1 Organizing Workflows

1.1 Ontology for Workflow Management

The objective of this chapter is to develop a reference framework. This framework has three functions in this book. First, it is used to define the business-management context within which workflow management systems operate. Second, it is used to model and analyze processes. And third, it is used to describe the functionality and architecture of workflow management systems. A reference framework is a system of straightforwardly defined terms that describe a particular field of knowledge. It is also known as an *ontology*.

The ontology in which we are interested is that of processes. The terms used are generic in nature and can be applied in virtually all working situations. In practice, however, many have various synonyms which are widely used; for the sake of clarity, we will try to use a single "preferred term" as often as possible. This will be in line with the terminology used by the Workflow Management Coalition. In this chapter, we first discuss the role of work in society. Then we examine processes, followed by the distribution of work. The relationship between the principal and the contractor plays an important role in this. Specifically in electronic business these relationships are extremely important. We then study organizational structures and the management of processes. Finally, we look at the role played by (computerized) information systems in the establishment and management of business processes.

1.2 Work

People work to live—even though some become so involved that they give the impression of living for their work. In fact, we work because we need products to maintain our lives (for example: food, clothing, a home, a means of transport, not to mention entertainment). We do not produce all the things that we need ourselves, because that is inefficient. It actually would be impossible to manufacture all the products that we use during our lives in a modern society, ourselves. We would have to learn so many different and complex skills that they alone would take up our entire lives. We would need many lifetimes just to make the tools needed to produce the necessities of life. This is why we are instead organized into specialized "business units," in which people produce a limited range of products in a highly efficient way, with the help of machines. These products are supplied to other people through a market mechanism and a distribution structure in exchange for money, which enables the producers to buy those products that they do not make themselves. With production distributed in such a way, there is also created work that would not exist if everybody was entirely self-sufficient in producing all the products they need. For example, managing money—what the banks do—and preparing advertising materials would not be necessary.

There have thus developed all kinds of services and products that do not make a direct contribution to keeping us alive, but are necessary to keep the organization operating. Despite this "burden," we are able to produce so efficiently that we have a large amount of free time—thus further stimulating the demand for entertainment. The leisure industry therefore is also a flourishing one.

Modern society has become so complex that nobody can entirely survey it any longer, and many people do not know what role their work plays in the overall scheme of things. This "alienation" is a major social problem that falls outside the scope of this book. But even within large companies there exists a high degree of work specialization, which results in the "big picture" being lost and employees not always realizing why they have to do the things they are told to do. Such alienation from work has a negative effect upon productivity. This is why many companies are organizing their work in such a way that their employees clearly understand that they are working for a particular customer. Among the objectives of such customer-oriented work is an increase in employees' motivation, and hence their productivity. The fact that we have moved from living in a supply-driven economy, in which the means of produc-

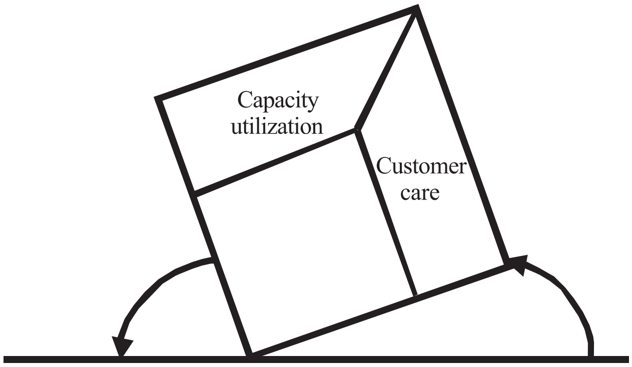

Figure 1.1
Organizational paradigm shift

tion were scarce, to a demand-driven one in which it is the customers who are scarce, has only served to reinforce this tendency. This shift of focus from the means of production to the customer is also known as "organizational paradigm shift" (see figure 1.1).

In order to make work "controllable" and to encourage communication between employees, *workflow management systems* have evolved. These are a new class of information system. They make it possible to build, in a straightforward way, a "bridge" between people's work and computer applications.

1.3 Business Processes

There are many different types of work, such as baking bread, making a bed, designing a house or collecting survey results to compile a statistic. In all of these examples, we can see the one tangible "thing" that is produced or modified: the bread, the bed, the house, or the statistic. In this book, we shall call such a "thing" a *case*. Other terms used are work, job, product, service, or item. A case does not need be a specific object; it can also be more abstract—like, say, a lawsuit or an insurance claim. A building project or the assembly of a car in a factory are also examples of cases.

Working on a case is discrete in nature. That is, every case has a beginning and an end, and each can be distinguished from every other case.

Each case involves a *process* being performed. A process consists of a number of *tasks* that need to be carried out and a set of *conditions* that determine the order of the tasks. A process can also be called a *procedure*. A task is a logical unit of work that is carried out as a single whole by one *resource*. A resource is the generic name for a person, machine or group of persons or machines that can perform specific tasks. This does not always mean to say that the resource necessarily carries out the task independently, but that it is responsible for it. We will examine this subject more closely in the next section.

As an example of a process, we shall examine how a (fictional) insurance company deals with a claim. We can identify the following tasks:

1. *recording* the receipt of the claim;
2. establishing the *type* of claim (for example, fire, motor vehicle, travel, professional);
3. checking the client's *policy*, to confirm that it does in principle cover what has been claimed for;
4. checking the *premium*, to confirm that payments are up to date;
5. *rejection*, if task 3 or 4 has a negative result;
6. producing a *rejection letter*;
7. estimating the *amount to be paid*, based upon the claim details;
8. appointment of an *assessor* to research the circumstances of the damage and to establish its value;
9. consideration of *emergency measures* to limit further damage or relieve distress;
10. provision of *emergency measures* if approved as part of task 8;
11. establishment or revision of *amount to be paid* and offer to client;
12. recording of client's *reaction*: acceptance or objection;
13. assessment of *objection* and decision to revise (task 11) or to take legal proceedings (task 14);
14. legal *proceedings*;
15. *payment* of claim; and
16. *closure* of claim: filing.

Here we can see sixteen tasks that do not necessarily need to be performed in the order shown. Two or more tasks that must be performed in a strict order are called a *sequence*. For some cases, certain tasks do not need to be carried out. One example is the appointment of an expert, if the claim report is clear and the amount of the claim is below a particular value, the involvement of an expert is not necessary. Other tasks that do not always need to be performed are taking emergency measures, assessing an objection, or taking legal proceedings. Sometimes, therefore, a choice between two or more tasks can be made. This we call a *selection*.

There are also tasks that can be performed *in parallel*, for example checking the policy and checking the premiums. These tasks must both be completed before the "rejection" task can begin. This is called *synchronization*.

This example of a process also includes *iteration*, or repetition—namely, the repeated assessment of an objection or the revision of the amount to be paid. In theory, this could go on forever. Figure 1.2 shows the order of the tasks as a *process diagram*: an arrow from task A to task B means that A must be done before B. We can also see that the diagram contains more information than the list of tasks. For example, it shows that a claim can only be closed once any emergency measures required have been taken. Each task is indicated by a rectangle. If a task has more than one successor task—that is, if it has more than one arrow leading from it—then precisely *one* of these subsequent tasks must be chosen during the task in question. If a task has more than one predecessor—more than one arrow leading to it—then *all* of these must be completed before that task can begin (synchronization). The circles indicate where particular workflows meet or split. The gray circles have *several* precursor tasks and only *one* subsequent task. They indicate that only one of the preceding tasks needs to be performed in order to continue. The black circles have *one* predecessor and *several* subsequent tasks. They show that all the subsequent tasks must be performed. (The circles can be regarded as "dummy" tasks.) Chapter 2 introduces a process notation which makes it easier to express such properties.

To summarize, we can identify four different basic mechanisms in process structures: sequence, selection, parallelization, and iteration. All are very commonplace in practice, and in principle all processes can be modeled using these four constructions. We shall consider them in greater detail in chapter 2.

Some tasks can be performed by a computer without human interference. Other tasks require human intelligence: a judgment or a decision. For instance, a bank employee decides if a client's loan request will be granted or not. Human workers need *knowledge* to execute tasks. This

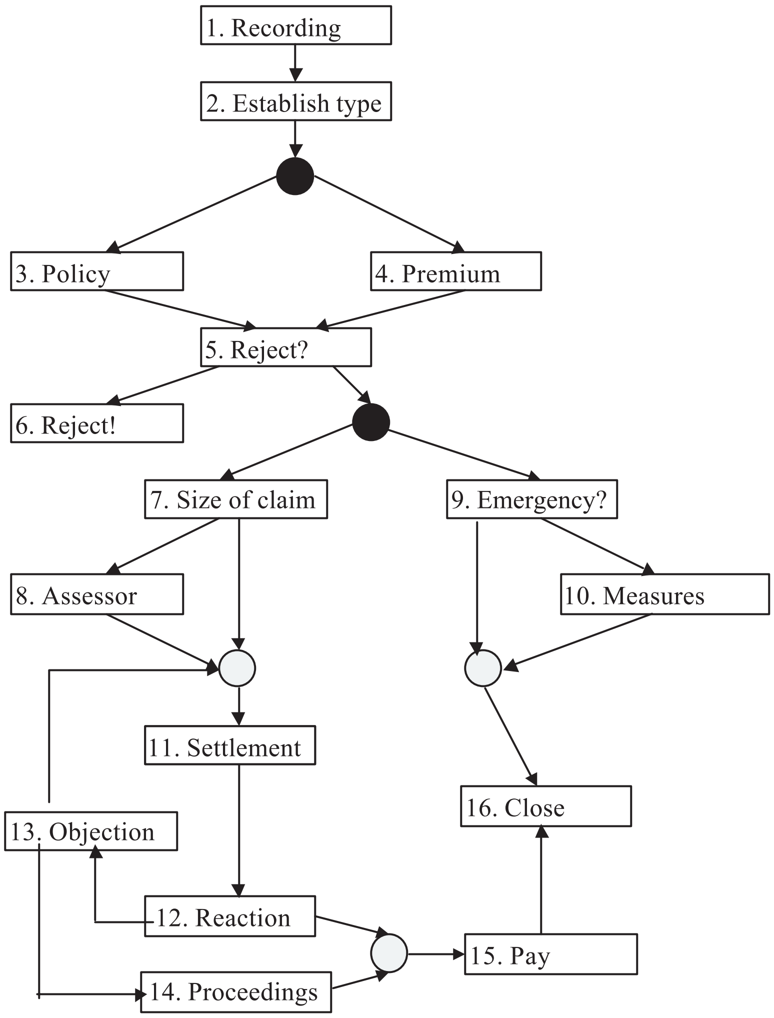

Figure 1.2
Insurance claim process

knowledge is stored in their minds by experience, the so-called *tacit knowledge*. Other forms of knowledge can be obtained by learning and information retrieval, the so-called *explicit knowledge*. *Knowledge management* is concerned with the acquisition, enrichment, and distribution of knowledge so that the right knowledge is at the right time with the person who has to fulfill a task.

A task can also be defined as a process that cannot be subdivided any further: an *atomic* process. There is a subjective element in this—what one person regards as a single task may be a nonatomic one to another. For an insurance company, for example, the compilation of an assessor's report of damage to a car is a single task, whereas for the expert himself it is a process comprising various tasks, such as checking the chassis, engine, and bodywork. A task is therefore an atomic process for the person defining or ordering it, but for the person carrying it out it is often a nonatomic one.

A single process is carried out on each case. We call the performance of a task by a resource an *activity*. Various cases may have the same process, but each case may follow a different route through that process. In the insurance company, for example, one claim may involve an objection and another not. The route taken depends upon the specific characteristics of the case—the *case attributes*. The number of processes in a company is (generally) finite and far smaller than the number of cases to be handled. As a result, a company can develop a routine for performing processes and thus operate efficiently.

This is clearly seen in the clothing industry: it is much faster to make one hundred skirts with the same pattern than one hundred skirts using different patterns. Off-the-rack is cheaper than made-to-measure. What's more, producing one thousand skirts of the same pattern is less expensive than ten times making one hundred in that pattern. This is called the economy of scale: the costs per case fall as the number of cases increases. Companies therefore endeavor to keep the number of processes small and to make the number of cases that each can perform as high as possible—at least, as long as they can earn something from each case. Profit, after all, is the ultimate objective.

An insurance company wants to keep the number of claims as low as possible—but this is generally a factor that it cannot control. It will also try to keep the number of processes low. There is, however, a catch:

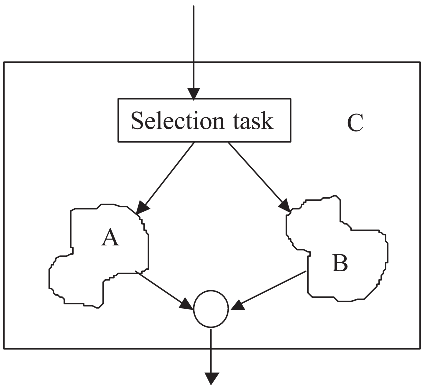

Figure 1.3
Combination of two processes into one

the processes must not become too complicated. It is better to have a few more, but simpler, processes than a few which are overly complex. Remember that, in theory, it is possible to combine two or more processes into one, as shown in figure 1.3. Processes A and B are joined to form a single process, C.

Here one additional task has been added: deciding what type of case we are dealing with and so choosing which of the processes to follow. This is therefore a false economy. In order to reach an efficient process structure, calculations need to be made which cannot generally be performed without the aid of computer simulations.

The situation that we have just described is the most common: a small number of processes with a lot of cases. There are, however, exceptions to this rule. A tailor, for example, produces every suit made-to-measure; one could therefore say that he must design and start up a new process for each case. This also applies to an architect who has to design every new house or office block from scratch. But we can also view this in a different way: both the tailor and the architect will certainly use a standard approach, and thus a process which they always follow. The tailor will start by taking the customer's measurements, then show him a number of patterns and try to establish with him which best matches his wishes, and then make changes to the pattern. Then the fabric is chosen and the tailor starts drawing the pattern. There are also many other tasks that can be identified as a part of each case. The same applies to the architect. What we can see here is that there is indeed a process, but the tasks performed are highly dependent upon the case. This is, therefore, a yardstick for the complexity of a process: the degree to which the tasks depend on the cases.

Although we shall deal primarily with situations in which many cases pertain to a single process, there are many situations in which a new process needs to be designed for each case. We call these "one of a kind" processes. In these, the first stage in tackling the case is the design of its specific process. Even here, there are frequently standard tasks from which the process is compiled. In such cases, we say that every case has its own *project*. The words "project" and "process" are here synonymous.

We have already seen that the work carried out on cases is of a discrete nature: each has a single beginning and a single end. However, there is also work of a continuous nature which does not clearly belong to a single case. Take, for example, a doorman whose work consists of assisting people to enter a building, or a policeman who has to guarantee security in a district by patrolling it. In both examples a case can still—with a little goodwill—be defined by identifying periods and regarding door keeping or patrolling for a particular period as one case. The employee thus automatically receives a continual sequence of cases, one for each period. Another way of regarding work of a continuous nature in case terms is to regard the work as a whole as one case comprising a continual repetition of tasks. In this book, we concentrate upon discrete work—but in doing so we do not exclude continuous work. It can serve as an extreme example with which the principles presented in the book can be put to the test.

To conclude this section, we shall subdivide processes into three categories: *primary*, *secondary*, and *tertiary*:

- Primary processes are those that produce the company's products or services. They therefore are known also as *production processes*. They deal with cases for the customer. As a rule, they are the processes that generate income for the company, and are clearly customer-oriented. Sometimes the customer is not yet known, as when firms produce to stock. Examples of primary processes are the purchase of raw materials and components, the sale of products and services, design and engineering, and production and distribution.

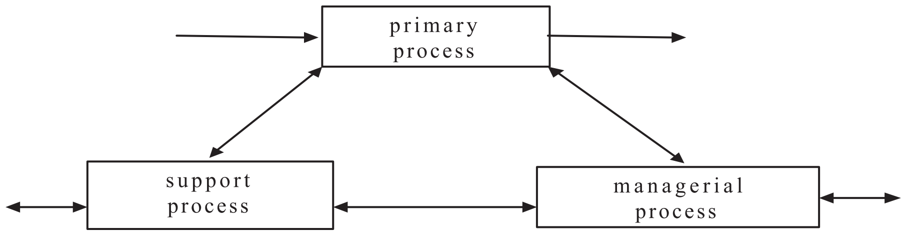

Figure 1.4
Links between the three types of processes

• Secondary processes are those that support the primary ones. They therefore are also known as *support processes*. One important group of secondary processes concentrates upon maintaining the means of production: the purchase and maintenance of machinery, vehicles, and premises. A comparable group of processes is that involving personnel management: recruitment and selection, training, work appraisal, payrolls, and dismissal. Financial administration is also a secondary process, as is marketing.

• Tertiary processes are the *managerial processes* that direct and coordinate the primary and secondary processes. During these, the objectives and preconditions within which the managers of the other processes must operate are formulated, and the resources required to carry out the other processes are allocated. The managerial processes also encompass the maintenance of contacts with financiers and other stakeholders.

Figure 1.4 shows the relationships between the three types of processes.

The managerial processes have objectives and capital as their input, and must deliver performance—often in the form of profit. Support processes receive, from the managerial processes, the means to buy in resources, and they dispose of resources which are no longer functioning. The resources managed by the secondary processes are placed at the disposal of the primary processes, which return them after use. As input, the primary processes receive orders on the one hand and raw materials and components on the other. As output, they deliver products and services. They receive assignments and purchasing budgets from the managerial processes. Support and primary processes report back to the managerial processes and submit their income.

The secondary and tertiary processes are often continuous in nature, although they may contain discrete subprocesses, whereas the primary processes are usually case driven and thus have a discrete character.

1.4 Allocating and Accepting Work

Animals and machines work on *orders*, or *assignments*, given by people. But most people's work is also assigned or outsourced to them by other people: their principals. Exceptions are artists, scientists, and politicians, who can—to some extent—decide for themselves what work they are going to do.

There are two forms of principals: the *boss* and the *customer*. Ultimately, assignments ordered by bosses are directly or indirectly related to work for customers. The relationship is "direct" if the work carried out results in a product or service for a customer, which may be unknown. This mainly applies to the primary processes. The relationship is "indirect" if the work involves maintaining or improving the production process: the secondary and tertiary processes.

In most organizations there exists a hierarchy under which assignments that people receive can (in part) be passed on to people further down the hierarchy. A person who is assigned a task is a *contractor*, also known as a *resource*. We mainly use the latter term because assignments can be carried out by machines—in particular, computer applications—as well as by people. Thus far we have discussed principals and contractors as if they are individual people, but they can in fact also be company departments or separate firms. We will therefore use the term *actor* to describe principals and contractors in general. An actor may play both roles—as a principal and a subcontractor (or resource)—at the same time.

A contractor does not necessarily carry out the work itself, but may redirect or subcontract it to third parties. But the contractor always *directs* the work which it accepts.

In larger organizations, employees carrying out an assignment often do not know for which customer the task is being performed. This is particularly the case when products are being produced to stock, because during production the identity of the customer is still unknown. (And sometimes there is eventually no customer at all for the product.)

As indicated before, a principal is either a customer or a boss. There is also a wide variety among customers. For the Prison Service, criminals (prisoners) are its customers; the Inland Revenue's customers are the taxpayers, a hospital's customers are its patients. The role of a customer is dependent upon the situation: the baker is the gardener's customer when the gardener looks after the baker's garden, but the gardener is the baker's customer when he buys bread.

In large organizations, there is a marked tendency to accentuate the role of the customer more clearly. The principle that "the customer is always right" is winning ground over "working for the boss." Customer awareness ensures that people are more conscious of who they are working for, which leads to a more careful approach to their work: after all, if they deliver poor quality work, they will be unsure whether the customer will order more. (For a prison "customer," this principle works the other way around.)

For all work a principal and a contractor exist who have a—sometimes unwritten—*contract* with one another about the case to be performed, the deadline for its completion, and the price to be paid. If the contractor is a separate company, then a communications process will be created between principal and contractor before the contract is entered into, and communications between the two actors may continue to be necessary during the performance of the task. When the relationship between the contractor and the principal is formalized, a *communications protocol* can be observed. This can be very complex. Figure 1.5 shows an example of a communications protocol.

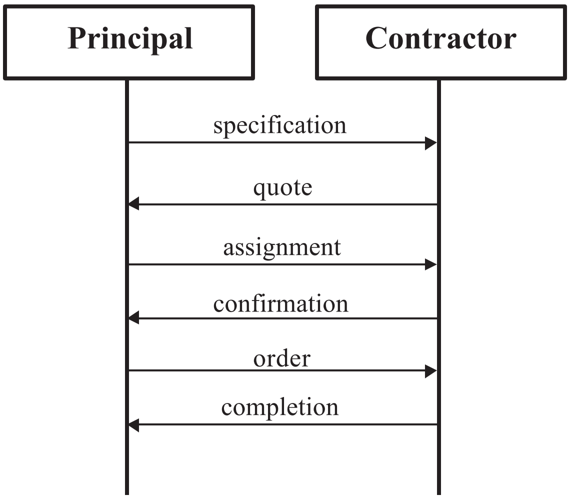

Figure 1.5
Communications protocol

In this example, we can see the successive steps in the relationship. The principal first provides a specification of the work to be carried out. Then the contractor produces a plan for performing the work and fixes a price. This is the "quote" that it submits to the principal. The latter studies the quote and orders the work in accordance with it. In practice, there can be a lot of discussion between the parties in the meantime, with the principal making supplementary demands—about the price, for example—and the contractor explaining how it intends to carry out the work. In many cases, the moment when the order is confirmed is not the same as when it actually begins. If the work forms part of a larger project that the principal is directing, then the work can only begin once other elements in the project have been completed; the principal thus determines at what point the work can start. The number of steps in a communications protocol between a principal and a contractor therefore can vary from case to case according to the specific characteristics and handling of each, and so does not need to be fixed in advance.

An actor responsible for a process may assign or outsource a task as a whole to a contractor or he may decompose it into a process, that is, a network of tasks, each of which he assigns to a contractor. At their turn these contractors may repeat this decomposition process. This decomposition leads to a *contract tree*. Execution of a task for a particular case requires the enactment of a communications protocol between principals and contractors. Instead of decomposing a task into a process and outsourcing the subtasks of this process for all cases that pass the task, it is also possible to do this for each case in a different way. Then the execution of a task for a particular case starts with a "design phase," in which the network of tasks is created and in which the (sub)contractors are selected. Figure 1.6 shows an example of this. In this example, the task is the transportation of a cargo from point A to point K. The principal P subdivides this work into two tasks: transportation from point A to point D, and transportation from point D to point K. Each of these tasks is subcontracted to a different contractor, that is, contractors Q and R. Each of the tasks is then subdivided again by these two: by principal/contractor Q into transportation from A to C and then C to D, and by principal/contractor R from D to J and then from J to K. This is illustrated in figure 1.6. Note that both Q and R act as principal *and* contractor.

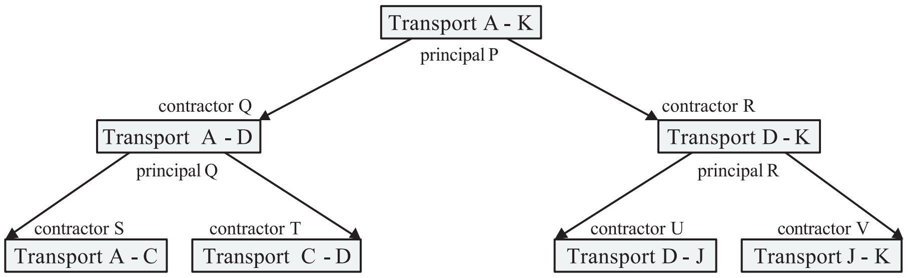

Figure 1.6
Contract tree

This tree contains "nodes," which are shown in the example as rectangles. "Branches" link two "nodes." The "nodes" show those actors who are responsible for a part of the work. In this example, the actors are identified by the tasks that they must perform. The "root" of the tree (which we actually show at the top of the diagram) receives the assignments directly from the principal. The "leaves" of the tree (that is, the lowest of the "nodes") are the actors who actually carry out the tasks. The other actors are both principals and contractors. An actor *X* is a subcontractor of another actor *Y* if there is an arc from *Y* to *X*. An actor is a principal if there is an arc leading from this actor to another actor. Consider for example figure 1.6. Actor *Q* is a subcontractor of *P* and a principal of *S* and *T*. Such decomposition and outsourcing processes occur frequently inside organizations but also between different organizations. In electronic business we try to automate/computerize these processes as much as possible. If we want to support business processes by information systems, we need very detailed and precise descriptions of these business processes. If we want to couple business processes of different organizations in an automatic/computerized way, this becomes even more important.

1.5 Organizational Structures

A great deal of literature has been published about organizational structures, and any attempt to summarize it in a few paragraphs is doomed to fail. Therefore we shall not try to do so. We shall, however, discuss those properties of the three most important forms of organizational structure that are relevant to workflow organization.

An organizational structure establishes how the work carried out by the organization in question is divided up amongst its staff. In most cases this does not mean the people themselves, but rather the *roles* or *functions* that they fulfill. A single person can fulfill several roles during her or his lifetime. Somebody can, for example, begin as an administrative assistant and end up as head of accounts. People may also fulfill different roles in time. It may be that the same person is both a driver and a messenger, delivering messages when there is nobody to be driven. One important aspect of an organizational structure is the division of authorities and responsibilities. If an executive has specific responsibilities, then he also has to have particular authorities. These often involve the authority to assign work to other members of staff—in other words, to outsource work to others. Conversely, an executive is responsible for ensuring that the work assigned to him by authorized colleagues actually is carried out.

The three most important forms of organizational structure—or rather, coordination mechanisms—are:

1. the hierarchical organization;
2. the matrix organization; and
3. the network organization.

The *hierarchical organization* is the best known of these, and is characterized by a "tree" structure. Such a structure is called an *organizational chart*. We already have encountered tree structures in the previous section in the form of contract trees. In an organizational chart, each node which is not a "leaf" indicates an individual role or function. The "leaves" of the tree usually represent groups of staff or departments. The "branches" show authority relationships: the person at the start (top) of the branch is authorized to order work from the person or department at the end (bottom) of it.

There is also another definition of the organizational chart that closely resembles ours but is, in fact, different. Under this definition, each "leaf" shows a person and each node at a higher level represents a department. The "root" node indicates the entire company, and every other node a part of that above it. The people indicated in each leaf thus belong to the department shown in the node immediately above them. Whereas the

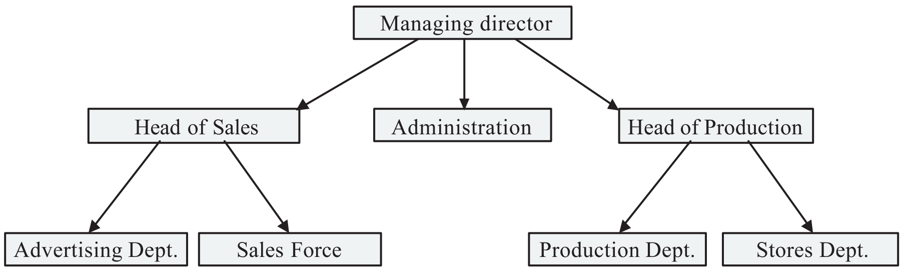

Figure 1.7
Organizational chart

first definition shows the person who is responsible for all the people below him in the tree for whom he represents the root, the second regards each of these collections of staff as one department. The similarity between organizational charts and contract trees is that both express principal-contractor relationships as "branches." The difference is that in an organizational chart this relationship is not linked to any specific case, whereas this relationship is very relevant for a transaction tree. In a strictly hierarchical organization, communication between two nodes always passes through their closest common predecessor. Figure 1.7 shows an example of an organizational chart.

In this example, formal communication between the sales force and the stores department must go through the head of sales, the managing director and the head of production. The "management" or "board" is often at the "root" of an organizational chart. Its "leaves" are the company's departments. One typical example of a hierarchical organization is the army. In practice, there exists a lot of informal communication between the various individual members of staff and departments, allowing communication to be quicker than if it were to follow hierarchical lines. Purely hierarchical organizations are virtually extinct now, since this structure is too inflexible. In many firms it is too unwieldy to allow the delegation of work only through fixed, hierarchical channels.

In designing a hierarchical organization, we are free to choose what departments are created and what management layers exist above them. In allocating staff into departments, we can select from three principles:

- *The capacity group*. Put people with the same skills together in the same department. In principle, such people are interchangeable. The task of the head of department is to keep its members "up-to-date"—through training, for example—and to do his best to "sell" them to other business units for whom they perform their work. Typical examples are typing pools and pools of maintenance engineers.

• *The functional department.* This performs an interdependent group of tasks, each often requiring the same skills. Responsibility for the work of the department rests with its head. Typical examples are departments like accounting, marketing, and maintenance.

• *Process or production departments.* In this case, the department is responsible for a complete business process or for the manufacturing of a product.

The first or second type of organization is often chosen for the secondary processes. In the primary ones, the third form begins to gain importance. Superseding the departments are the hierarchical management layers. In choosing these, the following question plays an important role: is the amount of coordination required between the departments large or small? There should be as few layers as possible between departments which need to coordinate to a great extent, so they should preferably have a single manager.

A manager has a maximum *span of control.* In other words, he cannot direct an unlimited number of subordinates. How large a particular manager's span of control is depends to a great extent upon the nature of the work and her own experience.

This is how the *matrix organization* came about. This form of organization is structured in accordance with two dimensions: the *functional* and the *hierarchical.* The hierarchical part is the same as described above and is usually based upon functional or capacity groups: people with the same skills belong to the same group. The functional part is based upon the tasks which have to be performed. (The terminology can be rather confusing.) Each person thus has a hierarchical boss—the head of the department to which he belongs—and a functional boss, who is responsible for the task to be carried out. The tasks—which in the context of matrix organizations are usually called "projects"—are unique; in other words, no fixed structure can be created based upon the tasks, so the hierarchical (fixed) structure is based upon the skills of the people concerned. The functional bosses are known as "project leaders."

Matrix organizations are found mostly in companies that operate on a project basis, such as building contractors, installation firms, and soft-

	Project-1	Project-2	Project-3
Supervisors	Louise	Anita	John
Carpenters	Pete	Karl	Geraldine
Masons	Henry	Tom	Jerry
Painters	Bert	Simone	Simone
Plasterers	Charles	Peter	Paul

Figure 1.8
Staff allocation in a matrix organization

ware houses: in other words, in businesses that do not carry out serial production but rather unique projects. The functional structure thus is constantly subject to change. It is quite possible that person A is for a while the leader of a project in which person B participates, and then a little later B becomes the leader of a project involving A. Figure 1.8 shows an example of staff allocation in a matrix organization. The columns show the functional allocation and the rows the hierarchical.

We can see how one person can take part in more than one project. Naturally, one person may be involved only in one project at a time, but it is equally possible for someone to work alternately on several projects during the same period. Often several people within one department work on the same project. In the matrix, this would mean more than one person being included in the same cell. For the sake of simplicity, this is not shown in figure 1.8. A form of organization which strongly resembles the matrix type occurs when processes are managed by a process manager and cases by a *case manager*. The former is responsible for the quality and efficiency of "her" process, whereas the latter ensures the rapid and correct completion of "her" cases. This can lead to a conflict of interests.

The last form of organization which we can identify is the network organization. In this, autonomous actors collaborate to supply products or services. To the customer, though, they appear to be one organization—which is why the network organization is sometimes called a *virtual organization*. The actors perform as principals and contractors. The autonomy means that there exists no formal permanent (employment) relationship, which means that an actor can choose whether or not she wishes to carry out a particular task. The actors required to perform each task therefore must be recruited individually on each occasion. This may be done through a protocol and a contract tree, as discussed in the previous section. This can be a time-consuming business, so "framework" contracts are often drawn up for regular assignments. Such a contract determines that a party is available upon request to perform a particular type of work. Just as in a matrix organization, party A can be party B's principal for one type of work but its subcontractor for another.

More and more network organizations are being created. There are two main reasons for this. First, firms are trying to keep their permanent workforce as small as possible instead making more extensive use of temporary staff and subcontractors. This, together with the fact that many people are now working part time, is known as the flexibilization of labor. In this way firms can control their fixed costs. The use of *co-makers* and *outsourcers*, which are examples of contractors, is very common in the building and motor industries. The second reason is that specialist companies, each with only a limited product range, can supply together an entire product. Examples are found in the construction industry—in which a range of actors join forces to build a bridge—and amongst consultancy firms, which package their individual knowledge to offer an integrated product incorporating, say, financial, legal, fiscal, and IT advice. A network organization is, to a certain degree, comparable with a matrix organization. After all, the resources for each project are assembled individually. The difference, however, is that in this case those resources do not have the same employer.

1.6 Managing Processes

One established way of studying the management of processes is to distinguish between a *management system* and a *managed system*. The word "system" here means all those people, machines, and computerized information systems that carry out particular processes. A managed system can even be further subdivided into a lower-level management system and a managed system (see figure 1.9). The managed system at the lowest level of this subdivision is an *enactment system*. At the highest level, a system is always part of a managed system. A management system can manage several systems, and in doing so, it ensures the ability of

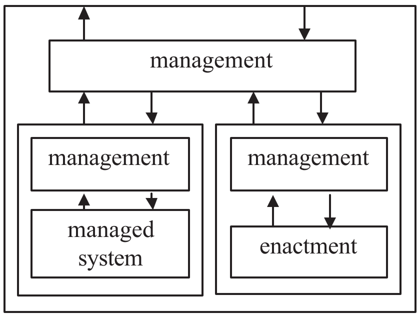

Figure 1.9
Recursive management paradigm: The whole entity is a managed system

the managed systems to communicate with one another and with the outside world—that is, the managed system at a higher level.

Between the management system and the managed system there occurs an exchange of information. This enables the management system to communicate objectives, preconditions, and decisions to the managed system, and the managed system—conversely—reports back to the management system. Based upon these reports, the management system may revise the objectives, preconditions, and decisions. This so-called *planning and control cycle* can be identified in every organization.

Process management has long been divided into four levels. The distinction between these is based upon the frequency and scope of the decisions to be made. By scope, we mean two things: the period of time over which the decision has an influence, and its (potential) financial impact. The four levels are as follows (see figure 1.10):

1. *Real-time management.* Decisions can be made very frequently (intervals range from microseconds to hours). The period of time during which the decision has an effect is very short, and the financial consequences of a wrong decision are small.
2. *Operational management.* Decisions are made very regularly (from hours to days) and their scope is limited. In other words, the influence of the decision is no longer noticeable after a short period.
3. *Tactical management.* Decisions are made periodically (from days to months), and their scope is limited.
4. *Strategic management.* Decisions are made only once, or no more than every couple of years, and their scope is wide. The influence of a strategic decision can remain noticeable for many years.

Management level	Time horizon	Financial impact	Type of decisions	Supporting methods
Real-time	Seconds–hours	Low	Equipment control	Control theory
Operational	Hours–days	Limited	Resource assignment	Combinatorial optimization (e.g., scheduling)
Tactical	Days–months	High	Resource capacity planning and budgeting	Stochastic models (e.g., queueing models)
Strategic	Months–years	Very high	Process design and resource types	Financial models, multi-criteria analysis

Figure 1.10
Four levels of process management

Another distinction between these levels of management is the types of decisions which are made. Real-time and operational management involve only dynamic aspects, not the structure of the business processes. Real-time management involves the control of machines and vehicles. Operational management mostly concerns the allocation of resources to cases and the routing of those cases. Typical examples of operational management are *production scheduling* and the *routing* of trains.

Tactical management concerns: *capacity planning* and *budgeting* for operational management. Capacity planning involves determining the quantities of resources required per type of case. This means not only human resources, but also the machines and raw materials used in performing the case. Stocks management is a typical example, involving not only the management of the raw-materials stocks themselves but also that of reserve resources. Budgeting concerns the allocation of financial means and the formulation of targets in financial terms.

Strategic management is concerned with the *structural* aspects of processes and types of resources. One strategic question is whether the company should carry out a particular process itself, or source it out. Another question is how the processes should be structured and what procedures should be followed.

Each management level, except for real-time management, also has the task to take care of exceptions to rules that are made for the lower levels. Tactical management may be involved if the resource allocation at the operational level does not succeed.

Decision making is an important feature of (process) management. The discipline of operations research (OR) searches for the best possible solutions to decision problems using mathematical techniques. Artificial intelligence (AI) tries to develop computer systems that can imitate human techniques for solving decision problems (heuristics). Organizational sociology tackles such things as methods by which people can cooperate to find a solution. Here, we shall confine ourselves to summarizing the four phases that are always passed through when solving decision problems:

1. *Definition* involves establishing exactly what the problem is and, in particular, within what scope a solution to it must be found. Drawing up optimization criteria often forms part of this phase.
2. *Creation* involves formulating one or more solutions that fall within the scope defined or satisfy an optimization criterion.

3. *Evaluation* involves assessing different solutions, for instance by multi-criteria analysis.
4. *Selection* involves selecting one solution that works in order to implement it.

In principle, computer support is available for all these tasks, particularly the second and third. This is sometimes possible using a simple spreadsheet but usually requires mathematical techniques or simulation models.

1.7 Information Systems for Business Processes

The organization of work, both within and between companies, is becoming more and more complicated. This is why (computerized) information systems have been developed that support the management of processes and their coordination. We shall first offer a method of classifying information systems. Then we shall outline how they have been developed in the past and how they will probably be developed in the near future.

Information systems can be categorized in many ways. The one we have chosen to use here is based upon the role played by the system in the processes. The list below is in ascending order of functionality: the first type of system listed contains very little knowledge of the processes and should only be used to support the people who actually do the work, whereas the final one can manage processes without any human intervention:

• *Office information systems.* These systems assist the staff responsible for carrying out and managing processes with basic information processing: writing, drawing, calculating, filing, and communication. They include word processors, drawing packages, spreadsheets, simple database management systems, and electronic mail. These systems do not themselves contain any knowledge of the processes. Although the information that they process may contain business knowledge, they themselves cannot do anything with this.

• *Transaction-processing systems.* These systems, also called registrational systems, register and communicate the relevant aspects of changes in the circumstances of a process and record these changes. Transaction-processing systems that specialize in communication between different organizations are called *interorganizational information systems.* These often use electronic data interchange (EDI) using standards for data exchange like XML. The heart of such a system generally is a database management system, but today a workflow management system also becomes an essential component. The latter type of system does have some knowledge of the processes, as proven—for example—by the fact that it can independently interpret incoming transactions and thus determine where and how the input data should be stored.

- *Knowledge-management systems*. These systems take care of acquisition and distribution of knowledge to be used by knowledge workers, either case workers or managers. The knowledge they handle is explicit knowledge that can be represented in digital form. One of the simplest forms of a knowledge-management system is a search engine coupled to a document-management system. With such a system, a knowledge worker is able to find relevant text fragments produced by himself or others by means of keywords or free-text search. A more advanced facility is a case-based reasoning system that searches through a database of best-practice cases and finds cases with a high level of similarity to the actual case. The solution presented by the cases found might be applicable for the actual case as well. Managers are interested mostly in aggregated data about the processing of cases or about the cases themselves. Here we often use *data warehouses* that are connected to tools for statistical analysis. A data warehouse is a database that stores aggregated data in multidimensional cells, for instance the number of customers that bought a typical kind of product in a specific time period and a geographical region.
- *Decision-support systems*. These compute decisions through interaction with people. There are two types of decision-support systems. The first type is based upon mathematical models. Examples include budgeting and investment systems and production-planning systems. The second type is based upon logical reasoning systems. They are also known as *expert systems*. One example is a system for establishing the cause of a defect in a machine. These systems are used at all levels of management (operational, tactical, and strategic).
- *Control systems*. Also known as programmed decision-making systems, these systems calculate and implement decisions entirely automatically, based upon the recorded state of a process. Examples are automatic ordering, climate control, and invoicing systems.

An information system is often a combination of the four types described above. From the viewpoint of efficiency, the control system appears to be the ideal because it requires no staff. In practice, the number of applications in which such systems can be used turns out to be very limited, and only well defined decision situations can be approached in this way. Nevertheless, they do work for some operational management

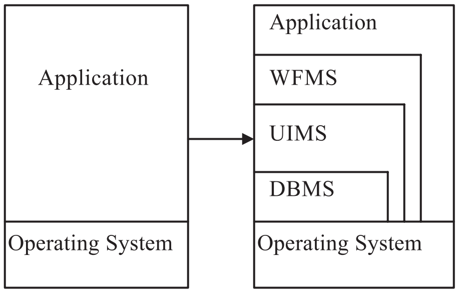

Figure 1.11
Decomposition of generic functionality

problems. The decision-support systems, which solve management problems through interaction with people, offer the most potential because they combine human insight with the computer's calculating power. We still have absolutely no idea how an information system should make a decision about many problems at the strategic level. In practice, most information systems are office-information and transaction-processing systems.

We shall now examine the way in which we develop information systems. This will be done by means of a historical summary. The boundaries of the time periods given should not be regarded as clear-cut, but that is not the most important point. The summary below highlights the influence of workflow management systems. What the history shows is that more and more generic tasks have been taken out of programs and put into decomposed management systems. Figure 1.11 illustrates this evolution.

1. *1965–1975: decompose applications.* During this period, information systems comprised decomposed applications, each with its own databases and definitions. The applications ran directly on the operating system and either had no user interface or one entirely of their own. Data were stored between two runs of the application program, originally on punch cards and paper tapes, and later on magnetic tape and in disk memory. There was no exchange of data between different applications. It thus was possible for a member of staff to have different names in the payroll program and the personnel program. It was impossible to achieve added value by combining different sources of data.
2. *1975–1985: database management—"take data management out of the applications."* This period is characterized by the rise of the *database*

management system (DMBS). Originally these were hierarchical and network databases, later relational ones. A database is a permanently available, integrated collection of data files which can be used by many applications. The use of databases has the advantages that data managed by different applications can be combined, that data structures only need to be defined once, that the organization of data can be handed over to a database management system, and that the same data item only needs to be stored once. A DBMS is a piece of generic software that can be used to define and use databases: to add, view, revise, and delete data. The use of database management systems has also radically changed the system-development process: once the database has been defined, different developers can work on designing applications on it at the same time. To do this, methods were developed for establishing data structures before the applications were defined. This is the data-oriented approach to system development. This period thus can be characterized as that during which the data organization was beginning to be extracted from application programs.

3. *1985–1995: user-interface management—"take the user interface out of the applications."* It was during this period that the next bottleneck in system development appeared. Because we were developing more and more interactive software, a great deal of time was being spent developing user interfaces. Originally these were designed by the developers screen by screen, field by field. Not only did this take up a lot of time, but also each designer had her own style, which meant that every system had to operate in a different way. There are now *user-interface management systems* (UIMS) that solve both these problems: a user interface can be defined rapidly and the designer is "invited" to do this in a standard way. In recent years, a transition has taken place from character-based user interfaces to graphics-based ones, and as a result the utilization of user-interface management systems has increased. Today the functions of user-interface management systems are integrated in other tools, like database management systems, program environments, and web browsers. During this period the user interfaces were extracted from the application programs.

4. *1995–2005: workflow management—"take the business processes out of the applications."* Now that data management and user interfacing have largely disappeared from applications themselves, it seems that much of the software is devoted to business processes (procedures) and the handling of cases. Therefore, it has become attractive to isolate this component now and find a separate solution for it. Not only can this accelerate the development of information systems, but it also offers the added advantage that the business processes become easier to maintain.

Today, it occurs frequently that management wants to change an administrative procedure, but this would have far-reaching consequences for the software. As a result, the change is not carried through. *Workflow systems* should solve such problems. A workflow system manages the workflows and organizes the routing of case data amongst the human resources and through application programs. Just as databases are developed and used with the assistance of a database management system, so *workflow management systems* (WFMS) can be used to define and use workflow systems. This period can be characterized as that during which the processes were extracted from the applications.

To put workflow management in historical perspective, we should mention some of the early work on workflow management. The idea to have generic tools, or at least generic methods, for supporting business processes emerged in the 1970s with pioneers such as Skip Ellis and Michael Zisman. Zisman completed his Ph.D. thesis "Representation, Specification, and Automation of Office Procedures" in 1997 (University of Pennsylvania). In the 1970s, Ellis and others worked at Xerox PARC on "Office Automation Systems." Ellis already used Petri-net-based workflow models (the so-called information control nets) in the late 1970s. One could wonder why it took such a long time before workflow management systems became established as a standard component for enterprise information systems. There are several reasons for this. First of all, workflow management requires users linked to a computer network. Only in the 1990s did workers become connected to the network. Second, many information systems evolved from systems that are unaware of business processes and the organization to systems that are aware; therefore, workflow was never considered as a really new piece of functionality. Finally, the rigid and inflexible character of the early (and some of the contemporary) products scared away many potential users.

A workflow management system can be compared with an operating system: it controls the workflows between the various resources—people or applications. It is confined to the *logistics* of case handling. In other words, a change to the content of case data is implemented only by people or application programs. A workflow management system has a number of functions that can be used to define and graphically track workflows, thus making both the progress of a case through a workflow and the structure of the flow itself easy to revise. It therefore is not remarkable that workflow management systems have become the ideal tool for achieving BPR.

In the above evolution, we can see that *disentangling* functions from applications is the way to improve efficiency. By separating certain functions, generic solutions (management systems) can be developed for them. In this way information systems can be made *component-based*, by first *configuring* the components and then *integrating* them (a process also known as *assembling*). Configuration is the setting of parameters, which may take all sorts of forms. The input of a database scheme into a database management system and the definition of a process scheme in a workflow management system are examples of component configuration.

For integration of components we have the so-called middleware. Some form of middleware just is a set of standards and language features that create a communication structure at compile time. Another form is a component that takes care of the communication needs of other components.

Alongside these developments, we also increasingly observe companies buying—for specific processes—standard software packages that combine a large number of the functions defined above. For a specific process, such *generic* software has to be configured; that is, parameters must be set. The advantage of a standard software package is that there are no development costs, but one drawback is that the system may not meet all the wishes of its users. This disadvantage could, though, be seen as a benefit, because it forces the organization to work in the tried and trusted way embedded in the package. In fact, such a software package contains a generic *company model* that can be adapted to a specific business situation.

EXERCISES

Exercise 1.1

A workflow is defined as a network of tasks with rules that determine the (partial) order in which the tasks should be performed.

(a) Which are these essential ordering principles?
(b) Show that iteration can be made by the other ordering principles.

Exercise 1.2

In this chapter we have seen (figure 1.2) some notation to describe a network of tasks. (This is not the notation we will use in the remainder of the book.) A task is represented by a rectangle and it has one or more direct predecessors and one or more direct successors. The rules are: all predecessors should be ready before the task may be executed and exactly one successor will be executed. Further there are two kinds of connectors: open and closed circles with rules for passing signals. Change these rules as follows: tasks have exactly one incoming and one outgoing arc. Connectors may have one or more incoming and outgoing arcs. Open circles pass the signal from only one incoming to one outgoing arc exactly. Closed circles require from all incoming arcs a signal and pass it to all outgoing arcs. Model the claim handling example of figure 1.2 with these new rules. (It is allowed to connect circles to each other.)

Exercise 1.3

The concept "task" has two meanings, depending on the point of view. Give these two meanings and explain them.

Exercise 1.4

Give the three principles to assign employees to departments in a hierarchical organization and give pros and cons for each choice.