rental process starts when a site engineer fills in an equipment rental request containing the following details:

- Name or identifier of the site engineer who initiates the request
- Requested start date and time of the equipment rental
- Expected end date and time of the equipment rental
- Project for which the equipment is to be rented
- Construction site where the equipment will be used
- Description of required equipment
- Expected rental cost per day (optional)
- Preferred supplier (optional)
- Supplier's equipment reference number (optional)
- Comments to the supplier (optional)

The rental request is taken over by one the clerks at the company's depot. The clerk consults the catalogs of the equipment suppliers and calls or sends e-mails to the supplier(s) in order to find the most cost-effective available equipment that complies with the request. Once the clerk has found a suitable piece of equipment available for rental, they recommend that it be rented. At this stage, the clerk must add the following data to the equipment rental request:

- Selected supplier
- Reference number of the selected equipment
- Cost per day

Equipment rental requests have to be approved by a works engineer (who also works at the depot). In some cases, the works engineer rejects the equipment rental request, meaning that no equipment will be hired. Of course, before rejecting a request in this way, the works engineer should first discuss their decision with the site engineer and also add an explanatory note to the equipment rental request. In other cases, the works engineer rejects the recommended equipment (but not the entire request) and asks the clerk to find an alternative equipment. Again, in this case the works engineer should communicate their decision to the clerk and add an explanatory note.

Rental requests where the cost per day is below 100 are automatically approved, without going through a works engineer.

Once a request is approved, a purchase order is automatically generated from the data contained in the approved rental request. The purchase order includes:

- Supplier's equipment identification
- Cost per day
- Construction site where the plant is to be delivered
- Delivery date and time
- Pick-up date and time
- Comments to the supplier (optional)

The supplier delivers the equipment to the construction site at the required date. The site engineer inspects the equipment. If everything is in order, they accept the equipment, add the date of delivery to the purchase order and optionally a note to indicate any issues found during the inspection. Similarly, when the equipment is picked up by the supplier at the end of the renting period, another inspection is performed, and the supplier marks the pick-up date in the purchase order (possibly with a pick-up note).

Sometimes, the site engineer asks for an extension of the rental period. In this case, the site engineer writes down the extended pick-up time into the purchase order, and the revised purchase order is automatically resent to the supplier. Prior to doing this change, the site engineer is expected to call the supplier in order to agree on the change of pick-up date.

A few days after the equipment is picked up, the supplier sends an invoice to the clerk by e-mail. The clerk records the following details:

- Supplier's details
- Invoice number
- Purchase order number
- Equipment reference number
- Delivery date and time
- Pick-up date and time
- Total amount to be paid

Having entered these invoice details, the clerk verifies the invoice details against the purchase order and marks the invoice as accepted or rejected. In case of rejection, the clerk adds an explanatory note (e.g. requesting the supplier to send a revised invoice). Eventually, the supplier may send a revised invoice if needed.

The accepted invoice is forwarded to the finance department for payment, but this part of the process is handled separately and is not part of this exercise.

Exercise 9.24 Define appropriate data types for the sales process shown in Fig. 9.13, and implement it using a BPMS of your choice.

Exercise 9.25 Define appropriate data types for the process shown in Figs. 4.32–4.34, and implement it from the perspective of BestLoans using a BPMS of your choice.

9.8 Further Reading

Van der Aalst and van Hee's book [95] offers an introduction to workflow management technology as of the early 2000s. The evolution from workflow management systems to BPMSs that took place during the 2000s is discussed at length by Weske [106]. As stated in this chapter, the WfMC reference model was instrumental in shaping the architectures of workflow management systems and later that of BPMSs. Details on the WfMC reference model can be found at http://wfmc.org/reference-model.html while Hollingsworth [34] gives a summary of the development and directions of this reference model.

A frequent critique of BPMSs is that they follow a Fordist paradigm, meaning that the BPMS forces process participants to act in a certain direction, i.e. exactly in the way it is captured in a process model. In the context of processes where unanticipated exceptions are common and where there is no predictable way to perform the process, a BPMS often ends up obstructing the work of process participants rather than supporting them. Approaches to support non-standardized or unpredictable processes are described by Reichert et al. [69]. One of those approaches is case handling as discussed in the Box "Types of BPMS". An introduction to this topic is given by Van der Aalst et al. [96], while a more comprehensive treatment of the subject is provided by Swenson [91].

A discussion on executable BPMN 2.0 is provided in Silver's book [87], as well as in the book by Freund and Rücker [19]. An in-depth coverage of process automation using the YAWL language is given by ter Hofstede et al. [92].

A gentle introduction to XML, XML Schema and XPath can be found in Møller and Schwartzbach's book [56]. Meanwhile, the topic of Web services is covered in-depth by Erl et al. [17]. The latter book also includes a discussion on WSDL 2.0—the default technology for implementing service interfaces in BPMN 2.0.

Chapter 10
Process Intelligence

If you can't measure something, you can't understand it. If you can't understand it, you can't control it. If you can't control it, you can't improve it.
H. James Harrington (1929–)

It is a central idea of BPM that processes are explicitly defined, then executed, and that information about process execution is prepared and analyzed. In this way, this information provides a feedback loop on how the process might be redesigned. Data about the execution of processes can stem from BPMSs in which processes are specified, but also from systems that do not work with an explicit process model, for instance ERP systems or ticketing systems. Data from those systems have to be transformed to meet the requirements of intelligent process execution analysis. This field is typically referred to as process mining.

This chapter deals with intelligently using the data generated from the execution of the process. We refer to such data as event logs, covering what has been done when by whom in relation to which process instance. First, we investigate the structure of event logs, their relationship to process models, and their usefulness for process monitoring and controlling. Afterwards, we discuss three major objectives of intelligent process analysis, namely transparency, performance and conformance. We discuss automatic process discovery as a technical step to achieve transparency of how the process is executed in reality. Then, we study how the analysis of event logs can provide insights into process performance. Finally, we discuss how the conformance between event logs and a process model can be checked.

10.1 Process Execution and Event Logs

In the previous chapter, we studied how a process model can be specified in a way that a BPMS can support its execution. Both process participants and process owners are involved in the execution of business processes. However, their perspective is quite different. Process participants work on tasks, which produce execution data as a side product. We call this data *event logs*. Process owners are particularly interested in drawing conclusions from such event logs. In this section, we discuss which

M. Dumas et al., *Fundamentals of Business Process Management*,
DOI 10.1007/978-3-642-33143-5_10, © Springer-Verlag Berlin Heidelberg 2013

questions can be answered using event data and how event logs and process models relate to each other.

10.1.1 The Perspective of Participants on Process Execution

When a process is executed on a BPMS or another piece of software, there is a clear separation between coordination and execution of tasks. The system usually takes care of coordinating individual cases informing participants about which tasks they need to work on. Accordingly, participants often see only those tasks that they are directly responsible for, while the system hides the complexity of the overall process. Each participant typically has a personal *worklist* that shows the set of *work items* that still need to be worked on. If an explicit process model exists, each of these work items corresponds to a task in the process model. However, there might exist multiple work items corresponding to a single task if several cases are currently being worked on. For example, at a specific point in time, Chuck as a process participant might see that four work items are in his worklist, all relating to the "Confirm order" task of the order fulfillment process: one work item relates to an order from customer A, one from customer B, and two from customer C.

The structure of a work item is defined in the executable process model or directly implemented in the software. This means that participants see those data fields that have been declared as input for a task. For each work item they are working on, they are supposed to document at least the completion. In this way, the system can keep track of the state of the process at any point in time. Among others, it is easy to record at which point in time somebody has started working on a work item, which input data were available, what output data were created, and who was the participant working on it. For example, when Chuck has confirmed the order of customer B, he enters the result in the system, and the system can decide automatically if next the invoice should be emitted or the order confirmation should be escalated to someone above Chuck. Most BPMSs and also other information systems record such data on what has been done at which point in time. The file in which these data is stored is called a *log file*, and the data in it is called *event logs*. Each time another task is completed, a new entry is added to the log file. That is, once Chuck has entered his data, the system appends a line in the log file stating that Chuck has confirmed an order with a corresponding timestamp.

10.1.2 The Perspective of the Process Owner on Process Execution

Event logs have the potential to reveal important management insights into how a process works in reality. Therefore, the process owner is most interested in analyzing it in a systematic way. In essence, we distinguish three major application scenarios for using event logs: automatic process discovery, performance analysis and conformance checking, all related to questions the process owner might ask.

What is the actual process model? Automatic process discovery is concerned with the question of how a process actually works in reality. In Chap. 5, we mentioned that the event logs can be used as an input to evidence-based process discovery. Automatic process discovery utilizes event logs for the generation of a corresponding process model. In this way, event logs are valuable to find a process model where no model existed before, and to adjust an existing model according to how the process really works.

What is the performance of the process? In Chap. 7, we discussed that process analyses such as flow analysis suffer from the fact that the average cycle time for each task in the process model needs to be estimated. Also, often strong assumptions are required such that the behavior of the process is not influenced by the load. Using event logs, we are able to inspect the actual behavior of a process and compare it with insights from process analysis. Furthermore, historic information about process execution can be leveraged for making operational decisions.

To which extent are the rules of the process model followed? Conformance checking is a collection of techniques that compare a set of event logs with a set of constraints or an existing process model. There are situations when process models are defined, but they are not strictly enforced by a corresponding BPMS. In these situations, conformance checking can be utilized in order to determine how often the process is executed as expected and, if not, at which stages deviations can be found. Here, event logs help to understand either where the model needs to be corrected or where the behavior of the participants working in the process has to be adapted.

By providing answers to these three types of question, we can get insights into the process, which may help to reposition it in the *Devil's Quadrangle*. Specifically, the dimension of time increases in transparency by investigating event logs: the timestamps show when tasks are executed and how long they take. There is also a strong association with cost if the working time of the participants working in the process can be assigned to a particular process instance. Flexibility can be analyzed based on the different paths a process takes. The historic set of actually used paths and their variety gives an indication for this dimension. Finally, also quality issues might be identified from event logs, for instance when inspecting the number of reworks and iterations required for a specific task.

The process owner can use event logs as input to two different control mechanisms: on an aggregated level and on an instance level. The mechanisms are called *process controlling* and *process monitoring*, respectively.

Process Controlling deals with the analysis of historic process execution. The input for process controlling are event logs that relate to a particular period of time, for instance a quarter or a full year. Process controlling provides insights into whether the general objectives of a process have been met and whether the KPIs are in line. Typically, process controlling is an *offline* activity, which involves logs of completed process executions.

Process Monitoring is concerned with the quality of currently running process instances. The input for process monitoring are the event logs of individual process instances or cases. Process monitoring works with objectives and rules that are formulated for these individual cases, and triggers counteractions when these rules are violated, for instance when a customer request is not responded in time. Typically, process monitoring is a continuous *online* activity which involves events of currently running instances.

Both process monitoring and process controlling play an important role in aligning the process with its overall business objectives. In that respect, they are closely related to ideas of quality management and the Plan-do-check-act (PDCA) cycle. PDCA can be regarded as an inspiration for the concept of a business process management lifecycle as discussed in the first chapter of this book. Process monitoring and controlling (check) investigate the data from executing processes (do) such that redesign measures (act) can be taken to realign the execution with the objectives (plan). All these concepts have inspired the idea of a *process cockpit* as a software tool where data on the execution of processes is provided online using charts and appropriate visualization (see Fig. 9.4 for an example). Often, these tools are also called Business Activity Monitoring (BAM) tools or Process Performance Measurement (PPM) tools.

10.1.3 Structure of Event Logs

Process monitoring and controlling strongly rely on event data to be recorded during the execution of processes. *Event logs* contain a set of events. Accordingly, we can understand an event log as a list of event recordings. Figure 10.1 gives an illustration of what data is typically stored in event logs. We can see that a single event has a unique event ID. Furthermore, it refers to one individual case, it has a timestamp, and it shows which resources executed which task. These may be participants (e.g. Chuck and Susi) or software systems (SYS1, SYS2, DMS). For several analysis techniques that we will discuss in this chapter, it is a minimum requirement that the events in the log refer to (i) one case, (ii) one task, and (iii) a point in time. Having these three pieces of information available, we can for instance discover a process model from the logs. In practice, there are often additional pieces of information stored for each event like costs, system being used, or data on the handled business case. These can be used for clustering, correlating or finding causal relationships in the event logs.

The event log of Fig. 10.1 is captured as a list in a tabular format.[1] The problem with event logs is that each vendor and software system defines individual log formats. In order to leverage the adoption of event log analysis tools such as the open

[1]For simplicity, we only consider one supplier in this example.

Case ID	Event ID	Timestamp	Activity	Resource
1	Ch-4680555556-1	2012-07-30 11:14	Check stock availability	SYS1
1	Re-5972222222-1	2012-07-30 14:20	Retrieve product from warehouse	Rick
1	Co-6319444444-1	2012-07-30 15:10	Confirm order	Chuck
1	Ge-6402777778-1	2012-07-30 15:22	Get shipping address	SYS2
1	Em-6555555556-1	2012-07-30 15:44	Emit invoice	SYS2
1	Re-4180555556-1	2012-08-04 10:02	Receive payment	SYS2
1	Sh-4659722222-1	2012-08-05 11:11	Ship product	Susi
1	Ar-3833333333-1	2012-08-06 09:12	Archive order	DMS
2	Ch-4055555556-2	2012-08-01 09:44	Check stock availability	SYS1
2	Ch-4208333333-2	2012-08-01 10:06	Check materials availability	SYS1
2	Re-4666666667-2	2012-08-01 11:12	Request raw materials	Ringo
2	Ob-3263888889-2	2012-08-03 07:50	Obtain raw materials	Olaf
2	Ma-6131944444-2	2012-08-04 14:43	Manufacture product	SYS1
2	Co-6187615741-2	2012-08-04 14:51	Confirm order	Conny
2	Em-6388888889-2	2012-08-04 15:20	Emit invoice	SYS2
2	Ge-6439814815-2	2012-08-04 15:27	Get shipping address	SYS2
2	Sh-7277777778-2	2012-08-04 17:28	Ship product	Sara
2	Re-3611111111-2	2012-08-07 08:40	Receive payment	SYS2
2	Ar-3680555556-2	2012-08-07 08:50	Archive order	DMS
3	Ch-4208333333-3	2012-08-02 10:06	Check stock availability	SYS1
3	Ch-4243055556-3	2012-08-02 10:11	Check materials availability	SYS1
3	Ma-6694444444-3	2012-08-02 16:04	Manufacture product	SYS1
3	Co-6751157407-3	2012-08-02 16:12	Confirm order	Chuck
3	Em-6895833333-3	2012-08-02 16:33	Emit invoice	SYS2
3	Sh-7013888889-3	2012-08-02 16:50	Get shipping address	SYS2
3	Ge-7069444444-3	2012-08-02 16:58	Ship product	Emil
3	Re-4305555556-3	2012-08-06 10:20	Receive payment	SYS2
3	Ar-4340277778-3	2012-08-06 10:25	Archive order	DMS
4	Ch-3409722222-4	2012-08-04 08:11	Check stock availability	SYS1
4	Re-5000115741-4	2012-08-04 12:00	Retrieve product from warehouse	SYS1
4	Co-5041898148-4	2012-08-04 12:06	Confirm order	Hans
4	Ge-5223148148-4	2012-08-04 12:32	Get shipping address	SYS2
4	Em-4034837963-4	2012-08-08 09:41	Emit invoice	SYS2
4	Re-4180555556-4	2012-08-08 10:02	Receive payment	SYS2
4	Sh-5715277778-4	2012-08-08 13:43	Ship product	Susi
4	Ar-5888888889-4	2012-08-08 14:08	Archive order	DMS

Fig. 10.1 Example of an event log for the order fulfillment process

source tool ProM,[2] the *IEEE Task Force on Process Mining* promotes the usage of the *eXtensible Event Stream* (*XES*) format. Several tools work with XES event logs or offer features to convert events logs into this format. The metamodel of XES is illustrated in Fig. 10.2. Each XES file represents a log. It contains multiple traces, and each trace can contain multiple events. All of them can contain different attributes. An attribute has to be either a string, date, int, float, or boolean element as a key-value pair. Attributes have to refer to a global definition. There are two global elements in the XES file, one for defining trace attributes, the other for defining event attributes. Several classifiers can be defined in XES. A classifier maps one of more attributes of an event to a label that is used in the output of the analysis tool. In this way, for instance, events can be associated with activities.

[2]The software is available at http://www.promtools.org.

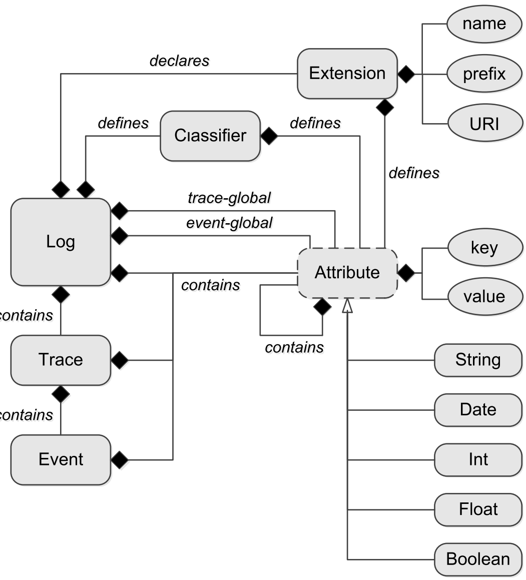


Fig. 10.2 Metamodel of the XES format

Fig. 10.3 Example of a file in the XES format

```
<log xes.version="1.0" xes.features="arbitrary-depth" xmlns="http://.../xes">
  <extension name=”Concept” prefix=”concept” uri=”http://.../xes/concept.xesext”/>
  <extension name=”Time” prefix=”time” uri=”http://.../xes/time.xesext”/>
  <global scope=”trace”>
    <string key=”concept:name” value=””/>
  </global>
  <global scope=”event”>
    <string key=”concept:name” value=””/>
    <date key=”time:timestamp” value=”1970-01-01T00:00:00.000+00:00”/>
    <string key=”resource” value=””/>
  </global>
  <classifier name=”Activity” keys=”concept:name”/>
  <float key=”log attribute” value=”2335.23”/>
  <trace>
    <string key=”concept:name” value=”1”/>
    <event>
      <string key=”concept:name” value=”Check stock availability”/>
      <date key=”time:timestamp” value=”2012-07-30T11:14:00:000+01:00”/>
      <string key=”resource” value=”Chuck”/>
    </event>
    <event>
      <string key=”concept:name” value=”Retrieve product from warehouse”/>
      <date key=”time:timestamp” value=”2012-07-30T14:20:00:000+01:00”/>
      <string key=”resource” value=”Rick”/>
    </event>
  </trace>
</log>
```

Figure 10.3 shows how parts of the information of the event log example from Fig. 10.1 can be stored in an XES file. From the first “global” element (scope = “trace”), we see that each trace element is expected to have a “concept:name” attribute. For the trace defined below, this attribute has the value 1. Furthermore, there are three different attributes expected for an event (“global” element with scope = “event”): “concept:name”, “time:timestamp”, and “resource”. In the trace defined below, we observe two events. The first one refers to “Check stock availability”, which was completed by SYS1 on the 30th of July 2012 at 11:14. The second event captures “Retrieve product from warehouse” conducted by Rick at 14:20.

10.1.4 Challenges of Extracting Event Logs

Event data that is available in some tabular format like that visualized in Fig. 10.1 can be readily converted to XES, and then analyzed using appropriate tools. In many cases though, the data which is relevant for event logs is not directly accessible in the required format, but has to be extracted from different sources and to be integrated. Therefore, we can identify five major challenges for log data extraction, potentially requiring considerable effort in a project. These challenges are:

1. **Correlation challenge:** This refers to the problem of identifying the case an event belongs to. Many information systems do not have an explicit notion of process defined. Therefore, we have to investigate which attribute of process-related entities might serve as a case identifier. Often, it is possible to utilize entity identifiers such as order number, invoice number, or shipment number.
2. **Timestamps challenge:** The challenge to work properly with timestamps stems from the fact that many information systems do not consider logging as a primary task. This means that logging is often delayed until the system has idle time or little load. Therefore, we might find sequential events with the same timestamp in the log. This problem is worsened when logs from different information systems potentially operating in different time zones have to be integrated. Partially, such problems can be resolved with domain knowledge, for example when events are known to always occur in a specific order.
3. **Snapshots challenge:** This point refers to the issue of having log data available for a certain period of time. For long running processes, we might not be able to observe all cases of the log with their full end-to-end duration in the considered period of time. It is a good idea to exclude such incomplete cases with missing head or tail. However, one should be aware that such a filtering might also introduce a bias, for instance that only brief cases are considered. Therefore, the time span reflected by the log should be significantly longer than the average duration of a case.
4. **Scoping challenge:** The scoping of the event spectrum is a challenge when the available information system does not directly produce event logs. Information systems such as ERP systems record an extensive amount of process-related events in numerous tables. Event logs have to be generated from the entries in these tables, which requires a detailed understanding of the data semantics. Such system expertise may not readily be available.
5. **Granularity challenge:** Typically, we are interested in conducting event log analysis on a conceptual level for which we have process models defined. In general, the granularity of event log recording is much finer such that each activity of a process model might map to a set of events. For example, an activity like “Retrieve product from warehouse” on the abstraction level of a process

model maps to a series of events like "Work item #1,211 assigned", "Work item #1,211 started", "Purchase order form opened", "Product retrieved" and "Work item #1,211 completed". Often, fine-granular events may show up repeatedly in the logs while on an abstract level only a single task is executed. Therefore, it is difficult to define a precise mapping between the two levels of abstraction.

Exercise 10.1 Consider the final assembly process of Airbus for their A380 series. The final assembly of this aircraft series is located at the production site in Toulouse, France. Large parts are brought by ship to Bordeaux and brought to Toulouse by waterway and road transport. What is the challenge when log data of the A380 production process has to be integrated?

10.2 Automatic Process Discovery

Automatic process discovery is a specific process mining technique. The goal of automatic process discovery is to construct a process model that captures the behavior of an event log in a representative way. The construction is meant to work automatically and generically, using an algorithm that should make minimal assumptions about properties of the log and the resulting process model. Being representative in this context loosely means that the constructed process model should be able to replay the cases of the event log and forbid behavior not found in the logs. In the following, we first discuss log data assumptions, then we present the α-algorithm as a basic discovery algorithm, and turn to the notion of representativeness in more detail.

10.2.1 Assumptions of the α-Algorithm

The *α-algorithm* is a basic algorithm for discovering process models from event logs. It is basic as it is less complex than other, more advanced algorithms. Beyond that, it makes certain assumptions about the provided event logs that we will later discuss as partially being problematic. These assumptions are the following:

- **Order of Events:** The events in the log are chronologically ordered. Such a chronological order can be defined based on timestamps.
- **Case Reference:** Each event refers to a single case.
- **Activity Reference:** Each event relates to a specific activity of the process.
- **Activity Completeness:** Each activity of the process is included in the log.
- **Behavioral Completeness:** The log is behaviorally complete in the sense that if an activity a can be directly followed by an activity b, then there is at least one case in the log where we observe ab.

Case ID	Event ID	Timestamp	Activity
1	Ch-468	2012-07-30 11:14	Check stock availability
1	Re-597	2012-07-30 14:20	Retrieve product from warehouse
1	Co-631	2012-07-30 15:10	Confirm order
1	Ge-640	2012-07-30 15:22	Get shipping address
1	Em-655	2012-07-30 15:44	Emit invoice
1	Re-418	2012-08-04 10:02	Receive payment
1	Sh-465	2012-08-05 11:11	Ship product
1	Ar-383	2012-08-06 09:12	Archive order
2	Ch-405	2012-08-01 09:44	Check stock availability
2	Ch-420	2012-08-01 10:06	Check materials availability
2	Re-466	2012-08-01 11:12	Request raw materials
2	Ob-326	2012-08-03 07:50	Obtain raw materials
2	Ma-613	2012-08-04 14:43	Manufacture product
2	Co-618	2012-08-04 14:51	Confirm order
2	Em-638	2012-08-04 15:20	Emit invoice
2	Ge-643	2012-08-04 15:27	Get shipping address
2	Sh-727	2012-08-04 17:28	Ship product
2	Re-361	2012-08-07 08:40	Receive payment
2	Ar-368	2012-08-07 08:50	Archive order

Letter	Actvities
a	Check stock availability
b	Retrieve product from warehouse
c	Check materials availability
d	Request raw materials
e	Obtain raw materials
f	Manufacture product
g	Confirm order
h	Get shipping address
i	Ship product
j	Emit invoice
k	Receive payment
l	Archive order

Workflow Log
a,b,g,h,j,k,i,l
a,c,d,e,f,g,j,h,i,k,l
...

Fig. 10.4 Definition of a workflow log

The first three assumptions refer to the information content of an event in the logs. These assumptions are rather general and non-restrictive. The activity completeness criterion refers to the fact that we can only include those activities in the generated process model that we observe in the logs. The behavioral completeness has the strongest implications. In practice, we can hardly assume that we find the complete set of behavioral options in a log. Advanced techniques try to make weaker assumptions on this point.

In line with these assumptions, we use a so-called *workflow log* as starting point for using the α-algorithm. Figure 10.4 shows how a workflow log can be constructed from an event log. In the following, we will work with letters as a reference to tasks. A workflow log is a collection of all unique execution sequences observed in the log. The α-algorithm does not distinguish how often a specific execution sequence was observed in a workflow log.

Exercise 10.2 Have a look at Fig. 10.1 and translate it into a workflow log following the same mapping rules as in Fig. 10.4.

10.2.2 The Order Relations of the α-Algorithm

The α-algorithm works in two phases in order to construct a process model. In the first phase, different order relations are extracted from the workflow log L. In the second phase, the process model is constructed in a stepwise fashion from these identified relations. The *order* relations refer to tasks which directly follow one another in the log. It provides the basis for the definition of three more specific relations that refer to *causality*, to potential *parallelism* and to *non-succession*. We refer to this set of relations as the α relations.

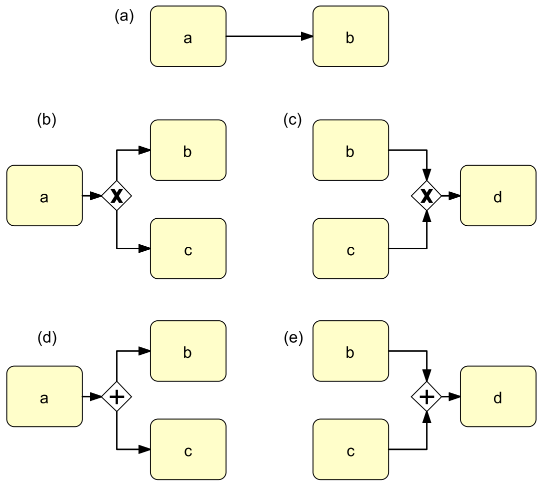


Fig. 10.5 Simple control flow patterns

- The basic order relation $a > b$ holds if we can observe in the workflow log L that an task a is directly followed by b.
- The causality relation $a \rightarrow b$ is derived from the basic relation. It holds if we observe in L that $a > b$ and that $b \not> a$.
- The relation of potential parallelism $a \parallel b$ holds if both $a > b$ and $b > a$ is observed in the workflow log L.
- The relation of no direct succession $a\#b$ holds if $a \not> b$ and $b \not> a$.

The reason why exactly these relations are used is shown in Fig. 10.5. There are five characteristic combinations of relations between the tasks in a workflow log that can be mapped to simple control flow patterns.

Pattern (a) depicts a sequence of tasks a and b. If we model them in this way, it should be guaranteed that in a workflow log we will find a followed by b, i.e. $a > b$, but never b followed by a, i.e. $b \not> a$. This means that the causality relation $a \rightarrow b$ should hold.

Pattern (b) also relates to a characteristic combination of relations. The workflow log should show that $a \rightarrow b$ and $a \rightarrow c$ hold, and that b and c would not be mutual successors, i.e. $b\#c$.

Pattern (c) also requires that b and c would not be mutual successors, i.e. $b\#c$, while both $b \rightarrow d$ and $c \rightarrow d$ have to hold.

Pattern (d) demands that $a \rightarrow b$ and $a \rightarrow c$ hold, and that b and c show potential parallelism, i.e. $b \parallel c$.

Pattern (e) refers to $b \rightarrow d$ and $c \rightarrow d$ while b and c show potential parallelism, i.e. $b \parallel c$.

The idea of the α-algorithm is to identify the relations between all pairs of tasks from the workflow log in order to reconstruct a process model based on Patterns (a) to (e). Therefore, before applying the α-algorithm, we first have to

Fig. 10.6 Footprint represented as a matrix of the workflow log $L = [\langle a,b,g,h,j,k,i,l\rangle, \langle a,c,d,e,f,g,j,h,i,k,l\rangle]$

	a	b	c	d	e	f	g	h	i	j	k	l
a	#	$\rightarrow$	$\rightarrow$	#	#	#	#	#	#	#	#	#
b	$\leftarrow$	#	#	#	#	#	$\rightarrow$	#	#	#	#	#
c	$\leftarrow$	#	#	$\rightarrow$	#	#	#	#	#	#	#	#
d	#	#	$\leftarrow$	#	$\rightarrow$	#	#	#	#	#	#	#
e	#	#	#	$\leftarrow$	#	$\rightarrow$	#	#	#	#	#	#
f	#	#	#	#	$\leftarrow$	#	$\rightarrow$	#	#	#	#	#
g	#	$\leftarrow$	#	#	#	$\leftarrow$	#	$\rightarrow$	#	$\rightarrow$	#	#
h	#	#	#	#	#	#	$\leftarrow$	#	$\rightarrow$	$\parallel$	#	#
i	#	#	#	#	#	#	#	$\leftarrow$	#	#	$\parallel$	$\rightarrow$
j	#	#	#	#	#	#	$\leftarrow$	$\parallel$	#	#	$\rightarrow$	#
k	#	#	#	#	#	#	#	#	$\parallel$	$\leftarrow$	#	$\rightarrow$
l	#	#	#	#	#	#	#	#	$\leftarrow$	#	$\leftarrow$	#

extract all basic order relations from the workflow log L. Consider the workflow log depicted in Fig. 10.4 containing the two cases $\langle a,b,g,h,j,k,i,l\rangle$ and $\langle a,c,d,e,f,g,j,h,i,k,l\rangle$. From this workflow log, we can derive the following relations.

- The basic order relations $>$ refer to each pair of tasks that directly follow one another. It can be directly read from the log:

$a > b$	$h > j$	$i > l$	$d > e$	$g > j$	$i > k$
$b > g$	$j > k$	$a > c$	$e > f$	$j > h$	$k > l$
$g > h$	$k > i$	$c > d$	$f > g$	$h > i$	

- The causal relations can be found when we check of each order relation whether it does not holds in the opposite direction. This holds for all pairs except (h, j) and (i, k), respectively. We get:

$a \rightarrow b$	$j \rightarrow k$	$a \rightarrow c$	$d \rightarrow e$	$f \rightarrow g$	$h \rightarrow i$
$b \rightarrow g$	$i \rightarrow l$	$c \rightarrow d$	$e \rightarrow f$	$g \rightarrow j$	$k \rightarrow l$
$g \rightarrow h$					

- The potential parallelism relation holds true for $h \parallel j$ as well as for $k \parallel i$ (and the corresponding symmetric cases).
- The remaining relation of no direction succession can be found for all pairs that do not belong to $\rightarrow$ and $\parallel$. It can be easily derived when we write down the relations in a matrix as shown in Fig. 10.6. This matrix is also referred to as the *footprint matrix* of the log.

Exercise 10.3 Have a look at the workflow log you constructed in Exercise 10.2. Define the relations $>$, $\rightarrow$, $\parallel$, # and the footprint matrix of this workflow log.

10.2.3 The α-Algorithm

The α-algorithm is a basic algorithm for automatic process discovery that takes an event log L and its α relations as a starting point. The essential idea of the algorithm is that tasks that directly follow one another in the log should be directly connected in the process model. Furthermore, if there is more than one task that can follow after another, we have to determine whether the set of succeeding tasks is partially exclusive or concurrent. An exception from the principle that tasks should be connected in the process model are those that are potentially parallel, i.e. those pairs included in $\|$. The details of the α-algorithm are defined according to the following eight steps.[3]

1. Identify the set of all tasks in the log as T_L.
2. Identify the set of all tasks that have been observed as the first task in some case as T_I.
3. Identify the set of all tasks that have been observed as the last task in some case as T_O.
4. Identify the set of all connections to be potentially represented in the process model as a set X_L. Add the following elements to X_L:
   a. Pattern (a): all pairs for which hold $a \rightarrow b$.
   b. Pattern (b): all triples for which hold $a \rightarrow (b\#c)$.
   c. Pattern (c): all triples for which hold $(b\#c) \rightarrow d$.
   Note that triples for which Pattern (d) $a \rightarrow (b \parallel c)$ or Pattern (e) $(b \parallel c) \rightarrow d$ hold are not included in X_L.
5. Construct the set Y_L as a subset of X_L by:
   a. Eliminating $a \rightarrow b$ and $a \rightarrow c$ if there exists some $a \rightarrow (b\#c)$.
   b. Eliminating $b \rightarrow c$ and $b \rightarrow d$ if there exists some $(b\#c) \rightarrow d$.
6. Connect start and end events in the following way:
   a. If there are multiple tasks in the set T_I of first tasks, then draw a start event leading to a split (XOR or AND depending on the relation between the tasks), which connects to every task in T_I. Otherwise, directly connect the start event with the only first task.
   b. For each task in the set T_O of last tasks, add an end event and draw an arc from the task to the end event.
7. Construct the flow arcs in the following way:
   a. Pattern (a): For each $a \rightarrow b$ in Y_L, draw an arc a to b.
   b. Pattern (b): For each $a \rightarrow (b\#c)$ in Y_L, draw an arc from a to an XOR-split, and from there to b and c.
   c. Pattern (c): For each $(b\#c) \rightarrow d$ in Y_L, draw an arc from b and c to an XOR-join, and from there to d.

[3]Note that the α-algorithm was originally defined for constructing Petri nets. The version shown here is a simplification based on the five simple control flow patterns of Fig. 10.5 in order to construct BPMN models.

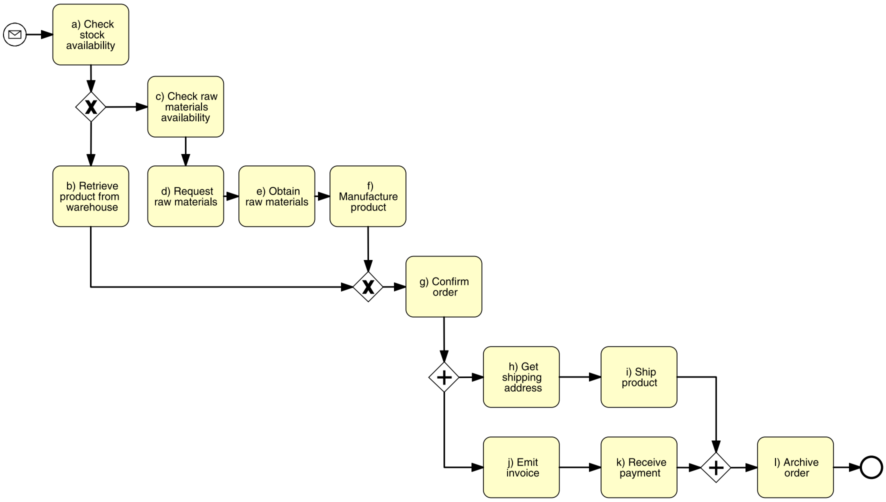


Fig. 10.7 Process model constructed by the α-algorithm from workflow log $L = [\langle a, b, g, h, j, k, i, l\rangle, \langle a, c, d, e, f, g, j, h, i, k, l\rangle]$

d. Pattern (d) and (e): If a task in the so constructed process model has multiple incoming or multiple outgoing arcs, bundle these arcs with an AND-split or AND-join, respectively.

8. Return the newly constructed process model.

Let us step through the α-algorithm with the workflow log $L = [\langle a, b, g, h, j, k, i, l\rangle, \langle a, c, d, e, f, g, j, h, i, k, l\rangle]$ as an example input. The Steps 1–3 identify $T_L = \{a, b, c, d, e, f, g, h, i, j, k, l\}$, $T_I = \{a\}$, and $T_O = \{l\}$. In Step 4a all causal relations are added to X_L including $a \to b$ and $a \to c$, etc. In Step 4b, we work row by row through the footprint matrix of Fig. 10.6 and check if there are cells sharing a $\to$ relation while relating to tasks that are pairwise in #. In the row a, we observe both $a \to b$ and $a \to c$. Also, $b\#c$ holds. Therefore, we add $a \to (b\#c)$ to X_L. We also consider row g and its relation to h and j. However, as $h \parallel j$ holds, we do not add them. In Step 4c, we progress column by column through the footprint matrix and see if there are cells sharing a $\to$ relation while relating to tasks that are mutually in #. In column g, we observe two $\to$ relations to b and f. Also, $b\#f$ holds. Accordingly, we add $(b\#f) \to g$ to X_L. We also check i and k that share the same relation to l. However, as $i \parallel k$ holds, we do not add them. There are no further complex combinations found in Step 4d.

In Step 5, we eliminate the basic elements in X_L that are covered by the complex patterns found in Steps 4b and 4c. Accordingly, we delete $a \to b$, $a \to c$, $b \to g$, and $f \to g$. In Step 6a we introduce a start event and connect it with a; in 6b, task l is connected with an end event. In Step 7, arcs and gateways are added for the elements of Y_L. Finally, in Step 8 the process model is returned. The resulting model is shown in Fig. 10.7.

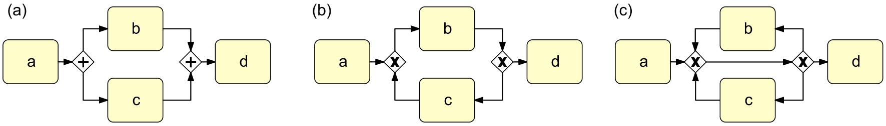


Fig. 10.8 Examples of two short loops, which are problematic for the α-algorithm

Exercise 10.4 Have a look at workflow log and the footprint you constructed in Exercises 10.2 and 10.3. Document progressing through the steps of the α-algorithm with this input and draw the resulting process model.

10.2.4 Robust Process Discovery

Clearly, the α-algorithm has its merits. It can reconstruct a process model from a behaviorally complete event log if that log has been generated from a structured process model. There are also limitations to be noted, though. The α-algorithm is not able to distinguish so-called *short loops* from true parallelism. As can be seen in Fig. 10.8, all three models can produce the workflow logs that yield $b \parallel c$ in the corresponding footprint. Several extensions to the α-algorithm have been proposed. The idea of the $\alpha+$-algorithm is to define the relation $\parallel$ in a stricter way such that $b \parallel c$ is only included if there is no sequence *bcb* in the logs. In this way, models (a) and (b) in Fig. 10.8 can be distinguished from each other in their generated logs. Furthermore, we can use preprocessing to extract direct repetition like *aa* or *bb* from the logs, note down the corresponding tasks, and continue with a log from which such repeated behavior is mapped to a single execution.

Further problems for the α-algorithm are *incompleteness* and *noise*. The notion of completeness assumed by the α-algorithm relates to the $>$ relation from which the other relations are derived. The number of required different cases increases with the factorial of the number of potentially concurrent tasks. Often, this number is low. But already for 10 concurrent tasks, we require $10! = 3,628,800$ cases. Therefore, it is desirable to use algorithms that can explicitly distinguish likely and unlikely behavior in order to generalize when logs are not complete. This direction also helps in addressing problems with noise. Event logs often include cases with a missing head, a missing tail, or a missing intermediate episode. Furthermore, there may be logging errors with events being swapped or recorded twice. Such unlikely behavior should not distort the process discovery result.

Several approaches have been defined for addressing problems of completeness and noise. In general, they try to balance four essential quality criteria, namely fitness, simplicity, precision, and generalization. *Fitness* refers to the degree of log behavior that a process model is able to replay. It can be defined based on the fraction of event patterns represented by the model or based on the fraction of cases that can be replayed in the model. *Simplicity* means that the resulting process model should be readily understandable. It can be measured using different complexity metrics for process models such as model size or degree of structuredness. *Precision* refers to the degree of behavior that is allowed by the model, but not observed in the logs. We can easily create a process model that allows the execution of all tasks in any arbitrary order with potential repetition. However, we can hardly learn any specifics of the process from such a model. *Generalization* refers to the ability of a process model to abstract from the behavior that is documented in the logs. A discovery technique that is able to generalize helps to work with incomplete behavior.

Exercise 10.5 Draw a model in BPMN with which you can replay any execution sequence that includes tasks a, b, c, d, e. Furthermore, discuss the fitness, simplicity, precision, and generalization of such a model for the trace $\langle a, b, c, d, e \rangle$.

10.3 Performance Analysis

In Sect. 7.1 we introduced the four performance dimensions of time, cost, quality and flexibility. In turn, Chap. 8 demonstrated that these four dimensions form a Devil's Quadrangle when we try to improve a process. These performance measures are usually considered to be generally relevant for any kind of business. Beyond this general set, a company should also identify specific measures. Often, the measures are industry-specific, like profit per square-meter in gastronomy, return rate in online shopping or customer churn in marketing. Any specific measure that a company aims to define should be accurate, cost-effective and easy-to-understand. Here, we focus on the four general performance measures of time, cost, quality and flexibility. The question of this section is how we can spot that a process does not perform well according to one of these dimensions. Event logs provide us with very detailed data that is relevant to performance. We will describe techniques that help us to measure and to visualize potential performance problems that relate to time, cost, quality and flexibility.

10.3.1 Time Measurement

Time and its more specific measures *cycle time* and *waiting time* are important general performance measures. Event logs typically show timestamps such that they can be used for time analysis. Time analysis is concerned with the temporal occurrence and probabilities of different types of event. The event logs of a process generally relate each event to the point in time of its occurrence. Therefore, it is straightforward to plot events on the time axis. Furthermore, we can utilize classifiers to group events on a second axis. A classifier typically refers to one of the attributes of an event, like case ID or participant ID. There are two levels of detail for plotting events in a diagram: *dotted charts* using the timestamp to plot an event, and *timeline chart* showing the duration of a task and its waiting time.

The dotted chart is a simple, yet powerful visualization tool for event logs. Each event is plotted on a two-dimensional canvas with the first axis representing its

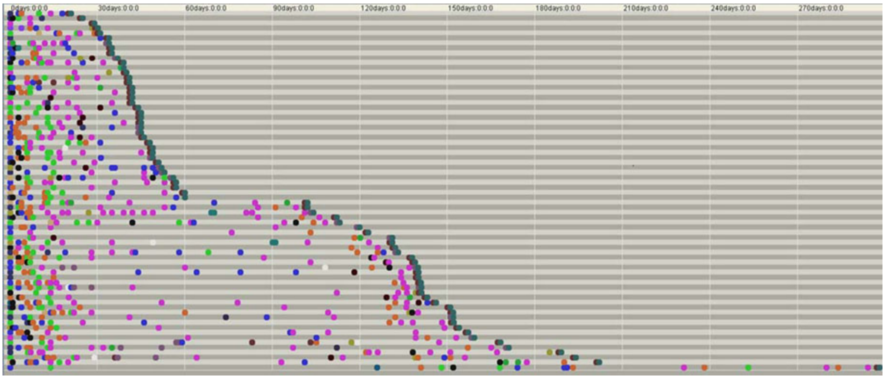


Fig. 10.9 Dotted chart of log data

occurrence in time and the second axis as its association with a classifier like a case ID. There are different options to organize the first axis. Time can be represented either *relative* such that the first event is counted as zero, or *absolute* such that later cases with a later start event are further right in comparison to cases that started earlier. The second axis can be sorted according to different criteria. For instance, cases can be shown according to their historical order or their overall cycle time.

Figure 10.9 shows the dotted chart of the logs of a healthcare process. The events are plotted according to their relative time and sorted according to their overall cycle time. It can be seen that there is a considerable variation in terms of cycle time. Furthermore, the chart suggests that there might be three distinct classes of cases: those that take no more than 60 days, those taking 60 to 210 days, and a small class of cases taking longer than 210 days. Such an explorative inspection can provide a good basis for a more detailed analysis of the factors influencing the cycle time.

Exercise 10.6 Draw a dotted chart of the event log in Fig. 10.1 showing relative cycle time and being sorted by cycle time.

The temporal analysis of event logs can be enhanced with further details if a corresponding process model is available and tasks can be related to a start and to an end event. The idea is to utilize the concept of token replay for identifying the point in time when a task gets activated. For tasks in a sequence, the activation time is the point in time when the previous task completed. For tasks after an AND-join, this is the point in time when all previous tasks have completed. For XOR-joins and splits it is the point when one of the previous tasks completes.

Using this information, we can plot a task not as a dot but as a bar in a timeline chart (see Fig. 10.10). A timeline chart shows a waiting time (from activation until starting) and a processing time (from starting until completion) for each task. The timelines of each task can be visualized in a similar way as a dot in the dotted chart. The timeline chart is more informative than the dotted chart since it shows the duration of the tasks. Furthermore, it is helpful to see the waiting times. Both

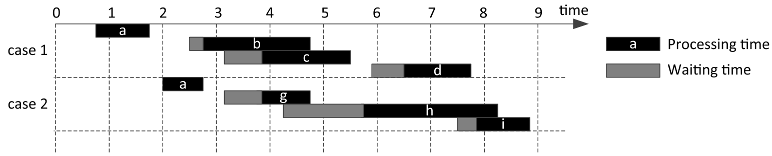


Fig. 10.10 Timeline chart of log data like PM 232

pieces of information are a valuable input for quantitative process analysis. When thousands of cases are available as a log, one can estimate the distribution of waiting time and processing time of each task. In this way, bottlenecks with long waiting times can be spotted and, similarly, tasks that are most promising to focus redesign efforts upon. Furthermore, this information can also be used for predicting execution times of running process instances, which is helpful for process monitoring.

Exercise 10.7 Calculate the waiting times required by a timeline chart for the event log in Fig. 10.1 using the process model of Fig. 10.7.

10.3.2 Cost Measurement

In a process context, cost measurement is mainly related to the problem of assigning indirect costs to cases. After all, direct costs like the purchasing costs of four wheels which are assembled on a car can be easily determined. Indirect labor or machine depreciation are more difficult. In accounting, the concept of *Activity-based Costing* (*ABC*) was developed to more accurately assign indirect costs to products and services, and to individual customers. The motivation of ABC is that human resources and machinery are often shared by different products and services, and they are used to serve different customers. For instance, the depot of BuildIT rents out expensive machinery such as bulldozers to different construction sites. On the one hand, that involves costs in terms of working hours of the persons working at the depot. On the other hand, machines like bulldozers lose in value over time and require maintenance. The idea of ABC is to use activities for distributing the indirect costs, e.g. associated with the depot.

Example 10.1

According to Fig. 1.6 in Chap. 1, the rental process of BuildIT contains five major activities. We observe the following durations in the event logs for the case of a bulldozer rental requested on 21st August:

- “Submit equipment rental request” is conducted by the site engineer. It takes the engineer 20 minutes to fill in the form on paper. The production of each paper form costs €1. The site engineer gets an annual salary of €60,000.
- The clerk receives the form and selects suitable equipment and checks the availability. Both activities together take 15 minutes. The clerk works at an annual rate of €40,000.
- The works engineer reviews the rental request (annual salary of €50,000). This review takes 10 minutes.

- The clerk is also responsible for sending a confirmation including a purchase Order for renting the equipment, which takes 30 minutes.

In order to work with these numbers we have to make certain assumptions. First, at BuildIT the actual working year contains 250 working days of 8 hours. Furthermore, all employees receive health insurance and pension contributions of 20 % on top of their salary. Finally, people are on average 10 days on a sick leave per year. Taking this into account, we can calculate the labor cost of each participant per minute as $\frac{salary \times 120\ \%}{(250-10)\times 8 \times 60}$. This is for the site engineer €0.63 per minute, for the clerk €0.42 per minute, and for the works engineer €0.52 per minute. Altogether, this case created costs of 20 minutes × €0.63 per minute + (15 + 30) minutes × €0.42 per minute + 10 minutes × €0.52 per minute, which sums up to €36.70.
Now consider a case of a street construction process. We observe the following durations from the event logs for these two activities:

- "Prepare the foundation" is conducted by four worker. It took one week at a specific construction case. The used excavator costs €100,000. It is written off in five years and has annual maintenance costs of €5,000. A worker gets an annual salary of €35,000.
- "Tar the road" is conducted by six workers. It took two days in this case. The tarring machine costs €200,000, is also written off in five years and costs €10,000 annual maintenance.

For this case, we can also take the costs of the machinery into account. The labor cost per day is €175 for one worker. The excavator costs €20,000 + 5,000 per annum for write-off and maintenance, which is €104.17 per working day. For the preparation of the foundation, this amounts to 4 × 5 × €175 + 5 × €104.17 = €4,020.85. For the tarring of the road, the costs are 6 × 2 × €175 + 2 × €208.34 = €2,516.68.

Exercise 10.8 Consider that the paper form is printed by the site engineer in the rental process, that the printer costs €300 written off in three years, and that a pile of 500 pieces of paper costs €10. Why could it make sense to include these costs in the calculation, why not?

An inherent problem of ABC is the detail of data that is required for tracking the duration of activities like renting out equipment or approving rental requests. Event data stored in process-aware information systems can help to provide such data. Also, technologies like Radio-frequency identification (RFID), which helps to track physical objects based on little RFID chips attached to it, are promising to overcome the tracking problem of ABC. Some systems only keep track of activity completion. However, ABC also requires the start of activities to be kept. This can be achieved by tracking timestamps of the point in time when a resource starts working on a specific task. What is important to take into account here is the cost of achieving additional transparency. There is a trade-off, and once it becomes overly expensive to gain more transparency, it is good to not include those costs in the calculation.

10.3.3 Quality Measurement

The quality of a product created in a process is often not directly visible from execution logs. However, a good indication is to check whether there are repetitions in the execution logs, because they typically occur when a task has not been completed successfully. Repetitions can be found in sequences of task. In Chap. 7, we saw that the loop of a rework pattern increases the cycle time of a task to $CT = \frac{T}{1-r}$ in comparison to T being the time to execute the task only once. The question is now how we can determine the repetition probability r from a series of event logs?

The first part of the answer to this question can be given by reformulating the equation such that it is solved for r. By multiplication with $1 - r$ we get $CT - r \times CT = T$. Subtraction of CT yields $-r \times CT = T - CT$, which can be divided by $-CT$ resulting in

$$r = 1 - \frac{T}{CT}.$$

Both CT and T can now be determined using the data of the event logs. Consider the five cases in which we observe the following execution times for task a:

1. 5 minutes, 10 minutes.
2. 10 minutes.
3. 20 minutes, 6 minutes, 10 minutes.
4. 5 minutes.
5. 10 minutes, 10 minutes.

The cycle time CT of a can now be calculated as the average execution time of a per case, while the average execution time T is the average execution time of a per instantiation. Both can be determined based on the sum of all executions of a, which is 86 minutes here. We have five cases, such that $CT = 86/5 = 17.2$. Altogether, a is executed nine times yielding $T = 86/9 = 9.56$. Hence, we get $r = 1 - \frac{86/9}{86/5} = 1 - \frac{5}{9} = 0.44$. Of course, this calculation is only an approximation of the real value for r. It builds on the assumption that the duration of a task always follows the same distribution, no matter if it is the first, the second or another iteration.

Exercise 10.9 Determine the probability of repetition r for the following execution times of task b:

1. 20 minutes, 10 minutes.
2. 30 minutes.
3. 30 minutes, 5 minutes.
4. 20 minutes.
5. 20 minutes, 5 minutes.
6. 25 minutes.

Also explain why the value is misleading for these logs.

In some information systems it might be easier to track repetition based on the assignment of tasks to resources. One example are *ticketing systems* that record which resource is working on a case. Also, the logs of these systems offer insights into repetition. A typical process supported with ticketing systems is incident resolution. For example, an incident might be a call by a customer who complains that the online banking system does not work. Such an incident is recorded by a dedicated participant, e.g. a call center agent. Then, it is forwarded to a first-level support team who tries to solve the problem. In case the problem turns out to be too specific, it is forwarded to a second-level support team with specialized knowledge in the problem domain. In the best case, the problem is solved and the customer notified. In the undesirable case, the team identifies that the problem is within the competence area of another team. This has the consequence that the problem is rooted back to the first-level team. Similar to the repetition of tasks, we now see that there is a repeated assignment of the problem to the same team. According log information can be used to determine how likely it is that a problem is rooted back.

10.3.4 Flexibility Measurement

Flexibility refers to the degree of variation that a process permits. This flexibility can be discussed in relation to the event logs the process produces. For the company owning the process, this is important information in order to compare the desired level of flexibility with the actual flexibility. It might turn out that the process is more flexible than what is demanded from a business perspective. This is the case when flexibility can be equated with lack of standardization. Often, the performance of processes suffers when too many options are allowed. Consider again the process for renting equipment at BuildIT. The process requires an equipment rental request form to be sent by e-mail. Some engineers, however, prefer to call the depot directly instead of filling the form. Since these engineers are often highly distinguished, it is not easy for the clerk to turn down these calls. As a result, the clerk fills out the form while being on the phone. Not only does this procedure take more time, but also, due to noise at the construction site, it increases the likelihood of errors. In practice, this means that the rental process has two options to submit a request: by form (the standard procedure) and via the phone.

Partially, such flexibility as described above can be directly observed in the event logs. We have seen that the workflow log of a process plays an important role for automatic process discovery. It can also be used to assess the flexibility of the process. The workflow log summarizes the essential behavior of the process. As it defines each execution as a sequence of tasks, it abstracts from the temporal distance between them. In this way, the workflow log contains a set of traces that have a unique sequence. This means that if two executions contain the same sequence of tasks, then they only result in a single trace to be included in the workflow log. This abstraction of process executions in terms of a workflow log makes it a good starting point for discussing flexibility. Accordingly, the number of *distinct executions DE* can be defined based on a workflow log L as

$$DE = |L|.$$

Exercise 10.10 Consider the event logs of the order fulfillment process in Fig. 10.1. What is the number of distinct executions *DE*?

The question arises whether the number of distinct executions always gives a good indication for flexibility. At times, the number of distinct executions might give an overly high quantification of flexibility. This might be the case for processes with concurrency. Such processes can be highly structured, but having only a small set of concurrent tasks results in a rich set of potential execution sequences. Consider the process model constructed by the α-algorithm in Fig. 10.7. The tasks i and h are concurrent to j and k. Indeed, there are six options to execute them:

1. i, h, j, k
2. j, k, i, h
3. i, j, k, h
4. j, i, h, k
5. i, j, h, k and
6. j, i, k, h

While the order is not strict, all of them must be executed. Therefore, it might be a good idea to additionally consider whether a task is optional. If T refers to the number of tasks that appear in the workflow log, then the set T_{opt} contains those tasks that are optional. Optionality according to the log means that for a particular task there exists at least one trace in which it does not occur. For the logs of Fig. 10.1, we observe that the tasks b to f depend upon the availability of raw materials. We can quantify the degree of *optionality OPT* as

$$OPT = \frac{T_{\text{opt}}}{T}.$$

We can also approach the question of flexibility from the perspective of the automatically discovered process model. This has some advantages since the degree of optionality does not reveal how many decisions have to be taken. If we consider the process model constructed by the α-algorithm in Fig. 10.7, we see that one decision node is included (XOR-split). It distinguishes the situation whether the product is in stock or not. This observation can be quantified as the number of *discovered decision points* (*DDP*). For instance, the α-algorithm can be used to this end.

10.4 Conformance Checking

While performance analysis is concerned with measuring performance indicators, conformance checking is concerned with the question whether or not the execution of a process follows predefined rules or constraints. This question can be answered by inspecting the event logs. If a particular constraint does not hold, we speak of a *violation*. Conformance checking is concerned with identifying these violations and making statements about the extent of them altogether. This information provides important insights for the process owner. Apparently, when rules are not followed, one should take corrective action. Violations might relate to one of the three process perspectives of control flow, data and resources in isolation or in combination. In the following, we describe how they can be specified.

10.4.1 Conformance of Control Flow

Conformance of control flow can be studied in two ways, either based on *explicit constraints* or based on a *normative process model*. Both can be related to each other, as many constraints can be automatically derived from a process model. First we will look into the approach assuming explicit constraints, after which we will discuss the use of a normative process model.

We focus on three types of control-flow related constraint: mandatoriness, exclusiveness, and ordering. All these three constraint types define how two activities are allowed to be related in a process. A company might want to define that certain activities are *mandatory* because they are required from a control perspective. Consider again the case of BuildIT and the equipment rental process. A works engineer is supposed to review the rental request. This activity serves as a control for securing that only appropriate equipment is rented. This measure might help to keep the rental costs in line. Such a control activity is a likely candidate for being a mandatory activity. On the level of the logs, mandatory activity violations can be found by searching for traces in which they are left out. *Exclusiveness* might be specified for activities that relate to a decision. If, for instance, a rental request is rejected, BuildIT wants to make sure that there is no way to overwrite this decision. On the level of the log, this means that it shall never be possible that a rejection of a request is followed by an approval. This exclusiveness can be checked by searching for traces in which both exclusive activities appear. The *order* of activities might be of specific importance to balance the performance of the process. In the equipment rental process, first the availability of the requested equipment is checked before the request is reviewed. This order constraint helps to relieve the workload of the works engineer who is supposed to review the request. Obviously, it is a waste of effort to review requests that cannot be met because the equipment is not available. Violations to order constraints can be found by searching for traces with the activities appearing in the wrong order.

Exercise 10.11 Consider the event logs of the order fulfillment process in Fig. 10.1. Which activities can be considered mandatory and exclusive to one another?

Conformance of control flow can also be checked by comparing the behavior observed in the logs with a normative process model. The idea here is to replay each trace in the workflow log and to record at each step whether an activity was allowed to be executed according to the rules of the model. Typically, this requires the replay of the logs in the process model. Based on the transition rules of BPMN, we can replay the case $\langle a, b, g, i, j, k, l\rangle$ on the model shown in Fig. 10.11. In the initial

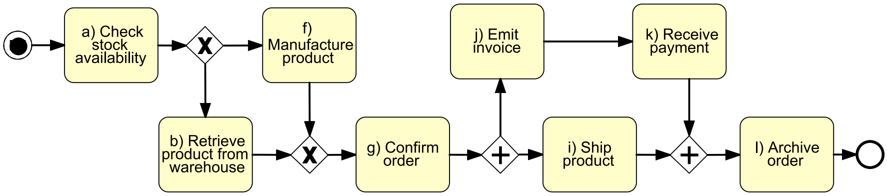


Fig. 10.11 BPMN model with token on start event for replaying the case $\langle a, b, g, i, j, k, l \rangle$

state, the process has a token on the start event. Once the case is started, this token moves to the output arc of the start event. This arc leads to activity a ("Check stock availability"), which means that the token enables this activity to be executed. The token moves to this activity while it is executed, and it is forwarded to the output arc once the activity is completed. Now, the XOR-split is activated, which means a decision has to be taken to either continue with b ("Retrieve product from warehouse") or with f ("Manufacture product"). For the considered case, we continue with b. In the same way, we can continue. After g ("Confirm order") has been completed, we arrive at an AND-split. An AND-split consumes a token from its input arc and creates one token on each of its output arcs. As a result, we have two tokens afterwards: one enabling i ("Ship product") and one enabling j ("Emit invoice"). In this state we can proceed either with i or j. These activities are concurrent. In order to replay the case, we first execute i and j afterwards. Once i and later k is completed, the AND-join is allowed to proceed. One token on each of its input arcs is required for that. Both these tokens are consumed and a single token is created on its output arc. As a result, l ("Archive order") can be executed.

Based on the concept of token replay, we can also assess the conformance of a trace to a process model. Consider the case $\langle a, b, i, j, k, l \rangle$ in which the confirmation of the order has been omitted. The idea is to compare at each step the number of tokens that are required for replaying an activity with tokens that are actually available. At each step, we might observe two situations of conformance and two of non-conformance. In case of conformance, we can count the following four facts for tokens:

- p: the number of tokens that are correctly produced
- c: the number of tokens that are correctly consumed
- m: the number of tokens that are missing for executing the next activity in the log, and
- r: the number of tokens remaining unconsumed after executing the final activity in the log

Figure 10.12 first shows the state before replaying b of the case $\langle a, b, i, j, k, l \rangle$. After replaying b, there is a token available to execute g, but it is not consumed. Instead, there is a missing token for firing the AND-split, which would activate i and j. The figure also shows the number of correctly produced and consumed tokens for each step until the completion. Using the four measures c, p, m and r, we can calculate the fitness measure as an indicator of conformance. It is defined

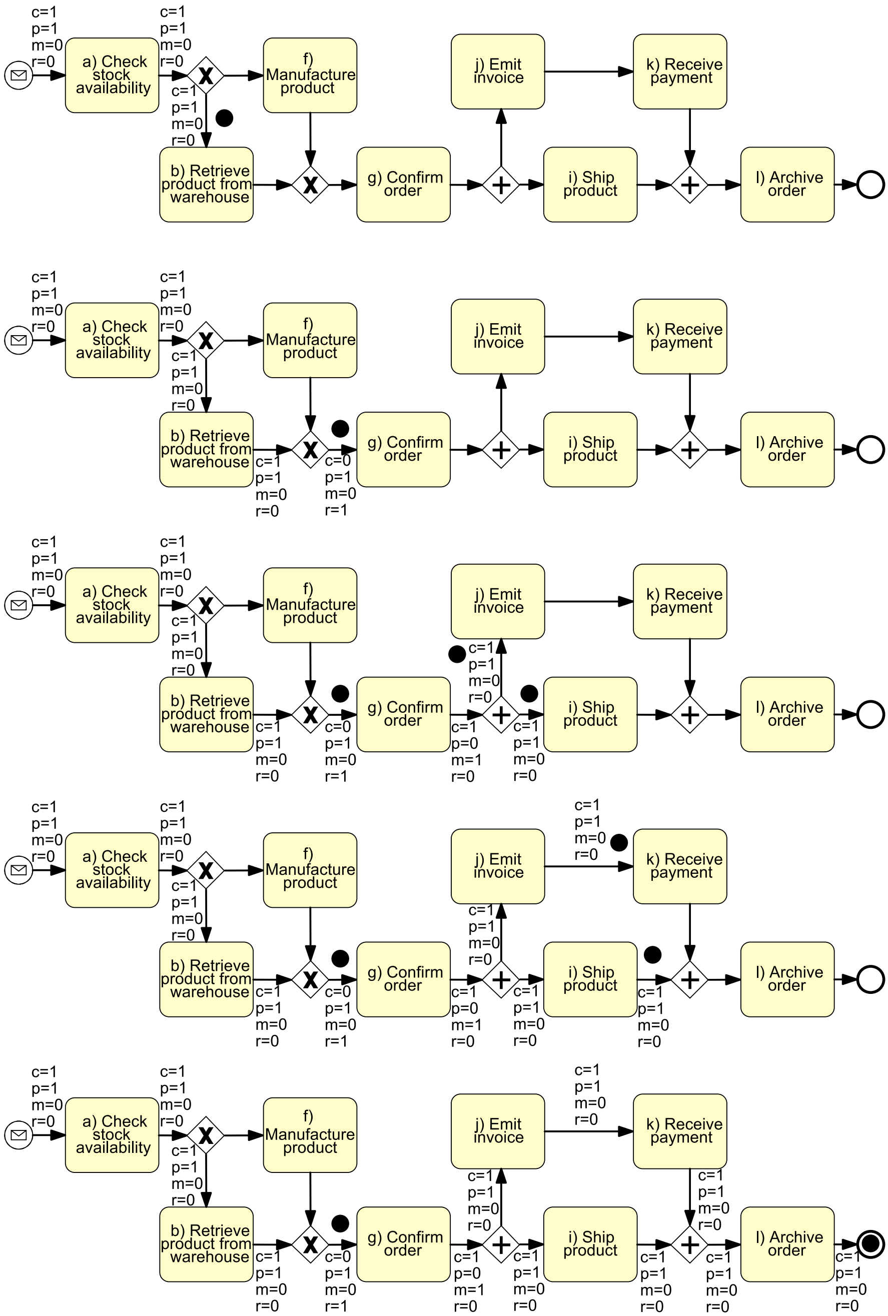


Fig. 10.12 Replaying the non-conforming case $\langle a, b, i, j, k, l \rangle$

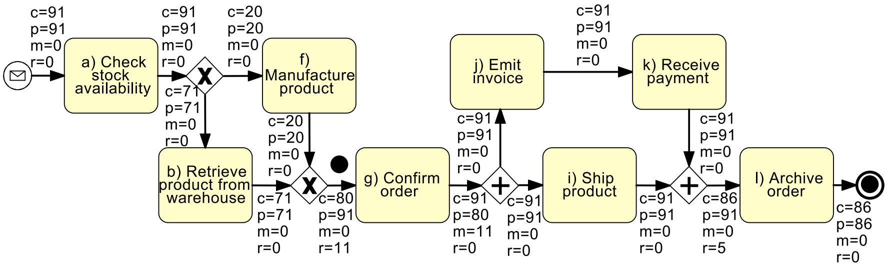


Fig. 10.13 Result of replaying cases in the process model

based on the fraction of missing tokens to correctly consumed tokens ($\frac{m}{c}$) and the fraction of remaining tokens to produced tokens ($\frac{r}{p}$) as

$$fitness = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right).$$

For our values $c = 12$, $p = 12$, $m = 1$ and $r = 1$, we get a fitness of $\frac{1}{2}(1 - \frac{1}{12}) + \frac{1}{2}(1 - \frac{1}{12}) = 0.8333$. When we consider a set of cases, not just a single case, we can easily calculate the fitness in the same way. The idea is to simply continue counting c, p, m and r by replaying the next case in the process model. Once we have replayed all cases, we get the resulting overall fitness of this set of cases.

The results of such a conformance analysis can be interpreted in two ways. First, we can use the overall fitness measure to get an idea of how accurately the process model matches the actually observed behavior as reflected by the set of cases. While the fitness as an overall measure is useful to this end, it does not help us to analyze the deviations in more detail. Therefore, and secondly, we can inspect at which arcs of the process model we have encountered missing or remaining tokens. Figure 10.13 shows the corresponding numbers from replaying several cases in the process model. It can be seen that apparently most deviations relate to activity g and some to activity l. This information can be utilized to interview process participants why g has been omitted for some cases. The goal of such an inquiry would be to find whether this omission is desirable. The question is whether the process participants have found a more efficient way to handle the confirmation, or whether an omission has to be considered bad practice which should be further discouraged. For the case of the archival (activity l), the omission is likely to be bad practice.

10.4.2 Conformance of Data and Resources

Beyond constraints on the control flow, there are often additional constraints on data and resources. Consider the situation that an expensive caterpillar is requested for a construction site of BuildIT. Many companies have extra rules for high expenses or risky commitments. In the case of BuildIT, the rent of the caterpillar requires the additional signature of a manager. Similar cases can be found in banks where a loan of more than €1,000,000 might require the additional sign-off from a director. There might also be constraints that providing a loan to a black-listed applicant is simply not allowed at all. Such constraints can be checked by searching for cases in the log where a certain data field takes a forbidden value.

The example of the additionally required signature already points to a combination of data and resource constraints. If a certain amount is exceeded, then there is a dedicated resource who is required to approve. However, there are also constraints that purely refer to the resource perspective. Participants usually require *permissions* to execute certain activities. For instance, the person signing off the caterpillar rent has to be a manager, and it is not allowed that this person is a construction worker. Typically, permissions are bundled for specific *roles*. For example, this can be done by explicitly listing what a manager and a construction worker are allowed to do. Violations of permissions can be checked by searching for each activity conducted by a participant whether or not an appropriate role or permission existed. Specific control rules which require two different persons to approve a business transaction are called *separation of duties* constraints. These rules do not necessarily involve supervisors. For example, it might be OK with a particular bank if a loan of €100,000 is signed by two bank clerks, while a €1,000,000 loan requires the signature of a clerk and a director. Such separation of duties constraints can be checked by searching for cases in the log where the same participant or two participants with the same role have approved the same transaction.

Exercise 10.12 Consider the intake process after the medical file redesign, which is shown in Fig. 8.3. Which separation of duties constraints would you define for this process?

10.5 Recap

In this chapter we discussed the topic of process intelligence. The starting point for process intelligence is the availability of event logs and, in general, data on the execution of processes. These data provides the basis for process monitoring and process controlling. Such event logs have to refer to a specific case, a task and a point in time in order to facilitate process-related analysis. In many cases, it is a challenge to extract data in such a way that it can readily support process intelligence.

An important area of process intelligence is automatic process discovery. Corresponding techniques like the α-algorithm yield process models describing how a process runs in reality according to the log data. The α-algorithm is a good example for how automatic process discovery works. However, it has some limitations in terms of robustness, which are addressed by more recent process mining algorithms.

Event logs also support the assessment of the performance of a process. We discussed the four dimensions of the Devil's Quadrangle. The time dimension of a process can be visualized as a dotted chart and further inspected using a timeline chart. Then, we demonstrated that the calculation of costs for a process heavily depends upon the level of detail with which execution times of tasks are captured. We turned to quality and related it to the number of repetitions that are encountered in a log. Finally, we discussed ways of quantifying the flexibility of a process based on the number of distinct executions, optionality and discovered decision points.

The area of conformance checking can be related to various types of constraint. There are different types of constraint that can be checked for control flow including mandatoriness, exclusiveness and ordering of activities. A notion of fitness can be calculated based on replaying the logs in a normative process model. Finally, we discussed important types of constraint for the data and the resource perspectives. Often, these are intertwined. They typically relate to additional requirements for approval beyond a certain monetary threshold or to enforce separation of duties in order to control risks of a business transaction.

10.6 Solutions to Exercises

Solution 10.1 It might be the case that parts of the production process are administered using different information systems. Accordingly, the event logs have to be integrated. In terms of correlation, this means that case identifiers from different systems have to be matched. If the timestamps of the events are recorded in different time zones, they have to be harmonized. Shipment might not be arranged by Airbus. Therefore, it might be the case that different snapshots of transportation might not be accessible. Also, information systems might not be directly producing case-related event logs. The data would then have to be extracted from the databases of these systems. Finally, the events might be recorded at diverging levels of granularity, ranging from a detailed record of production steps to coarse-granular records of transportation stages.

Solution 10.2 The workflow log considers the order of events for each case. We use letters a to l for referring to the tasks. The workflow log L containing three elements as the first and the fourth execution sequence are the same. Therefore, we get $L = [\langle a, b, g, h, j, k, i, l\rangle, \langle a, c, d, e, f, g, j, h, i, k, l\rangle, \langle a, c, f, g, j, h, i, k, l\rangle]$.

Solution 10.3 The following basic relations can be observed:

$a > b$	$h > j$	$i > l$	$d > e$	$g > j$	$i > k$
$b > g$	$j > k$	$a > c$	$e > f$	$j > h$	$k > l$
$g > h$	$k > i$	$c > d$	$f > g$	$h > i$	$c > f$

The matrix shows the resulting relations.

	a	b	c	d	e	f	g	h	i	j	k	l
a	$\#$	$\rightarrow$	$\rightarrow$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$
b	$\leftarrow$	$\#$	$\#$	$\#$	$\#$	$\#$	$\rightarrow$	$\#$	$\#$	$\#$	$\#$	$\#$
c	$\leftarrow$	$\#$	$\#$	$\rightarrow$	$\#$	$\rightarrow$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$
d	$\#$	$\#$	$\leftarrow$	$\#$	$\rightarrow$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$
e	$\#$	$\#$	$\#$	$\leftarrow$	$\#$	$\rightarrow$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$
f	$\#$	$\#$	$\leftarrow$	$\#$	$\leftarrow$	$\#$	$\rightarrow$	$\#$	$\#$	$\#$	$\#$	$\#$
g	$\#$	$\leftarrow$	$\#$	$\#$	$\#$	$\leftarrow$	$\#$	$\rightarrow$	$\#$	$\rightarrow$	$\#$	$\#$
h	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\leftarrow$	$\#$	$\rightarrow$	$\parallel$	$\#$	$\#$
i	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\leftarrow$	$\#$	$\#$	$\parallel$	$\rightarrow$
j	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\leftarrow$	$\parallel$	$\#$	$\#$	$\rightarrow$	$\#$
k	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\parallel$	$\leftarrow$	$\#$	$\rightarrow$
l	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\leftarrow$	$\#$	$\leftarrow$	$\#$

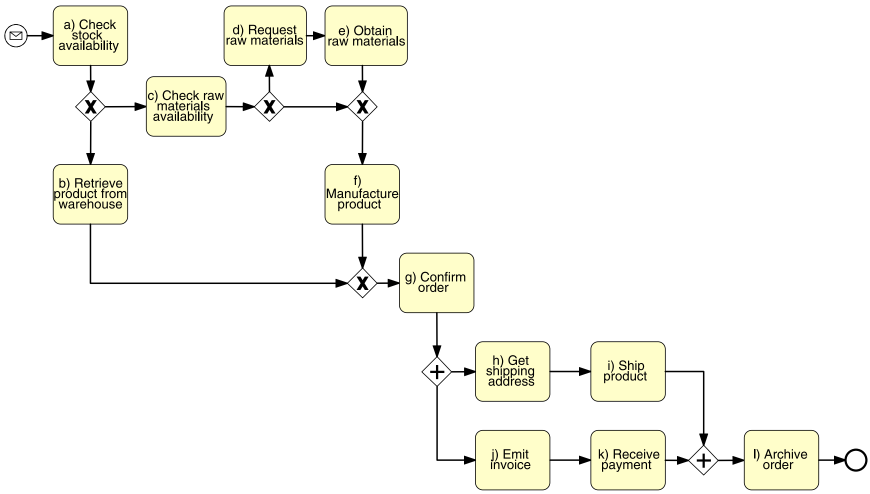


Fig. 10.14 Process model constructed by the α-algorithm

Solution 10.4 The α-algorithm stepwise yields the following sets:

1. $T_L = \{a, b, c, d, e, f, g, h, i, j, k, l\}$.
2. $T_I = \{a\}$.
3. $T_O = \{l\}$.
4. $X_L = Z_1 \cup Z_2$ with $Z_1 = \{a \rightarrow b, a \rightarrow c, b \rightarrow g, c \rightarrow d, c \rightarrow f, d \rightarrow e, e \rightarrow f, f \rightarrow g, g \rightarrow h, g \rightarrow j, h \rightarrow i, i \rightarrow l, j \rightarrow k, k \rightarrow l\}$ and $Z_2 = \{a \rightarrow (b\#c), c \rightarrow (d\#f), (c\#e) \rightarrow f, (b\#f) \rightarrow g\}$
5. $Y_L = Z_2 \cup \{d \rightarrow e, g \rightarrow h, g \rightarrow j, h \rightarrow i, j \rightarrow k, i \rightarrow l, k \rightarrow l\}$.
6. Add start event pointing to a and end event following after l.
7. Construct process model based on Y_L with XOR- and AND-gateways.
8. Return process model, see Fig. 10.14.

Solution 10.5 Include a, b, c, d, e in an XOR-block and put this XOR-block in an XOR-loop. This model shows a perfect fitness to the execution sequence as it is able to replay any occurrence of a to e at any stage. The model is not completely simple as it includes four gateways for characterizing the behavior of five activities. Also, the model is not very precise in this sense that it does not introduce specific constraints on the behavior: any occurrence of a to e is allowed at any stage. Generalization refers to the ability of the model to abstract. As the model does not constrain the behavior, there is hardly any general insight that we can learn from it.

Solution 10.6 First, we have to determine the cycle time. Case 1 takes roughly two hours less than seven days, case 2 one hour less than six days, case 3 takes four days and 20 minutes, and case 4 takes four days and six hours. Therefore, the relative order must be case 3, case 4, case 2 and case 1. Each event has to be visualized according to the time elapsed since the first event of the case.

Solution 10.7 The timeline diagram follows the same principle as the dotted chart. Additionally, it shows waiting times as soon as an activity becomes enabled in the process model.

Solution 10.8 In general, it is a good idea to include all expenses in the cost calculation as it provides transparency on what is spent where. In this case, however, it might not be a good idea since the cost per paper is relatively low with €0.02. It has to be kept in mind though that the decision whether to record certain costs should not be governed by the unit piece, but by the relative impact on cost calculation. A cost of paper of €0.02 is small as compared to overall process costs of several thousand Euros. However, it can be relatively high if the overall process produces €0.30 costs per instance and millions of instances are processed per day. Think of the situation of processing bank transactions using paper forms versus using online banking. The high amount of transactions per day might result in considerable costs per year.

Solution 10.9 The formula yields $r = 1 - \frac{T}{CT} = 1 - \frac{18.3}{27.5} = 0.333$. Apparently, this result is misleading for the figures because every second task required rework. However, the rework time was in general much smaller as the first try duration. This has the effect that T appears to be relatively small in comparison to CT, which results in a low value for r.

Solution 10.10 We have to consider four different cases. However, as the first and the fourth case show the same sequence of tasks, there are three distinct executions.

Solution 10.11 There are several activities that can be observed in all cases. These mandatory activities are a, g, h, i, j, k, l. The other activities b, c, d, e, f are not mandatory. Exclusiveness relationships hold between b and c, d, e, f pairwise.

Solution 10.12 The intake process demands meeting two intakers. The persons conducting "Meet with first intaker" and "Meet with second intaker" should be mutually exclusive. Therefore, we can identify a separation of duty constraint that the participant conducting "Meet with first intaker" should be another one as the participant conducting "Meet with second intaker".

10.7 Further Exercises

Exercise 10.13 Download the process mining tool ProM, write down the workflow log L in XES, and run the α-algorithm.

Exercise 10.14 Apply the α-algorithm and document the different steps for the workflow log L containing four cases $L = [\langle a, b, c, d\rangle, \langle a, b, d, c\rangle, \langle a, b, d, c, e\rangle, \langle a, c, d, e\rangle]$.

Exercise 10.15 Apply the α-algorithm and document the different steps for the workflow log L containing four cases $L = [\langle a, b, c, d, e, f\rangle, \langle a, b, d, c, e\rangle, \langle a, b, d, c, e, f\rangle, \langle a, b, c, d\rangle]$.

Exercise 10.16 Consider there is an AND-split in a process model with two subsequent activities a and b. Which kind of pattern do these activities show on the timeline chart?

Exercise 10.17 Consider the workflow log, which you created for Exercise 10.2 from the cases shown in Fig. 10.1. Replay these logs in the process model of Fig. 10.13. Note down consumed, produced, missing and remaining tokens for each arc, and calculate the fitness measure. Assume that activities not shown in the process model do not change the distribution of tokens in the replay.

10.8 Further Reading

Process intelligence relates to various streams of current research. The workflow resource patterns give an extensive account of strategies that can be used for effectively assigning work items to participants at the time of execution. Process mining has been a very influential area of research in the recent years, offering various tools and techniques for process intelligence. An excellent overview of this research area is provided in the book by van der Aalst on process mining [94]. Among others, it summarizes work on leveraging shortcomings of the α-algorithm, namely the heuristic miner, the fuzzy miner, and the genetic miner. The idea of the *heuristic miner* is to generate so-called heuristic nets in which the arc probabilities are included. Using appropriate threshold values, the user can eliminate unlikely behavior. The *fuzzy miner* takes correlations between events and behavior into account. In this way, tasks can be clustered and inspected at different levels of abstraction. The *genetic miner* generates a population of potential models and stepwise manipulates these models using change operations and quality metrics in order to improve the approximation of the log. This process is repeated until a model with a predefined quality level is found.

Various guiding principles and research challenges for process mining have been formulated by the IEEE Task Force on Process Mining in the Process mining manifesto [37]. There are dedicated tools that help with putting process mining into practice (see http://www.processmining.org). Process mining tools usually cover automatic process discovery and techniques for conformance checking.

A good summary of current research on process performance analysis is provided by the BPM Handbook [102, 103] and, most notably, its chapters on process performance management and on business process analytics [33, 111]. A good overview on process controlling and its relationship to process automation is the book by zur Muehlen [110]. More generally, the book by Harmon provides a good perspective on how to define process measures within a process governance framework [32]. A good book on foundations on performance from an operations management point of view is the book by Anupindi et al. [4]. Various metrics for event logs are discussed in the Ph.D. thesis of Günther [25].

Conformance checking is also nicely covered in the book by van der Aalst on process mining [94]. Research with a focus on generating constraints from process models for conformance checking is conducted by Weidlich et al. [104, 105]. Case studies on process mining and conformance checking are, among others, reported in [13, 22, 97]. Separation of duties is traditionally discussed as a topic within the research area of role-based access control (RBAC). A summary of essential RBAC concepts and their relationship to process-centered concepts is provided in [90].

References

1. I. Adan, J. Resing, *Queueing Theory* (Eindhoven University of Technology, Eindhoven, 2002)
2. T. Allweyer, BPMN 2.0. Books on Demands (2010)
3. A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, A. Guizar, N. Kartha, C.K. Liu, R. Khalaf, D. Koenig, M. Marin, V. Mehta, S. Thatte, D. van der Rijn, P. Yendluri, A. Yiu, Web services business process execution language version 2.0. Committee specification, 31 January 2007, OASIS (2007)
4. R. Anupindi, S. Chopra, S.D. Deshmukh, J.A. van Mieghem, E. Zemel, *Managing Business Process Flows* (Prentice Hall, New York, 1999)
5. J. Becker, M. Rosemann, C. von Uthmann, Guidelines of business process modeling, in *Business Process Management. Models, Techniques, and Empirical Studies*, ed. by W.M.P. van der Aalst, J. Desel, A. Oberweis (Springer, Berlin, 2000), pp. 30–49
6. J. Becker, M. Kugeler, M. Rosemann, *Process Management: a Guide for the Design of Business Processes* (Springer, Berlin, 2011)
7. B.L. Berg, H. Lune, *Qualitative Research Methods for the Social Sciences* (Pearson, Boston, 2004)
8. S. Conger, Six sigma and business process management, in *Handbook of Business Process Management*, vol. 1, ed. by J. vom Brocke, M. Rosemann (Springer, Berlin, 2010), pp. 127–148
9. T. Curran, G. Keller, *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model* (Prentice Hall, Upper Saddle River, 1997)
10. T.H. Davenport, *Process Innovation: Reengineering Work Through Information Technology* (Harvard Business School Press, Boston, 1993)
11. T.H. Davenport, J.E. Short, The new industrial engineering: information technology and business process redesign. Sloan Manag. Rev. **31**(4), 11–27 (1990)
12. R.B. Davis, E. Brabander, *ARIS Design Platform: Getting Started with BPM* (Springer, Berlin, 2007)
13. J. De Weerdt, M. De Backer, J. Vanthienen, B. Baesens, A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. Inf. Syst. **37**(7), 654–676 (2012)
14. R. Dijkman, I. Vanderfeesten, H.A. Reijers, The road to a business process architecture: an overview of approaches and their use. BETA Working Paper Series, WP 350. Eindhoven University of Technology, Eindhoven (2011)
15. D.J. Elzinga, T. Horak, C.Y. Lee, C. Bruner, Business process management: survey and methodology. IEEE Trans. Eng. Manag. **42**(2), 119–128 (1995)

M. Dumas et al., *Fundamentals of Business Process Management*,
DOI 10.1007/978-3-642-33143-5,

16. G. Engels, A. Förster, R. Heckel, S. Thöne, Process modeling using UML, in *Process-Aware Information Systems*, ed. by M. Dumas, W.M.P. van der Aalst, A.H.M. ter Hofstede (Wiley, New York, 2005). Chapter 5
17. T. Erl, A. Karmarkar, P. Walmsley, H. Haas, *Web Service Contract Design and Versioning for SOA* (Prentice Hall, New York, 2008)
18. P.J.M. Frederiks, T.P. van der Weide, Information modeling: the process and the required competencies of its participants. Data Knowl. Eng. **58**(1), 4–20 (2006)
19. J. Freund, B. Rücker, *Real-Life BPMN: Using BPMN 2.0 to Analyze, Improve, and Automate Processes in Your Company*. CreateSpace Independent Publishing Platform (2012)
20. V. Frolov, D. Megel, W. Bandara, Y. Sun, L. Ma, Building an ontology and process architecture for engineering asset management, in *Proceedings of the 4th World Congress on Engineering Asset Management (WCEAM)*, Athens, Greece, September 2009 (Springer, Berlin, 2009)
21. D. Fürstenau, *Process Performance Measurement* (GRIN, Santa Cruz, 2008)
22. S. Goedertier, J. De Weerdt, D. Martens, J. Vanthienen, B. Baesens, Process discovery in event logs: an application in the telecom industry. Appl. Soft Comput. **11**(2), 1697–1710 (2011)
23. E.M. Goldratt, *The Goal: A Process of Ongoing Improvement* (North River Press, Great Barrington, 1992)
24. A. Greasley, A redesign of a road traffic accident reporting system using business process simulation. Bus. Process. Manag. J. **10**(6), 635–644 (2004)
25. C. Günther, Process mining in flexible environments. PhD thesis, Technische Universiteit Eindhoven (2009)
26. M. Hammer, Reengineering work: don't automate, obliterate. Harv. Bus. Rev. **68**(4), 104–112 (1990)
27. M. Hammer, *Beyond Reengineering: How the Process-Centered Organization Is Changing Our Work and Our Lives* (HarperBusiness, New York, 1997)
28. M. Hammer, What is business process management, in *Handbook of Business Process Management*, vol. 1, ed. by M. Rosemann, J. vom Brocke (Springer, Berlin, 2010)
29. M. Hammer, J. Champy, *Reengineering the Corporation: A Manifesto for Business Revolution* (HarperCollins, New York, 1993)
30. P. Hanafizadeh, M. Moosakhani, J. Bakhshi, Selecting the best strategic practices for business process redesign. Bus. Process. Manag. J. **15**(4), 609–627 (2009)
31. P. Harmon, *Business Process Change: A Guide for Business Managers and BPM and Six Sigma Professionals*, 2nd edn. (Morgan Kaufmann, San Mateo, 2007)
32. P. Harmon, Analyzing activities. BPTrends Newsletter **1**(4), April 2003. http://www.bptrends.com
33. D. Heckl, J. Moormann, Process performance management, in *Handbook on Business Process Management 2* (Springer, Berlin, 2010), pp. 115–135
34. D. Hollingsworth, in *The Workflow Reference Model: 10 Years on the Workflow Handbook 2004* (Workflow Management Coalition, Cohasset, 2004), pp. 295–312
35. D. Hollingsworth, The Workflow Reference Model. TC00-1003 Issue 1.1, Workflow Management Coalition, 24 November 1994
36. R. Hull, Artifact-centric business process models: brief survey of research results and challenges, in *On the Move to Meaningful Internet Systems: OTM 2008* (2008), pp. 1152–1163
37. IEEE TaskForce on Process Mining. Process mining manifesto. http://www.win.tue.nl/ieeetfpm/doku.php?id=shared:process_mining_manifesto. Accessed: November 2012
38. F. Johannsen, S. Leist, G. Zellner, Implementing six sigma for improving business processes at an automotive bank, in *Handbook of Business Process Management*, vol. 1, ed. by J. vom Brocke, M. Rosemann (Springer, Berlin, 2010), pp. 361–382
39. R.S. Kaplan, D.P. Norton, The balanced scorecard—measures that drive performance. Harv. Bus. Rev. **70**(1), 71–79 (1992)
40. W.D. Kelton, R.P. Sadowski, N.B. Swets, *Simulation with Arena*, 5th edn. (McGraw-Hill, New York, 2009)

41. W.J. Kettinger, J.T.C. Teng, S. Guha, Business process change: a study of methodologies, techniques, and tools. Manag. Inf. Syst. Q. **21**, 55–80 (1997)
42. J. Krogstie, G. Sindre, H.D. Jørgensen, Process models representing knowledge for action: a revised quality framework. Eur. J. Inf. Syst. **15**(1), 91–102 (2006)
43. M. Laguna, J. Marklund, *Business Process Modeling, Simulation and Design* (Prentice Hall, New York, 2004)
44. S. Limam Mansar, H.A. Reijers, F. Ounnar, Development of a decision-making strategy to improve the efficiency of bpr. Expert Syst. Appl. **36**(2), 3248–3262 (2009)
45. O.I. Lindland, G. Sindre, A. Sølvberg, Understanding quality in conceptual modeling. IEEE Softw. **11**(2), 42–49 (1994)
46. R.L. Manganelli, M.M. Klein, *The Reengineering Handbook: A Step-by-Step Guide to Business Transformation* (Amacom, New York, 1994)
47. S.L. Mansar, H.A. Reijers, Best practices in business process redesign: validation of a redesign framework. Comput. Ind. **56**(5), 457–471 (2005)
48. S.L. Mansar, H.A. Reijers, Best practices in business process redesign: use and impact. Bus. Process. Manag. J. **13**(2), 193–213 (2007)
49. K. McCormack, The development of a measure of business process orientation and its relationship to organizational performance, April 1999. Online tutorial available at http://www.prosci.com/mccormack.htm
50. J. Mendling, *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*. Lecture Notes in Business Information Processing, vol. 6 (Springer, Berlin, 2008)
51. J. Mendling, Empirical studies in process model verification, in *Transactions on Petri Nets and Other Models of Concurrency II, Special Issue on Concurrency in Process-Aware Information Systems*, vol. 5460 (2009), pp. 208–224
52. J. Mendling, H.A. Reijers, J. Recker, Activity labeling in process modeling: empirical insights and recommendations. Inf. Syst. **35**(4), 467–482 (2010)
53. J. Mendling, H.A. Reijers, W.M.P. van der Aalst, Seven process modeling guidelines (7PMG). Inf. Softw. Technol. **52**(2), 127–136 (2010)
54. J. Mendling, L. Sánchez-González, F. García, M. La Rosa, Thresholds for error probability measures of business process models. J. Syst. Softw. **85**(5), 1188–1197 (2012)
55. J. Mendling, M. Strembeck, J. Recker, Factors of process model comprehension—findings from a series of experiments. Decis. Support Syst. **53**(1), 195–206 (2012)
56. A. Møller, M.I. Schwartzbach, *An Introduction to XML and Web Technologies* (Addison-Wesley, Reading, 2006)
57. D. Müller, M. Reichert, J. Herbst, A new paradigm for the enactment and dynamic adaptation of data-driven process structures, in *Advanced Information Systems Engineering* (Springer, Berlin, 2008), pp. 48–63
58. M. Netjes, R.S. Mans, H.A. Reijers, W.M.P. Aalst, R.J.B. Vanwersch, Bpr best practices for the healthcare domain, in *Business Process Management Workshops* (Springer, Berlin, 2010), pp. 605–616
59. A. Nigam, N.S. Caswell, Business artifacts: an approach to operational specification. IBM Syst. J. **42**(3), 428–445 (2003)
60. Object Management Group, Unified Modeling Language (UML) Version 2.4.1 (2011)
61. Object Management Group, Business Process Model and Notation (BPMN), Version 2.0, January 2011. http://www.omg.org/spec/BPMN/2.0
62. P. O'Neill, A.S. Sohal, Business process reengineering a review of recent literature. Technovation **19**(9), 571–581 (1999)
63. A. Ottensooser, A. Fekete, H.A. Reijers, J. Mendling, C. Menictas, Making sense of business process descriptions: an experimental comparison of graphical and textual notations. J. Syst. Softw. **85**(3), 596–606 (2012)
64. M.A. Ould, *Business Process Management: A Rigorous Approach*. British Informatics Society Ltd (2005)

65. C. Ouyang, W.M.P. van der Aalst, M. Dumas, A.H.M. ter Hofstede, M. La Rosa, Semantic web services: theory, tools and applications, in *Service-Oriented Processes: An Introduction to BPEL*, ed. by J. Cardoso (IGI Publishing, Hershey, 2007), pp. 155–188
66. M. Petre, Why looking isn't always seeing: readership skills and graphical programming. Commun. ACM **38**(6), 33–44 (1995)
67. M.E. Porter, *Competitive Advantage: Creating and Sustaining Superior Performance* (Free Press, New York, 1985)
68. G. Redding, M. Dumas, A.H.M. ter Hofstede, A. Iordachescu, A flexible, object-centric approach for business process modelling. Serv. Oriented Comput. Appl. **4**(3), 191–201 (2010)
69. M. Reichert, B. Weber, *Enabling Flexibility in Process-Aware Information Systems* (Springer, Berlin, 2012)
70. H.A. Reijers, Product-based design of business processes applied within the financial services. J. Res. Pract. Inf. Technol. **34**(2), 110–122 (2002)
71. H.A. Reijers, *Design and Control of Workflow Processes: Business Process Management for the Service Industry* (Springer, Berlin, 2003)
72. H.A. Reijers, S. Liman Mansar, Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics. Omega **33**(4), 283–306 (2005)
73. H.A. Reijers, J. Mendling, A study into the factors that influence the understandability of business process models. IEEE Trans. Syst. Man Cybern., Part A, Syst. Hum. **41**(3), 449–462 (2011)
74. H.A. Reijers, T. Freytag, J. Mendling, A. Eckleder, Syntax highlighting in business process models. Decis. Support Syst. **51**(3), 339–349 (2011)
75. H.A. Reijers, J. Mendling, R.M. Dijkman, Human and automatic modularizations of process models to enhance their comprehension. Inf. Syst. **36**(5), 881–897 (2011)
76. S.-H. Rhee, N.W. Cho, H. Bae, Increasing the efficiency of business processes using a theory of constraints. Inf. Syst. Front. **12**(4), 443–455 (2010)
77. J.J. Rooney, L.N. vanden Heuvel, Root cause analysis for beginners. Qual. Prog. **37**(7), 45–53 (2004)
78. M. Rosemann, Potential pitfalls of process modeling: Part A. Bus. Process. Manag. J. **12**(2), 249–254 (2006)
79. M. Rosemann, Potential pitfalls of process modeling: Part B. Bus. Process. Manag. J. **12**(3), 377–384 (2006)
80. G.A. Rummler, A.P. Brache, *Improving Performance: Managing the White Space on the Organizational Chart* (Jossey-Bass, San Francisco, 1990)
81. G.A. Rummler, A.J. Ramias, A framework for defining and designing the structure of work, in *Handbook of Business Process Management*, vol. 1, ed. by M. Rosemann, J. vom Brocke (Springer, Berlin, 2010)
82. A.-W. Scheer, *ARIS Business Process Modelling* (Springer, New York, 2000)
83. K.D. Schenk, N.P. Vitalari, K.S. Davis, Differences between novice and expert systems analysts: what do we know and what do we do? J. Manag. Inf. Syst. **15**(1), 9–50 (1998)
84. A. Schwegmann, M. Laske, As-is modeling and process analysis, in *Process Management: A Guide for the Design of Business Processes* (Springer, Berlin, 2011), pp. 133–156
85. I. Seidman, *Interviewing as Qualitative Research: A Guide for Researchers in Education and the Social Sciences* (Teachers College Press, New York, 2006)
86. A. Sharp, P. McDermott, *Workflow Modeling: Tools for Process Improvement and Application Development*, 2nd edn. (Artech House, Norwood, 2008)
87. B. Silver, *BPMN Method and Style*, 2nd edn. (Cody-Cassidy Press, Aptos, 2011)
88. J. Stirna, A. Persson, K. Sandkuhl, Participative enterprise modeling: experiences and recommendations, in *Proceedings of the 19th Conference on Advanced Information Systems Engineering (CAiSE 2007)*, Trondheim, Norway, ed. by J. Krogstie, A.L. Opdahl, G. Sindre. Lecture Notes in Computer Science, vol. 4495 (Springer, Berlin, 2007), pp. 546–560
89. D. Straker, *A Toolbook for Quality Improvement and Problem Solving* (Prentice Hall, New York, 1995)

90. M. Strembeck, J. Mendling, Modeling process-related rbac models with extended uml activity models. Inf. Softw. Technol. **53**(5), 456–483 (2011)
91. K.D. Swenson, *Mastering the Unpredictable: How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done* (Meghan-Kiffer Press, Tampa, 2010)
92. A.H.M. ter Hofstede, W.M.P. van der Aalst, M. Adams, N. Russell (eds.), *Modern Business Process Automation: YAWL and Its Support Environment* (Springer, Berlin, 2010)
93. W.M.P. van der Aalst, Verification of workflow nets, in *Application and Theory of Petri Nets 1997*, ed. by P. Azéma, G. Balbo. Lecture Notes in Computer Science, vol. 1248 (Springer, Berlin, 1997), pp. 407–426
94. W.M.P. van der Aalst, *Process Mining—Discovery, Conformance and Enhancement of Business Processes* (Springer, Berlin, 2011)
95. W.M.P. van der Aalst, K. van Hee, *Workflow Management: Models, Methods, and Systems* (MIT Press, Cambridge, 2004)
96. W.M.P. van der Aalst, M. Weske, D. Grünbauer, Case handling: a new paradigm for business process support. Data Knowl. Eng. **53**(2), 129–162 (2005)
97. W.M.P. van der Aalst, H.A. Reijers, A.J.M.M. Weijters, B.F. van Dongen, A.K. Alves de Medeiros, M. Song, H.M.W.(E.) Verbeek, Business process mining: an industrial application. Inf. Syst. **32**(5), 713–732 (2007)
98. W.M.P. van der Aalst, M. Rosemann, M. Dumas, Deadline-based escalation in process-aware information systems. Decis. Support Syst. **43**(2), 492–511 (2007)
99. W.M.P. van der Aalst, J. Nakatumba, A. Rozinat, N. Russell, Business process simulation, in *Handbook of Business Process Management*, vol. 1, ed. by J. vom Brocke, M. Rosemann (Springer, Berlin, 2010), pp. 313–338
100. I. Vanderfeesten, H.A. Reijers, W.M.P. van der Aalst, Product-based workflow support. Inf. Sci. **36**(2), 517–535 (2011)
101. L. Verner, The challenge of process discovery. BPM Trends, May 2004
102. J. vom Brocke, M. Rosemann, *Handbook on Business Process Management 1: Introduction, Methods, and Information Systems*, vol. 1 (Springer, Berlin, 2010)
103. J. vom Brocke, M. Rosemann, *Handbook on Business Process Management 2: Strategic Alignment, Governance, People and Culture*, vol. 2 (Springer, Berlin, 2010)
104. M. Weidlich, A. Polyvyanyy, N. Desai, J. Mendling, M. Weske, Process compliance analysis based on behavioural profiles. Inf. Syst. **36**(7), 1009–1025 (2011)
105. M. Weidlich, H. Ziekow, J. Mendling, O. Günther, M. Weske, N. Desai, Event-based monitoring of process execution violations, in *Business Process Management—9th International Conference, BPM 2011, Proceedings*, Clermont-Ferrand, France, August 30–September 2, 2011, ed. by S. Rinderle-Ma, F. Toumani, K. Wolf. Lecture Notes in Computer Science, vol. 6896 (Springer, Berlin, 2011), pp. 182–198
106. M. Weske, *Business Process Management: Concepts, Languages, Architectures*, 2nd edn. (Springer, Berlin, 2012)
107. S.A. White, D. Miers, *BPMN Modeling and Reference Guide* (Future Strategies Inc., Lighthouse Point, 2008)
108. Workflow Patterns Initiative, Workflow Patterns home page (2001). http://www.workflowpatterns.com
109. Y. Yang, M. Dumas, L. García-Bañuelos, A. Polyvyanyy, L. Zhang, Generalized aggregate quality of service computation for composite services. J. Syst. Softw. **85**(8), 1818–1830 (2012)
110. M. zur Muehlen, *Workflow-Based Process Controlling. Foundation, Design, and Implementation of Workflow-Driven Process Information Systems*. Advances in Information Systems and Management Science, vol. 6 (Logos, Berlin, 2004)
111. M. zur Muehlen, R. Shapiro, Business process analytics, in *Handbook on Business Process Management 2* (2010), pp. 137–157
112. M. zur Muehlen, D.E. Wisnosky, J. Kindrick, Primitives: design guidelines and architecture for BPMN models, in *ACIS 2010 Proceedings* (2010)

Index

M. Dumas et al., *Fundamentals of Business Process Management*,
DOI 10.1007/978-3-642-33143-5,