2

Decision support systems

A *decision support system* (DSS) is an interactive computer-based application that combines data and mathematical models to help decision makers solve complex problems faced in managing the public and private enterprises and organizations. As described in Chapter 1, the analysis tools provided by a business intelligence architecture can be regarded as DSSs capable of transforming data into information and knowledge helpful to decision makers. In this respect, DSSs are a basic component in the development of a business intelligence architecture.

In this chapter we will first discuss the structure of the decision-making process. Further on, the evolution of information systems will be briefly sketched. We will then define DSSs, outlining the major advantages and pointing out the critical success factors relative to their introduction. Finally, the development phases of a DSS project will be described, addressing the most relevant issues concerning its implementation.

2.1 Definition of system

The term *system* is often used in everyday language: for instance, we refer to the solar system, the nervous system or the justice system. The entities that we intuitively denominate *systems* share a common characteristic, which we will adopt as an abstract definition of the notion of system: each of them is made up of a set of components that are in some way connected to each other so as to provide a single collective result and a common purpose.

Every system is characterized by boundaries that separate its internal components from the external environment. A system is said to be *open* if its boundaries can be crossed in both directions by flows of materials and information.

Business Intelligence: *Data Mining and Optimization for Decision Making* *Carlo Vercellis*
 ISBN: 978-0-470-51138-1

When such flows are lacking, the system is said to be *closed*. In general terms, any given system receives specific input flows, carries out an internal transformation process and generates observable output flows.

As can be imagined, this abstract definition of system can be used to describe a broad class of real-world phenomena. For example, the logistic structure of an enterprise is a system that receives as input a set of materials, services and information and returns as output a set of products, services and information. More generally, even an enterprise, taken as a whole or in part, may be represented in its turn as a system, provided the boundaries as well as input and output flows are clearly defined.

Figure 2.1 shows the structure that we will use as a reference to describe the concept of system. A system receives a set of *input* flows and returns a set of *output* flows through a *transformation* process regulated by *internal conditions* and *external conditions*. The effectiveness and efficiency of a system are assessed using measurable performance indicators that can be classified into different categories. The figure shows the main types of metrics used to evaluate systems embedded within the enterprises and the public administration.

A system will often incorporate a *feedback* mechanism. Feedback occurs when a system component generates an output flow that is fed back into the system itself as an input flow, possibly as a result of a further transformation. Systems that are able to modify their own output flows based on feedback are called *closed cycle systems*. For example, the closed cycle system outlined in Figure 2.2 describes the development of a sequence of marketing campaigns.

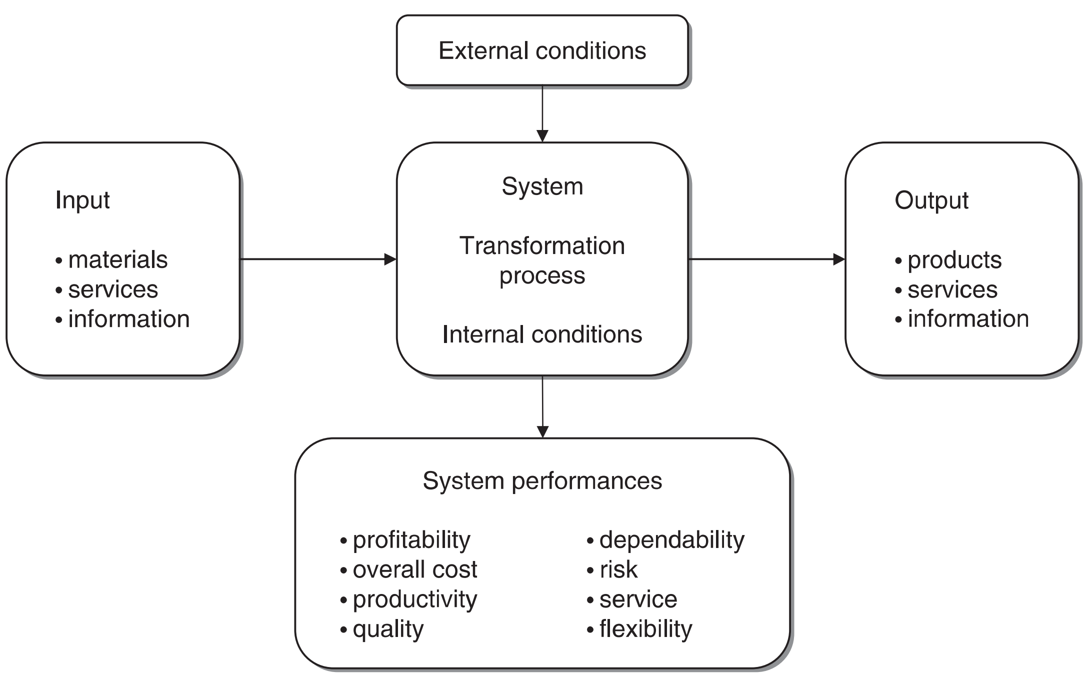

Figure 2.1 Abstract representation of a system

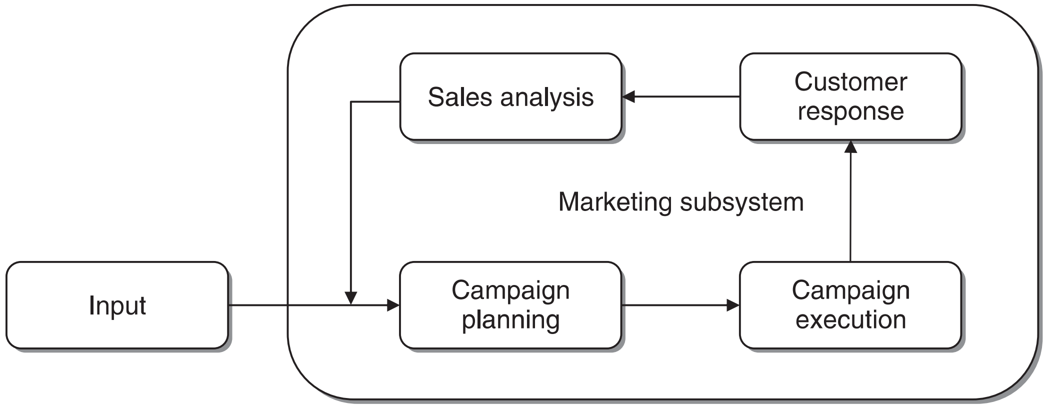

Figure 2.2 A closed cycle marketing system with feedback effects

The sales results for each campaign are gathered and become available as feedback input so as to design subsequent marketing promotions.

In connection with a decision-making process, whose structure will be described in the next section, it is often necessary to assess the performance of a system. For this purpose, it is appropriate to categorize the evaluation metrics into two main classes: *effectiveness* and *efficiency*.

Effectiveness. Effectiveness measurements express the level of conformity of a given system to the objectives for which it was designed. The associated performance indicators are therefore linked to the system output flows, such as production volumes, weekly sales and yield per share.

Efficiency. Efficiency measurements highlight the relationship between input flows used by the system and the corresponding output flows. Efficiency measurements are therefore associated with the quality of the transformation process. For example, they might express the amount of resources needed to achieve a given sales volume.

Generally speaking, effectiveness metrics indicate whether the *right* action is being carried out or not, while efficiency metrics show whether the action is being carried out in the *best* possible way or not.

2.2 Representation of the decision-making process

In order to build effective DSSs, we first need to describe in general terms how a decision-making process is articulated. In particular, we wish to understand the steps that lead individuals to make decisions and the extent of the influence exerted on them by the subjective attitudes of the decision makers and the specific context within which decisions are taken.

2.2.1 Rationality and problem solving

A *decision* is a choice from multiple alternatives, usually made with a fair degree of rationality. Each individual faces on a continual basis decisions that can be more or less important, both in their personal and professional life. In this section, we will focus on decisions made by knowledge workers in public and private enterprises and organizations. These decisions may concern the development of a strategic plan and imply therefore substantial investment choices, the definition of marketing initiatives and related sales predictions, and the design of a production plan that allows the available human and technological resources to be employed in an effective and efficient way.

The decision-making process is part of a broader subject usually referred to as *problem solving*, which refers to the process through which individuals try to bridge the gap between the current operating conditions of a system (*as is*) and the supposedly better conditions to be achieved in the future (*to be*). In general, the transition of a system toward the desired state implies overcoming certain obstacles and is not easy to attain. This forces decision makers to devise a set of alternative feasible options to achieve the desired goal, and then choose a decision based on a comparison between the advantages and disadvantages of each option. Hence, the decision selected must be put into practice and then verified to determine if it has enabled the planned objectives to be achieved. When this fails to happen, the problem is reconsidered, according to a recursive logic.

Figure 2.3 outlines the structure of the problem-solving process. The *alternatives* represent the possible actions aimed at solving the given problem and helping to achieve the planned objective. In some instances, the number of alternatives being considered may be small. In the case of a credit agency that has to decide whether or not to grant a loan to an applicant, only two options exist, namely acceptance and rejection of the request. In other instances, the number of alternatives can be very large or even infinite. For example, the development of the annual logistic plan of a manufacturing company requires a choice to be made from an infinite number of alternative options.

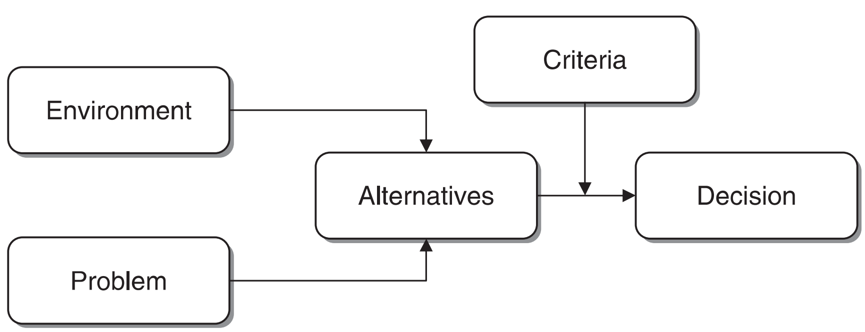

Figure 2.3 Logical flow of a problem-solving process

Criteria are the measurements of effectiveness of the various alternatives and correspond to the different kinds of system performance shown in Figure 2.1. A *rational* approach to decision making implies that the option fulfilling the best performance criteria is selected out of all possible alternatives. Besides economic criteria, which tend to prevail in the decision-making process within companies, it is however possible to identify other factors influencing a rational choice.

Economic. Economic factors are the most influential in decision-making processes, and are often aimed at the minimization of costs or the maximization of profits. For example, an annual logistic plan may be preferred over alternative plans if it achieves a reduction in total costs.

Technical. Options that are not technically feasible must be discarded. For instance, a production plan that exceeds the maximum capacity of a plant cannot be regarded as a feasible option.

Legal. Legal rationality implies that before adopting any choice the decision makers should verify whether it is compatible with the legislation in force within the application domain.

Ethical. Besides being compliant with the law, a decision should abide by the ethical principles and social rules of the community to which the system belongs.

Procedural. A decision may be considered ideal from an economic, legal and social standpoint, but it may be unworkable due to cultural limitations of the organization in terms of prevailing procedures and common practice.

Political. The decision maker must also assess the political consequences of a specific decision among individuals, departments and organizations.

The process of evaluating the alternatives may be divided into two main stages, shown in Figure 2.4: *exclusion* and *evaluation*. During the exclusion stage, compatibility rules and restrictions are applied to the alternative actions that were originally identified. Within this assessment process, some alternatives will be dropped from consideration, while the rest represent feasible options that will be promoted to evaluation. In the evaluation phase, feasible alternatives are compared to one another on the basis of the performance criteria, in order to identify the preferred decision as the best opportunity.

2.2.2 The decision-making process

A compelling representation of the decision-making process was proposed in the early 1960s, and still remains today a major methodological reference. The

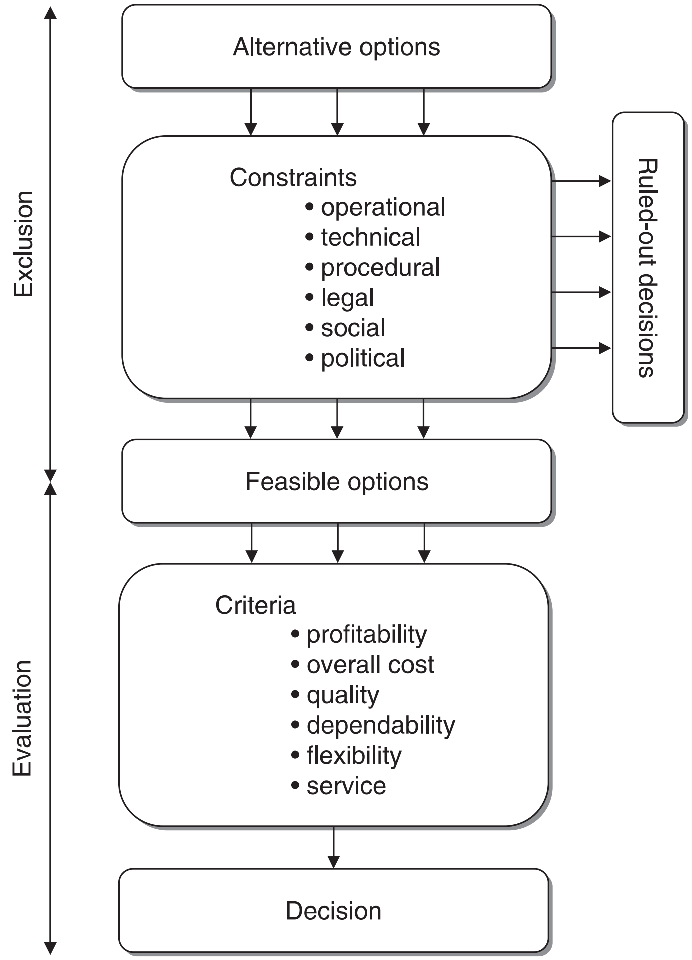

Figure 2.4 Logical structure of the decision-making process

model includes three phases, termed *intelligence*, *design* and *choice*. Figure 2.5 shows an extended version of the original scheme, which results from the inclusion of two additional phases, namely *implementation* and *control*.

Intelligence. In the *intelligence* phase the task of the decision maker is to identify, circumscribe and explicitly define the problem that emerges in the system under study. The analysis of the context and all the available information may allow decision makers to quickly grasp the signals and symptoms pointing to a corrective action to improve the system performance. For example, during the execution of a project the intelligence phase may consist of a comparison between the current progress of the activities and the original development plan. In general, it is important not to confuse the problem with the symptoms. For example, suppose that an e-commerce bookseller receives a complaint concerning late delivery of a book order placed on-line. Such inconvenience may be interpreted as the problem and be tackled by arranging a second delivery by priority shipping to circumvent the dissatisfaction of the customer. On the other

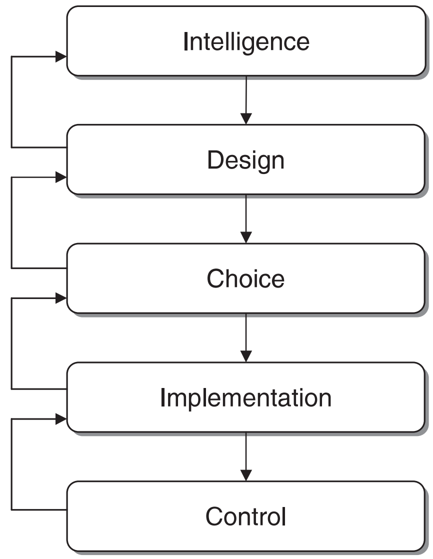

Figure 2.5 Phases of the decision-making process

hand, this may be the symptom of a broader problem, due to an understaffed shipping department where human errors are likely to arise under pressure.

Design. In the *design* phase actions aimed at solving the identified problem should be developed and planned. At this level, the experience and creativity of the decision makers play a critical role, as they are asked to devise viable solutions that ultimately allow the intended purpose to be achieved. Where the number of available actions is small, decision makers can make an explicit enumeration of the alternatives to identify the best solution. If, on the other hand, the number of alternatives is very large, or even unlimited, their identification occurs in an implicit way, usually through a description of the rules that feasible actions should satisfy. For example, these rules may directly translate into the constraints of an optimization model.

Choice. Once the alternative actions have been identified, it is necessary to evaluate them on the basis of the performance criteria deemed significant. Mathematical models and the corresponding solution methods usually play a valuable role during the *choice* phase. For example, optimization models and methods allow the best solution to be found in very complex situations involving countless or even infinite feasible solutions. On the other hand, decision trees can be used to handle decision-making processes influenced by stochastic events.

Implementation. When the best alternative has been selected by the decision maker, it is transformed into actions by means of an *implementation* plan. This involves assigning responsibilities and roles to all those involved into the action plan.

Control. Once the action has been implemented, it is finally necessary to verify and check that the original expectations have been satisfied and the effects of the action match the original intentions. In particular, the differences between the values of the performance indicators identified in the choice phase and the values actually observed at the end of the implementation plan should be measured. In an adequately planned DSS, the results of these evaluations translate into experience and information, which are then transferred into the data warehouse to be used during subsequent decision-making processes.

The most relevant aspects characterizing a decision-making process can be briefly summarized as follows.

- Decisions are often devised by a group of individuals instead of a single decision maker.
- The number of alternative actions may be very high, and sometimes unlimited.
- The effects of a given decision usually appear later, not immediately.
- The decisions made within a public or private enterprise or organization are often interconnected and determine broad effects. Each decision has consequences for many individuals and several parts of the organization.
- During the decision-making process knowledge workers are asked to access data and information, and work on them based on a conceptual and analytical framework.
- Feedback plays an important role in providing information and knowledge for future decision-making processes within a given organization.
- In most instances, the decision-making process has multiple goals, with different performance indicators, that might also be in conflict with one another.
- Many decisions are made in a fuzzy context and entail risk factors. The level of propensity or aversion to risk varies significantly among different individuals.
- Experiments carried out in a real-world system, according to a *trial-and-error* scheme, are too costly and risky to be of practical use for decision making.
- The dynamics in which an enterprise operates, strongly affected by the pressure of a competitive environment, imply that knowledge workers need to address situations and make decisions quickly and in a timely fashion.

2.2.3 Types of decisions

Defining a taxonomy of decisions may prove useful during the design of a DSS, since it is likely that decision-making processes with similar characteristics may be supported by the same set of methodologies. Decisions can be classified in terms of two main dimensions, according to their *nature* and *scope*. Each dimension will be subdivided into three classes, giving a total of nine possible combinations, as shown in Figure 2.6.

According to their nature, decisions can be classified as *structured*, *unstructured* or *semi-structured*.

Structured decisions. A decision is structured if it is based on a well-defined and recurring decision-making procedure. In most cases structured decisions can be traced back to an algorithm, which may be more or less explicit for decision makers, and are therefore better suited for automation. More specifically, we have a structured decision if input flows, output flows and the transformations performed by the system can be clearly described in the three phases of intelligence, design and choice. In this case, we will also say that each component phase is structured in its turn. Actually, even decisions that appear fully structured require in most cases the direct intervention of decision makers to cope with unexpected events, caused for example by unusual values of some input flows.

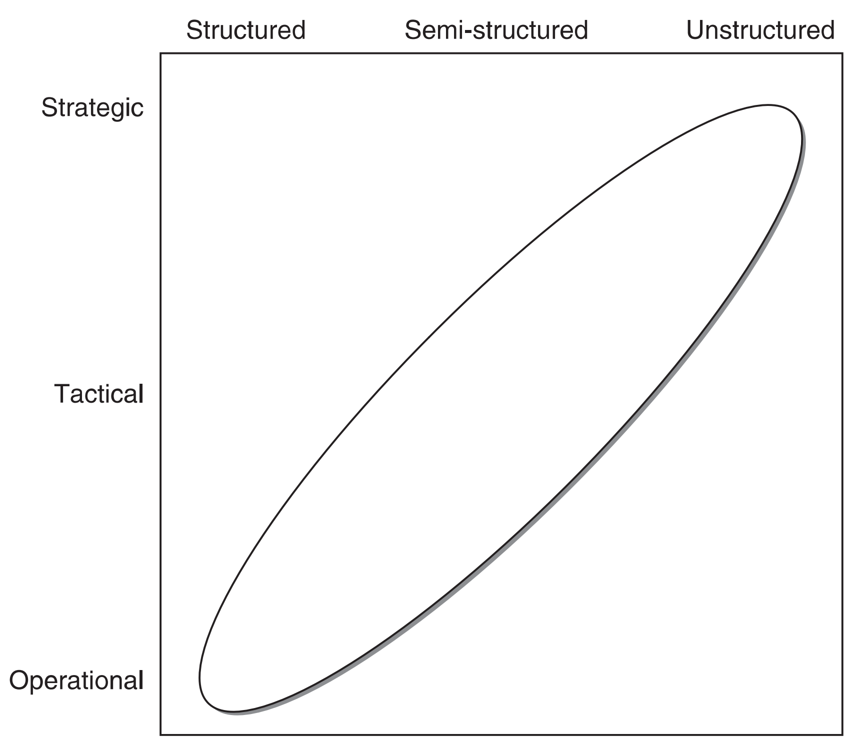

Figure 2.6 A taxonomy of decisions

Unstructured decisions. A decision is said to be unstructured if the three phases of intelligence, design and choice are also unstructured. This means that for each phase there is at least one element in the system (input flows, output flows and the transformation processes) that cannot be described in detail and reduced to a predefined sequence of steps. Such an event may occur when a decision-making process is faced for the first time or if it happens very seldom. In this type of decisions the role of knowledge workers is fundamental, and business intelligence systems may provide support to decision makers through timely and versatile access to information.

Semi-structured decisions. A decision is semi-structured when some phases are structured and others are not. Most decisions faced by knowledge workers in managing public or private enterprises or organizations are semi-structured. Hence, they can take advantage of DSSs and a business intelligence environment primarily in two ways. For the unstructured phases of the decision-making process, business intelligence tools may offer a passive type of support which translates into timely and versatile access to information. For the structured phases it is possible to provide an active form of support through mathematical models and algorithms that allow significant parts of the decision-making process to be automated.

Sometimes situations may arise where the nature of a decision cannot be easily identified unambiguously. When facing the same problem, such as establishing the sale price of a product, different decision makers operating in different organizations may come up with dissimilar choices. For example, a first decision maker may believe that the best sale price can be obtained by comparing cost and price–demand elasticity curves. As a consequence, such decision maker may consider the choice phase of the decision-making process as structured. By contrast, a second decision maker may believe that the elasticity curve does not reflect all the factors influencing the response of the market to price variations since some of these elements cannot be quantified. For this individual the choice phase turns out to be unstructured or at most semi-structured. Examples 2.1, 2.2 and 2.3 describe structured, semi-structured and unstructured decisions, respectively.

Example 2.1 – A structured decision. A paper mill produces for the company warehouse paper sheets in different standard sizes that are subsequently cut to size for customers. Specifically, customers submit orders in terms of type of paper, quantity and size. The sizes specified in the orders are usually smaller than standard sizes and must be cut out of these.

The paper mill is therefore forced to consider how the sizes required to fulfill orders should best be combined and cut from standard sizes so as to minimize paper waste. This decision is common to many industries (paper, aluminum, wood, steel, glass, fabric) and can be very well supported by optimization models. However, even in connection with such structured decisions, particular circumstances and specific input values may require intervention by the decision maker to modify the plans obtained by means of optimization models. For example, the company may wish to favor a specific request of a customer considered strategic, introducing a *fast-processing lane* in the cutting plan, even if this may involve more wasted material during the cutting stage.

Example 2.2 – A semi-structured decision. The logistics manager of a manufacturing company needs to develop an annual plan. The logistic plan determines the allocation to each plant of the production volumes forecasted for the different market areas, the purchase of materials from each supplier with the related volumes and delivery times, the production lots for each manufacturing stage, the stock levels of sub-assemblies and end items, and the distribution of end items to the market areas. These decisions have a great economic and organizational impact that might greatly benefit from the adoption of a DSS based on large-scale optimization models. However, it is likely that in a real situation some elements are left to discretion of the decision makers, who may prefer a given logistic plan over another, even if it implies moderately higher costs compared to the optimal plan proposed by the model. For example, it might be appropriate to maintain unaltered the supply of parts purchased from a given supplier who is considered strategic for the future even though this supplier is less competitive than others, that are instead preferred by the optimization model in terms of minimum cost.

Example 2.3 – An unstructured decision. Consider an enterprise that is the target of a hostile takeover by a public offer made by a direct competitor. There are various possible defensive decisions and actions that are strongly dependent on the context in which the enterprise operates and the offer is made. It is difficult to envisage a systematic description of the decision process that might be later reproduced in other similar cases.

From the above examples it emerges that the nature of a decision process depends on many factors, including:

- the characteristics of the organization within which the system is placed;
- the subjective attitudes of the decision makers;
- the availability of appropriate problem-solving methodologies;
- the availability of effective decision support tools.

Depending on their scope, decisions can be classified as *strategic*, *tactical* and *operational*.

Strategic decisions. Decisions are strategic when they affect the entire organization or at least a substantial part of it for a long period of time. Strategic decisions strongly influence the general objectives and policies of an enterprise. As a consequence, strategic decisions are taken at a higher organizational level, usually by the company top management.

Tactical decisions. Tactical decisions affect only parts of an enterprise and are usually restricted to a single department. The time span is limited to a medium-term horizon, typically up to a year. Tactical decisions place themselves within the context determined by strategic decisions. In a company hierarchy, tactical decisions are made by middle managers, such as the heads of the company departments.

Operational decisions. Operational decisions refer to specific activities carried out within an organization and have a modest impact on the future. Operational decisions are framed within the elements and conditions determined by strategic and tactical decisions. Therefore, they are usually made at a lower organizational level, by knowledge workers responsible for a single activity or task such as sub-department heads, workshop foremen, back-office heads.

The characteristics of the information required in a decision-making process will change depending on the scope of the decisions to be supported, and consequently also the orientation of a DSS will vary accordingly. Figure 2.7 shows variations in the characteristics of the information as the scope of the decisions changes. The scheme can be used as an assessment tool when designing a DSS: once the scope of the decisions for which the system is intended

	Operational	Tactical	Strategic
Accuracy	High	↔	Low
Level of detail	Detailed	↔	Aggregate
Time horizon	Present	↔	Future
Frequency of use	High	↔	Low
Source	Internal	↔	External
Scope of information	Quantitative	↔	Qualitative
Nature of information	Narrow	↔	Wide
Age of information	Present	↔	Past

Figure 2.7 Characteristics of the information in terms of the scope of decisions

has been set, the scheme can be used to establish whether the decision-making process is adequately supported by the right information.

Although nature and scope are not perfectly correlated, most real-world decisions fall within the ellipse shown in Figure 2.6: most strategic decisions are unstructured, while most operational decisions are structured and most tactical decisions are semi-structured. This empirical remark is useful when defining in advance the characteristics of a DSS to facilitate a decision-making process of specific nature and scope.

2.2.4 Approaches to the decision-making process

As observed above, the subjective orientation of decision makers across an organization strongly influences the structure of the decision-making process. In this section we will review the major approaches that prevail in the management of complex organizations, and examine the implications when designing a DSS.

A preliminary distinction should be made between a *rational* approach and a *political-organizational* approach.

Rational approach. When a rational approach is followed, a decision maker considers major factors, such as economic, technical, legal, ethical, procedural and political, also establishing the criteria of evaluation so as to assess different options and then select the best decision. In this context, a DSS may help both in a passive way, through timely and versatile access to information, and in an active way, through the use of mathematical models for decision making.

Political-organizational approach. When a political-organizational approach is pursued, a decision maker proceeds in a more instinctual and less systematic way. Decisions are not based on clearly defined alternatives and selection criteria. As a consequence, a DSS can only help in a passive way, providing timely and versatile access to information. It might also be useful during discussions and negotiations in those decision-making processes that involve multiple actors, such as managers operating in different departments.

Within the rational approach we can further distinguish between two alternative ways in which the actual decision-making process influences decisions: *absolute rationality* and *bounded rationality*.

Absolute rationality. The term 'absolute rationality' refers to a decision-making process for which multiple performance indicators can be reduced to a single criterion, which therefore naturally lends itself to an optimization model. For example, a production manager who has to put together a medium-term logistic plan may be able to convert all performance indicators into monetary units, and therefore subsequently derive the solution with the minimum cost. This implies that non-monetary indicators, such as stock volumes or the number of days of delay in handling a given order, should be transformed into monetary measurement units. From a methodological perspective, this implies that a multi-objective optimization problem is transformed into a single-objective problem by expressing all the relevant factors in a common measurement unit that allows the heterogeneous objectives to be added together.

Bounded rationality. Bounded rationality occurs whenever it is not possible to meaningfully reduce multiple criteria into a single objective, so that the decision maker considers an option to be satisfactory when the corresponding performance indicators fall above or below prefixed threshold values. For instance, a production plan is acceptable if its cost is sufficiently low, the stock quantities are within a given threshold, and the service time is below customers expectations. Therefore, the concept of bounded rationality captures the rational choices that are constrained by the limits of knowledge and cognitive capability.

Most decision-making processes occurring within the enterprises and the public administration are aimed at making a decision that appears acceptable with respect to multiple evaluation criteria, and therefore decision processes based on bounded rationality are more likely to occur in practice.

2.3 Evolution of information systems

Decision support systems combine data and mathematical models to help decision makers in their work. To some extent they show connections with the information systems of an organization. Hence, it is worth describing in this section a time-line for the evolution of information systems, since this highlights how data processing has developed and has been used within companies.

Digital computers made their appearance in the late 1940s, and soon began to be applied in the business environment. The first decades saw a rush toward information technology development, usually under the mantra of *data processing*. They were characterized by the widespread diffusion of applications that achieved an increase in efficiency by automating routine operations within companies, especially in administration, production, research and development.

In the 1970s there began to arise within enterprises increasingly complex needs to devise software applications, called *management information systems* (MIS), in order to ease access to useful and timely information for decision makers. However, attempts to develop such systems were hampered by the state of information technologies at the time. The mainframe computers of those days lacked graphic visualization capabilities, and communicated with users through character-based computer terminals and dot printers. A further difficulty lay in the organizational structure of companies, based on a highly centralized information systems department, usually resulting in very long and frustrating time delays in implementing changes or extensions to the available applications.

From the late 1980s the introduction of personal computers with operating systems featuring graphic interfaces and pointing devices, such as a mouse or an optical pen, had two major consequences. On the one hand, it became possible to implement applications capable of sophisticated interactions and graphic presentation of results, a prerequisite for providing decision makers with really useful support tools. On the other hand, knowledge workers could rely on autonomous processing tools which made them substantially independent of the company information systems department, thus avoiding the above-mentioned time lag in data access. This led the most proactive knowledge workers to create local databases and develop simulation models, for example by means of spreadsheets, which can be regarded as true ancestors of today's business intelligence architectures. The increase in independent processing capabilities held by users, usually referred to as *end user computing*, was a critical enabling factor for future developments, as it helped circumscribe the importance of information systems departments.

Meanwhile, the initial concept of *decision support system* was also introduced, whose exact meaning will be described in the next section. Later developments brought to light new types of applications and architectures: *executive information systems* and *strategic information systems* were first introduced toward the late 1980s to support executives in the decision-making process. Such systems were intended for unstructured decision-making processes and therefore represented passive support systems oriented toward timely and easy access to information.

From the early 1990s, network architectures and distributed information systems based on *client–server* computing models began to be widely adopted. Moreover, there arose the need to logically and physically separate the databases intended for DSSs from operational information systems. This brought about the concepts of *data warehouses* and *data marts*, which will be described in Chapter 3.

Finally, toward the end of the 1990s, the term *business intelligence* began to be used to generally address the architecture containing DSSs, analytical methodologies and models used to transform data into useful information and knowledge for decision makers, as discussed in Chapter 1.

2.4 Definition of decision support system

Since the late 1980s, a decision support system has been defined as an interactive computer system helping decision makers to combine data and models to solve semi-structured and unstructured problems. This definition entails the three main elements of a DSS shown in Figure 2.8: a database, a repository of mathematical models and a module for handling the dialogue between the

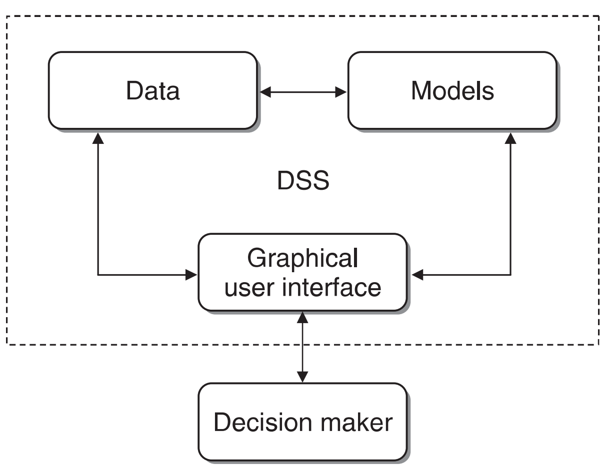

Figure 2.8 Structure of a decision support system

system and the users. It thus highlights the role of DSSs as the focal point of evolution trends in two distinct areas: on the one hand, data processing and information technologies; and on the other hand, the disciplines addressing the study of mathematical models and methods, such as operations research and statistics.

Indeed, despite significant improvements achieved in both areas, the actual implementation of applications that could be used by knowledge workers has been troublesome. Information technology in the early 1970s, as discussed in the previous section, mainly consisted of applications for accounting and administration, their capabilities restricted to the processing of large amounts of transactions and the production of summary reports. The partial failure of management information systems indicated that even the essential objective of attaining timely and versatile access to information was not within easy reach. On the other hand, although mathematical models were made more flexible by resort to innovative user interaction techniques, they were still mostly based on regulatory methods, suitable for operational decisions characterized by unique objectives rather than tactical and strategic decisions, usually less structured. As we have observed, semi-structured and unstructured decision processes are heavily influenced by subjective opinions, so that knowledge workers have to directly analyze multiple performance indicators which it would be difficult to reduce to a single decision-making criterion.

It is worth highlighting the relevant features of a DSS in order to circumscribe the definition given above and to better understand its role.

Effectiveness. Decision support systems should help knowledge workers to reach more effective decisions. In this respect they are a fundamental component of business intelligence architectures. Note that this does not necessarily imply an increased efficiency in the decision-making process. In fact, the adoption of a DSS may entail a more accurate analysis and therefore require a greater time investment by decision makers. However, the greater effort required will usually result in better decisions.

Mathematical models. In order to achieve more effective decisions, a DSS makes use of mathematical models, borrowed from disciplines such as operations research and statistics, which are applied to the data contained in data warehouses and data marts. The use of analytical models to transform data into knowledge and provide active support is the main characteristic that sets apart a DSS from a simple information system.

Integration in the decision-making process. A DSS should provide help for different kinds of knowledge workers, within the same application domain, particularly in respect of semi-structured and unstructured decision processes, both of an individual and a collective nature. Further, a DSS is intended for decision-making processes that are strategic, tactical and operational in scope. Moreover, decision makers should have the opportunity to integrate in a DSS their preferences and competencies, adapting it to their needs rather than passively accepting what comes out of it. In this way, a DSS may progressively take the role of a key component in the problem-solving methodology adopted by decision makers, enabling a *proactive* and *perceptive* decision-making style, instead of a *reactive* and *by-exception* attitude, in order to anticipate any rapidly evolving dynamic phenomenon.

Organizational role. In many situations the users of a DSS operate at different hierarchical levels within an enterprise, and a DSS tends to encourage communication between the various parts of an organization. By providing support for sequential and interdependent decision processes, a DSS can keep track of the analysis and the information that led to a specific decision.

Flexibility. A DSS must be flexible and adaptable in order to incorporate the changes required to reflect modifications in the environment or in the decision-making process. Moreover, it should be easy to use, with user-friendly and intuitive interaction methods and high-quality graphics for presenting the information extracted or generated. It is becoming increasingly common for DSSs to feature a web-browser interface to communicate with users.

The structure of the DSS shown in Figure 2.8 is extended in Figure 2.9 to include some new components.

Data management. The data management module includes a database designed to contain the data required by the decision-making processes to which the DSS is addressed. In most applications the database is a data mart, as we will see

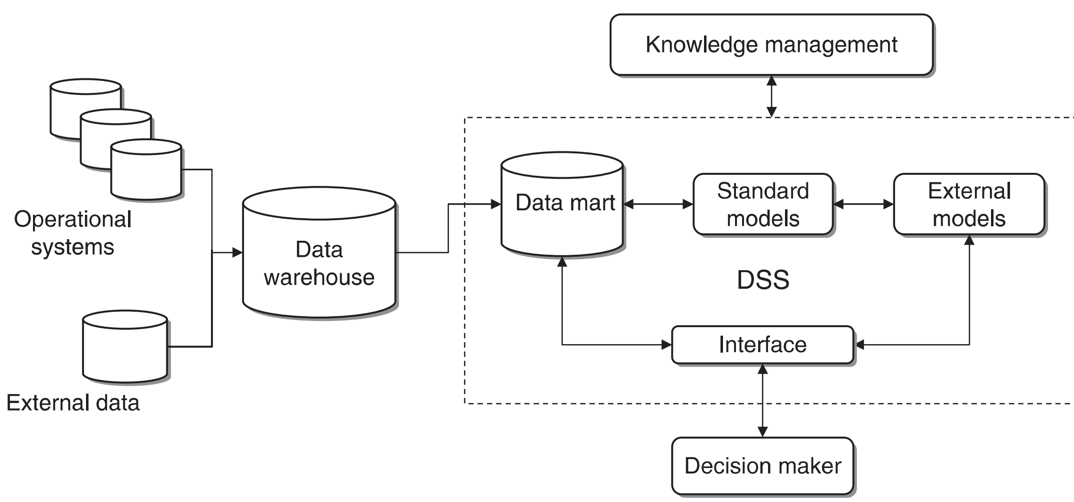

Figure 2.9 Extended structure of a decision support system

in Chapter 3. Keeping in mind current business intelligence architectures, the data management module of a DSS is usually connected with a company data warehouse, also described in Chapter 3, which represents the main repository of the data available to develop a business intelligence analysis.

Model management. The model management module provides end users with a collection of mathematical models derived from operations research, statistics and financial analysis. These are usually relatively simple models that allow analytical investigations to be carried out that are very helpful during the decision-making process. To illustrate the role played by the model management module one can think of the analytical functions offered by current spreadsheets. These include simple optimization models, financial and actuarial analysis models and statistical functionalities. Moreover, the model management module helps the activities of knowledge workers by means of high-level languages for the development of *ad hoc* models. In certain applications such a module may be integrated with more complex models, referred to as *external models* in Figure 2.9, created to carry out specific analysis tasks. For example, a large-scale optimization model formulated to develop the annual logistic plan of a manufacturing company falls in this category.

Interactions. In most applications, knowledge workers use a DSS interactively to carry out their analyses. The module responsible for these interactions is expected to receive input data from users in the easiest and most intuitive way, usually through the graphic interface of a web browser, and then to return the extracted information and the knowledge generated by the system in an appropriate graphical form.

Knowledge management. The knowledge management module is also interconnected with the company knowledge management integrated system. It allows decision makers to draw on the various forms of collective knowledge, usually unstructured, that represents the *corporate culture*.

This section concludes with a summary of the major potential advantages deriving from the adoption of a DSS:

- an increase in the number of alternatives or options considered;
- an increase in the number of effective decisions devised;
- a greater awareness and a deeper understanding of the domain analyzed and the problems investigated;
- the possibility of executing scenario and what-if analyses by varying the hypotheses and parameters of the mathematical models;

- an improved ability to react promptly to unexpected events and unforeseen situations;
- a value-added exploitation of the available data;
- an improved communication and coordination among the individuals and the organizational departments;
- more effective development of teamwork;
- a greater reliability of the control mechanisms, due to the increased intelligibility of the decision process.

2.5 Development of a decision support system

In this section we will describe the development phases of a DSS. Unlike other software applications, such as information systems and office automation tools, DSSs are usually not available as standard programs. Multidimensional analysis environments have facilitated and standardized the access to passive business intelligence functions. However, in order to develop most DSSs a specific project is still required.

Figure 2.10 shows the major steps in the development of a DSS. The logical flow of the activities is shown by the solid arrows. The dotted arrows in the opposite direction indicate revisions of one or more phases that might become necessary during the development of the system, through a feedback mechanism. We describe now in detail how each phase is carried out.

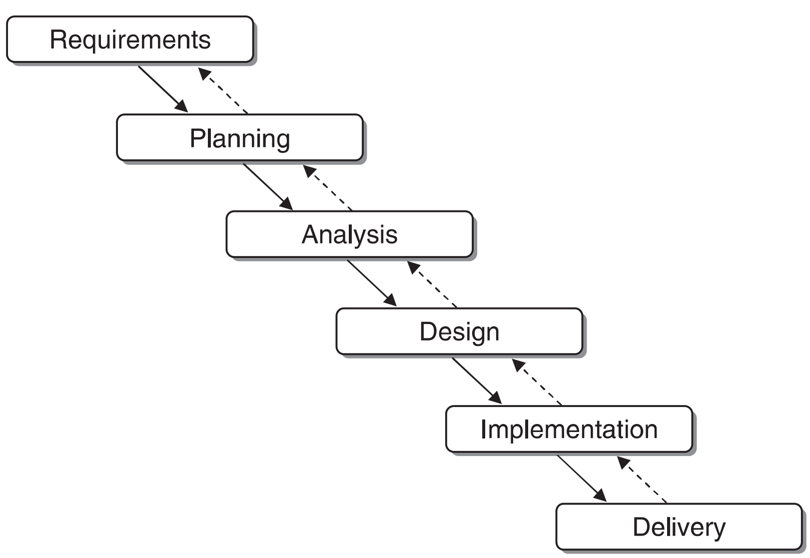

Figure 2.10 Phases in the development of a decision support system

Planning. The main purpose of the planning phase is to understand the needs and opportunities, sometimes characterized by *weak signals*, and to translate them into a project and later into a successful DSS. Planning usually involves a *feasibility study* to address the question: *Why do we wish to develop a DSS?* During the feasibility analysis, general and specific objectives of the system, recipients, possible benefits, execution times and costs are laid down. It is not easy to identify the benefits of a DSS. As already observed, the main advantage for most DSSs is not so much an increased effectiveness as an improvement in decision quality, which is difficult to predict a priori. The elements required to decide whether or not the system should be implemented become available at the end of the feasibility study. A negative decision will call a halt to the development of the project, although of course this may be reconsidered at a later time. If one decides to proceed with the system, the planning phase should be followed by the definition of the activities, tasks, responsibilities and development phases, for which classical project management methodologies should be used.

Analysis. In the analysis phase, it is necessary to define in detail the functions of the DSS to be developed, by further developing and elaborating the preliminary conclusions achieved during the feasibility study. A response should therefore be given to the following question: *What should the DSS accomplish, and who will use it, when and how?* To provide an answer, it is necessary to analyze the decision processes to be supported, to try to thoroughly understand all interrelations existing between the problems addressed and the surrounding environment. The organizational implications determined by a DSS should be assessed. The analysis also involves mapping out the actual decision processes and imagining what the new processes will look like once the DSS is in place. Finally, it is necessary to explore the data in order to understand how much and what type of information already exists and what information can be retrieved from external sources.

Design. During the design phase the main question is: *How will the DSS work?* The entire architecture of the system is therefore defined, through the identification of the hardware technology platforms, the network structure, the software tools to develop the applications and the specific database to be used. It is also necessary to define in detail the interactions with the users, by means of input masks, graphic visualizations on the screen and printed reports. In recent years the web browser has become an important interaction tool, and has certainly contributed to the harmonization of, and to the simplification of the problems related to, communication between knowledge workers and computers. A further aspect that should be clarified during the design phase is the *make-or-buy* choice – whether to subcontract the implementation of the DSS to third parties, in whole or in part.

Implementation. Once the specifications have been laid down, it is time for implementation, testing and the actual installation, when the DSS is rolled out and put to work. Any problems faced in this last phase can be traced back to project management methods. A further aspect of the implementation phase, which is often overlooked, relates to the overall impact on the organization determined by the new system. Such effects should be monitored using *change management* techniques, making sure that no one feels excluded from the organizational innovation process and rejects the DSS.

Sometimes a project may not come to a successful conclusion, may not succeed in fulfilling expectations, or may even turn out to be a complete failure. However, there are ways to reduce the risk of failure. The most significant of these is based on the use of *rapid prototyping development* where, instead of implementing the system as a whole, the approach is to identify a sequence of autonomous subsystems, of limited capabilities, and develop these subsystems step by step until the final stage is reached corresponding to the fully developed DSS.

Rapid prototyping development offers clear advantages. Each subsystem can be actually developed more quickly and therefore is more readily available. Moreover, when a subsystem is released to users it is possible to verify its conformity with the intended purpose and test its functions even if these are still not fully developed. Hence, the evolutionary development of a DSS allows the risk of failure to be minimized. Furthermore, intermediate tests make it possible to promptly correct most design errors. In a situation where there is a clear discrepancy between the features of the prototype developed and the users' expectations it is even possible to come to an early interruption of the project.

Other development methods can be effectively adopted in order to speed up the implementation of the software. These include *agile development techniques* and *extreme programming techniques*.

A further aspect that should not be overlooked is the periodic administration and revision of the DSS. Since no software application should be considered completely finished and unchangeable over time, this is even more critical for a tool intended to support knowledge workers in the decision-making process in rapidly changing dynamic situations. As a consequence, it is necessary to develop a DSS by making provision for future changes and adjustments.

In this respect it is possible to identify two clear advantages offered by the development methods described so far. First, if a DSS has been developed using rapid prototyping techniques, it is likely to be more amenable to periodic extensions and revisions. Furthermore, the mathematical models used to develop a

DSS are suitable for incorporating evolving changes and updates which often reduce to simple modifications in the expressions that describe the model.

We conclude this section by considering the major critical factors that may affect the degree of success of a DSS.

Integration. The design and development of a DSS require a significant number of methodologies, tools, models, individuals and organizational processes to work in harmony. This results in a highly complex project requiring diverse competencies. The role of a *system integrator* is thus essential. A system integrator is an expert in all the aspects involved in the development of a DSS, such as information system architectures, decision-making processes, mathematical models and solution methods. This role is usually performed by a third party who may also exert a positive influence as an *agent of innovation* capable of overcoming most of the resistance to change that often arises in every organization.

Involvement. The exclusion or marginalization from the project team of knowledge workers who will actually use the system once it is implemented is a mistake that is sometimes made during the design and development of a DSS. In many cases this happens because a DSS is mistakenly considered a mere computer application, the development of which is assigned solely or primarily to the information systems department. Conversely, the involvement of decision makers and users during the development process is of primary importance to avert their inclination to reject a tool that they perceive as alien. It is also necessary to promote informal communication processes, especially during the design and implementation phases of the first prototypes.

Uncertainty. In general, costs are not a major concern in the implementation of a DSS, and the advantage of devising more effective decisions largely offsets the development costs incurred. Of course, it is appropriate to reduce the project uncertainty through prototyping, user friendliness, system tests during the preliminary stages and an evolutionary implementation.

2.6 Notes and readings

Among the many volumes devoted to decision support systems, we wish to suggest the following readings: Mallach (2000), Turban *et al*. (2005), Klein and Methlie (1995) and Dhar and Stein (1997). For a recent review, see Shim *et al*. (2002). Moss and Atre (2003) analyze the process of developing a DSS. Among the major contributions dating back to the foundations of DSSs, see Simon (1969, 1977), Gorry and Scott Morton (1971), Keen and Scott Morton (1978) and Sprague (1980).

3

Data warehousing

As observed in Chapter 2, from the mid-1990s the need was felt for a logical and material separation between the databases feeding input data into decision support systems and business intelligence architectures on the one hand, and operational information systems on the other.

In this chapter we will describe the features of *data warehouses* and *data marts*, illustrating the factors that led to their conception, and highlighting the major differences between them and operational systems, and discussing the requirements concerning data quality. Then we will examine the architecture of a data warehouse, pointing out the role of ETL tools and metadata. The last part of the chapter will be devoted to on-line analytical processing operations and analyses that can be performed by using multidimensional cubes and hierarchies of concepts.

We will focus our discussion on the goals and functions of a data warehouse, deliberately avoiding technical issues relating to their development. For these latter, readers may refer to more specific books fully devoted to the subject, indicated in the last section of the chapter.

3.1 Definition of data warehouse

As its name suggests, a data warehouse is the foremost repository for the data available for developing business intelligence architectures and decision support systems. The term *data warehousing* indicates the whole set of interrelated activities involved in designing, implementing and using a data warehouse.

It is possible to identify three main categories of data feeding into a data warehouse: *internal data*, *external data* and *personal data*.

Business Intelligence: *Data Mining and Optimization for Decision Making* *Carlo Vercellis*
 ISBN: 978-0-470-51138-1

Internal data. Internal data are stored for the most part in the databases, referred to as *transactional systems* or *operational systems*, that are the backbone of an enterprise information system. Internal data are gathered through transactional applications that routinely preside over the operations of a company, such as administration, accounting, production and logistics. This collection of transactional software applications is termed *enterprise resource planning* (ERP).
The data stored in the operational systems usually deal with the main entities involved in a company processes, namely customers, products, sales, employees and suppliers. These data usually come from different components of the information system:

- *back-office systems*, that collect basic transactional records such as orders, invoices, inventories, production and logistics data;
- *front-office systems*, that contain data originating from call-center activities, customer assistance, execution of marketing campaigns;
- *web-based systems*, that gather sales transactions on e-commerce websites, visits to websites, data available on forms filled out by existing and prospective customers.

External data. There are several sources of external data that may be used to extend the wealth of information stored in the internal databases. For example, some agencies gather and make available data relative to sales, market share and future trend predictions for specific business industries, as well as economic and financial indicators. Other agencies provide data market surveys and consumer opinions collected through questionnaires.
A further significant source of external data is provided by *geographic information systems* (GIS), which represent a set of applications for acquiring, organizing, storing and presenting territorial data. These contain information relative to entities having a specific geographic position. Each entity is therefore associated with latitude and longitude coordinates, along with some other attributes, usually originating from relational databases and actually depending on the application domain. Hence, these data allow to subject-specific analyses to be carried out on the data associated with geographic elements and the results to be graphically visualized.

Personal data. In most cases, decision makers performing a business intelligence analysis also rely on information and personal assessments stored inside worksheets or local databases located in their computers. The retrieval of such information and its integration with structured data from internal and external sources is one of the objectives of knowledge management systems.

Software applications that are at the heart of operational systems are referred to as *on-line transaction processing* (OLTP). On the other hand, the whole set of tools aimed at performing business intelligence analyses and supporting decision-making processes go by the name of *on-line analytical processing* (OLAP). We can therefore assume that the function of a data warehouse is to provide input data to OLAP applications.

There are several reasons for implementing a data warehouse separately from the databases supporting OLTP applications in an enterprise. Among them, we recall here the most relevant.

Integration. In many instances, decision support systems must access information originating from several data sources, distributed across different parts of an organization or deriving from external sources. A data warehouse integrating multiple and often heterogeneous sources is then required to promote and facilitate the access to information. Data integration may be achieved by means of different techniques – for example, by using uniform encoding methods, converting to standard measurement units and achieving a semantic homogeneity of information.

Quality. The data transferred from operational systems into the data warehouse are examined and corrected in order to obtain reliable and error-free information, as much as possible. Needless to say, this increases the practical value of business intelligence systems developed starting from the data contained in a data warehouse.

Efficiency. Queries aimed at extracting information for a business intelligence analysis may turn out to be burdensome in terms of computing resources and processing time. As a consequence, if a 'killer' query were directed to the transactional systems it would risk severely compromising the efficiency required by enterprise resource planning applications, with negative consequences on the routine activities of a company. A better solution is then to direct complex queries for OLAP analyses to the data warehouse, physically separated from the operational systems.

Extendability. The data stored in transactional systems stretch over a limited time span in the past. Indeed, due to limitations on memory capacity, data relative to past periods are regularly removed from OLTP systems and permanently archived in off-line mass-storage devices, such as DVDs or magnetic tapes. On the other hand, business intelligence systems and prediction models need to access all available past data to be able to grasp trends and detect recurrent patterns. This is possible due to the ability of data warehouses to retain historical information.

In light of the previous remarks, we can define a data warehouse as a collection of data supporting decision-making processes and business intelligence systems having the following characteristics.

Entity-oriented. The data contained in a data warehouse are primarily concerned with the main entities of interest for the analysis, such as products, customers, orders and sales. On the other hand, transactional systems are more oriented toward operational activities and are based on each single transaction recorded by enterprise resource planning applications. During a business intelligence analysis, orientation toward the entities allows the performance of a company to be more easily evaluated and any potential source of inefficiencies to be detected.

Integrated. The data originating from the different sources are integrated and homogenized as they are loaded into a data warehouse. For example, measurement units and encodings are harmonized and made consistent.

Time-variant. All data entered in a data warehouse are labeled with the time period to which they refer. We can fairly relate the data stored in a data warehouse to a sequence of nonvolatile snapshot pictures, taken at successive times and bearing the label of the reference period. As a consequence, the temporal dimension in any data warehouse is a critical element that plays a predominant role. In this way decision support applications may develop historical trend analysis.

Persistent. Once they have been loaded into a data warehouse, data are usually not modified further and are held permanently. This feature makes it easier to organize read-only access by users and simplifies the updating process, avoiding concurrency which is of critical importance for operational systems.

Consolidated. Usually some data stored in a data warehouse are obtained as partial summaries of primary data belonging to the operational systems from which they originate. For example, a mobile phone company may store in a data warehouse the total cost of the calls placed by each customer in a week, subdivided by traffic routes and by type of service selected, instead of storing the individual calls recorded by the operational systems. The reason for such consolidation is twofold: on one hand, the reduction in the space required to store in the data warehouse the data accumulated over the years; on the other hand, consolidated information may be able to better meet the needs of business intelligence systems.

Denormalized. Unlike operational databases, the data stored in a data warehouse are not structured in normal form but can instead make provision for redundancies, to allow shorter response time to complex queries.

Granularity represents the highest level of detail expressed by the primary data contained in a data warehouse, also referred to as *atomic data*. Obviously, the granularity of a data warehouse cannot exceed that of the original data

Table 3.1 Differences between OLTP and OLAP systems

Characteristic	OLTP	OLAP
volatility	dynamic data	static data
timeliness	current data only	current and historical data
time dimension	implicit and current	explicit and variant
granularity	detailed data	aggregated and consolidated data
updating	continuous and irregular	periodic and regular
activities	repetitive	unpredictable
flexibility	low	high
performance	high, few seconds per query	may be low for complex queries
users	employees	knowledge workers
functions	operational	analytical
purpose of use	transactions	complex queries and decision support
priority	high performance	high flexibility
metrics	transaction rate	effective response
size	megabytes to gigabytes	gigabytes to terabytes

sources. In general, it is strictly lower due to consolidation aimed at reducing storage occupancy, as described above.

The design philosophy behind data warehouses is quite different from that adopted for operational databases. Table 3.1 summarizes the main differences between OLTP and OLAP systems.

3.1.1 Data marts

Data marts are systems that gather all the data required by a specific company department, such as marketing or logistics, for the purpose of performing business intelligence analyses and executing decision support applications specific to the function itself. Therefore, a data mart can be considered as a functional or departmental data warehouse of a smaller size and a more specific type than the overall company data warehouse.

A data mart therefore contains a subset of the data stored in the company data warehouse, which are usually integrated with other data that the company department responsible for the data mart owns and deems of interest. For example, a marketing data mart will contain data extracted from the central data warehouse, such as information on customers and sales transactions, but also additional data pertaining to the marketing function, such as the results of marketing campaigns run in the past.

Data warehouses and data marts thus share the same technological framework. In order to implement business intelligence applications, some companies prefer to design and develop in an incremental way a series of integrated data marts rather than a central data warehouse, in order to reduce the implementation time and uncertainties connected with the project.

3.1.2 Data quality

The need to verify, preserve and improve the quality of data is a constant concern of those responsible for the design and updating of a data warehouse. The main problems that might compromise the validity and integrity of the data are shown in Table 3.2.

More generally, we can identify the following major factors that may affect data quality.

Accuracy. To be useful for subsequent analyses, data must be highly accurate. For instance, it is necessary to verify that names and encodings are correctly represented and values are within admissible ranges.

Completeness. In order to avoid compromising the accuracy of business intelligence analyses, data should not include a large number of missing values. However, one should keep in mind that most learning and data mining techniques are capable of minimizing in a robust way the effects of partial incompleteness in the data.

Consistency. The form and content of the data must be consistent across the different data sources after the integration procedures, with respect to currency and measurement units.

Table 3.2 Data integrity: problems, causes and remedies

Problem	Cause	Remedy
incorrect data	data collected without due care	systematic checking of input data
	data entered incorrectly	data entry automation
	uncontrolled modification of data	implementation of a safety program for access and modifications
data not updated	data collection does not match user needs	timely updating and collection of data
		retrieval of updated data from the web
missing data	failure to collect the required data	identification of data needed via preliminary analysis and estimation of missing data

Timeliness. Data must be frequently updated, based on the objectives of the analysis. It is customary to arrange an update of the data warehouse regularly on a daily or at most weekly basis.

Non-redundancy. Data repetition and redundancy should be avoided in order to prevent waste of memory and possible inconsistencies. However, data can be replicated when the denormalization of a data warehouse may result in reduced response times to complex queries.

Relevance. Data must be relevant to the needs of the business intelligence system in order to add real value to the analyses that will be subsequently performed.

Interpretability. The meaning of the data should be well understood and correctly interpreted by the analysts, also based on the documentation available in the metadata describing a data warehouse, as illustrated in Section 3.2.2.

Accessibility. Data must be easily accessible by analysts and decision support applications.

3.2 Data warehouse architecture

The reference architecture of a data warehouse, shown in Figure 3.1, includes the following major functional components.

- The data warehouse itself, together with additional data marts, that contains the data and the functions that allow the data to be accessed, visualized and perhaps modified.

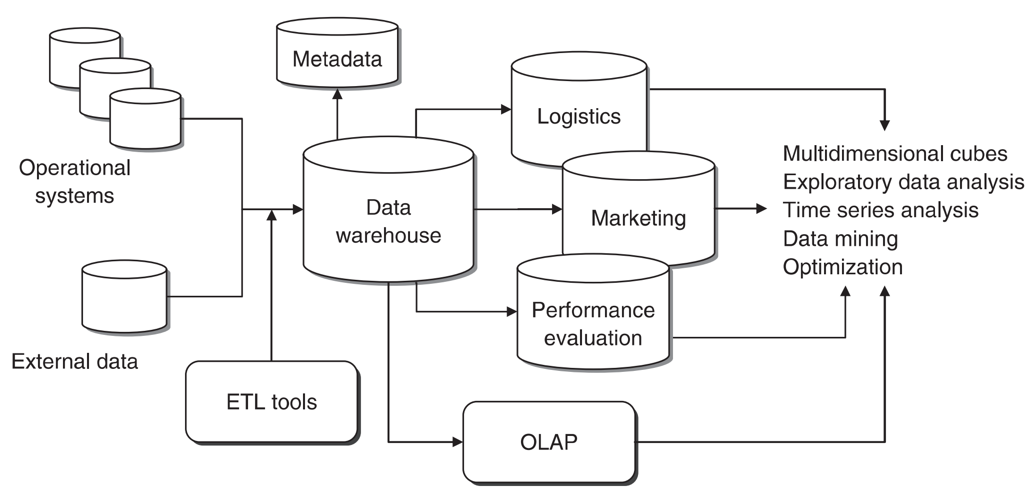

Figure 3.1 Architecture and functions of a data warehouse

- Data acquisition applications, also known as *extract, transform and load* (ETL) or *back-end* tools, which allow the data to be extracted, transformed and loaded into the data warehouse.

- Business intelligence and decision support applications, which represent the *front-end* and allow the knowledge workers to carry out the analyses and visualize the results.

The three-level distinction applies to the architecture shown in Figure 3.1 even from a technological perspective.

- The level of the data sources and the related ETL tools that are usually installed on one or more servers.

- The level of the data warehouse and any data mart, possibly available on one or more servers as well, and separated from those containing the data sources. This second level also includes the metadata documenting the origin and meaning of the records stored in the data warehouse.

- The level of the analyses that increase the value of the information contained in a data warehouse through query, reporting and possibly sophisticated decision support tools. The applications for business intelligence and decision support analysis are usually found on separate servers or directly on the client PC used by analysts and knowledge workers.

The same *database management system* platforms utilized to develop transactional systems are also adopted to implement data warehouses and data marts. Due to the response requirements raised by the complex queries addressed to a data warehouse, database management system platforms used for data warehousing are subject to different structuring and parameterizations with respect to transactional systems.

A data warehouse may be implemented according to different design approaches: *top-down*, *bottom-up* and *mixed*.

Top-down. The top-down methodology is based on the overall design of the data warehouse, and is therefore more systematic. However, it implies longer development times and higher risks of not being completed within schedule since the whole data warehouse is actually being developed.

Bottom-up. The bottom-up method is based on the use of prototypes and therefore system extensions are made according to a step-by-step scheme. This approach is usually quicker, provides more tangible results but lacks an overall vision of the entire system to be developed.

Mixed. The mixed methodology is based on the overall design of the data warehouse, but then proceeds with a prototyping approach, by sequentially implementing different parts of the entire system. This approach is highly practical and usually preferable, since it allows small and controlled steps to be taken while bearing in mind the whole picture.

The steps in the development of a data warehouse or a data mart can be summarized as follows.

- One or more processes within the organization to be represented in the data warehouse are identified, such as sales, logistics or accounting.
- The appropriate granularity to represent the selected processes is identified and the atomic level of the data is defined.
- The relevant measures to be expressed in the fact tables for multidimensional analysis are then chosen, as described in Section 3.3.
- Finally, the dimensions of the fact tables are determined.

3.2.1 ETL tools

ETL refers to the software tools that are devoted to performing in an automatic way three main functions: *extraction*, *transformation* and *loading* of data into the data warehouse.

Extraction. During the first phase, data are extracted from the available internal and external sources. A logical distinction can be made between the initial extraction, where the available data relative to all past periods are fed into the empty data warehouse, and the subsequent incremental extractions that update the data warehouse using new data that become available over time. The selection of data to be imported is based upon the data warehouse design, which in turn depends on the information needed by business intelligence analyses and decision support systems operating in a specific application domain.

Transformation. The goal of the cleaning and transformation phase is to improve the quality of the data extracted from the different sources, through the correction of inconsistencies, inaccuracies and missing values. Some of the major shortcomings that are removed during the data cleansing stage are:

- inconsistencies between values recorded in different attributes having the same meaning;
- data duplication;
- missing data;
- existence of inadmissible values.

During the cleaning phase, preset automatic rules are applied to correct most recurrent mistakes. In many instances, dictionaries with valid terms are used to substitute the supposedly incorrect terms, based upon the level of similarity. Moreover, during the transformation phase, additional data conversions occur in order to guarantee homogeneity and integration with respect to the different data sources. Furthermore, data aggregation and consolidation are performed in order to obtain the summaries that will reduce the response time required by subsequent queries and analyses for which the data warehouse is intended.

Loading. Finally, after being extracted and transformed, data are loaded into the tables of the data warehouse to make them available to analysts and decision support applications.

3.2.2 Metadata

In order to document the meaning of the data contained in a data warehouse, it is recommended to set up a specific information structure, known as *metadata*, i.e. data describing data. The metadata indicate for each attribute of a data warehouse the original source of the data, their meaning and the transformations to which they have been subjected. The documentation provided by metadata should be constantly kept up to date, in order to reflect any modification in the data warehouse structure. The documentation should be directly accessible to the data warehouse users, ideally through a web browser, according to the access rights pertaining to the roles of each analyst.

In particular, metadata should perform the following informative tasks:

- a documentation of the data warehouse structure: layout, logical views, dimensions, hierarchies, derived data, localization of any data mart;
- a documentation of the data genealogy, obtained by tagging the data sources from which data were extracted and by describing any transformation performed on the data themselves;
- a list keeping the usage statistics of the data warehouse, by indicating how many accesses to a field or to a logical view have been performed;
- a documentation of the general meaning of the data warehouse with respect to the application domain, by providing the definition of the terms utilized, and fully describing data properties, data ownership and loading policies.

3.3 Cubes and multidimensional analysis

The design of data warehouses and data marts is based on a multidimensional paradigm for data representation that provides at least two major advantages: on the functional side, it can guarantee fast response times even to complex queries, while on the logical side the dimensions naturally match the criteria followed by knowledge workers to perform their analyses.

The multidimensional representation is based on a *star schema* which contains two types of data tables: *dimension tables* and *fact tables*.

Dimension tables. In general, *dimensions* are associated with the entities around which the processes of an organization revolve. Dimension tables then correspond to primary entities contained in the data warehouse, and in most cases they directly derive from *master tables* stored in OLTP systems, such as customers, products, sales, locations and time. Each dimension table is often internally structured according to *hierarchical* relationships. For example, the temporal dimension is usually based upon two major hierarchies: {day, week, year} and {day, month, quarter, year}. Similarly, the location dimension may be hierarchically organized as {street, zip code, city, province, region, country, area}. Products in their turn have hierarchical structures such as {item, family, type} in the manufacturing industry and {item, category, department} in the retail industry. In a way, dimensions predetermine the main paths along which OLAP analyses will presumably be developed.

Fact tables. Fact tables usually refer to transactions and contain two types of data:

- links to dimension tables, that are required to properly reference the information contained in each fact table;
- numerical values of the attributes that characterize the corresponding transactions and that represent the actual target of the subsequent OLAP analyses.

For example, a fact table may contain sales transactions and make reference to several dimension tables, such as customers, points of sale, products, suppliers, time. The corresponding measures of interest are attributes such as quantity of items sold, unit price and discount. In this example the fact table allows analysts to evaluate the trends of sales over time, either total, or referred to a single customer, or referred to a group of customers, that can be identified through any hierarchy induced by the dimension table associated with the customers. The analyst may also evaluate the trend over time of sales percentages relative to customers located in a specific region.

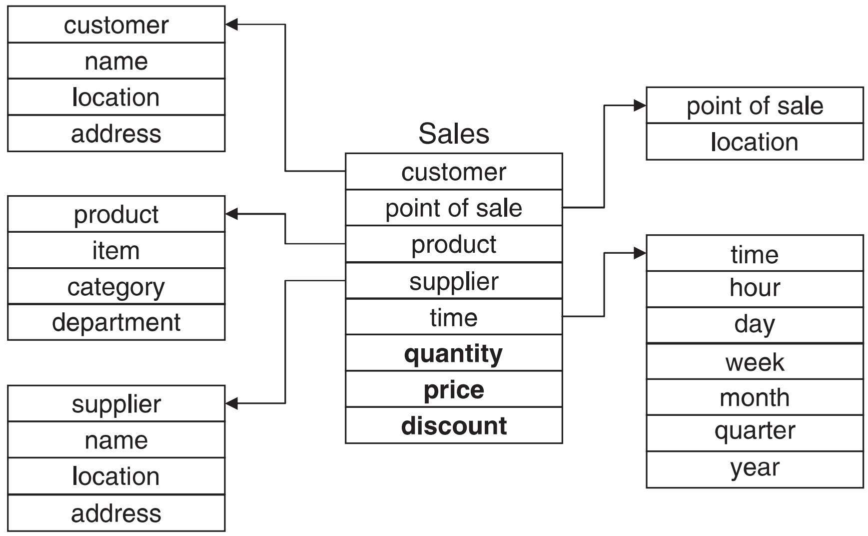

Figure 3.2 Example of a star schema

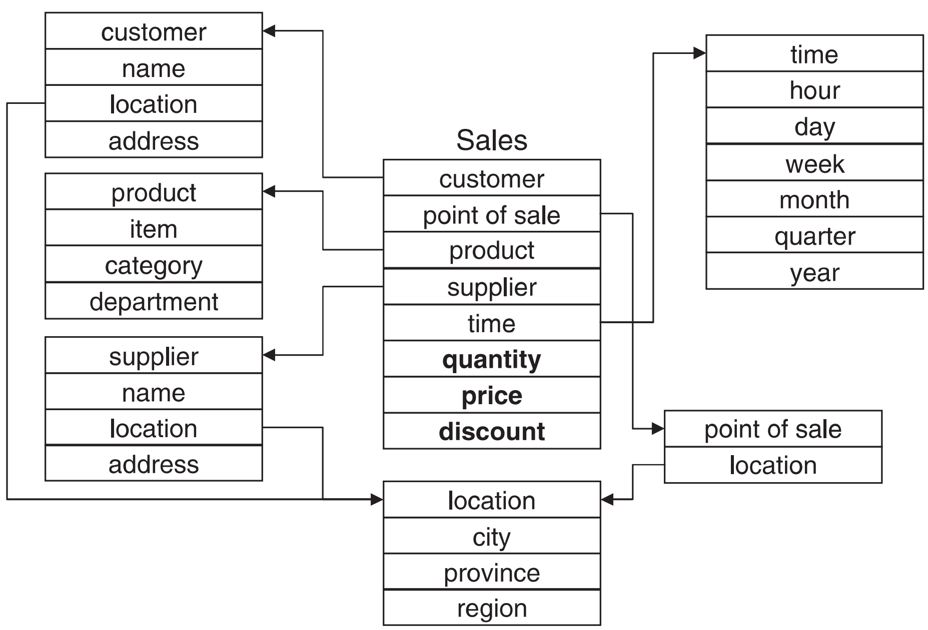

Figure 3.3 Example of a snowflake schema

Figure 3.2 shows the star schema associated with the fact table representing sales transactions. The fact table is placed in the middle of the schema and is linked to the dimension tables through appropriate references. The measures in the fact table appear in bold type.

Sometimes dimension tables are connected in their turn to other dimension tables, as shown in Figure 3.3, through a process of partial data standardization, in order to reduce memory use. In the given example the dimension table

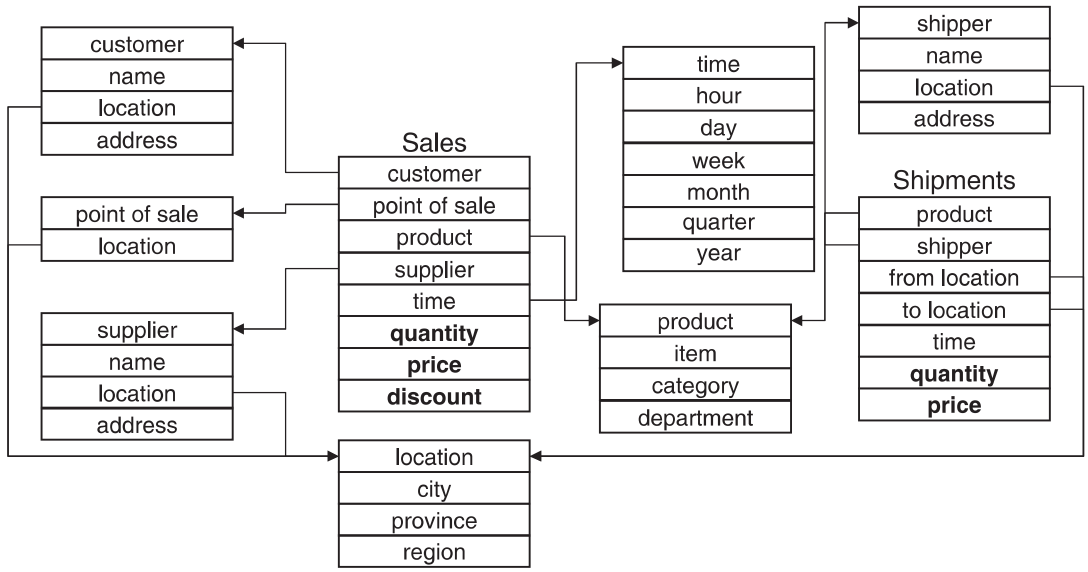

Figure 3.4 Example of a galaxy schema

referring to the location is in turn hierarchically connected with the dimension table containing geographical information. This brings about a *snowflake schema*.

A data warehouse includes several fact tables, interconnected with dimension tables, linked in their turn with other dimension tables. The latter type of schema, shown in Figure 3.4, is termed a *galaxy schema*.

A fact table connected with n dimension tables may be represented by an n-dimensional *data cube* where each axis corresponds to a dimension. Multidimensional cubes are a natural extension of the popular two-dimensional spreadsheets, which can be interpreted as two-dimensional cubes. For instance, consider a sales fact table developed along the three dimensions of {time, product, region}. Suppose we select only two dimensions for the analysis, such as {time, product}, having preset the region attribute along the three values {USA, Asia, Europa}. In this way we obtain the three two-dimensional tables in which the rows correspond to quarters of a year and the columns to products (see Tables 3.3–3.5). The cube shown in Figure 3.5 is a three-dimensional illustration of the same sales fact table. Atomic data are represented by 36 cells that can be obtained by crossing all possible values along the three dimensions: time {Q1, Q2, Q3, Q4}, region {USA, Asia, Europa} and product {TV, PC, DVD}. These atomic cells can be supplemented by 44 cells corresponding to the summary values obtained through consolidation along one or more dimensions, as shown by the cube in the figure.

Suppose that the sales fact table also contains a fourth dimension represented by the suppliers. The corresponding data cube constitutes a structure in

Table 3.3 Two-dimensional view of sales data in the USA

region = USA			
	product		
time	TV	PC	DVD
Q1	980	546	165
Q2	765	456	231
Q3	879	481	192
Q4	986	643	203

Table 3.4 Two-dimensional view of sales data in Asia

region = Asia			
	product		
time	TV	PC	DVD
Q1	789	456	187
Q2	654	732	157
Q3	623	354	129
Q4	756	876	231

Table 3.5 Two-dimensional view of sales data in Europe

region = Europe			
	product		
time	TV	PC	DVD
Q1	638	576	192
Q2	876	723	165
Q3	798	675	154
Q4	921	754	201

four-dimensional space and therefore cannot be represented graphically. However, we can obtain four logical views composed of three-dimensional cubes, called *cuboids*, inside the four-dimensional cube, by fixing the values of one dimension.

More generally, starting from a fact table linked to n dimension tables, it is possible to obtain a lattice of cuboids, each of them corresponding to a different level of consolidation along one or more dimensions. This type of aggregation is equivalent in *structured query language* (SQL) to a query *sum* derived from a *group-by* condition. Figure 3.6 illustrates the lattice composed by the cuboids obtained from the data cube defined along the four dimensions {time, product, region, supplier}.

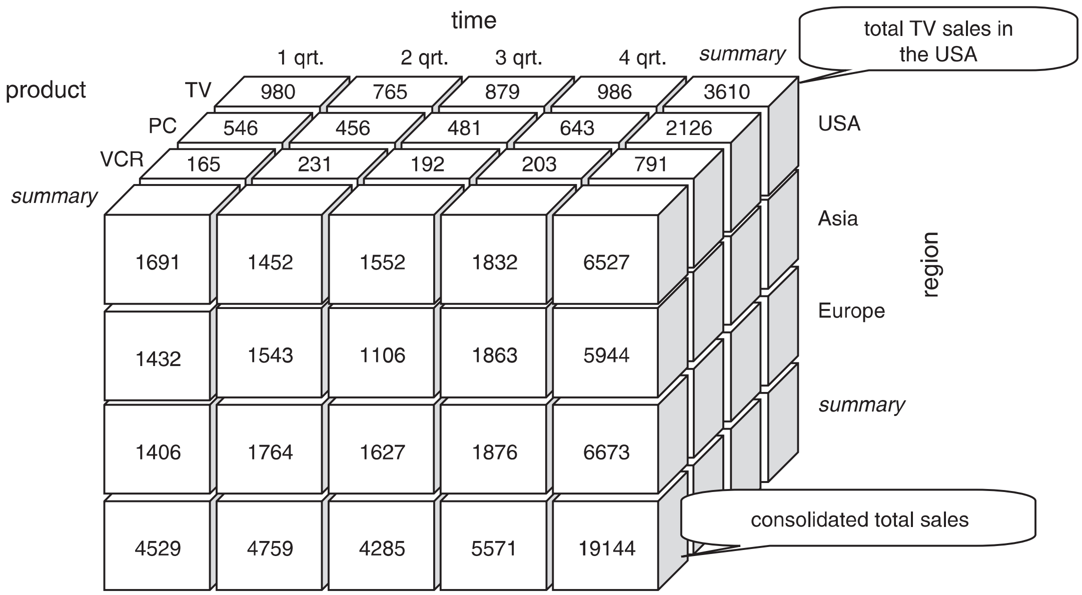

Figure 3.5 Example of a three-dimensional cube

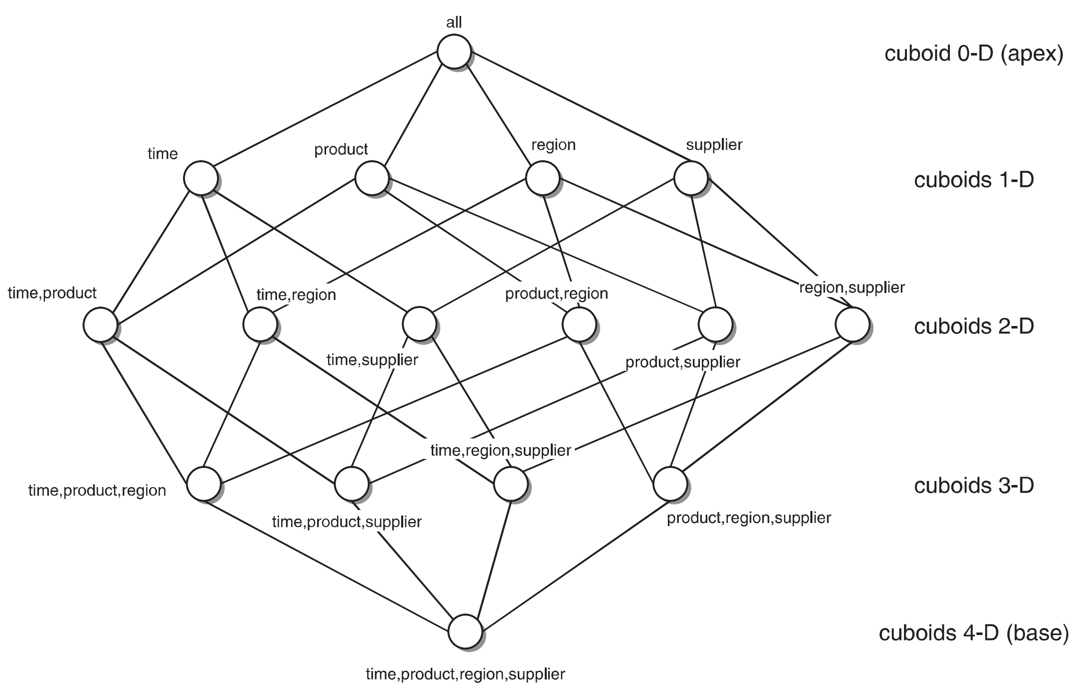

Figure 3.6 Lattice of cuboids derived from a four-dimensional cube

The cuboid associated with the atomic data, which therefore does not imply any type of consolidation, is called a *base cuboid*. At the other extreme, the *apex cuboid* is defined as the cuboid corresponding to the consolidation along all dimensions, therefore associated with the grand total of the measure of interest.

3.3.1 Hierarchies of concepts and OLAP operations

In many instances, OLAP analyses are based on *hierarchies of concepts* to consolidate the data and to create *logical views* along the dimensions of a data warehouse. A concept hierarchy defines a set of maps from a lower level of concepts to a higher level.

For example, the {location} dimension may originate a totally ordered hierarchy, as shown in Figure 3.7, developing along the {address, municipality, province, country} relationship. The temporal dimension, on the other hand, originates a partially ordered hierarchy, also shown in Figure 3.7.

Specific hierarchy types may be predefined in the software platform used for the creation and management of a data warehouse, as in the case of the dimensions shown in Figure 3.7. For other hierarchies it is necessary for analysts to explicitly define the relationships among concepts.

Hierarchies of concepts are also used to perform several visualization operations dealing with data cubes in a data warehouse.

Roll-up. A *roll-up* operation, also termed *drill-up*, consists of an aggregation of data in the cube, which can be obtained alternatively in the following two ways.

- Proceeding upwards to a higher level along a single dimension defined over a concepts hierarchy. For example, for the {location} dimension it is possible to move upwards from the {city} level to the {province} level and to consolidate the measures of interest through a *group-by* conditioned sum over all records whereby the city belongs to the same province.

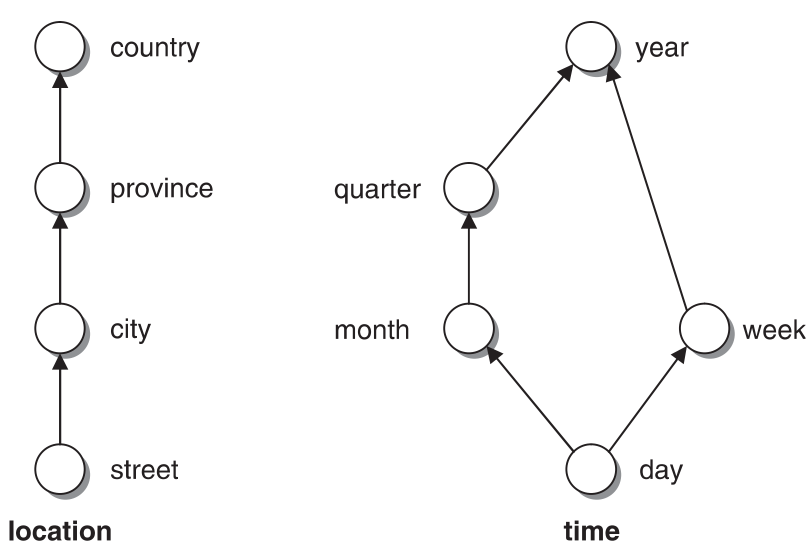

Figure 3.7 Hierarchies of concepts

- Reducing by one dimension. For example, the removal of the {time} dimension leads to consolidated measures through the sum over all time periods existing in the data cube.

Roll-down. A *roll-down* operation, also referred to as *drill-down*, is the opposite operation to roll-up. It allows navigation through a data cube from aggregated and consolidated information to more detailed information. The effect is to reverse the result achieved through a roll-up operation. A drill-down operation can therefore be carried out in two ways.

- Shifting down to a lower level along a single dimension hierarchy. For example, in the case of the {location} dimension, it is possible to shift from the {province} level to the {city} level and to disaggregate the measures of interest over all records whereby the city belongs to the same province.
- Adding one dimension. For example, the introduction of the {time} dimension leads to disaggregate the measures of interest over all time periods existing in a data cube.

Slice and dice. Through the *slice* operation the value of an attribute is selected and fixed along one dimension. For example, Table 3.3 has been obtained by fixing the region at the {Usa} value. The *dice* operation obtains a cube in a subspace by selecting several dimensions simultaneously.

Pivot. The *pivot* operation, also referred to as *rotation*, produces a rotation of the axes, swapping some dimensions to obtain a different view of a data cube.

3.3.2 Materialization of cubes of data

OLAP analyses developed by knowledge workers may need to access the information associated with several cuboids, based on the specific queries and analyses being carried out. In order to guarantee adequate response time, it might be useful to design a data warehouse where all (or at least a large portion of) values of the measures of interest associated with all possible cuboids are pre-calculated. This approach is termed *full materialization* of the information relative to the data cubes.

Observe that where hierarchies of concepts are missing, it is possible to form 2^n distinct cuboids from all possible combinations of n dimensions. The existence of hierarchies along different dimensions makes the number of distinct cuboids even greater. If L_i denotes the number of hierarchical levels associated with the ith dimension, for an n-dimensional data cube it is possible to calculate the full number of cuboids, given by

$$T = \prod_{i=1}^{n} (L_i + 1). \tag{3.1}$$

For example, if a data cube includes 5 dimensions, and if each of these dimensions includes 3 hierarchical levels, the number of cuboids is equal to $4^5 = 2^{10} \approx 10^3$. It is clear that the full materialization of the cuboids for all the cubes associated with the fact tables of a data warehouse would impose storage requirements that could be hardly sustained over time, considering the rate at which new records are gathered.

For all of the above reasons, it is necessary to strike a balance between the need for fast access to information, which would suggest the full materialization of the cuboids, and the need to keep memory use within reasonable limits. As a consequence, preventive materialization should be carried out only for those cuboids that are most frequently accessed, while for the others the computation should be carried out on demand only when actual queries requesting the associated information are performed. This latter approach is referred to as *partial materialization* of the information relative to the data cubes.

3.4 Notes and readings

More in-depth studies of data warehouses, including technical aspects which have deliberately been omitted from this chapter, are provided by Kimball (1996), Kimball *et al*. (1998), Dyche (2000), Inmon (2002), Kimball and Ross (2002) and Nemati *et al*. (2002).