Managing the Evolution of Corporate Databases

BUTLER COX FOUNDATION

Research Report 64, June 1988

BUTLER COX
FOUNDATION

Managing the Evolution of Corporate Databases

Research Report 64, June 1988

Butler Cox & Partners Limited

LONDON
AMSTERDAM MUNICH NEW YORK PARIS

Published by Butler Cox & Partners Limited
Butler Cox House
12 Bloomsbury Square
London WC1A 2LL
England

Availability of reports
Members of the Butler Cox Foundation receive three copies of each report upon publication; additional copies and copies of earlier reports may be purchased by members from Butler Cox.

Photoset and printed in Great Britain by Flexiprint Ltd., Lancing, Sussex.

BUTLER COX FOUNDATION

Managing the Evolution of Corporate Databases

Research Report 64, June 1988

Contents

Management Summary

A Management Summary of this report has been published separately and distributed to all Foundation members. Additional copies of the Management Summary are available from Butler Cox

Chapter 1

The data-management approach

Corporate databases should not be implemented without a data-management approach to systems. In a previous Foundation Report on this subject (Report 32 — Data Management, published in November 1982) we described the data-management approach and its benefits. We defined data management as the activity of planning and controlling the use of data and we highlighted the need for a long-term strategy to exploit the use of data. In that report, we recommended a two-stage plan for introducing data management. The first-stage activities were preliminary work, designed to pave the way. In particular, we recommended that an appropriate data-oriented systems-development method should be introduced, together with a data-dictionary system. The method should be selected according to its ability to allow developers to design applications according to their use of data. The second-stage activities concerned the establishment of a data-administration function, led by a data administrator responsible for all aspects of the management of the organisation's data. The primary purpose of the function is to develop a high-level plan for organising and controlling the enterprise's computer-held data.

In our research for this report, we found that the use of data-oriented methods (the first stage of introducing data management) is now commonplace. Today, it is unusual if the systems development department in a major organisation does not produce, as a matter of course, a data (or entity) model as an important part of the application-development process. (Data modelling, and other terms used in this report, are defined in the glossary at the end of the report.) We also found that a high proportion of Foundation members (65 per cent) had reached the second stage of introducing data management — the appointment of a data administrator and the application of data management at a corporate level. Many of these organisations have made substantial progress with introducing the data-management approach and with establishing the associated databases, database management systems, and applications.

However, the progress to date has been based on first-generation database software products and tools, and many organisations are now planning to make major changes to their database systems over the next five years. In particular, they are planning (or considering) the move to second-generation data-management techniques and tools — relational databases, advanced data-dictionary systems, and distributed-database systems. The aim of these changes is to enable applications that better meet the needs of the business to be developed, to provide end users with easy access to databases, and to achieve better control over the use of data.

The main purpose of the data-management approach is to create a shared corporate data resource that is available for the benefit of the organisation as a whole. The data resource is implemented as one or more corporate databases that store data describing the organisation's resources, its products, and its dealings with customers, suppliers, and so forth. The data resource, with its corporate-wide definitions of the meaning of data items, provides the basis for a common understanding, by all users, of the data used in all the computer systems. In addition to unambiguous definitions of data, the data-management approach requires the use of software facilities for data management and for database access.

Several Foundation members are well advanced in their use of the data-management approach; two case histories are described briefly in Figures 1.1 and 1.2 overleaf. The second of these (British Gas North Western) has reached an advanced stage in the data-management approach and has, in effect, created a corporate database. In our research, however, we found that few organisations had reached such an advanced stage. Our research has shown that there are several reasons that make it expensive and difficult to adopt data management on a corporate-wide basis. The three main reasons are:

— Converting existing databases to new database technology is a major problem. There are significant costs involved, not only in the conversion itself, but in the disruption of the service provided to users.

Figure 1.1 Data management in Electricité de France

Electricité de France (EDF) is the national supplier of electricity and gas to the whole of France. Computer and database systems are used mainly for management information and reporting, customer billing, supply network maintenance, power generation, load dispatch, and distribution. EDF's distribution division has more than 100 data processing centres throughout France, grouped around 18 regional centres. Major hardware configurations are based on IBM and Bull equipment, and there is a wide range of medium to large systems.

The distribution division uses three types of database management systems — hierarchical systems for managing supplies, work-in-progress, and network management; SOCRATE-CLIO (supplied by SYSECA) used at local centres for billing large corporate customers; and relational databases — DB2 and Oracle — for general services such as the preparation of corporate-wide statistical analyses.

The databases are designed centrally using Merise, a data-oriented design method. Generic versions of the database are distributed to each of the remote data processing centres, together with a data dictionary that specifies the design of the database. This ensures that the same database design is used throughout the organisation, even though the data stored is specific to the locality. EDF organises its databases into six subject areas — customer service, personnel, distribution network support, physical resource management, accounting, and management support.

EDF believes that its data-management approach enables successful distributed applications to be developed and supported centrally, whilst also providing a degree of local autonomy.

Figure 1.2 Data management at British Gas North Western

British Gas North Western is responsible for the supply and distribution of gas in the north west of England. It runs a chain of high street shops, operates a fleet of 3,500 vehicles, and has a large spares inventory. A large central corporate database system (based on ICL mainframes) is linked to satellite databases in 16 regional offices.

British Gas North Western has used data-modelling methods to create a clear structure for all its shared data. Moreover, the structure has been designed to facilitate the use of distributed databases and to enable the interfaces between applications and the databases to be controlled. The data-administration function has been established for seven years, and the descriptions and names of all shared data are now controlled via the ICL Data Dictionary System (DDS). The data-administration function sets standards for the corporate database systems and approves the data models produced by application developers.

— The systems department often finds it difficult to justify corporate-wide data management because of the long-term nature of many of the benefits and the lack of understanding by senior management of the need to adopt a data-management approach.

— Much of the existing data-management software cannot support fully all of the functions implied by a corporate-wide data-management approach. The technology is new and complex, advances occur rapidly, and products are oversold.

THE KEY STAGES IN THE EVOLUTION OF CORPORATE DATABASES

Given the enormous investment in existing business applications, few organisations can afford to rebuild their database systems completely without taking account of their existing systems and databases. Instead, new database software and concepts will usually be introduced in an evolutionary way, based on a master plan for managing corporate data. The purpose of this report is to provide guidance for systems and business managers as they manage the evolution of their organisations' databases. Our research has shown the best way to proceed is to:

— Create a corporate data architecture.

— Establish the technical basis.

— Improve the facilities for user access to databases.

— Plan a cost-effective migration path.

— Extend the role of the data-administration function.

Each of these stages is described briefly below, and is then discussed in detail in the remaining chapters of the report.

CREATE A CORPORATE DATA ARCHITECTURE

The first stage in the evolution is to create the overall data architecture within which new databases and applications will be developed. The data architecture is part of the information systems plans of the organisation and it acts as a 'blueprint' for the data-management activities. Without a data architecture on which to base the development of new database systems, many of the benefits of the new data-management techniques and tools will be difficult to achieve. Indeed, the move to new data-management techniques and tools often provides a unique opportunity to develop a corporate data architecture.

A data architecture describes the logical structure of the data that is used by the organisation. Such an architecture is essential to the development of integrated and flexible corporate databases. The process of town planning provides a useful analogy to the process of developing a data architecture. The purpose of town planning is to place different parts of the town (factories, houses, parks, shopping areas, and so on) in separate areas, but with easy access routes between them. The overall plan provides a framework for developing the town's areas and the relationships between them, and ensures that the detailed plans for each area can be integrated into a consistent whole. A data architecture performs a similar purpose by setting the framework for the development of corporate databases and by providing the basis for the development and integration of individual applications and databases. The designers of a data architecture should note that its main purpose is to create corporate databases that will support business applications. A data architecture will not, by itself, identify the applications with the highest payoff.

In Chapter 2 we describe the elements of a data architecture in more detail and discuss several ways in which organisations have approached the task of developing such an architecture. The most comprehensive, but most expensive, approach is to design a data architecture in a top-down manner. There are, however, several less-expensive, and less-comprehensive, methods that can be used when the top-down approach is not appropriate.

ESTABLISH THE TECHNICAL BASIS

The data-management approach and corporate databases must be supported by powerful data-management products such as data dictionaries, relational databases, and distributed databases. The challenge is to select and use an integrated set of data-management systems that allow corporate databases to be created and managed and that provide improved user access to the databases.

A data-dictionary system is one of the most important data-management tools. A dictionary is used to document and control data and plays an important role in database planning and applications development. The other two types of data-management system are relational database management systems and distributed database management systems. Relational systems provide the flexibility to allow a variety of access paths to the databases, and distributed systems manage databases located in several different computer systems. We call these three types 'second-generation' data-management systems in this report because they provide new data-management capabilities.

Care must be taken in selecting data-management systems, timing their introduction, and choosing suitable applications. The concepts underlying the new generation of database products are very seductive, but the reality often falls short of the theory. In Chapter 3 we describe the promises and actual performance of these data-management systems and describe how they can best be used.

IMPROVE THE FACILITIES FOR USER ACCESS TO DATABASES

Our research showed that Foundation members consider user-driven database-access tools to be of increasing importance. If the full potential of second-generation data-management systems is to be exploited, organisations need to match these tools to the users' requirements and capabilities. In order to use data resources effectively, users need to be able to access the required data easily, to understand its meaning accurately, and to receive data in a form that meets their needs. We describe in Chapter 4 the problems that impede user access to corporate databases, and the new methods that can be used to overcome the problems.

PLAN A COST-EFFECTIVE MIGRATION PATH

Designing a new data architecture, and developing and implementing new databases, is an expensive task that can disrupt the work both of users and of the systems department. Over the next five years, most systems departments are likely to be involved in converting their existing database systems to second-generation data-management systems. The benefits of new corporate databases will be achieved sooner and the conversion costs will be minimised by carefully selecting and planning the migration path. In Chapter 5 we describe the major problems that will be encountered when converting databases to conform with a new data architecture and to use new data-management systems, and we discuss how to select a migration path.

EXTEND THE ROLE OF THE DATA-ADMINISTRATION FUNCTION

The evolution of corporate databases will not be trouble-free unless the data-administration function is sufficiently powerful and influential to control the development and use of corporate databases. The data administrator is the custodian of the corporate data architecture and resolves conflicts about the use of data. We expect the role of the data administrator to evolve to include additional responsibilities in data and information management. We describe the major new data-administration tasks in Chapter 6.

SCOPE OF THE RESEARCH CARRIED OUT

The conclusions and recommendations of this report are supported by an extensive programme of research carried out during the last quarter of 1987 and the first quarter of 1988. The research was led by Martin Langham, a senior consultant with Butler Cox in London who specialises in selecting and using databases. He was assisted by Tony Brewer, Director of the Foundation in the United Kingdom and the author of the previous Foundation report on data management, and by Simon Forge, a senior consultant in Butler Cox's Paris office.

The research began with a review of the extensive published material in the field of data management. (A bibliography is included at the end of the report.) The views and practices of Foundation members were gathered in face-to-face and telephone interviews, and by analysing the responses to a detailed questionnaire about present data-management activities and future plans. In all, 120 Foundation members replied to the questionaire. We also met with suppliers of database systems to establish the current capabilities of their products and their product plans. In addition, we sought the views of leading experts in the fields of relational database and data dictionaries, including Ted Codd (acknowledged as the founding father of relational database theory) and his partner, Chris Date. In the field of data dictionaries, we sought the views of David Gradwell, founder of Data Dictionary Systems Ltd. He has played a leading role in Europe in the development of data-dictionary systems and is the Rapporteur for the ISO (International Standards Organisation) committee for IRDS (Information Resource Dictionary Standard).

We also held focus groups with Foundation members in Belgium and the Netherlands to seek out their views on issues such as corporate data management, access to corporate databases, and the problems of migrating to a new data architecture and new databases.

Chapter 2

Creating a corporate data architecture

In the previous chapter we said that the first stage in the evolution of corporate databases is to create a corporate data architecture. In this chapter we describe the key elements of such an architecture and the top-down approach that must be used to develop a complete data architecture. However, creating a data architecture in this way is expensive and time-consuming, and many organisations will prefer to use a less-rigorous approach. We therefore discuss four alternative approaches, used successfully by several Foundation members, that can deliver some of the benefits of a data architecture, but at less expense.

Developing and maintaining a corporate data architecture is a challenging task and, at the outset, the benefits are not immediately obvious to business managers. We therefore also describe how the case for developing a data architecture can be made and how to plan and to carry out the exercise.

THE TOP-DOWN APPROACH TO BUILDING A THREE-LEVEL DATA ARCHITECTURE

The aim is to develop a data architecture that clearly models and supports the main business functions, that consists of easily managed subject-area data models, and that forms a framework for consistent descriptions of data. The most comprehensive and rigorous way of doing this is to investigate the important business processes of the organisation and to apply data-analysis techniques in a top-down manner. These techniques produce a model showing the data used by the organisation and the relationships between major data elements. Data analysis at the corporate level results in an enterprise model. This enterprise model forms the basis of the data architecture. (The concepts of data analysis and enterprise models are described in the glossary, and the bibliography lists a recommended text on data-analysis techniques.)

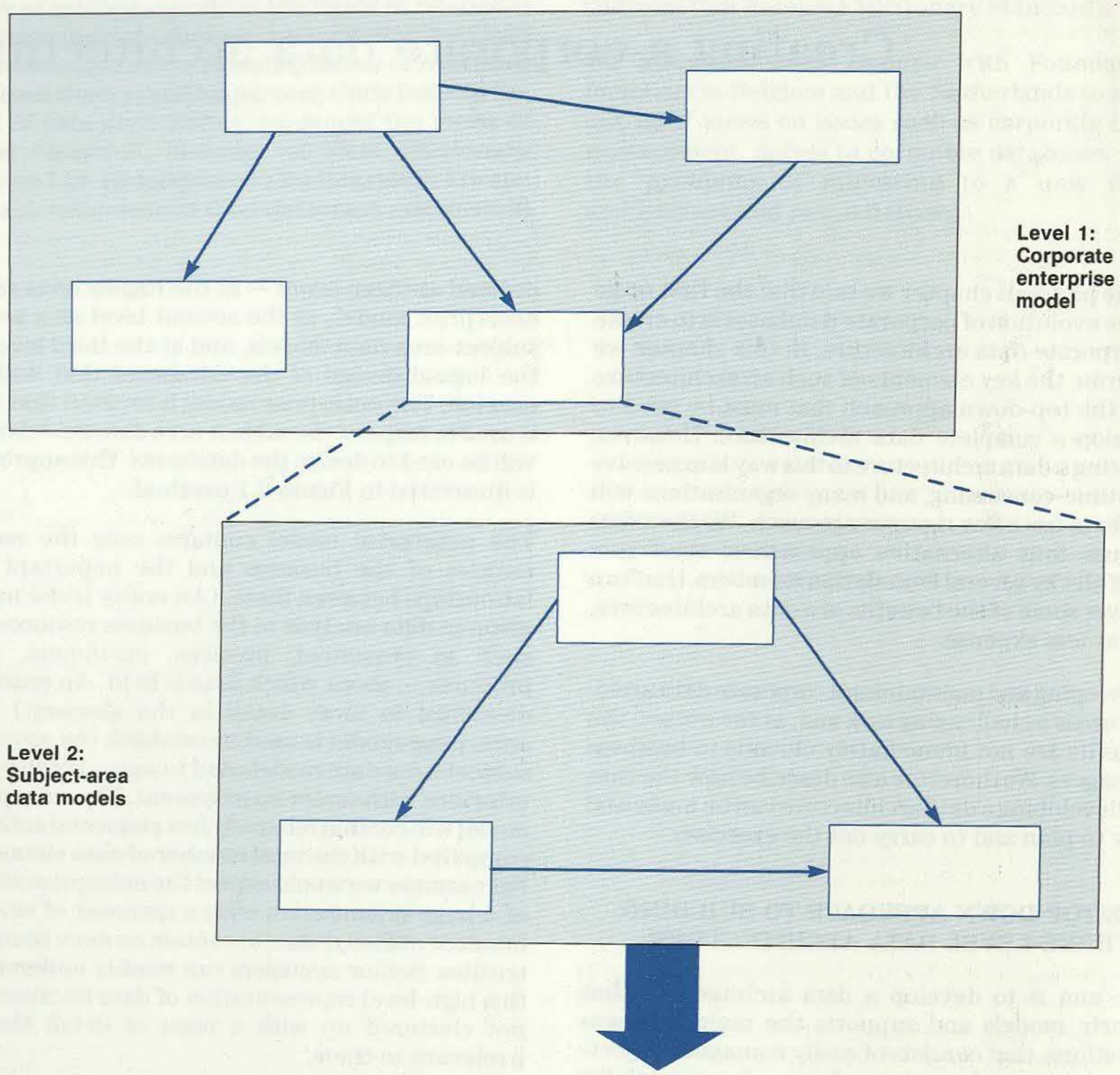
The production of an enterprise model is a demanding and time-consuming task that runs the risk of becoming overwhelmed by detail. One of the most important developments in the methods for developing corporate data architectures has been the recognition that a data architecture should be defined at three levels — at the higher level as an enterprise model, at the second level as a set of subject-area data models, and at the third level as the logical design of the databases that will be created. The enterprise model is created first and is used to map out the subject-area data models that will be used to design the databases. This approach is illustrated in Figure 2.1 overleaf.

The enterprise model contains only the major entities of the business and the important relationships between them. (An entity is the name given in data analysis to the business resources — such as personnel, invoices, customers, and products — about which data is held. An entity is described in more detail in the glossary.) The enterprise model is used to establish the scope of subject-area data models and to agree application priorities with senior management. The enterprise model will contain relatively few elements (entities) compared with the total number of data elements. For example we would expect the enterprise model of a large organisation with a turnover of several hundred million dollars to contain no more than 100 entities. Senior managers can readily understand this high-level representation of data because it is not cluttered up with a mass of detail that is irrelevant to them.

Any model of the business will become out of date as the business changes. However, experience has shown that an enterprise model remains valid for longer than other types of models because it is based on just the fundamental activities and resources of the organisation — products, warehouses, orders, invoices, and so on. These are not likely to change unless the whole basis of the business changes. This is in contrast to, for example, models of information flows, which change frequently as information needs change.

The second level subject-area data models are made up of sets of data that are associated with significant business activities (such as accounting) or with major business entities, such as products. The subject areas are chosen to be as self contained as possible so that the need to share data between subject-area data models is minimised. They contain more detailed descriptions of the data than the

Figure 2.1 Three-level data architecture



The boxes at level 1 represent major business entities in the corporate enterprise model, and the boxes at level 2 represent the data model of a specific subject area

enterprise model and are chosen to be of a suitable size to form manageable sets of implementation projects. By segmenting the enterprise model into subject-area data models, the resulting data architecture and databases are easier to implement and easier to control.

Subject-area data models differ from the application-oriented data models (and the resulting databases) that many organisations have created in the past. Each application database was built for a specific application, and data was often duplicated in several application databases. By basing applications on subject-area data models, which are derived from the corporate enterprise model, such duplication can be identified and reduced, leading to a more consistent definition and usage of data.

At the third level of the data architecture, the subject-area data models are further expanded and refined during a detailed study of each business area, and the detailed descriptions of the data items used in the data models are collected into a data dictionary and rationalised. It is at this stage that the logical design of the databases is carried out. An effective data-administration function is thus a vital prerequisite for the successful execution of this stage.

The enterprise model, together with the set of subject-area data models corresponding to important business activities, and the resulting logical database designs, defines the corporate data architecture that will be used as the framework for developing and implementing corporate databases. The final stage of implementing the data architecture is to map the logical database designs to a particular hardware and software environment.

The approach of creating a three-level data architecture before implementing any corporate databases should be contrasted with the approach commonly used today. At present, data analysis is carried out as subject-area applications and databases are logically designed. This means that existing databases have, by and large, been developed independently of each other and do not conform to an overall corporate data architecture. By extending data-analysis techniques to the corporate level to construct an enterprise model and subsidiary subject-area data models, it is possible to ensure that there are no inconsistencies in the resulting corporate databases.

Figure 2.2 describes how one large retailing company used the top-down approach to develop a corporate data architecture.

OTHER APPROACHES TO CREATING A DATA ARCHITECTURE

For many organisations, top-down development of a data architecture is not appropriate. Typical difficulties are:

— Top-down development and implementation of a data architecture is expensive and requires a considerable amount of senior management time and effort.

— The benefits of a data architecture for the whole organisation are achieved slowly and the organisation cannot wait for new business applications.

— The organisation may not have (or be able to spare) the high level of skill required to develop a data architecture. Even if outside consultants are used, in-house expertise is still needed to maintain and enhance the data architecture after it has been created.

Because of these difficulties, many organisations have not used the top-down approach to developing a data architecture. Instead, they have used pragmatic approaches that address their specific needs. For example, a paper published by the Center for Information Systems Research at MIT's Sloan School of Management reported that, for the reasons described above, few organisations had used a top-down approach. (The paper is based on an in-depth

Figure 2.2 B&Q used a top-down approach to create a corporate data architecture

B&Q is the largest do-it-yourself retail group in Europe. It has 220 stores, employs 10,000 staff and has a turnover in excess of £500 million. The group is expanding rapidly — opening a new store every two weeks.

There are 130 staff in the systems department, organised in four sections:

— The business-systems section, responsible for the information systems strategy and the applications portfolio. The business-systems account managers have full responsibility for ensuring that the systems needs of the company's business divisions are met.

— The development section, which builds the applications specified by the account managers.

— The technical-strategy section, which defines common requirements such as the network architecture, the hardware and software strategy, and the data architecture. The data administrator, who is responsible for the data architecture, is a member of this section, as is the database administrator, who is responsible for the physical implementation of the databases.

— The computer-services section, which runs the computers and communications network.

B&Q has developed a three-level corporate data architecture consisting of a corporate data model (the enterprise model), subject-area data models, and project data models. The company describes the corporate data model as the high-level model of the whole organisation's use of data. This high-level model includes about 50 of the major business entities such as products and suppliers — less important entities are excluded from the model at this level. An entity is included in the corporate model only if it is used in more than one subject area, and no attributes of the entities are included. (Thus, the corporate model does not specify that a customer record will always contain the customer's name and address, for example.) B&Q uses the corporate data model to provide an overview of the company's data for senior executives and to control the lower levels of the data architecture. The corporate data model also forms the basis of B&Q's plans for a management information database.

The subject-area data models cover specific subject areas of the company and typically map onto the organisation structure. Examples of B&Q's subject areas are personnel, property, and buying and merchandising. The starting point for developing a subject-area data model is a segment of the corporate data model. This is then enhanced and refined by further analysis of the subject area. At all times, the subject-area and the corporate data models are kept consistent. Changes resulting from refinements at the subject level are reflected in the high-level model, and changes in the high-level model lead to changes in the subject-area data models.

The third and lowest level of the data architecture is the project data models. A project data model comprises all or part of a subject-area model, and forms a working data model used by the development section. Project data models are checked by the data-administration function for quality and consistency.

B&Q believes that its multilevel corporate data architecture is essential to ensure that database applications are aligned to the business strategy and that the development of corporate databases can be controlled without the data administrator becoming a bottleneck in the development process. The company also believes that it is essential to use software tools such as a dictionary and diagramming aids. Without such tools, it is very difficult to control changes in the architecture and to communicate the results to all parties.

survey of 20 large US corporations carried out during 1985 and 1986.)

We now describe four alternative approaches to the top-down approach to developing a data architecture — basing the architecture on a high-level business analysis, focusing the architecture development on particular business objectives, basing the architecture on a 'packaged' data model, and using a bottom-up approach. Three of these have been used successfully by organisations reported in the US study or interviewed in our own research. The packaged-base approach is relatively new and has been used by a few organisations in specific business sectors. Each of these approaches provides fewer benefits than an architecture developed with the top-down approach, but costs less. One of these approaches may be more appropriate than the top-down approach in particular circumstances. Figure 2.3 shows when each of them is applicable.

HIGH-LEVEL BUSINESS ANALYSIS

As in many areas of planning, the first 20 per cent of the planning effort often yields 80 per cent of the results. By restricting the development of a data architecture to a rapid high-level analysis of the business, taking one or two months at most, a reasonably sound data architecture can be produced quickly. The results of such an exercise are used to identify and define several subject-area data models that are critical to the organisation's applications requirements.

This approach requires those developing the architecture to have considerable business knowledge and data-analysis expertise. There is also a danger that inconsistencies will later be found in the architecture.

Figure 2.3 Alternatives to the top-down approach to developing a data architecture

Circumstances when applicable	Approach: High-level business analysis	Focused	Packaged based	Bottom up
Considerable change expected or required			✓	
Commitment level is low				✓
Need to change is low		✓		
Cost is a major factor	✓	✓		✓
Business and data modelling skills in short supply			✓	

We found that several organisations had initiated their development of a complete data architecture in this way. One of these was British Gas North Western, whose experiences were described in Figure 1.2.

FOCUSED APPROACH

Some organisations have urgent and specific business needs. For example, a holding company might be concerned primarily with particular needs such as financial-reporting procedures and personnel management. Other organisations may wish to support particular business functions, such as a production and delivery system, or to create a database to meet a specific information need.

With the focused approach, the development of the data architecture is concentrated exclusively on these business functions, and a data model is created to coordinate the development of a family of applications. This approach is suitable when it is necessary to demonstrate an early success in the application of data-management techniques, or when cost and time severely constrain the resources available.

However, this method of developing a data architecture can only produce part of the enterprise model because vital information needs may well be omitted from the scope of the study.

PACKAGED-BASED APPROACH

In some industries, such as banking and finance, sophisticated software packages are available to handle many of the functions of the organisation. Packages such as these are, of necessity, very flexible and sometimes have their own built-in (and documented) data models. Some database software suppliers (Oracle, for example) also supply packaged data models that can be tailored to meet the needs of a particular business sector. For example, in our research we talked with two financial-services companies that were using this approach to develop a new data architecture (although each was using a different package).

By using a package-based approach, the organisation benefits from the experience of the package designer and obtains a documented data model. This approach is particularly useful when the need for change is urgent and when business and data-analysis skills are in short supply. It also means that the data architecture can be implemented much earlier than with other approaches.

Sometimes it is possible to base the high-level enterprise model on models for similar organisations, thereby speeding up the development of the data architecture. For example, the enterprise models of local-government authorities will have many common characteristics. An expert data analyst can quickly develop an initial enterprise model of these organisations, which can serve as a guide to the development of subject-area data models.

Although the package-based approach is relatively new and untried, we believe that it may well be applicable in a range of industries. We suggest that when Foundation members embark on a data-architecture exercise they should, as a matter of course, investigate whether a package-base approach could be used.

BOTTOM-UP APPROACH

Sometimes, organisations are too large and/or too decentralised for any of the above methods to be practical. The development of a high-level enterprise model is unlikely to be supported by all parts of such an organisation. In this situation, an alternative approach is to set standards for describing the data and gradually to integrate individual data models as the application teams develop them. This approach means that conflicts about data sharing will have to be resolved on a day-to-day basis, which means that the data architecture may have to be compromised. It will also be necessary to ensure that application developers use a dictionary of shared data items.

The advantages of such a bottom-up approach are that the lengthy process of storing descriptions in the data dictionary is started immediately and that duplicate and inconsistent data is no longer created. The disadvantages are that inconsistencies in data shared by existing applications will not be resolved and it is unlikely that a set of subject-area data models will be created. This approach is often used by large, diverse organisations with many operating companies. In such organisations, there is usually limited senior management commitment to a corporate-wide data architecture, or the decentralised philosophy mitigates against a corporate data architecture.

MAKING THE CASE FOR DEVELOPING A DATA ARCHITECTURE

The systems department has to overcome two major obstacles when it sets out to develop a data architecture for the organisation. First, it has to persuade senior management (both within and without the systems department) that the benefits of an architecture will repay the investment in effort required by the systems department and by user departments to develop it. Although the effort is not large compared with major system-development projects, many organisations find it difficult to justify expenditure that cannot be related to specific projects or business areas and that will not produce benefits until some time in the future. Second, it has to use an approach that results in a sound architecture and that the user departments can understand and support.

Often, the need for data management is not understood by senior managers and, as a result, they may raise objections to the development of a corporate data architecture. One objection is that it is not possible to plan the use of data because the resulting data model would not be able to reflect the high rate of change experienced by the organisation. In practice, the high-level enterprise model describes the fundamental aspects of the business, and these tend to remain the same. It is the information requirements of business applications that change rapidly as different aspects of the business become more or less important.

A second objection to developing a data architecture is that the organisation cannot afford the resulting costs and disruption. Instead, new data-management systems are introduced without a data architecture having been developed first. Although this approach does, initially, avoid the cost of developing a data architecture, we believe that it is shortsighted. Without a data architecture, the full benefits of new data-management systems will not be obtained. In the longer term, the organisation is likely to have to spend much more on rationalising its databases and systems than it would have done by investing in a data-architecture project at the outset. The introduction of new data-management systems provides a unique opportunity to make the case for building a new data architecture.

Sometimes, the systems department is unfamiliar with data-analysis and modelling methods and, as a result, the benefits that would derive from developing a corporate data architecture are not identified. In one organisation, the task of redeveloping the databases became so urgent that it was taken out of the hands of the systems department and the responsibility was given to another manager outside the systems department. (The individual concerned had previously worked in the systems department, but was now assigned as a 'troubleshooter', reporting to the chief executive.)

One of the strongest arguments for overcoming these objections is to point out that many of the difficulties of evolving to second-generation data management are caused *because* first-generation databases were not developed in accordance with a sound data architecture.

Implementing the data-management approach is a long-term process and it is important to manage the expectations of the organisation's senior managers so that the project to develop a data architecture retains its momentum. If they do not appreciate the nature of the task they will become disappointed at the apparent lack of progress being made and will question the need for the project to continue. It is important to ensure that senior management understands that major benefits are derived from data management only once a critical mass of data is held in a data dictionary, and several subject-area databases, conforming to the overall corporate data architecture, have been developed. However, Foundation members that were in the process of implementing a corporate data architecture have identified two benefits that result almost immediately from adopting a corporate data-management approach:

— It makes it easier for users to find relevant data. By establishing an inventory of data, based on a data dictionary, the data administrator of one organisation had been able to reduce the time taken by users to locate relevant data from weeks to hours.

— It reduces the time required to design new applications. A major manufacturer is developing a corporate data architecture for its European operations, and provides a 'starter' data model to business analysts whenever a new application is initiated. This model is derived from a European divisional data model (equivalent to the enterprise model) and contains definitions of many of the items of data required by the application. The company claims that this approach can reduce the time required to design an application by as much as 50 per cent because it removes much of the requirements-analysis effort previously needed to identify and describe the data used by the application.

The longer-term benefits of establishing a well-managed corporate data architecture are more difficult to quantify and more diffuse in their impact, but, ultimately, they are more significant. In particular, a data architecture makes it easier to integrate systems across several different departments of the organisation and to build applications that can be used to access the databases easily and that can be changed easily when the data structure is amended. Two examples of ensuing practical business benefits are described below:

— A large manufacturing company found that, in order to be able to compete successfully, it was essential to integrate information from the whole of its product cycle (sales, design, production, and delivery). The development of an organisation-wide data architecture was the key to this integration, even though it was costly to implement.

— Another organisation (a conglomerate operating primarily in the leisure, hotel, and brewing industries) adopted a corporate-wide data architecture because it was continually subjected to organisational change as parts of the organisation were acquired or sold-off, and because it was faced with even greater changes in the future due to new legislation. The existing applications could not cope with changes on this scale because of inconsistencies in their data descriptions. Data analysis was used to produce a data architecture that ensured there were consistent data definitions across the organisation and that could provide the flexibility to cope with the changes.

If data is to be managed as a shared corporate resource, then key managers need to understand why data is important and why they need to view the corporate databases as essential ingredients of their business success. Where different aspects of business processes are carried out by many departments, then staff in those departments need to understand each other's data and, at least in outline, the activities that create and maintain the data. To develop this common understanding, management and staff therefore need to understand the parts of the corporate enterprise model that affect their work.

A data architecture can also improve the availability of information for management and decision-support purposes and can lead to improved application-development productivity.

PLANNING AND EXECUTING A DATA-ARCHITECTURE PROJECT

The development of a corporate data architecture is a very difficult task to carry out successfully. Data-architecture projects have frequently failed because basic common-sense rules have been ignored in setting up the project and in selecting the methods used to carry out the project. Some organisations have avoided these difficulties by ensuring that the project steering committee includes senior business management representatives. This arrangement has two main benefits:

— It gives the project a sufficiently high profile to ensure that all parts of the organisation cooperate fully with the team members.

— It enables the project team to resolve problems concerning changes to the organisational structure and scope, and the ownership and definitions of data. If the team is unable to resolve these problems, the success of the project will be jeopardised.

A key task in planning the development of a data architecture is to define the scope of the project. The scope may cover the whole organisation, a company in the organisation, or a business unit. In defining the scope, it is necessary to balance the potential for data sharing against the difficulties and expense of managing a larger project.

Senior user management should be involved closely with the project. Unless senior user managers are involved, the resulting data architecture will be perceived as a technical requirement that is irrelevant to the organisation as a whole. Data-analysis skills are vital to the success of the study but it is important that these are provided by staff with significant business knowledge, and who are not seen as data processing technicians.

User departments should also be involved in the data-architecture project — helping the analysts to understand their data requirements and helping to make the design decisions during the development of the enterprise model. In this way, user departments will gain a valuable understanding of the data architecture as it is developed and will feel that they 'own' their part of the architecture.

The process of developing a data architecture for a large organisation will generate a lot of information. In order to avoid being overwhelmed by paper, it is sensible to use a suitable analysts' workbench software tool and/or a data-dictionary system. Using aids such as these has the additional advantage of generating a data dictionary that can then be used as the starting point for the development of database applications.

Finally, many data architectures are ignored after they have been completed, and the expense of developing them is wasted. Implementing a data architecture once it has been defined requires considerable commitment. This commitment is easier to gain if there is a powerful champion who is able to see the implementation of the architecture through to completion.

Chapter 3

Establishing the technical basis for corporate databases

In Chapter 1 we said that three new types of data-management system — data dictionaries, relational databases, and distributed databases — were necessary to provide the basis for the construction of corporate databases. (We refer to these as second-generation data-management systems.) The new systems are required because the new corporate databases developed from the corporate data architecture need to be managed by more powerful and flexible facilities than were available with previous database implementations.

Data-dictionary systems have been available in various forms for more than a decade, but it is only now that they are acquiring the capabilities to allow them to come into widespread use. We expect that a data-dictionary system will become one of the most important elements of corporate data-management systems.

Relational database management systems, on the other hand, are less mature, but their capability is advancing rapidly as they are being enthusiastically taken up by corporate systems departments. Relational databases are now well established in decision-support applications, and we expect that, by the end of the 1980s, major operational applications will begin to make use of relational databases. Relational databases will provide the means of structuring and accessing data in the flexible way that will be needed if the use of corporate databases is to evolve with the organisation's information needs.

Distributed database management systems are the least mature type of data-management system, both in the capabilities of the products that are available and in organisations' understanding of their capabilities and potential applications. Most of the theoretical problems associated with distributed databases have now been solved, but the widespread implementation of distributed databases is still several years away. Nonetheless, there are now dozens of successful implementations of partially distributed databases, and, as we discuss later, distributed database management systems will play an important role in supporting the corporate databases of the future.

In the remainder of this chapter we discuss the underlying concepts of the three types of data-management systems and assess their current and potential capabilities in detail.

DATA-DICTIONARY SYSTEMS

In her book on data analysis (which is listed in the bibliography), Rosemary Rock-Evans writes that the data dictionary is ''Management services' own database and has such potential as a provider of information that one can scarcely be in danger of overselling it.'' Our research confirmed that systems departments are indeed beginning to recognise the data dictionary as a vital facility that can support many of the activities of the department. In our view, systems departments that do not have access to such a facility are, in effect, 'flying blind'. Data dictionaries will also play an increasingly important role in providing users with direct access to databases. This latter point is discussed further in the next chapter.

A 'critical mass' of data descriptions must be entered into the data dictionary before the benefits of corporate databases can be fully realised. Unfortunately, however, it often takes several years to load all data descriptions into a data dictionary. Many Foundation members confirmed that creating a data dictionary is a lengthy and laborious task that can only be progressed as new applications are developed. If the long-term benefits of data management are to be realised, it is therefore vital that data descriptions from new applications are documented in the data dictionary. In order to exploit the potential of data-dictionary systems, it is important to understand their characteristics and their potential.

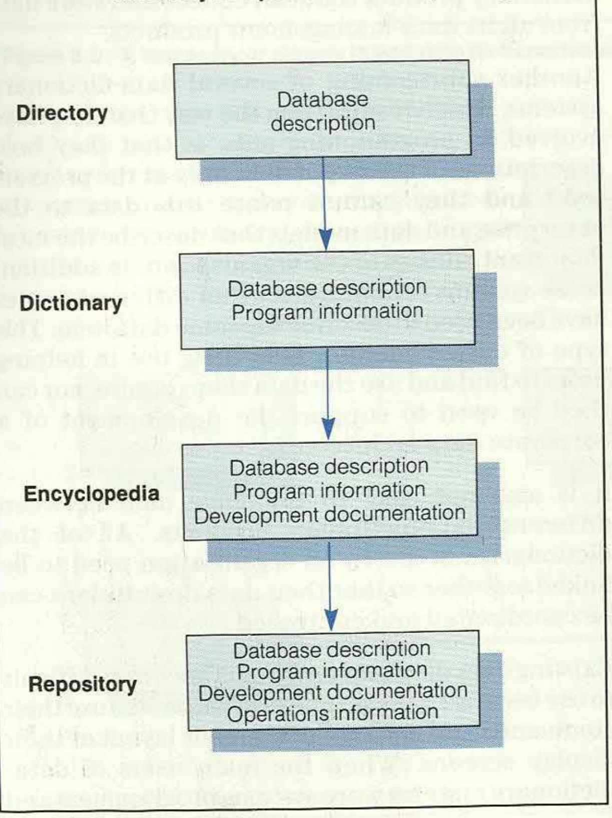
THE VARIOUS TYPES OF DATA DICTIONARY

Although data-dictionary systems have been in use for more than a decade, they have had less attention paid to them than other types of data-management system. Both suppliers and user organisations have devoted insufficient resources to them to enable them to reach their full potential. However, they are now evolving into very extensive systems for controlling the resources of an organisation. Figure 3.1 shows how they are developing in terms of the information that they control and their use.

The most basic form of data dictionary is a *data directory*. A directory holds descriptions of the data used by the database management system, and it is used solely by the database administrator to manage the database. The next stage in the evolution is to use the system as a *data dictionary* that also contains information about the applications that use the data (programs, screen layouts, report layouts, and so on). Recording this information in a data dictionary allows the effects of changes to applications or data to be assessed easily. DEC's Common Data Dictionary (CDD) product is typical of such data dictionaries. This product provides a definition language that is used to describe the entries in the dictionary and allows the information stored in the dictionary to be shared by several Vax data-management systems.

The third stage in the evolution of data dictionary systems is to use them as an *encyclopaedia* that holds business-analysis data as well as applications and database information. The information held in a data encyclopaedia provides support for all stages of the system development life cycle, from the initial analysis to applications testing and maintenance. ICL has been a leader in developing this type of dictionary system with its Data Dictionary System (DDS). Several Foundation members told us that DDS is a vital element of their application-development environment.

Figure 3.1 Evolution of data dictionaries from a simple directory to a powerful data repository



Other data-encyclopaedia products are emerging from relational database suppliers such as Oracle and from the suppliers of integrated Case (computer-aided systems engineering) tools, whose products provide data-dictionary facilities as a matter of course. Such tools are used throughout the life cycle of a system and can be used to create the data architecture and the resulting data models and databases. These facilities are usually based on a relational database management system. (There is still a great deal of confusion about what Case is and is not. The forthcoming Foundation Report on the topic of Case will clarify the situation.)

The final stage in the evolution of data dictionaries is to use them as data *repositories*. A data repository contains information about data and applications and supports the whole of the development life cycle, together with information about some operations activities such as change management, operating-systems management, and network management. A data repository is thus a large and complex database that collects data from several data-management systems and from other software systems. No data-repository products are available at present, although several suppliers (including IBM) plan to introduce this type of product by the beginning of 1990. Others (ICL and Software AG, for example) are enhancing their existing data-dictionary products to enable them to be used as data repositories. Figure 3.2 shows the content and use of a data repository.

Data dictionaries allow data-management systems and data-management tools to be linked together to form an integrated set of products with common

Figure 3.2 Contents and use of a data repository

Contents
- Enterprise model
- Subject-area data models
- Data item descriptions
- Data integrity and security rules
- Logical database designs
- Physical database designs
- Application specifications
- Hardware and software inventory

Use
- Business analysis
- System development
- Data architecture development
- Operating system management
- Database operation
- Change management
- Data security and privacy
- Distributed data management
- User access to databases
- Database extracts
- Decision support

data descriptions. An integrated set of data-management systems and tools simplifies the task of developing corporate databases. Dictionaries are already important tools for data administration and for applications development. They are also beginning to be used to help users access corporate databases, and for applications conversion.

Data administration
A data encyclopaedia is often used during the development of a data architecture to support the development of a high-level enterprise model. During our research we encountered one organisation where the team developing the enterprise model had installed an ICL computer mainly to use the ICL Data Dictionary System. Such encyclopaedias also provide an essential link between the high-level enterprise model and the more detailed descriptions of the subject-area data models required for applications development.

Applications development
Few applications developers would contemplate developing major new applications without the use of a dictionary to ensure that data items are consistently named and used. Experience shows that dictionaries save considerable amounts of application-development effort. Dictionaries are now the central component of Case tools that assist developers to design applications. In a Case tool, the dictionary supports a complete development environment, with the design documents of each development stage being stored in the dictionary, and screen and report definitions being generated from data stored in the dictionary. Case tools can even use the dictionary to generate test data.

User access
At present, the use of data dictionaries is mainly confined to the systems department, although they are increasingly being used in a user environment for prototyping, as decision-support tools (in Focus from Information Builders, for example) and as part of natural-language interfaces. In each of these areas, data dictionaries allow the meaning of the data items to be described in terms that will be understood by users and enable users to understand the 'language' of the databases.

An interesting example of the use of a data dictionary in a user environment is provided by an insurance organisation that uses the ICL DDS data dictionary as a development tool. The user representatives assigned to development projects were able to access the applications design, which was held in the dictionary. Using the dictionary in this way allowed the user representatives to assess and comment on the design as it was generated by the analysts. The result was much better participation by the users in the development process.

Applications conversion
A data dictionary that contains a well-documented description of an existing application can often allow the application to be redeveloped at a much lower cost than would otherwise be the case. In theory, after an application has been completely defined in a data encyclopaedia an advanced Case tool could generate the actual programs and database automatically. This approach provides a low-cost method of converting applications to new computer systems and databases. Thus, an organisation can use a data dictionary to document its existing applications over a period of time, and then transfer the applications to a new computing environment.

LIMITATIONS OF DATA DICTIONARIES

Although data dictionaries have been available for several years, they still have some serious shortcomings. One of the main difficulties is that many existing dictionary products hold only a subset of the total data used by the systems department, which means that it is difficult to manage and coordinate changes in the programs that access the databases.

Another problem is that the scope of a dictionary is often too narrow because it does not collect data from all the data-management systems and tools that are likely to be used in an organisation. IBM, for example, provides several separate dictionary products for IMS, for the CSP application-generator product, and for DB2. It does not yet provide a single dictionary product that can collect and store data from all its data-management products.

Another shortcoming of several data-dictionary systems, which results from the way that they have evolved as programming aids, is that they hold descriptions of the use of data only at the program level and they cannot relate this data to the enterprise and data models that describe the most important entities of the organisation. In addition, these systems cannot detect that different names have been used to describe the same data item. This type of data dictionary is of little use in helping users to find and use the data they require, nor can they be used to support the development of a corporate data architecture.

It is also not easy to exchange data between different data-dictionary products. All of the dictionaries in use in an organisation need to be linked together so that their data descriptions can be coordinated and controlled.

Existing data dictionary products are often difficult to use because it takes time to learn how to use their commands and because of the poor layout of their display screens. When the main users of data-dictionary systems were systems programmers and database designers, a poor user interface was less of a problem. Now that a wider range of people need access to data dictionaries, ease-of-access is becoming an important factor. The popularity of Case tools that use graphics-based interfaces to their encyclopaedias illustrates the value of easy-to-use interfaces.

PLANNING FOR THE INTRODUCTION OF A DATA DICTIONARY

A data dictionary is the central element of data management, and organisations need to plan carefully for the introduction and development of this important resource. We expect data dictionaries to continue to evolve as they are used to manage more of the data and resources of the systems department. Particular aspects that need to be taken into account in developing a data-dictionary plan are the selection of a suitable data-dictionary system, the need for active and passive data dictionaries, the introduction of international data-dictionary standards, and the increasing use of data dictionaries for operational support.

Selecting a data-dictionary system

When introducing a new data dictionary, its selection should be given as much attention as the selection of a database management system or systems-development method. The product chosen should ideally overcome the limitations of data dictionaries discussed earlier — in particular:

- It should interface with all the other data-management products and tools used by the organisation.
- It should be easy to use by non-experts.
- It should be flexible so that it can be tailored to the unique requirements of the organisation.
- It should be capable of exchanging data with other data dictionaries.

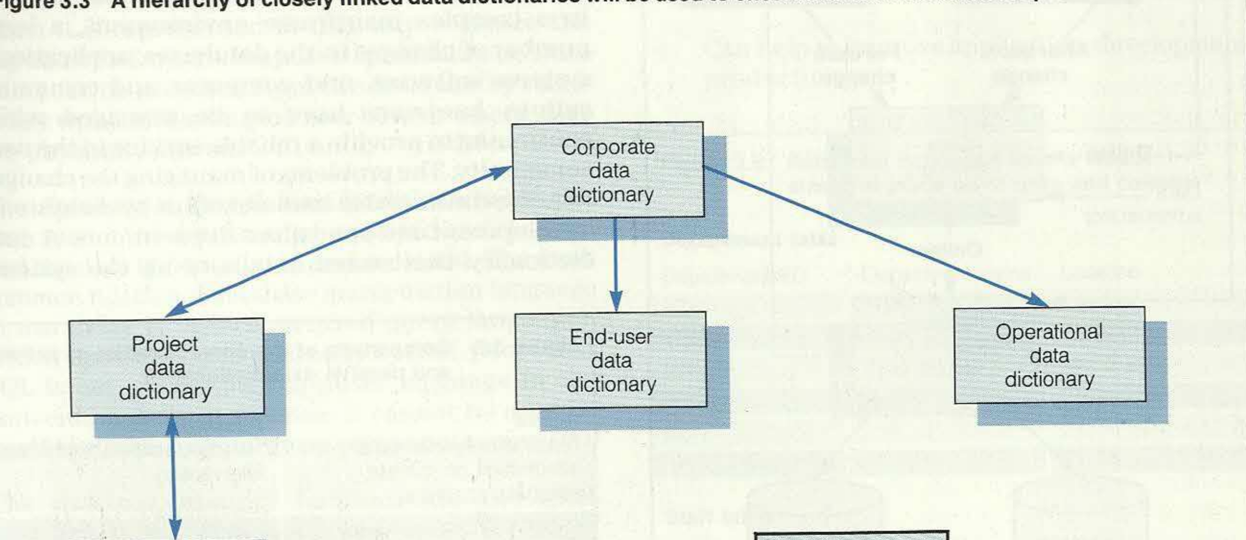
The last of these requirements is necessary because different data dictionaries will be used to meet different needs. For example, two major database management systems suppliers (Oracle and Relational Technology Inc, the supplier of Ingres) provide separate dictionaries for controlling the operational database and for supporting applications development. Furthermore, Case tools and other products such as Intellect and Focus all have their own data dictionaries, and separate data dictionaries are often used with decision-support databases. Each of these data dictionaries needs to be linked to a master dictionary that records the complete resources of the systems department. A typical hierarchy of dictionaries is shown in Figure 3.3.

Some data-dictionary suppliers have built interfaces between certain dictionaries. MSP's Data Manager, for example, can exchange data with Cullinet's IDMS/R and IDD and with IBM's DB2 and SQL/DS.

The need for active and passive dictionaries

The different types of data dictionaries in use in an organisation will be used either as active or passive dictionaries. An active data dictionary is updated automatically by data-management systems and

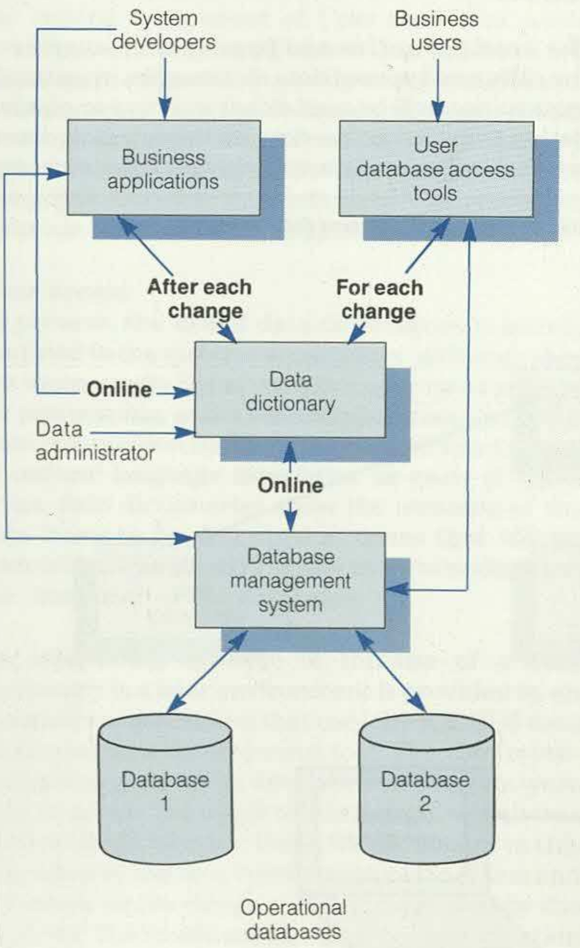
Figure 3.3 A hierarchy of closely linked data dictionaries will be used to ensure consistent data descriptions



application programs whenever the data stored in the dictionary is changed. Thus, an active dictionary would monitor user queries and immediately include any changes made to the database structure — a new user view, for example. Figure 3.4 shows the way in which an active data dictionary is used. A passive data dictionary, on the other hand, is not updated as the database and applications are changed. Instead, it has to be updated separately by the data administrator and application developers and by transferring data to it from other data dictionaries.

An active data dictionary that is fully integrated with the data-management systems and the associated data-access tools provides a completely accurate picture of the current state of the corporate databases. However, only data dictionaries that are supplied with a particular data-management system can provide this level of integration, which means that the scope of the data dictionary has to be limited when several data-management systems are in use. On the other hand, a passive data dictionary can be used to hold descriptions from several data-management systems, which means that the scope of the dictionary can be wider, although the data will not be updated in realtime. The advantages and disadvantages of active and passive data dictionaries are compared in Figure 3.5. The hierarchy of data dictionaries illustrated in Figure 3.3 would consist of several active, specialised data dictionaries that transfer data, under the control of the data administrator, to a passive, master data dictionary.

Figure 3.4 An active data dictionary is updated automatically as changes to the databases are made by the data administrator, system developers, and business users



International standards for data dictionaries
In the area of international standards for data dictionaries, a common data-dictionary standard, known as the Information Resource Dictionary Standard (IRDS), is being developed by the ISO and ANSI standards organisations. The aim of this standard is to make it easier to transfer data between data dictionaries. IRDS is based on the relational database model, and defines the structure of a standard data dictionary, and a standard computer interface for data-management systems used to access and manipulate data in the dictionary. It is being developed as an 'open' standard so that data can also be exchanged with many other types of systems such as language compilers, Case tools, application generators, database management systems, and applications themselves. The concept of IRDS is of great interest to suppliers and users of data-management software and is being progressed, although a standard is unlikely to be agreed before 1990.

Using data dictionaries for operational support
In a complex mainframe environment, a large number of changes to the databases, applications, systems software, and computer and communications hardware have to be managed while continuing to provide a reliable service to the user community. The problems of managing the changes requires considerable time and effort by the systems development and operations departments. A data dictionary that stores details of all the systems

Figure 3.5 Comparison of the characteristics of active and passive data dictionaries

Active	Passive
Automatically maintains information on current database contents	Needs considerable manual intervention
A single dictionary may form a bottleneck in the development process	Can be used as a migration tool
Integrated with a single database management system	Can be used with several database management systems
Operational systems depend on the particular data-dictionary system	Operational systems independent of the data-dictionary system

resources (hardware, software, operating-system parameters, user identifiers, and so on) will be a great help in reducing the problems to a manageable size. Software AG is one of several suppliers of data-management software that plans to extend its dictionary system to manage this type of data. Its Predict data-dictionary system will document and control the system configuration for the IBM S/370 architecture and will store data about the performance of system components and resources, such as IBM's VTAM software.

RELATIONAL DATABASE MANAGEMENT SYSTEMS

Relational database management systems are based on a rigorous mathematical theory developed in the early 1970s by Ted Codd whilst he was working for IBM. Translating the theory into usable products has taken a long time, both because of limitations of technology and because of scepticism about the practicality of relational theory. However, today there are more than 50 relational database products, and tens of millions of dollars are being spent on implementing relational databases.

A relational database has three essential elements — the logical database structure, the database-manipulation facilities, and the database-integrity facilities. The structure of a relational database is made up of tables containing rows and columns as illustrated in Figure 3.6. Each row contains the data (attributes) of a particular entity, and each column contains similar data items — staff salaries, for example. People find it easy to understand data when it is represented in this way — hence the popularity of microcomputer spreadsheets. Data items can be accessed from the database by name or key without needing to know how, or where, they are physically stored.

The database-manipulation facilities enable the structure of the database to be created and changed and the data to be updated and accessed. The most common relational database manipulation language in use today is SQL (structured query language), which was developed originally by IBM. (Note that SQL is not, in fact, a computer language in the conventional sense, because it cannot be used by itself to develop application programs.)

The database-integrity facilities are the most critical and the least-well implemented aspect of relational databases. They ensure that the rules the database designer included in the data model are followed by the database users — for example, ensuring that, in a personnel application, a department of the organisation cannot be removed from the database until all the staff in that department have been transferred to another department. These facilities are the most difficult part of the relational concept to implement because they have an adverse affect on performance. Hence, they are missing from many of today's relational database products.

We now review the benefits of relational databases compared with first-generation databases and provide guidelines for selecting and using relational products. We also emphasise that a move to relational technology should be made with caution because the use of the technology is still relatively immature.

BENEFITS OF RELATIONAL DATABASES

Many observers of the IT industry claim that relational databases merely provide the facilities that suppliers claimed were available with first-generation database systems. Thus, the difference between a relational database management system and database management systems such as IMS and IDMS is more one of degree than of type. Nevertheless, relational databases do provide substantial benefits compared with earlier systems. In particular, relational database management systems:

— Enable a subject-area data model to be implemented clearly and accurately.

— Provide an easier method for changing the structure of databases and the ways in which they are accessed.

— Enable users to access databases directly and easily.

— Can help to improve application-development productivity.

Figure 3.6 Relational databases have a tabular structure made up of rows and columns

Department table

Department I/D	Department name	Location
001	Accounts	London
002	Sales	Birmingham
003	Production	Manchester
004	Research	Sheffield
etc.		

Staff table

Staff code	Department	Surname	First name	Date of birth	Salary
123	001	Adams	John	10/10/51	$30,000
124	004	Burns	Steven	5/5/55	$36,000
125	001	Cook	Alan	8/8/48	$24,000
etc.					

Clear implementation of a subject-area data model
First-generation data-management systems often could not be used to implement the complete logical structure of a data model because doing so would have led to excessive complexity and operational inefficiency. The table structure of relational databases makes it almost impossible not to implement the complete logical structure.

The benefits of implementing the complete logical structure are that it makes it easy for users to understand the database, the database can easily be related to the overall data model (particularly the enterprise model), and it is easier to ensure that all the databases are consistent.

Easier database management and access
Many of the Foundation members that attended our focus groups considered lack of flexibility to be a major problem with their existing database systems. Their databases and applications were very difficult to change and the changes took a long time to implement.

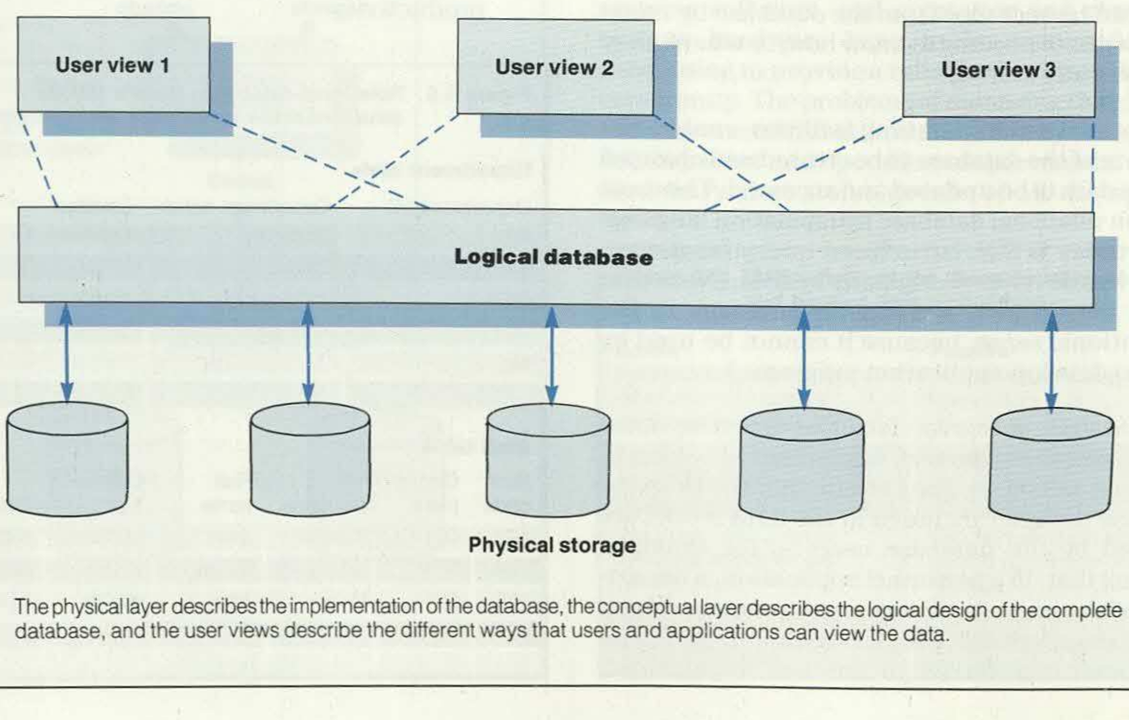
Relational databases and the applications that use them are more flexible because they are implemented by using a three-layer model of the database. Implementing a database with independent layers insulates the user from changes to the data architecture and insulates the data architecture from changes to the physical database storage. The three layers are:

— The physical level that manages the way data is stored on magnetic discs.
— The conceptual-schema level that defines the logical database (corresponding to the third level of the data architecture discussed in the previous chapter).
— The user- and application-view level that provides sets of database records tailored for a particular use.

Figure 3.7 shows the three independent layers of an advanced relational database. Changes can be made in each of the layers without affecting the other layers. One advantage is that the physical design of the database can be optimised after the applications have been implemented, when usage and data volumes are better known. Another advantage is that applications are not affected by changes to parts of the database that they do not use. It is therefore much easier to maintain relational database applications. If, for example, the logical model defined at the conceptual-schema level is changed to add more details about customers, then existing applications that do not need to access the extra details do not need to be changed.

Easier access to databases by users
In a relational database, each user can be provided with a 'view' (or subset) of the parts of the database specific to his or her information requirements. A view is constructed by the user or by the data administrator and consists of a set of data items and

Figure 3.7 The three layers used by advanced relational database management systems



The physical layer describes the implementation of the database, the conceptual layer describes the logical design of the complete database, and the user views describe the different ways that users and applications can view the data.

an extract from the complete database. Figure 3.8 shows an example of a user view that consists of the names of staff located in one department of the organisation. A view therefore enables a database user to work with familiar record structures while hiding the more complex underlying structure. A view can also be used to control access to the database by prohibiting access by certain users to specific parts of the database.

With the most advanced relational systems, a view can even be updated by the user. The system ensures that all the relevant parts of the database are updated. With first-generation databases, users need to be more aware of the underlying physical structure in order to retrieve the data.

Improved application-development productivity

Many of the Foundation members that responded to the questionnaires sent out at the beginning of the research expected that the use of relational database systems would lead to improvements in application-development productivity. The productivity gain attributable to relational databases is difficult to quantify because the move to a relational database is often accompanied by the introduction of an advanced system-building tool and/or a data dictionary. However, relational databases provide two major systems-development benefits — they can improve programmer productivity and they simplify the task of the database designer.

Relational databases improve programmer productivity because they make the task of accessing the database much simpler compared with first-generation databases. With first-generation databases, programmers required a detailed knowledge of the physical structure of the database in order to write the correct database-access code. With a relational database, programmers can work with logical data descriptions and they need only deal with a simplified view (subset) of the whole database. Furthermore, programmers access a relational database using SQL, the high-level database-access language. SQL is much more powerful than first-generation database-access languages and can select, retrieve, and update a set of records at a time. Relational databases therefore simplify the task of database programming and reduce the amount of code that programmers have to write.

Figure 3.8 User views are constructed from the tables of a relational database

Staff table

Staff code	Surname	First name	Date of birth	Salary
123	Adams	John	10/10/51	$30,000

User view

Staff code	Surname	First name
123	Adams	John

Development productivity is also improved by relational databases because database designers do not have to consider physical constraints when the database is being logically designed. With relational database management systems, the logical design of the database is created first and the physical design is developed and refined as more becomes known about the applications workload. Logical- and physical-design tasks can thus be considered separately, thereby simplifying the design process.

SHORTCOMINGS OF RELATIONAL DATABASE MANAGEMENT SYSTEMS

Although relational databases are a considerable improvement over their first-generation counterparts, relational products are still relatively immature, particularly in terms of their operational performance. It is therefore important to consider the shortcomings of relational products when choosing a relational database management system and selecting suitable applications. The major drawbacks relate to operational performance, to database integrity, and to the current implementations of SQL.

Relational database performance

User organisations are concerned that the operational performance of relational databases will be inferior to that for first-generation database products. Our research shows that relational databases still provide slower retrieval than a well-designed first-generation database. The relative differences between the performance of first-generation and relational databases cannot be quantified exactly because the actual performance depends so much on the particular applications and the software being used. In general, relational database management systems perform better when they are used for complex data-retrieval purposes. Their performance is not as good when they are used to process a high number of simple, and repetitive, updates to an operational database. Sharon Weinberg, President of the Codd and Date Consulting Group, which specialises in relational database technology, estimates that release 2 of Version 1 of DB2 is 40 per cent slower than an equivalent IMS application. However, benchmark tests indicate that performance varies considerably between different relational database products. The best perform up to seven times faster than the worst in these tests.

The first releases of relational software were, without doubt, very inefficient. However, their operational performance has been improved considerably and there is still considerable scope for further improvements. For example, the latest version of the Oracle product is up to 40 times faster than the version available six years earlier. Further performance improvements will be made because most suppliers of relational database management systems have not yet fully optimised their physical storage structures, because advanced software is being introduced to optimise database-access requests, and because most relational databases can use the larger amounts of random-access memory now available with many computer systems to speed up database access. (IBM provides up to 2 gigabytes of expanded memory to improve the performance of DB2, for example.) As a result of these changes, relational database suppliers are able to promise that each new product release will improve performance by up to 40 per cent. An example of the improvements that can be achieved is provided by a version of the Oracle relational database product that is optimised for transaction processing. This version, which was released in early 1988, is claimed to provide a six-fold performance improvement over previous versions.

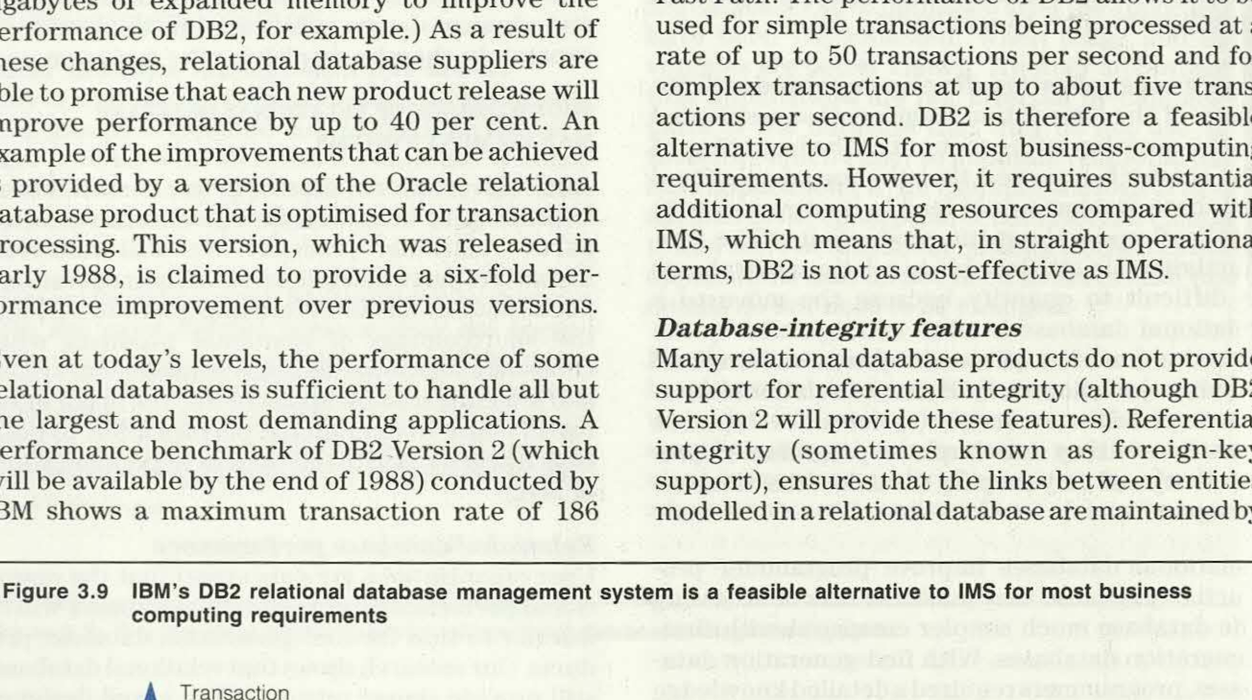
Even at today's levels, the performance of some relational databases is sufficient to handle all but the largest and most demanding applications. A performance benchmark of DB2 Version 2 (which will be available by the end of 1988) conducted by IBM shows a maximum transaction rate of 186 transactions per second at 90 per cent CPU utilisation on an IBM 3090-600E. (Few commercial online systems need to process this workload volume, however.) And Tandem claims that its NonStop SQL product is capable of processing more than 200 transactions per second when it is used with multiple processors to access a single-image database. These figures support the suppliers' claims that relational databases should eventually perform at least as well as first-generation databases.

Figure 3.9 shows the range of applications for which IBM's DB2 can now be used instead of IMS and IMS Fast Path. The performance of DB2 allows it to be used for simple transactions being processed at a rate of up to 50 transactions per second and for complex transactions at up to about five transactions per second. DB2 is therefore a feasible alternative to IMS for most business-computing requirements. However, it requires substantial additional computing resources compared with IMS, which means that, in straight operational terms, DB2 is not as cost-effective as IMS.

Database-integrity features

Many relational database products do not provide support for referential integrity (although DB2 Version 2 will provide these features). Referential integrity (sometimes known as foreign-key support), ensures that the links between entities modelled in a relational database are maintained by

Figure 3.9 IBM's DB2 relational database management system is a feasible alternative to IMS for most business computing requirements



applications using the database. For example, in an order-processing database it would usually be incorrect to add an order to the database if a corresponding customer record does not exist. The customer record must be established first. In addition, it should not be possible to delete the customer record if order records for the customer are still held in the database. Without referential-integrity features that ensure these types of checks are made, the database is vulnerable to corruption. Moreover, application programmers must include additional logic in their programs to provide the necessary integrity checks.

Referential-integrity features are included as part of the specification for the SQL database-access language. Unfortunately, the current SQL standard defines referential integrity only in an addendum and many suppliers have not implemented the features because of the performance penalties that would be incurred. SQL2, the next version of the standard, will include referential integrity as an essential element of the language. The SQL2 standard is expected to be published in 1989, although many suppliers are likely to anticipate the standard and provide referential-integrity features earlier than this. At present, Sybase (marketed by the company of the same name) and Cincom's Supra are the leading products in the area of referential integrity. Although referential-integrity features will enhance the utility of relational database management systems, their introduction is likely to increase significantly the hardware resources required to run such systems.

SQL standards

The SQL language is the means by which relational databases are created and manipulated. SQL is a very important element of relational databases because it offers the prospect of portability — the ability to transfer applications between database systems and to 'mix and match' database-access tools and relational database management systems.

Before complete portability can be provided, however, all suppliers will have to provide a standard version of the SQL language. A standard for SQL has been established by ANSI, the American National Standards Institute, and IBM claims that it will support the ANSI SQL standard as part of its systems applications architecture (SAA). Many other suppliers of database-management software are also planning to support this standard. The current standard for SQL (Level 1) does not specify all aspects of the language (for example the description of errors in database access requests). The next version of the ANSI standard (Level 2) will address many of these shortcomings.

In the meantime, the IBM DB2 version of SQL is likely to become an interim *de facto* standard. It is certainly the version of SQL that IBM will be marketing most vigorously. (IBM provides three different versions of SQL at present.) This version, in the near future, will become the basis for practical links between separate data-management systems. SQL is unlikely to be superseded by any other relational database-access and maintenance language, and eventually will evolve into a powerful international standard that supports a high degree of portability.

TIMING THE MOVE TO RELATIONAL DATABASES

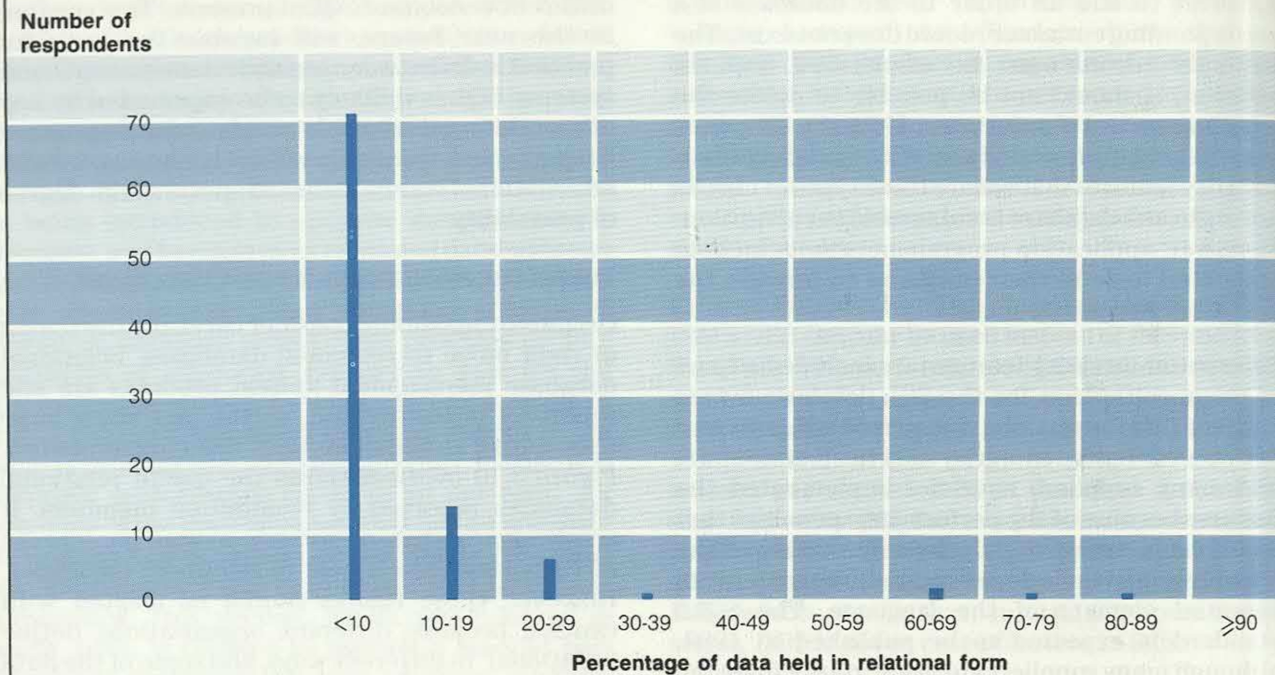
Organisations should consider carefully the timing of their move to relational databases. Relational database management system products are still relatively immature and, as yet, not many large operational applications have been implemented. Figure 3.10 overleaf shows the use of relational databases reported by Foundation members. It shows clearly that, at present, a small percentage of corporate data is held in relational databases. However, these figures should be treated with caution because different organisations define 'relational' in different ways, and some of the data included in Figure 3.10 refers to databases that provide much of the functionality of relational systems without adhering strictly to their concepts.

We showed earlier (in Figure 3.9) that many applications are practical within the current levels of relational database performance. However, it should be noted that, as databases grow in size, the performance of relational databases decreases in a nonlinear way that is hard to predict.

Nevertheless, we believe that the time has now come for many organisations to begin using relational technology. An organisation about to embark on a major overhaul of its applications portfolio would be well advised to move to relational technology at the same time. Delaying any longer will only postpone the inevitable move and will make the task of converting existing databases and applications even more difficult than it would be today. First-generation databases, and their associated tools, have effectively reached the end of their life cycle. From now on, new data-management tools will depend on relational technology. Some leading organisations have already recognised the advantages of moving to relational technology sooner rather than later. For example, Marks and Spencer plc, a leading UK retail chain, has created a very large database using DB2, and Amro Bank in the Netherlands will now use DB2 for all new development work.

Other organisations will prefer to wait until they can make a convincing cost-benefits case for investing in relational databases for major applications. In doing this, they will need to compare the additional

Figure 3.10 Most corporate data is still held in nonrelational form



(Source: Survey of Foundation members)

costs of using relational databases with the benefits of improved development productivity and earlier applications implementation. Maintenance costs will also be reduced because relational databases are more flexible and hence easier to change.

The projected cost-benefits of relational technology for a particular application can be depicted on the type of graph shown in Figure 3.11. This shows that the costs of developing a relational database application are lower than a nonrelational database application, but that the running costs are significantly higher. A relational database is therefore a suitable choice for applications with a short anticipated operational life, for applications with complex database requirements that will benefit particularly from the improved applications-development productivity available with relational databases, for those that are likely to change rapidly and will thus benefit from the improved flexibility provided by relational databases, and for those with small processing volumes.

DISTRIBUTED DATABASES

A distributed database enables data to be stored at many different sites without the user needing to be aware of the location of the data when it is updated or accessed. The data can either be partitioned (that is, each site holds part of the database) or replicated, where the same data is held at each site. Although distributed-database technology is relatively immature, rapid progress is being made and practical applications are now possible.

Distributed database systems will be used mainly for three technical purposes:

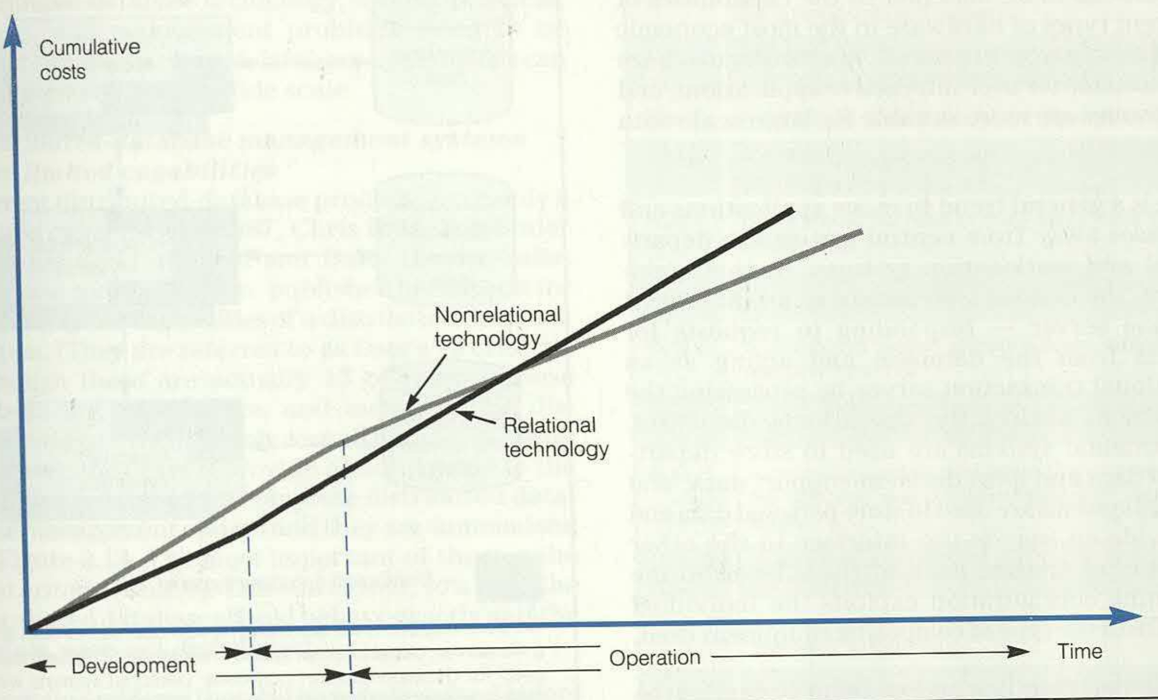
— To integrate data stored in different databases and to distribute the data among the databases.
— To overcome operational-performance problems.
— To match applications to the most suitable configuration of mainframes, minicomputers, and microcomputers.

Each of these three areas is now discussed in turn.

INTEGRATING AND DISTRIBUTING DATA

As computers and databases proliferate throughout the organisation, it is quite likely that individual departments will create their own 'islands' of data. As a consequence, there will be an increasing need to integrate and distribute data on a regular basis. Distributed-database systems are designed to distribute data reliably and efficiently and they will provide much better facilities than the existing crude and inefficient database-extract facilities now used by many organisations. A particular problem with present extract facilities is the need to halt operational systems when extracts are made, in order to obtain data that is consistent at a particular point in time. This requirement places

Figure 3.11 Relational technology allows applications to be implemented earlier and at lower cost, but the running costs are higher than nonrelational technology



serious constraints on the frequency of updating and on the currency of database extracts used for decision-support purposes.

Another problem experienced by many organisations today concerns the difficulty of maintaining separate databases that contain different versions of the same data. The process of ensuring that these databases are in step with each other can be very time-consuming because it is often necessary to copy large portions of one database to another. Moreover, this copying usually has to be done during a weekend, or at best overnight. Existing copy-management software often does not provide adequate procedures for controlling different generations of data, with the result that it is difficult to ensure that the data held in different databases is consistent. Distributed-database management systems provide better copy-management facilities because they use more advanced techniques.

An example of such a facility is DEC's Vax Data Distributor. This product can create and automatically manage several extracts from a database. Two modes of operation are available — extraction, which creates a complete new version of the remote database, and replication, which transmits only the changes to the remote database. Tandem provides a similar product known as the Remote Duplicate Database Facility. This product has been used to keep two IBM databases in step with each other to within one or two minutes. Facilities such as these can reduce greatly the management and performance problems of controlling several database copies, and they provide the means of obtaining up-to-the-minute management information without affecting the performance of the operational database.

OVERCOMING PERFORMANCE PROBLEMS

Some organisations need to process very high volumes of transactions against large volumes of data (more than 50 transactions per second accessing a database containing tens of gigabytes of data). These applications are becoming increasingly difficult to handle as single-image databases on one mainframe because the growth in mainframe performance is being outstripped by the growth in the workload of many organisations. Mainframe processing power is increasing by about 20 per cent a year while most organisations' workload is growing at 40 per cent a year. The result is longer response times for large database applications.

These performance problems can be reduced by distributing data and transactions to multiple systems, each of which is used to handle a part of the load. Moreover, distributed databases can provide much greater scope for incremental capacity growth than a single mainframe system. The distributed-database approach allows the Tandem NonStop SQL product to achieve the impressively high performance level we described earlier on page 20.

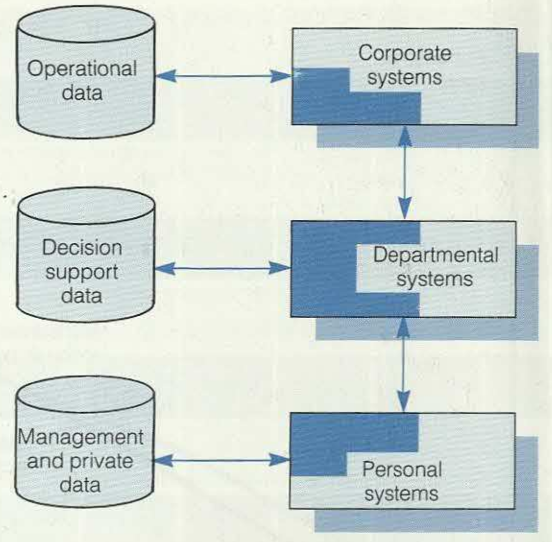
MATCHING APPLICATIONS TO THE COMPUTING CONFIGURATION

Distributed database management systems permit applications to be matched to the capabilities of different types of hardware in the most economic and effective way. In general, microcomputers are more suitable for user-interactive applications, and mainframes are more suitable for large-scale data management.

There is a general trend to move applications and databases away from central systems to departmental and workstation systems. In this architecture, the central systems act as an intelligent database server — responding to requests for extracts from the database and acting as an operational transaction server by processing the transactions against the operational databases. Departmental systems are used to store departmental data and local decision-support data, and personal systems are used to store personal data and to provide an easy-to-use interface to the other two types of system. Each of these levels in the computing configuration exploits the individual strengths of the type of computing equipment used.

A distributed database management system can be used to manage the links between the three database levels so that the user does not need to know where data is located in the overall configuration. The database manager can therefore store data according to the facilities required to manipulate it and the degree to which it needs to be shared. Figure 3.12 shows an example of a three-level architecture for distributed-database applications. Products such as NonStop SQL, Sybase, and Ingres/Star can support this type of architecture.

Figure 3.12 Three-level architecture for distributed database applications



= Distributed database systems

Corporate systems will store and handle operational data as a file server; departmental systems will store extracted data used for decision-support purposes; personal systems will store management and personal data. All the elements will be interlinked by a distributed database system, which will include distributed data-dictionary facilities.

Microcomputer workstations will play a central role in this new configuration by supporting cooperative processing. This term is used to describe the situation where two or more systems cooperate in performing a task so that the methods of connecting the systems, and the operating systems in the respective systems, are transparent to the programs performing the task. In cooperative-processing systems, the workstation handles the input and display of information and sends requests for data and directs processor-intensive operations to the central system. This approach is being supported strongly by the suppliers of microcomputer database management systems because the graphics capabilities and the predictable performance of workstations provide better user interfaces, whilst the central system provides access to a large, powerful, and well-managed database.

Rapid developments are occurring in the facilities provided by microcomputer database systems. The requirements are for data-management systems that can provide relational interfaces between the user and the microcomputer database and that can support cooperative processing between the microcomputer database and remote databases. This implies the need to interlink the products of several suppliers, which means that standards are a prerequisite for cooperative processing. At present, a combination of ANSI standards and IBM's DB2 SQL is the most common approach. However, the next level of the ANSI SQL standard and the development of the Remote Data Access (RDA) standard by ISO will provide an even clearer definition of the required interfaces.

Several microcomputer database suppliers are enhancing their products so they will be able to interface with mainframe databases in this type of way. For example, Paradox (which is marketed by Borland) will provide a back-end SQL interface that can access and retrieve data from mainframe relational databases. Oracle and Relational Technology also have PC versions of their products that can query remote databases using the SQL language. And Ashton-Tate, Microsoft, and Sybase have entered into an agreement to develop an SQL local area network database server that will work with products such as dBase IV. The SQL data manager embedded in IBM's extended edition of the OS/2 workstation operating system is another example of this type of product.

SHORTCOMINGS OF CURRENT DISTRIBUTED-DATABASE SYSTEMS

Although great progress has been made in distributed-database technology, several technical, design, and management problems need to be solved before distributed-database applications can be implemented on a wide scale.

Distributed-database management systems have limited capabilities

Current distributed-database products have only a limited capability. In 1987, Chris Date, cofounder with Ted Codd of Codd and Date, the specialist database consulting firm, published his criteria for evaluating the capabilities of a distributed database system. (They are referred to as Date's 12 criteria, although there are actually 13 of them.) These criteria are very severe, and most existing distributed systems meet only a small number of them. However, the criteria provide a useful guide to the facilities required by a complete distributed database management system and they are summarised in Figure 3.13. The most important of them is the first criteria (Rule 0). This states that, to a user, the distributed database should behave exactly as if the system were not distributed.

The main problems that still have to be solved before distributed-database systems can meet all of Date's criteria are:

- The ability to record up-to-date information about the contents, currency, and location of data in a distributed database, so that users do not need to know the location of their data in order to access it (Rule 4).
- The ability to update replicated data simultaneously (Rule 6).
- The ability to provide acceptable response times for ad hoc queries that access several distributed databases connected by low-speed data communications links (Rule 7).
- The ability to maintain the consistency of a distributed database after a failure (Rule 8).

Several relational database suppliers (Tandem, Sybase, Oracle, and Relational Technology in particular) provide useful distributed-database capabilities. Sybase, for example, provides the ability to update and retrieve data stored at several sites, but the user (or the application developer) has to know where the required data is located. Relational Technology plans to release the third generation of its distributed database management system (known as Ingres/Star) during 1988. The functions of this release are designed to meet all 13 of Date's criteria.

Distributed-database applications are difficult to design

Designing a distributed-database application is a demanding task and few methods are available to assist the application designer. The key decisions that have to be made concern the partitioning of the database and the physical location of the data. Wrong decisions made at the design stage will result in poor operational performance caused by the communications bottleneck between the different parts of the distributed database. In order to decide on the optimum way of distributing a database, it is first necessary to prepare an accurate data model and to carry out a detailed analysis of the application characteristics and transaction volumes. Data can be distributed in two ways — by placing a copy of it at each location that requires access to it (replication) or by subdividing the data and placing parts of it at each location (partitioning). Figure 3.14 overleaf provides guidelines for deciding whether to partition or replicate data, or whether it is better to store all the data centrally.

Figure 3.13 Chris Date's 12 criteria for evaluating a distributed database

Rule 0	To the user, a distributed system should look exactly like a nondistributed system
Rule 1	The sites of a distributed system should have local autonomy for local data and operations
Rule 2	A distributed system should not rely on a central site
Rule 3	A distributed system should be capable of continuous operation
Rule 4	Users should not need to know the physical location of their data
Rule 5	A distributed system should support fragmentation of relations (records)
Rule 6	A distributed system should support replication of relations (records)
Rule 7	A distributed system should be able to optimise distributed query processing
Rule 8	A distributed update transaction should always leave the database in a consistent state after success or failure
Rule 9	A distributed system should be independent of hardware type
Rule 10	A distributed system should be independent of the operating system type
Rule 11	A distributed system should be independent of the network characteristics
Rule 12	A distributed system should be independent of the database management system type

Distributed-database applications are difficult to control

Controlling the consistency and currency of a distributed database is much more difficult than for a central database. A distributed database can hold several copies of data, or the database may be partitioned and stored at several sites. For example, in a personnel database the individual personnel records may be partitioned and stored at several sites, while the salary scale is replicated at each site. A distributed-database controller — usually the data administrator — has to ensure that the records of a staff member are stored in the right database

Figure 3.14 Design guidelines for distributed databases

Ratio of updates to read-only access	Proportion of updates that would need to access more than one site if database is partitioned		Database storage strategy
High	High	→	Centralised
High	Low	→	Partitioned
Low	High	→	Replicated
Low	Low	→	Partitioned

The most important criteria for deciding how to distribute data are the ratio of database updates to read-only accesses and the number of updates needing to access more than one site if the database is partioned.

and that replicated records are identical (consistency), and that each database stores the latest salary scale (currency). The distributed-database controller therefore needs a powerful data-dictionary system that works in conjunction with the distributed database management system to automate the detection and control of these types of discrepancy. (Most leading distributed database management systems provide this type of facility.)

THE NEED FOR AN INTEGRATED SET OF DATA-MANAGEMENT SYSTEMS AND TOOLS

So far in this chapter we have discussed the three main types of data-management system — data dictionary, relational database, and distributed database — in isolation.

However, in selecting new data-management systems and tools it is important to note that the most powerful and effective products are usually part of an integrated set where each system can exchange information with, and support the functions of, the other systems. Within such an integrated set there is an emerging division between data-management systems and data-management tools. Data-management tools are used by system developers and users to design, access, and manipulate databases. The tools include products such as Case tools, application generators, and facilities that allow users to access corporate databases.

Several suppliers are planning to develop sets of integrated data-management systems and tools. Their objective is to provide a common set of interfaces between each system and tool so that data can be easily exchanged (to and from the data dictionary for example), and to present a common interface both to users and to applications programs. Notable developments include elements of IBM's systems application architecture (SAA), DEC's Vax information architecture (VIA) and Software AG's software information architecture (SIA).

Architectures such as these provide considerable benefits for application developers because several integrated tools can share the same data descriptions across the set of data-management systems. The interfaces between the products allow, for example, text-manipulation and electronic-mail functions to be accessed from a data processing application. They also allow a single data dictionary to be used for a range of databases covering operational systems and systems-development activities. Moreover, distributed applications are easier to implement because the architecture ensures there is a consistent interface between the different data-management systems and tools.

An architecture that has been defined by a major hardware or software supplier can also open the way for a wide range of third-party products. For example, DEC provides a well-defined interface (SRDI — standard relational database interface) that allows third-party suppliers to link their products to VIA, thereby gaining access to the data dictionary and relational databases. This facility will provide user organisations with a wide choice of products, such as system-building tools and user-access tools, that conform with an integrated approach to data-management systems and tools.

The link between these two classes of products — data-management systems and data-management tools — is provided by the database-access language SQL. SQL provides a standard interface between the two types of product, so that an organisation can 'mix and match' a set of tools and data-management systems to provide the required data-management facilities. Thus, provided that the basic data-management systems are well-integrated, the data-management tools can be selected from a wide variety of products.

SUMMARY

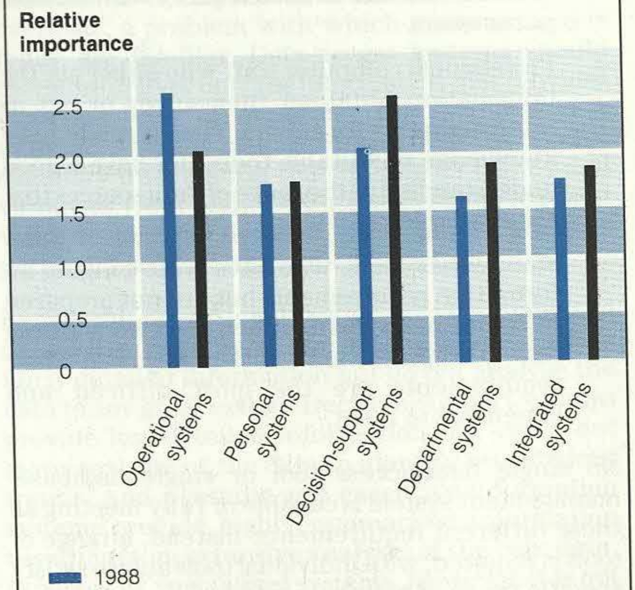
In this chapter, we have reviewed the developments in the three main types of second-generation data-management systems (data dictionaries, relational database management systems, and distributed databases). We have provided advice about how to select appropriate products and when to use them. The focus of the chapter has been on using the systems for creating and maintaining corporate databases. However, our research has shown that there will be an increasing requirement for business users to have direct access to the databases. We address this subject in the next chapter.

Chapter 4

Providing facilities for user access to databases

One of the most important objectives when developing corporate databases is to satisfy the growing requirement for management information. Many organisations have now solved, or are rapidly solving, the problems of providing operational systems to support the day-to-day running of the business. Instead, much of their systems effort is now directed at satisfying the need for ad hoc information. In our survey of Foundation members for this report, we found that satisfying requirements for improved decision-support systems was expected to be the top priority in five years' time. Figure 4.1 shows that decision-support requirements will displace operational-support requirements as the highest priority for the systems department.

Figure 4.1 The focus of attention of the systems department is changing from operational to decision-support systems



Respondents were asked to rate the importance of each type of system both today (1988) and in five years time on a scale 0 (unimportant) to 4 (important).

(Source: Survey of Foundation members)

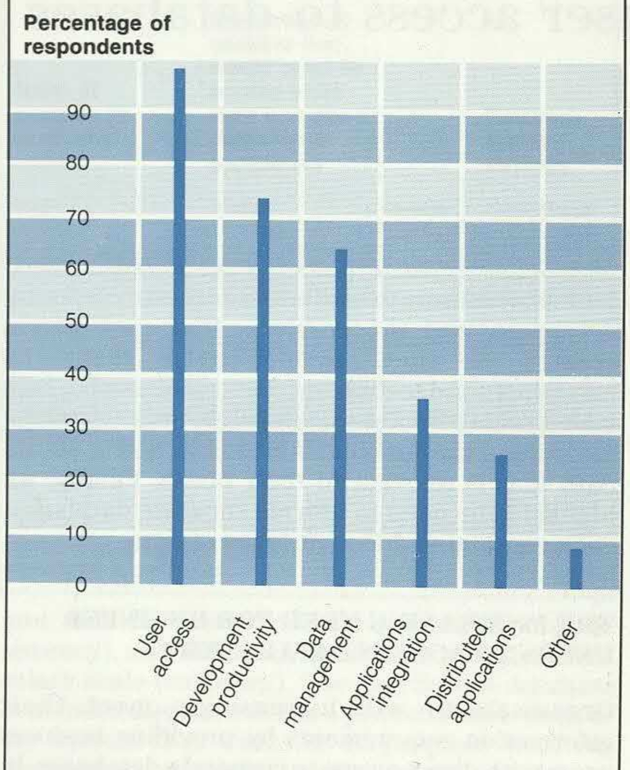
In this chapter, we first identify the increasing need for business users to be allowed access to corporate databases, and the problems that such access creates. We then provide advice about the procedures and techniques that can be used to help users access corporate databases, to understand the meaning of the data that is available, and to obtain data that is relevant to their needs. Finally, we highlight the need to provide separate databases, each containing different types of data.

THE INCREASING NEED FOR BUSINESS USERS TO ACCESS DATABASES

Organisations will increasingly meet their information requirements by providing business users with direct access to corporate databases. In our survey, we asked Foundation members what were the three most important benefits of new database systems. Figure 4.2 overleaf shows that members rated 'improved user access to the databases' as the most important benefit of new database systems. One reason for this trend is that business information requirements are becoming more complex. Furthermore, ordinary business people are becoming 'computer literate' and so are able to handle their own data-access requirements without having to use the systems department or the information centre as an intermediary. Opening corporate databases to the user community provides benefits because the systems department no longer has to spend time on satisfying simple information-retrieval requests and can concentrate its efforts on the task of building the databases.

The UK brewing division of Grand Metropolitan plc provides an example of the type of access to databases with which users are now being provided. This organisation does not automatically include a wide range of predefined reports when it develops new applications. Instead, it plans to set up systems that can extract data from the operational database and that make the data available for use with an appropriate PC-based or mainframe-based information management tool. Such tools provide business users with a much more flexible method of interrogating the database that can be adapted to their information needs.

Figure 4.2 Most organisations expect improved user access to be the most important benefit of new database technology



Respondents were asked to state the three most important benefits they expected to gain from new database technology

(Source: Survey of Foundation members)

THE PROBLEMS OF USER-DRIVEN DATABASE ACCESS

Database-access facilities have to provide the means of transferring the data held and described in the databases to the business user who wishes to access it. In order for the transfer to be completed successfully, three conditions must be satisfied — the data must be received correctly, the recipient must understand the meaning of the data that is delivered, and the data must be relevant to his or her needs.

ACCESSING AND RETRIEVING THE DATA

Ensuring that data is accessible and can be retrieved easily are major considerations in developing user-driven information systems. Users may not know what data is available, or they may not be able to find data they know exists, or they may not have the skills required to retrieve the data.

User-driven information systems often expect users to provide the system names of the data items that they wish to retrieve. This means that users face considerable difficulties when they try to obtain information by using applications with which they are unfamiliar. The ideal data-access facility will provide users with a description of the information that is available for the area they are interested in. A data dictionary that can be used by business users is one way to meet this requirement.

Users in a large organisation often face a formidable problem in finding relevant information. Usually, they cannot refer to a single source to find out what data is available. As a consequence, the source of the data used for decision-support purposes is often computer printouts — these are the only data sources readily available to many business people. Using data from printouts in this way is fraught with difficulties because the meanings of data items are not well defined. The data itself may have been processed through several systems, and it may be out of date or inconsistent. This means that the resulting analyses based on the data are often ambiguous or contradictory.

Business users can most successfully access the data they require when the access facilities are matched to their skills and information needs. These skills cover a range, including:

— Business users whose information requirements are largely predictable. Such users need access to well-defined parts of the database and need to use a simple access method — for example, a user view extracted from a relational database or a fixed query with variable parameters.

— Professional computer staff who either use the information obtained themselves or act as intermediaries for others. They already have knowledge of suitable tools and have a good understanding of the range of data sources that are available.

— Managerial users, who have wide-ranging information requirements but are not prepared to spend more than an hour or two learning how to access the information they need. Their requirements are the most difficult and expensive to meet.

No single data-access tool or single database-management system is capable of fully meeting all these different requirements. Instead, a range of tools is required, with individual tools able to satisfy specific types of requirement.

UNDERSTANDING THE DATA

An important task in providing user-driven data-access systems is to educate users about the contents and structure of the parts of the corporate databases that are relevant to their needs. As part of this task, it will be necessary to create a data dictionary that describes the data items to which users will have access. To access the databases themselves, users need to know far more about the structure and content of the databases than they did when they were provided with prespecified reports produced by the systems department.

Providing users with an adequate understanding of the content of databases, and the meaning of data items, is a major challenge. The precise meaning of a data item and its relationship with, and differences from, other data items is often not immediately apparent. A good example of the problems of understanding the meaning of data was provided at one of our focus groups. A senior manager at a leading insurance syndicate was trying out a new database-access facility and wished to know the number of members in the syndicate. He knew the answer and was disappointed to find that the computer produced a wildly inaccurate figure. The error was caused by conflicting definitions of the term 'member'. The computer understood 'member' to mean both past and present, but the senior manager meant only present members. To overcome these problems, the data dictionary needs to provide a detailed and precise description of the data items.

ENSURING THE INFORMATION IS RELEVANT

One of the major challenges for designers of user-driven systems is to build systems that deliver relevant information. Providing people with large amounts of irrelevant data causes 'information overload', a problem with which many managers today are familiar. Data-access systems should therefore deliver only the information that provides the right level of detail for the recipient. For example, a senior manager might want to receive summary information that highlights trends and variances from plan. Other business users might want to review detailed information about a particular stock item or sales territory.

In general, operational systems that support the day-to-day running of the organisation provide fairly detailed information but do not analyse the data to any great extent. Decision-support systems provide less-detailed information but carry out more analyses of the data to identify less-obvious trends. And planning and executive-information systems provide highly summarised information resulting from extensive analyses of the base data, typically by specialised systems. However, it is not always true that senior management is only interested in summary information. Once the summary information has highlighted a particular area of concern, senior managers often need to 'zoom-in' to the relevant parts of the database to examine the underlying data in detail. Providing this facility is particularly demanding.

ASSISTING BUSINESS USERS TO ACCESS DATA

The data administrator, in conjunction with the information centre (or business-support function), should play a key role in publishing details about the organisation's databases and information resources and in assisting business users to identify and access the information they need. Various computer-aided tools, some of which are based on expert-systems techniques, are already being used both to assist in the search for information and to simplify the task of accessing the data.

To help the data administrator in this task, it is sensible to make a start on creating an inventory of all the information in the organisation that is used for decision-support purposes. This inventory will not only include shared corporate data, but will also cover external data, planning data, and so forth. Several of the organisations we spoke with are developing information inventories that will be used to assist users to locate the information they need. These inventories need to describe the information resources in terms that users understand — for example, by referring to data by names that are in common use in the user community even if these are not the names used by the systems department.

The data-dictionary system used by the Inland Revenue department in the United Kingdom provides an example of the way in which business users can use this type of system. The dictionary, which is based on the Status text-retrieval system, can be searched by data-item description as well as by name. Separate, lower-level dictionaries running on ICL and IBM computers provide the information to update this dictionary. The department's statisticians (and other staff) can use the dictionary to locate data either by searching for the textual description of the data items or by using the data names. Considerable reductions have been achieved in the time taken to locate data items that are needed for new statistical reports.

Some organisations are beginning to introduce advanced systems based on artificial-intelligence techniques to assist the users to find the data they need. Two examples of such products are Infosearch from ISTEL (a UK systems and software company) and Nemasis, a product used by ICL for internal systems. Infosearch assists users to access more than 1,000 public databases. It provides three levels of access support — for inexperienced users, for occasional users, and for professional users. The first level allows the user to ask a question in general terms and Infosearch then selects the database that will provide the required information. The second level allows a standard database-access language to be used for all the databases. An expert system is used to translate the common language into the native database-access language of the required database. The Nemasis system used by ICL is also based on an expert system. This system provides guidance for ICL staff accessing the large number of separate internal business databases in the organisation. Such systems are very useful in decision-support systems that require information from both external and internal sources.

Expert systems can also simplify the task of accessing databases by allowing the queries to be expressed in natural language. The English-like sentences are then translated into database-access commands. The system then asks the user to verify that it has interpreted the query correctly before retrieving the required information. (The syntax capabilities of a typical PC-based natural-language product, Q&A — provided by Symantec — were shown on page 37 of the previous Foundation Report — No 63, The Future of the Personal Workstation.)

Another natural-language system is Datatalker, available from Natural Language Inc. in Berkeley, California. Datatalker allows the user to query a Sybase relational database using a restricted form of English. This product is less expensive and more effective than most other natural-language software because it uses a combination of traditional natural-language-processing and artificial-intelligence techniques. The artificial-intelligence-based concepts library and deductive system is used to resolve ambiguous sentences by taking the context of the query into account.

Although natural-language interpreters do make access to databases easier, it is difficult to construct an accurate dictionary (or, to use the jargon of natural-language systems, a lexicon) for a particular database. In the December 1987 edition of *Byte*, Gary Hendrix of Symantec, the developer of Q&A, described this problem as the 'natural-language processing (NLP) wall'. A natural-language interpreter can be made to understand the language of the user fairly easily, but it is much harder for it to capture the meaning or 'language' of the database. A large amount of effort is required to describe the database to the natural-language interpreter, and this difficulty is known as the NLP wall. The Q&A product partially overcomes this problem by providing a 'teaching' facility that enables the user to describe the database to Q&A with a series of menus.

ASSISTING USERS TO UNDERSTAND DATA

The problem of the NLP wall described above shows how important it is for users to understand the meaning of the data stored in the databases. Without such an understanding, any analyses or inferences based on the data will be suspect. (To use an analogy, the analyses would be as reliable as a data-access program written by a programmer who did not fully understand the database's contents and structure.) Thus, a prerequisite to opening up corporate databases to access by business users is the existence of comprehensive database documentation.

One method for users to gain an understanding of the databases is via a general or a specially tailored data dictionary. Ideally, the dictionary will contain a description of the underlying data model as well as the individual data items, so that users can understand the context of their queries and avoid the problems caused by ambiguous data definitions.

One data-management system that provides extensive facilities for describing databases in terms that can be understood by business users is the Semantic Information Manager (SIM) supplied by Unisys. SIM's data dictionary is linked closely with the database manager and is used in the analysis of the users' information requests. The data-dictionary component holds considerably more information about the relationships and structure of the database than is held in a basic relational database. The data dictionary contains information about the meaning of the data in the database and presents this when it displays the database structure. It also contains data derived from the basic data items and it can create subclassifications of data — classifying retailers as supermarkets and small shops, for example. Users work directly with this information and are guided through a view of the logical database structure by a graphics-based query facility that is used in conjunction with the data dictionary. SIM therefore helps users to learn about the structure of the database as they formulate their queries. In addition, the database and data-dictionary rules help to ensure that the data extracted is relevant to the users' needs. (SIM is described in more detail in the appendix.)

We expect that many more user-driven database-access systems will provide this type of facility in the future. IBM, for example, said in its recent announcement of referential-integrity features for DB2 that these features will allow users to update a database without compromising the integrity of the database.

ASSISTING USERS TO OBTAIN RELEVANT DATA

Different types of staff have very different data-access requirements and use the data extracted from corporate databases for very different purposes. In our research, we found that organisations are meeting these requirements by developing three different types of system, which we call management-information systems, decision-support systems, and the newest type, known as executive-information systems. Each of these types of system addresses different information requirements and is intended for use by different types of user. Figure 4.3 compares the characteristics of each type, and each is described below in more detail.

MANAGEMENT-INFORMATION SYSTEMS

Management-information systems typically provide routine information from the operational systems of the organisation. Thus, they provide regular summary reports that give a detailed snapshot of the organisation's status at the time of reporting. Users of such systems are interested in looking in some detail at the current status of business activities. They need to access the databases for the purposes of:

- Monitoring. Standard reports that summarise the daily transactions, and that are produced on a routine basis to a fixed schedule, are normally sufficient for this purpose.
- Exception reporting. Systems that produce predefined reports about predefined exception conditions are used for this purpose.
- Enquiring. Systems that provide a flexible enquiry capability, enabling users to design and change their own reports as they wish, are used for this purpose.

DECISION-SUPPORT SYSTEMS

Decision-support systems are used to answer questions such as "What will be the effect on profits if we increase all salaries by 5 per cent?" The data used in these systems is a mixture of data extracted from the operational databases, additional data (such as budgets and plans) entered by users, historical data, and even personal data.

These systems provide powerful facilities for analysing data in order to support decision making. The facilities include optimisation routines, modelling systems, simulation systems, and statistical analyses. In addition, decision-support systems often provide powerful graphics-display features so that the results of the analyses can be presented in an attractive way.

Figure 4.3 Characteristics of management-information, decision-support, and executive-information systems

Type of system	Ease of use	Detail	Graphics	Modelling	Access to external information	Access to derived information
Management information	✓✓	✓				
Decision support	✓		✓	✓		✓
Executive support	✓✓	✓	✓		✓	✓

✓: Good
✓✓: Excellent

Many organisations require access to large databases for decision-support purposes. Examples include the insurance industry, where decision makers need to analyse policy-holders' data, and the retail industry, where there is a need to analyse retail-sales data. A few organisations are now using database machines for this type of application in order to improve the operational performance of their systems. For example, during the 1988 Foundation Study Tour visit to Teradata, delegates heard Cahl Rahmquist, manager of end-user support services at Transamerican Insurance Group, describe how his company uses the Teradata DBC/1012 database machine.

In 1982/83, Transamerican examined available data-retrieval tools and selected Intellect because it is user friendly and because its commands use basic English. It was used initially by the four top insurance managers and subsequently by managers throughout the United States. There are now more than 750 authorised users, about 350 of whom use the system regularly.

In late 1985, these users were accounting for around 8 per cent of the capacity of the company's IBM 3090-400. As a consequence, Mr Rahmquist decided to investigate Teradata as an alternative means of handling enquiries under DFAM. He carried out benchmark tests and a pilot project during 1986 and purchased a Teradata system in November that year.

In the test, various kinds of enquiry made via Intellect/DFAM were compared with the same enquiries made via Intellect/Teradata. With Teradata, the response times for simple and moderately simple enquiries were slightly slower, but the response times for moderately complex and complex enquiries were very much faster. For a typical month (July 1986), the actual mix of enquiries processed with the Teradata machine reduced operational costs by $26,000 — a reduction of 60 per cent. File loading was also significantly faster.

Mr Rahmquist said that the number of Intellect enquiries had increased considerably since the

Teradata system was installed and had grown in complexity, but that the saving in the cost of IBM processing time, and the much faster response times, easily justified the investment in the Teradata equipment.

(Those readers interested in more information about the current status of database machines should refer to the 1988 Study Tour Presentation Summaries. That document contains details of the presentations made by Teradata and by Nucleus International Corporation, suppliers of an innovative relational database engine.)

EXECUTIVE-INFORMATION SYSTEMS

Executive-information systems are a relatively new development. They are aimed at senior managers, who, in general, have been poorly supported by management-information and decision-support systems. The aim of an executive-information system typically is to provide information about business performance and external factors that influence the business environment. Systems of this type are very powerful, but they are expensive to set up and operate. For this reason, their use is likely to be restricted to key personnel, at least in the immediate future. (Executive-information systems, or boardroom systems, were discussed in detail in Foundation Report 53 — Using Information Technology to Improve Decision Making.)

Compared with management-information and decision-support systems, the main distinguishing feature of executive-information systems is the power and ease-of-use of their user interface. They are very easy to use because they have been designed for direct use by senior managers. Executive-information systems provide summaries of large volumes of information that are presented in an easily understood format using exception reporting and graphical techniques. However, when required, they can 'zoom-in' to examine the supporting data in detail when further information is requested.

Many examples of this type of system are based on relational database systems, but several, such as Resolve from Metapraxis, Acumen from EFFEM Management Services Limited, and Cubit from ICL, are based on a matrix approach to data management. Matrix database systems are often more suitable for the specialised information needs of senior managers. The matrix has several dimensions, with each dimension providing a different reporting 'spectrum'. For example, one dimension may show data by country, another data by product, another data by month, and so on. Figure 4.4 shows the structure of a typical matrix database system.

THE NEED TO PROVIDE MULTIPLE DATABASES

The three different types of system described above are most useful when they are based on several separate databases each containing different types of data. Separate databases are preferable to a single database because:

— Unrestricted use of management-information systems, decision-support systems, and executive-information systems can seriously degrade the performance of operational databases, but constraints on usage are not acceptable to the user community.

— Many of the databases used for decision-support and executive-information purposes contain external and derived data that does not easily fit in with the operational database and may 'contaminate' it.

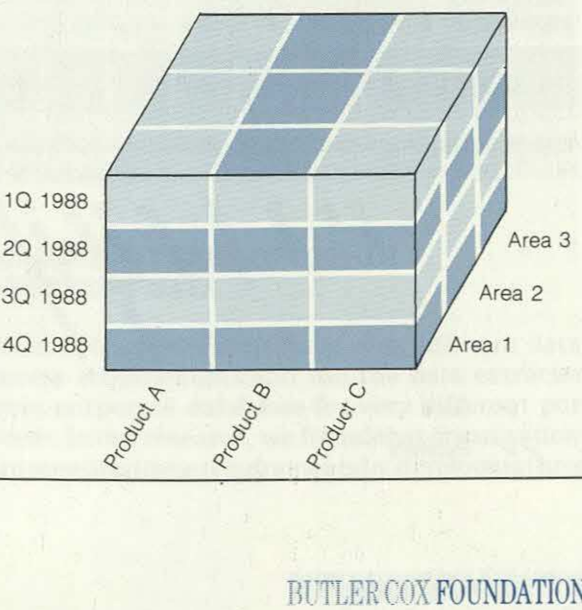
— The structure of decision-support and executive-information databases may be different from operational databases. (The database structure used with Metapraxis's system, for example, is different from that used with conventional databases.)

— Decision-support and executive-information databases need to hold large amounts of historical information that would not normally be held in an operational database.

— The use made of databases for decision-support and executive-information purposes may need to change easily and quickly without being constrained by operational systems.

An additional advantage of providing separate databases is that the organisation can achieve many of the decision-support and executive-information

Figure 4.4 A matrix database structure allows information to be summarised easily in a variety of ways



benefits of advanced database systems without having to disturb the existing, well-tried, operational systems.

For these reasons it is sensible to establish a set of databases that can support the use of management-information systems, decision-support systems, and executive-information systems. The need for this type of database architecture was predicted in a previous Foundation Report on the topic of corporate databases — Report 50, Unlocking the Corporate Data Resource. The separate databases are often known as 'data pools' by those organisations that have adopted this approach.

AN EXAMPLE OF SEPARATE DATABASES

Rolls-Royce plc, the Derby-based aero-engine manufacturer, is one of the many organisations that plans to implement the data-pool approach. The systems department at Rolls-Royce will provide four separate database areas for use by 3,500 staff in the organisation. These areas are:

- Operational DB2 and IMS databases, and VSAM files, all of which are professionally derived and managed. All access to these databases and files is by programs written by the systems department.
- A DB2 database containing 'snapshot' data derived from the operational databases. This area is known as 'professionally derived data' or the 'data warehouse'. Users can access this data but cannot change it.
- Local or departmental shared data, together with professionally derived data from the first area, that can be modified by users as they seek answers to 'what-if' questions. This area is controlled by the originators and users of the data.
- Personal data that belongs to, and is only used by, an individual.

The first two areas are true database areas. Rolls-Royce refers to the second two areas as 'information bases', because the data is used in a specific context.

THE PROBLEMS OF MANAGING DATA POOLS

Although a series of specialised databases is the best way to provide flexible data-access facilities, this approach can give rise to several problems. In particular, it is necessary to update the databases on a regular basis by copying across relevant portions of the operational databases. Often, this process takes an unacceptable time and can be run only at the weekend or at best overnight. The result is that the information held in the specialised databases is always out of date.

Distributed-processing techniques can help to overcome these problems, however. For example we described in the previous chapter (on page 23) how distributed databases can greatly reduce the problems of managing and controlling several database copies, and can provide the means of obtaining up-to-the-minute information without disrupting the operational databases.

Another problem is that several copies of the data dictionary will often be needed to support user access to databases. The copies may differ from the main data dictionary in that they might support different 'views' of the database, or include additional information, or provide more detailed descriptions. The data administrator will need to manage these data dictionaries, ensuring that they are up to date and that the addition of new entries into a master dictionary is controlled carefully.

Users have considerable freedom to manipulate the data stored in their local data pools. For this reason, local data should not be transferred to the operational databases. Finally, professionally managed and derived data should be kept separate from data that is originated and controlled by users.

SUMMARY

So far in this report we have described the tools that are available for defining, creating, and maintaining corporate databases and for providing business users with direct access to the databases. To select and make the best use of these tools is not a trivial task — even if a database environment is being created for the first time. It is even more complicated when an existing database environment has to be taken into account as well. We now turn to the problems of migrating from existing first-generation data-management systems to a new environment based on advanced data dictionaries and relational databases.

Chapter 5

Planning a cost-effective migration path to new corporate databases

Most organisations will not start with a clean sheet when they begin to plan their move to second-generation databases. Instead, they need to create a migration path that will allow them to move from their existing applications portfolio and databases to a situation where databases and applications conform to a corporate data architecture of the type we described in Chapter 2. The migration plans must therefore allow for applications to be re-developed or converted, for descriptions of the data to be rationalised and documented in a data dictionary, and for the databases to be converted into a new format. The process of migrating to a new database environment thus requires substantial resources and it is important to select an approach that minimises the costs and the time required, and that causes the least disruption to existing systems. Many organisations will find that their ability to exploit the second generation of data-management systems is constrained by the resources they can allocate to the process of migrating to a new data architecture.

Despite the high costs of conversion, many organisations are now considering making major changes to their database environment. Our survey of Foundation members for this report found that the existing use of relational databases was low, but that many members planned to move an increasing proportion of their databases to relational systems over the next five years.

In this chapter, we identify the major problems that organisations will encounter as they carry out this conversion. We describe four methods for converting databases to second-generation data-management systems and we describe the two problems (and methods for solving them) that are common to all these conversion paths.

THE PROBLEMS OF DATABASE CONVERSION

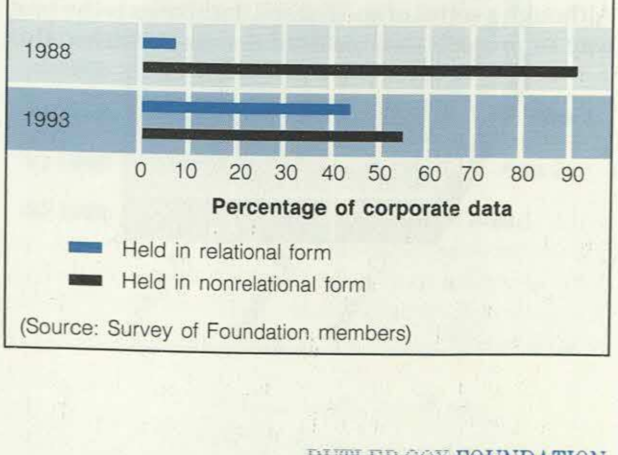
Figure 5.1 shows how members expect to move from holding between them just 8 per cent of their corporate data in relational form to 46 per cent in five years. Bearing in mind the massive investments that have been made in first-generation database systems, these plans indicate that systems departments will be faced with a huge task of converting applications and databases over the next five years.

To date, few major database conversion exercises have been carried out, so most organisations are not familiar with the problems associated with database conversion. The problems are much greater than those encountered when converting file-orientated batch applications because it is not possible to run the old and new systems in parallel. Often, there are no short cuts that can be taken when converting from a database environment (IMS, for example) to an incompatible environment such as DB2. The most usual problems occur in the following areas:

— A great deal of disruption is caused while moving from the old to the new environment. The applications for the new environment often cover a different range of functions from the old applications, which causes complicated interface problems during a phased conversion. The result is that both the old and the new databases need to be managed during the conversion process, and updates to them need to be synchronised.

— The databases being converted are normally used with online systems. As a result, the 'window' of time available for switching the operational systems from the old to the new

Figure 5.1 Foundation members plan a major move to relational databases during the next five years



(Source: Survey of Foundation members)

databases is normally small. Moreover, it is extremely difficult to phase in the new database systems gradually.

— The existing applications are often badly documented and the existing data descriptions are poorly defined. Sometimes, the only source of information about an application and the data it uses is the program source code and, often, this is almost impossible to convert. When one organisation was converting a large online application, an analyst who was already very familiar with the application took a year to decipher the meaning of the data by inspecting the program code.
— Major effort is required to translate and convert the old databases into the required new formats. Sometimes, conversion can only proceed over a protracted timescale because there are insufficient clerical resources available to speed-up the process.

During the research, we met with Mr Frans Blommers, data processing manager at Westland-Utrecht Hypotheebank, a Dutch company specialising in mortgage banking, real estate, and life insurance. Westland-Utrecht is one of the few organisations that has carried out a major database conversion exercise. This organisation uses Unisys/Burroughs computers, and its applications are based on a central database. There are about 250 terminals used with applications developed in third-generation languages to access four databases.

The plan was to implement the new systems over a period of time, during which elements of both the old and new systems would be operational. The objectives of the conversion were to develop a new, segmented data architecture, to provide better recovery facilities, to improve the maintainability of the systems, and to meet new business requirements.

The conversion took two years and required 18 work-years of effort. The new database systems required a fifth of all the existing applications to be rewritten. The new data architecture has a normalised structure and the subject-area databases have rationalised entity descriptions. All the applications now use DMS II, an advanced and flexible database management system. Westland-Utrecht considers that DMS II will enable it to cope with future changes in an evolutionary way and that no more large conversion exercises will be required.

CHOOSING A MIGRATION PATH

Organisations should choose their database migration path depending on the quality both of their existing application systems and of the conversion tools that are available. The quality of the applications affects the amount of work that is needed. In order to assess the quality of the existing applications, it is necessary to answer the following questions:

— Are the existing database systems based on a sound data architecture with suitable subject-area databases that match the organisational structure?
— Do the existing applications still meet the users' requirements, or are they now inappropriate in their scope or in the functions they support?
— Is the technical quality of the existing applications sound, or should they be rewritten?
— Are the existing data descriptions consistent and suitable for relational databases, or do they need to be redefined?

The answers to these questions will indicate the extent of the conversion work that must be undertaken.

In addition, the conversion tools that are available will, to some extent, govern the choice of migration path. Some relational database suppliers recognise the value of powerful database conversion tools and provide facilities for converting existing databases to their formats. (Examples of several products are quoted later in the chapter.)

There are four possible migration paths — to redesign and rewrite the applications completely, to convert the applications, to interface existing applications to new databases via database-transparency facilities, or to re-create existing applications if they were written with an application generator or Case tool that can be used with both the old and new database environments.

If the databases and the applications are not sound, then the only course of action is to redesign and rewrite the application systems. If the database has a sound architecture and the application programs are technically sound, then the systems department can convert the applications. If the only change needed is to move to a relational database, the database-software supplier may provide transparency facilities that will convert or interface existing applications to the new database with very little effort. Finally, if the existing applications make use of an application generator or a Case tool that can be used both with the old and new database management systems, the systems department can implement the new databases merely by re-creating existing applications. Figure 5.2 summarises the conditions under which each of these migration paths should be used.

REDESIGNING APPLICATIONS COMPLETELY

Organisations need to redesign and rewrite their applications completely if the applications no

Figure 5.2 Selecting a database migration path

Condition	Convert	Database(1) transparency	Re-create(2)	Redesign(3)
Sound data architecture	✓✓	✓✓	✓✓	
Applications meet user requirements	✓✓	✓✓	✓✓	
Good technical quality of applications		✓	✓✓	
Good quantity data descriptions		✓	✓✓	

Migration path

✓✓: Fully meets the condition

✓: Partially meets the condition

(1) Database transparency requires suitable software products to enable an existing application to access the new databases.

(2) Re-creation requires an applications generator that can re-create an existing application so it can be used with the new databases.

(3) If none of the conditions apply, the only choice is to redesign and rewrite the applications.

longer meet the business requirements or if changes in the business have made them obsolete. Because of the time and effort required for a complete rewrite, applications are usually redesigned for the new database management systems as they fall due for renewal. A complete rewrite of all the applications is an expensive undertaking, but it does have the advantage of producing a new architecture that is not compromised by the need for compatibility with the old structure.

Often, the lack of information about the existing applications leads the systems department to discard the existing applications and to redevelop them completely from the requirements-definition stage onwards. However, it should not be forgotten that the old applications are a valuable source of information about the real user requirements because they have been modified over a period of time to match the users' needs. There are considerable benefits in 'debriefing' the old applications rather than starting again from the beginning with a new analysis of requirements.

CONVERTING APPLICATIONS

If the data used in the existing systems is basically sound, if the database and applications architecture is robust and coherent, and if the applications broadly meet the users' requirements, then much of the existing applications logic can be converted for use with the new database system and the conversion effort can be reduced considerably. With this approach, it is possible to migrate gradually to new database management systems. There is less disruption to the existing applications and development resources are used more evenly.

The technique of 'systems re-engineering' (or 'reverse engineering') is often used when converting applications logic to a new database environment. Systems re-engineering uses an array of tools both to extract the underlying logical database structures from existing applications and to turn the old unstructured code into structured, documented, and maintainable code. The stages of systems re-engineering are:

- To stabilise the application to produce a solid basis for the changes. At this stage, the application is analysed and restructured using proprietary tools.
- To upgrade the quality of the application without changing its functionality. Large applications and databases are split into smaller and more manageable sections and changes are made to improve the operational performance of the applications.
- To make technical improvements, including changing the application so it can be used with a new database management system.
- To provide a series of releases of the application, each of which provides functional improvements to enhance the application.

Several systems re-engineering tools are becoming available. Their purpose is to assist development staff to analyse existing applications, to document them, and to improve their maintainability. For example, Bachman Information Systems provides two database analysis tools, Data Analyst and Database Administrator, that can extract the physical structure of an IMS database, modify it to meet new requirements, and generate a model of a DB2 database automatically. New databases can thus be built rapidly from existing applications. Bachman Information Systems plans to introduce in 1988/89 another set of integrated tools that will extract the underlying logic of a Cobol program and generate new code. Other database management systems suppliers provide facilities that can convert the descriptions and data of a DL/1 database into relational form. One example of such a tool is Cullinet's Escape.

In our research, we discussed systems re-engineering with a large systems house that had completed five re-engineering projects and that was currently working on a project which would cost up to $20 million to rewrite the complete systems using conventional development methods. This project involved converting a large order-processing application to new database technology using much of the existing applications logic. The application was fundamental to the client's business and it was important to convert it as quickly as possible. Rather than rewriting the application completely, the existing application was used to create a set of requirements so that a replica of the application could be built. The systems house believes that proceeding in this way is more likely to produce a system that matches the users' real requirements. To rewrite the system would have meant producing a new requirements definition, which would inevitably have been inaccurate or incomplete.

Working with the users, the conversion began by using Case tools to 'debrief' the existing application in order to construct a data model for the application and store it in a data dictionary. Existing applications are usually a valuable (and hopefully complete) source of current user requirements. By analysing the existing application in great detail, the systems house was able to benefit from the considerable maintenance and enhancement work that had been carried out since the application was originally implemented.

The application was then rebuilt using Cincom's Supra database and fourth-generation language, Mantis. This approach took considerably less time and cost several million dollars less than a complete rewrite would have done.

USING DATABASE-TRANSPARENCY SOFTWARE

The database-transparency technique allows existing applications to use a new relational database, either online or in batch mode. This technique is used as follows:

— Old database structures are converted into relational database tables and loaded into the relational database data dictionary.

— Programs are automatically developed to convert the old data to a relational format and load it into the relational database.

— At run time, database-access requests from the existing applications are intercepted by the database-transparency software and are converted to the equivalent relational database-access calls.

The database-transparency technique therefore allows systems departments to move old applications to the new database quickly, and avoids the effort and expense of supporting both the old and new database management systems. This approach is only practical when the old application is well-structured and the database contents are well-defined. Moreover, once the new database has been created, and its structure and contents have been described in a data dictionary, it is then possible to redevelop gradually the database structures to meet new business requirements and to remove inconsistencies in the database.

One well-established product of this type is ADR/DL1 Transparency, supplied by ADR Corporation, which also supplies the ADR/Datacom/DB relational database product. This software product creates an ADR/Datacom/DB relational database definition from DL/1 database definitions and also creates the entries in a data dictionary. Another program is then used to create the actual database automatically. At run time, the Intercept subsystem takes control of the DL/1 environment and intercepts and interprets the DL/1 calls for the converted database. ADR's transparency product supports mixed DL/1 and ADR databases. ADR also provides a VSAM transparency product. IBM plans to release a transparency product in the near future that will initially provide VSAM transparency.

Figure 5.3 summarises the database-transparency migration path.

RE-CREATING APPLICATIONS SO THEY CAN USE OLD AND NEW DATABASES

Organisations that have a large proportion of their applications written in very-high level languages (such as Natural and Powerhouse), or defined in system encyclopaedias generated by Case tools, can avoid many database-conversion problems by

Figure 5.3 The data-transparency migration path allows existing database applications to use the new databases

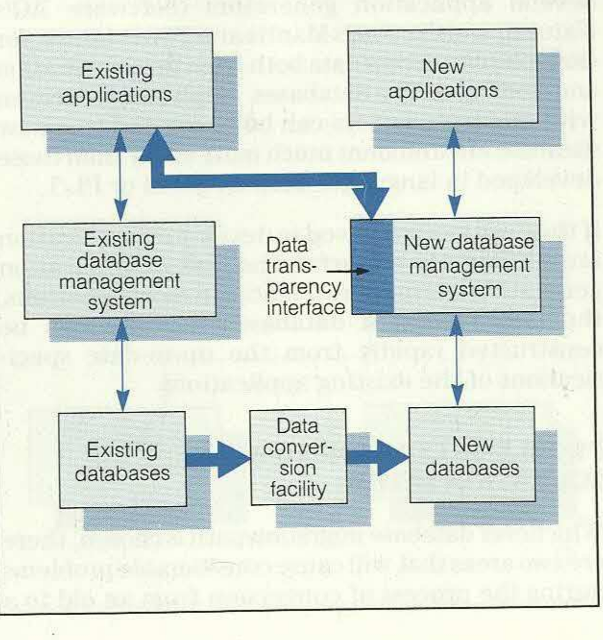
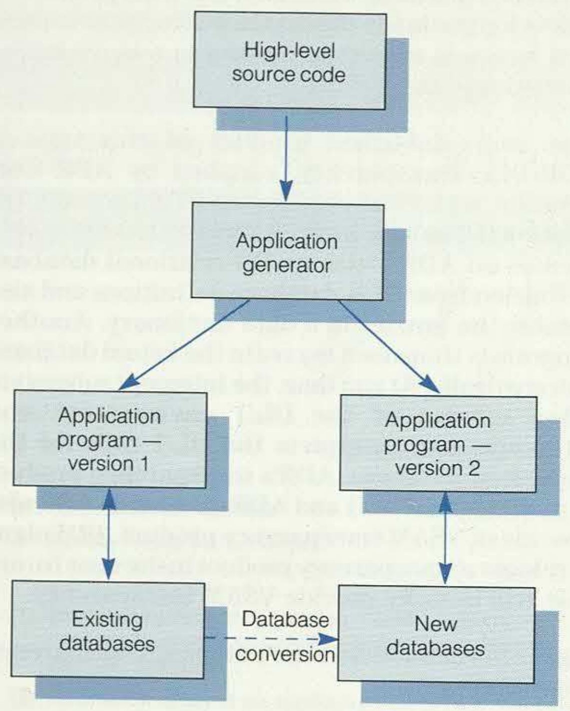


Figure 5.4 Some application generators allow existing applications to be re-created so they can be used with the new databases



exploiting the portability that these facilities provide. In effect, these facilities permit applications programmers to interface with databases at a logical level. At this level, the logical structure of the old and new databases should be the same, so the same applications logic can be used in both cases. This database migration path is depicted in Figure 5.4.

Several application generators (Software AG's Natural, and Cincom's Mantis and Powerhouse, for example) can access data both from first-generation and from relational databases. Applications written with these generators can be converted to a new database environment much more easily than those developed in languages such as Cobol or PL/1.

If the Case facilities used to develop an application are sufficiently powerful, the task of application generation is largely automatic. In most situations, the new relational database structure can be constructed rapidly from the up-to-date specifications of the existing applications.

PROBLEMS COMMON TO ALL MIGRATION PATHS

Whichever database migration path is chosen, there are two areas that will cause considerable problems during the process of conversion from an old to a new database environment. The first problem occurs because it is not possible to convert the whole applications portfolio in one go. The conversion exercise has to proceed on an application-by-application basis. In the interim stages, the old and new databases need to be linked so that the same data is stored in both of them.

The second problem concerns the conversion of the data itself. Often, the existing data is poorly defined and its accuracy may be doubtful. These problems sometimes mean that attempts to convert large databases containing several hundreds of megabytes have to be abandoned. The conversion effort cannot keep pace with the rate at which the database contents change.

Methods for dealing with these two problem areas are described below.

LINKING OLD AND NEW DATABASES

Usually, a new data architecture and applications portfolio does not match the existing applications on a one-to-one basis. Applications in the new portfolio may replace only parts of one or more existing applications. Existing 'monolithic' databases cause particular problems in this area. These are databases that have no detectable internal divisions because each part of the database is inextricably interwoven with the rest. Converting these databases is extremely difficult because bespoke interfaces between the old and the new databases are needed for each application. The representative of one Foundation member organisation that is tackling this problem described the process as being "Like slum clearance, but with several historic buildings that have to be preserved."

During most database-conversion exercises, there is a period when data is shared between both new and old applications. The systems department therefore has to maintain two separate databases containing the same data, with the attendant problems of ensuring that the two databases are kept in step with each other. The problem is even more acute when an online database is being converted because the old and new databases have to be synchronised exactly.

There are three ways to establish links between the old and new databases — by transferring batches of data between the database systems, by using distributed-database techniques, or by integrating the old applications and databases closely with the new relational database management system.

Products such as IBM's DXT can be used to extract batches of data from the IMS, DL/1 VSAM, DB2, and SQL/DS databases in a standard information exchange format (IXF). The extracted data can then be transferred to a new DB2 or SQL/DS database. The data definitions are extracted from Cobol or PL/1 source code or from IMS database descriptions. This is the most common method of linking old and new databases but it cannot synchronise the two databases completely because of the delays inherent in batch-processing techniques.

As we described in Chapter 3, distributed-database techniques can be used to link old and new databases. Changes made in one database can then be quickly passed to the other database. Few products are available to do this because the interfaces need to be tailored for each first-generation database management system. Tandem has a Remote Duplicate Database Facility (RDF) and DEC has a Vax to IBM Data Architecture (VIDA) product, both of which provide the basis for this method of linking old and new databases.

The third alternative is to link the databases using the inherent capabilities of relational databases. Most relational databases allow the logical database structure to be independent of the physical structure of the database. Relational database theory allows any physical database structure to be used to support the logical database structure. Cincom makes use of these relational database capabilities to allow its Supra product to access data held in Total and VSAM databases. Cincom considers that a true relational database should be able to map any physical data structure onto the logical database structure. It claims that its products can achieve this, even to the extent of resolving the problems of duplicate physical data items held in different physical file structures. In Supra, a logical database can be superimposed over a set of different physical database and file-management systems. Any duplication of data is handled by the relational database management system and by the integrated data dictionary. This method is known as the 'native-mode' approach to database conversion and is illustrated in Figure 5.5. At present, Supra can convert Total and VSAM files and Cincom expects to release a version of the product during 1988 that can handle 80 per cent of the structures used in a DL/1 database.

CONVERTING THE DATA TO RELATIONAL FORM

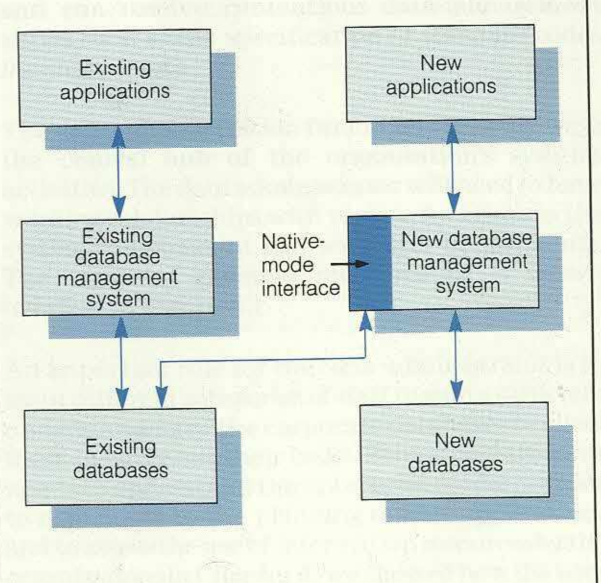
Before data can be loaded into a relational database it must first be converted into a suitable relational form. Data in this form is known as 'normalised' data. Old applications will usually contain a great deal of data that is not normalised, as well as data with other problems. The database has to be converted to a relational form to achieve the full benefits of a flexible relational database. Typical problems with existing data formats are:

- Several coding schemes may have been used for the same data item. For example, the sales and production departments may use different product codes. A single code for use by everyone must be agreed and the old codes need to be mapped to the new codes.
- Some data items in the old applications may not be held in a coded form but stored as unvalidated and unstructured text. They cannot be used by the new applications in this form. These data items have to be coded by hand from the text descriptions before they can be loaded into the new database.
- Existing data items may be used to store more than one fact. For example, a customer number may also indicate the customer's location. These facts have to be held as separate data items before they can be loaded into the relational database.
- Some facts are implied by the position of a field in the record. For example, monthly data may be held in a set of 12 fields. The position of the field in the set indicates the particular month. Implicit facts such as these have to be described explicitly in the new relational database.

Overcoming these problems may mean that it is necessary to write special programs to clean-up and reformat the data so that it corresponds with the new data descriptions. Converting operational data can be particularly difficult if it requires human intervention to translate between the old and new data descriptions.

We know of a large insurance company that made use of an expert system as it converted its major systems to a new database environment. The expert

Figure 5.5 A native-mode interface allows new database applications to work with existing databases



system was able to use a set of rules to read previously stored text and to decide how to code it so it could be stored as data. The result was by no means perfect and still had to be checked by clerical staff, but this method was very much faster than the alternative of manually converting all the data.

SUMMARY

In this chapter we have described the four possible paths for migrating from an existing database environment to a new relational database environment and have provided advice on when to use each path. We have also highlighted the problems that will be encountered in moving to new databases and have shown how these problems can be solved. The solutions to the problems will require a strong data-administration function, however. The next, and final, chapter of the report describes the role of the data administrator in the evolution of corporate databases.

Chapter 6

Extending the role of data administration

The process of moving to a new corporate database environment is not a single event. Instead, it is a continuous activity during which the databases are developed to match business requirements and to satisfy business information needs. Moreover, the databases continue to change in response to changing business requirements. In order to manage this process, a powerful and independent data-administration function is required to control the construction of the corporate databases and to educate the organisation in their use.

Organisations are becoming increasingly dependent on information. As they strive to obtain a competitive advantage, they are recognising the need to manage all their information resources — not just those stored in corporate databases. The data administrator will therefore take on further responsibilities in the future, particularly for the management of text and other media and the control of information flows to and from the organisation.

However, the appointment of a data administrator does not in itself solve the problems of data management. Many data administrators are still grappling with the internal problems of data ownership and of data administration.

In this chapter, we describe the current data-administration role and the problems faced by data administrators today. We conclude the chapter, and the report, by identifying the new data-administration tasks that will be necessary as organisations evolve their databases to a new, relational, environment.

THE CURRENT DATA-ADMINISTRATION ROLE

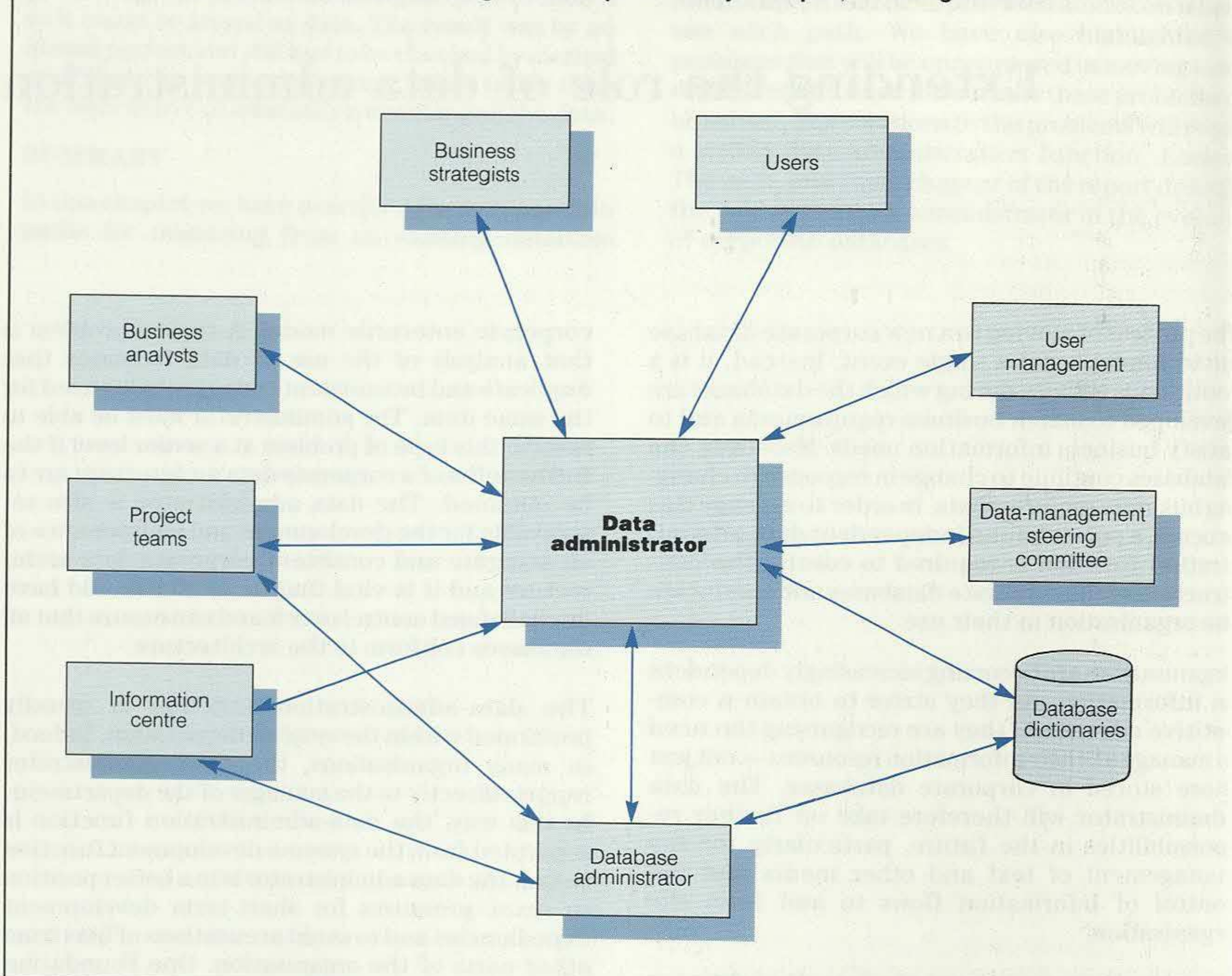
Today, the data administrator has two key tasks — to control the corporate databases at the strategic-planning level, and to educate the organisation about using and managing the databases. To carry out these tasks effectively, the data-administration function must be positioned at a high level in the reporting structure. If it is positioned at too low a level, the data administrator will not be in a position to resolve the strategic issues that are likely to arise during the evolution of the corporate databases, particularly during the construction of the corporate enterprise model. A typical problem is that analysis of the use of data indicates that duplicate and inconsistent codes are being used for the same item. The administrator must be able to resolve this type of problem at a senior level if the full benefits of a corporate data architecture are to be obtained. The data administrator is also responsible for the development and maintenance of an accurate and consistent corporate data architecture and it is vital that he or she should have unchallenged control over it and can ensure that all databases conform to the architecture.

The data-administration function is usually positioned within the systems department. Indeed, in many organisations, the data administrator reports directly to the manager of the department. In this way, the data-administration function is separated from the systems-development function so that the data administrator is in a better position to resist pressures for short-term development expediencies and to avoid accusations of bias from other parts of the organisation. One Foundation member ensured that the data-administration function has sufficient seniority by establishing a data-management steering group that is made up of senior managers from user departments. This group can direct the work of the data administrator and can resolve contentious data-management issues such as the specification of standard codes for shared data.

The data-administration function is in many ways the central hub of the organisation's systems activities. The data administrator will need to have working relationships with various functions in the systems department and with user departments. The range of possible relationships is shown overleaf in Figure 6.1.

An important role for the data administrator is to assist different categories of staff to gain a sufficient understanding of the corporate databases to allow them to carry out their tasks. Senior management needs to understand the enterprise model in order to contribute to the planning of new applications and to assess the use of information resources by the organisation. In Chapter 4, we showed how the user community needs to understand the organisation's

Figure 6.1 The data administrator has working relationships with many other parts of the organisation



data in order to extract for itself the data required for management-information and decision-support purposes.

Educating the systems department is an equally important task. Unless the majority of development staff have been trained in how to use data-management tools and techniques, it will not be possible to establish and maintain a data-oriented culture in the systems-development function. Several of the organisations we met during the research assign data-administration staff to work alongside the application-development staff during the early stages of a development project. This has two advantages — the quality of the work is improved because the data-administration staff provide a high level of expertise, and the project team is given practical training in the use of data-management techniques.

CURRENT DATA-ADMINISTRATION PROBLEMS

There are two particular areas of data administration that often cause problems — resolving data-ownership problems and managing data in large organisations.

RESOLVING DATA-OWNERSHIP PROBLEMS

In business, as in many other walks of life, the possession of information confers an advantage on its owner. As a consequence, some of the most contentious issues for the data-administration function arise because of the difficulties of convincing people that 'their' data belongs to the organisation and should be shared with others. Nevertheless it is vital to establish which department or individual 'owns' shared data items. Many organisations have found that the accuracy of shared data (and hence the quality of the databases) improves once the ownership of the data has been established.

Data-ownership responsibilities should not be seen as residing either with the systems department or with the user community. Rather, it is a question of assigning the responsibilities to the most suitable user departments. However, the data administrator should act as the overall custodian or 'guardian' of the organisation's data. He or she will be responsible for policing the data-sharing policy and for ensuring that the data owner is carrying out the agreed security arrangements.

The problems that can be caused by data-ownership disputes are illustrated by the experiences of a mineral-extraction company that had two points of contact with its customers. In this organisation, the sales department handled sales negotiations and the technical department dealt with technical queries that often resulted in sales. Each department maintained its own database of customer and sales information. The databases, which were incompatible with each other, were regarded as the personal property of each department. Attempts to develop a common database were frustrated by their unwillingness to share data, and, as a result, opportunities to exploit all the information about a particular customer were lost. In the end, the solution adopted was to nominate one of the departments as the lead department (and therefore as the data owner) and to encourage the other department to convert their applications and merge their data with that of the lead department.

Disputes over data ownership need firm handling and high-level management support if they are to be resolved successfully. Such disputes are much easier to resolve if the data administrator can apply sanctions against the departments that are being awkward.

DATA-MANAGEMENT PROBLEMS IN LARGE ORGANISATIONS

Corporate-wide data management is less common in large organisations that are composed of several separate trading divisions. There are four main reasons why corporate-wide data management is difficult to achieve in this situation:

— Corporate policies may make it difficult for the divisions to cooperate with each other.

— Each division may wish to own and manage its own data.

— Such organisations may consider they are too large for a centralised data-management function to be effective.

— They may believe they cannot derive any benefits from a centralised approach.

However, if the divisions have integrated product lines or share customers and markets, even the largest of organisations can benefit from a corporate approach to data management. In our research, we talked with several large organisations from a wide range of industries that were investing considerable effort in achieving corporate-wide data management.

For data management to be effective in a large organisation, a more complex data-administration function is required than in other types of enterprise. The number of data items in use in all the parts of such organisations is too great for a single administrator to manage. Furthermore, the workload for a single data administrator would cause a severe bottleneck in the systems-development process. The solution is to distinguish between corporate, shared data and data that is of interest only to a single division. Only a small proportion of the data will need to be shared across the whole organisation. Each division should have its own data administrator responsible for designing and managing divisional databases that conform to the data-management standards set by the corporate data administrator. These standards include the definitions of the codes, data descriptions, and meanings of the small proportion of data items that are shared between the divisions.

An example of corporate-wide data management in a large organisation is provided by a manufacturing company. This company has established a European data-administration function whose role is to manage the high-level business (enterprise) model. Each division of the company has a data manager who is responsible for producing the detailed data model for his or her division. However, the divisional data managers are also responsible for ensuring that their data models conform to the definitions of the shared entities. To help them in this task, they have online access to the European data administrator's data dictionary.

This approach has produced benefits by standardising the key shared data items and by providing a clear definition of the scope of divisional applications and of the interfaces both between applications and between the divisions. The latter definitions are particularly important to this highly integrated organisation. An immediate benefit is the ability to shorten the analysis time required for a new application because the relevant subset of the corporate data dictionary can be extracted for use by the application analysts.

NEW DATA-ADMINISTRATION TASKS

We said at the beginning of this chapter that data administrators will increasingly be called on to manage text and image data and to provide the basis for integrating all information. The data administrator will also be required to manage the use of external data and, in the future, may well become the custodian of knowledge bases. The appendix, which discusses the likely future developments in database management systems, shows that today's databases are likely to evolve gradually into the knowledge bases that will form the basis for expert-system applications. In parallel with these developments, the data-administrator's role will evolve into that of a knowledge administrator. The knowledge-administration role will include the management and integration of knowledge bases throughout the organisation. The traditional skills of data administration will be used to establish the scope of the individual knowledge bases, to establish standards, and to ensure that they can eventually be integrated both with each other and with the corporate databases.

MANAGING TEXT AND OTHER FORMS OF INFORMATION

The data administrator will need to manage text databases and data stored in other forms such as graphics and images. Without widening the scope of data administration in this way, it will be difficult to avoid the creation of 'islands' of information and it will be more difficult to plan for the eventual convergence of all of the forms of information into one storage and delivery system. Many organisations are developing text-based applications in response to the demand for hybrid systems (which contain elements of hitherto separate office systems, data processing systems, and telecommunications systems) and the ability of disc technology to store the volumes of text required. The rationale for establishing an overall 'information-administration' function mirrors that for establishing a single communications-management function.

In managing text databases, however, it is important to realise that data and text differ in several significant respects. Figure 6.2 shows the main differences. These differences mean that text management is not merely an extension to data management, but requires new skills and new technology.

Before making text databases available, the data administrator must ensure that they are structured in a format suitable for their users. The ease with which a text database is manipulated and interrogated depends on the degree of structure in the text. Someone who is not used to accessing a text database finds it difficult to search an unstructured text database thoroughly. For this reason, intermediaries such as information specialists are often used to conduct these searches. A computer database, on the other hand, has a very well-defined structure because it must be 'understood' by application programs as well as by people. Between these two extremes, there is a range of options for the management of data and text databases.

Figure 6.2 Main differences between data and text storage

Data	Text
Record oriented	Document oriented
High level of change	Low level of change
Complex relationships between records	Usually simple hierarchical relationships between documents
Automated retrieval	Human intervention required for retrieval
Precise search criteria	'Fuzzy' search criteria
Concurrent access and update	Single-user access and update
Many records with the same structure	Each document has a unique structure

There are considerable differences between the types of text-management system that are likely to be used. The main types are systems that embed text fields in conventional databases, free-text retrieval systems, and document-storage systems. Whichever types of text-management system are used, the data administrator needs to establish policies and standards that will ensure they are as consistent and meaningful as conventional databases. Thus, the data administrator will have to establish a consistent corporate document-naming and keyword system that enables documents to be indexed and retrieved easily by users. In addition, where separate data and text databases exist, the data administrator will need to establish and manage the links between the databases and the text documents.

Embedded text fields in database records

Often there is a need to hold unformatted (free) text fields within a conventional database record and to access them either by database-retrieval methods or by searching the text itself. At present, most database management systems provide poor facilities for integrating data and text. A typical problem is the lack of word processing facilities for editing text fields in a database. Oracle is planning to introduce more facilities in its database management system for supporting mixed text and data processing applications. These facilities will include enhancements to the SQL commands, to be known as SQL Text, that will enable text fields embedded in a conventional database to be searched in the same way as a free-text-retrieval system.

Free-text-retrieval systems

Free-text-retrieval systems are the best choice when users need to examine the contents of a document in great depth and retrieve parts of the document. These types of system are expensive to run because they require a great deal of computing resources to construct the indexes and to search the database, and they are much more difficult to integrate with a conventional database. Furthermore, specialist skills are required to construct the thesaurus required to make free-text retrieval systems useful.

Document-storage systems

For large documents containing 100 pages or so, it is usually sufficient to identify and retrieve them by their document identifier or a set of keywords associated with each document. There is usually no requirement to search the text within such documents. Document-storage systems are usually part of the office-system infrastructure and are often difficult to integrate with database management software. However, the need for integration will become more important as hybrid systems consisting of linked data and text-processing applications are developed. A product from Software AG (called Con-nect) provides a way to handle this integration. Con-nect has electronic-mail and text-processing components that can be accessed by data processing applications written in Software AG's high-level application generator, Natural.

MANAGING THE USE OF EXTERNAL INFORMATION

The data administrator should also manage both the use that is made of external data resources and the release of the organisation's data to the outside world. Organisations are becoming increasingly interdependent and are, as a result, exchanging ever-increasing volumes of data in a machine-readable form. They are also becoming increasingly aware that their public image can be affected by the quality of the data provided to the world at large. The data administrator should therefore also be responsible for ensuring that all external data transactions are consistent and accurate. Typical examples of such external transactions include electronic data interchange (EDI) transactions, access to public databases, use of external information services such as stock-exchange prices, and the use of interorganisation networks.

The data administrator is well placed to manage the organisation's use of external data. He or she needs to ensure the quality, the privacy, and the security of data published by the organisation, and to assess (and, if necessary, negotiate) the quality of data being received. These tasks increasingly will affect the competitive position of the organisation. For example, in negotiations about common industry formats, the data administrator should take care not to agree to arrangements that will place the organisation at a commercial disadvantage. The data administrator will also be in the best position to ensure that the requirements of the relevant data-protection legislation are complied with.

REPORT CONCLUSION

This report began with the premise that during the next five years, most Foundation members will be moving away from first-generation databases (such as IMS and IDMS) to second-generation relational systems. In the report, we have provided advice about how to manage the evolution that will occur. In particular, we have highlighted the need to define an overall data architecture — the framework within which the new databases will be designed and used.

We have also provided advice about how to select and use the three main types of data-management systems — data dictionaries, which will play an increasingly important role both within the systems department and as a means of providing users with direct access to databases; relational databases, whose operational performance can now match that of first-generation databases; and distributed databases, which although they are still relatively immature, can provide technical benefits.

An increasingly important element of the database environment will be the provision of tools that enable users to access databases directly. The report has described the tools that are (and will be) available and has provided advice about when and how to use them.

The major difficulty for many organisations in moving to second-generation data-management systems is planning how to convert from their existing database environment. Both the old and the new environments will need to operate in parallel during the conversion period. We have identified the possible migration paths and provided advice about when each is suitable.

Finally, the report has re-emphasised the role of the data administrator. The potential benefits of the new data-management systems will be put at risk if there is not a strong data-administration function charged with the responsibility of creating the corporate data architecture and ensuring that the new databases and systems conform with the architecture.

Appendix
Future directions in database management systems

Several significant developments are expected to take place in database management systems in the next few years. These developments will not supersede relational concepts. Rather they will build on the firm foundation that relational concepts have provided. The main area of development will result from the considerable research effort into the relationships between databases, knowledge bases, and object-oriented systems. We expect this effort to lead to new, advanced database systems that will dramatically improve development productivity and that will provide powerful and much easier-to-use database-access facilities.

In particular, the research is leading towards the production of a unified concept of information and knowledge management. We discuss these developments under the headings of capturing more meaning in the data model, and the increasing overlap between knowledge-base and database concepts.

CAPTURING MORE MEANING IN THE DATA MODEL

The purpose of corporate databases is to store and make available information about business activities. In other words, databases model the activities. However, most existing database management systems do not capture all the information about a business activity to which a computer application needs to have access. Relational databases are often the most deficient in this respect because many of them do not capture and enforce referential-integrity rules. But even relational databases that provide referential-integrity features are not capable of capturing all the details of a business activity because the integrity rules describe only the basic relationships between entities.

For example, a database may store the fact that the price of a given product is $30. In addition, the database structure may contain the fact that every product must have a price (and this condition may well be enforced by the referential-integrity rules). However, the fact that each product price must be more than $20 may be held only in the application-program logic or even in the heads of the users accessing the database. This extra information about the business activity is known as 'semantic' data because it relates to the meaning of the data.

Semantic databases aim to capture this additional information about the meaning of data. They have evolved from research work on general database theory, on artificial intelligence, and on programming languages. Some of the concepts of semantic databases come from work in the field of linguistics, particularly about how people translate perceptions into language. Relationships between entities in a semantic database are expressed in more detail than in previous types of data models. In a semantic data model, integrity rules are held within the database and are not provided by individual applications and queries. Semantic data models are, in effect, a superset of the relational data model. As more meaning is captured in a database management system, the dictionary, which stores all of this meaning, becomes the most important aspect of the software, and the relational database becomes merely the engine that drives the mechanism for providing access to the data.

Although many suppliers consider these concepts too advanced to be of practical application, a few are beginning to introduce them. Cincom, for example, claims to "incorporate many of the principles of semantics as discussed in the extended relational model" in its Supra database product. And Software AG plans to support the entity-relationship model developed by Chen, another leading database researcher, in its Adabas database product.

The most significant advance in this area so far is the Semantic Information Manager (SIM) developed by Unisys. This database management system is based on the work of Michael Hammer and Dennis Mcleod (who have published several papers on semantic database theory). Unisys claims that SIM is the first commercially available implementation of a semantic data model. SIM is one element of Unisys's Information Executive (InfoExec) Series of database-management products. In addition to SIM, InfoExec includes:

— ADDS (Advanced Data Dictionary System).

- OCM (Operations Control Manager).
- IQF (Interactive Query Facility), which supports ad hoc requests within the InfoExec environment.
- WQF (Workstation Query Facility), an end-user database-access tool.

All of these products rely on SIM for information. The InfoExec environment is the umbrella for all the products that operate with SIM.

SIM has an advanced three-level architecture comprising an external level, a conceptual level, and a physical level. The InfoExec environment provides the external level. SIM itself is the conceptual level that manages the formal description of the data and the semantics. The physical level is based on an advanced version of the well-established DMS II database system. The data administrator uses the ADDS facility to define a SIM database, and the physical database structures are generated from this description of the database.

These types of database differ from relational databases because they are integrated closely with a more powerful data dictionary system that controls all aspects of the database and its semantic content. All users (including the data administrator and application programmers) access the database via the data dictionary. We believe that semantic databases will be introduced in the future by many database suppliers in order to simplify user access to databases and to reduce the problems of controlling the integrity of corporate databases.

CONVERGENCE OF DATABASE AND KNOWLEDGE-BASE CONCEPTS

There has been much recent activity in investigating the potential for merging the separate concepts of knowledge bases, databases, and object-oriented systems. The primary purpose of knowledge bases and databases is the same because they both aim to model real-world activities. However, knowledge bases include rules for reasoning with the facts they store, whereas databases simply reflect their designers' views of the world. Both would benefit from a common set of concepts. Databases would gain very sophisticated semantic modelling facilities of the type described above and knowledge bases would gain by better operational performance, by better facilities for controlling the information they held, by improved security features, and by enhanced features for data-storage control.

Object-orientated systems show great promise as a means of handling data that does not fit easily into conventional item definitions. In data processing, objects are character strings, integers, floating-point numbers, and possibly dates, times, and money. But, in its widest sense, data includes documents, representations of three-dimensional objects, bit maps corresponding to pictures, arrays of scientific data, and so on. As databases are used with more multimedia applications, they will need object-management facilities to handle all forms of data.

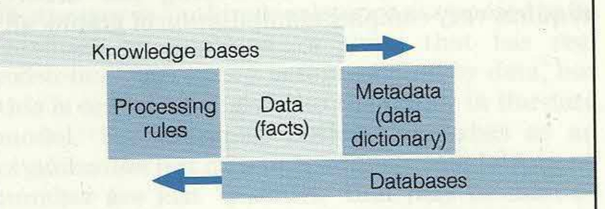
KNOWLEDGE BASES

The differences and overlap between knowledge bases and databases are shown in Figure A.1. Knowledge bases hold much information about the data and its meaning but the information is poorly structured. In fact, the structure of a knowledge base is defined mainly by the processing rules. Databases, on the other hand, have a well-defined structure but are deficient in their ability to store the processing rules or semantics of the business activity being modelled. In the foreseeable future, we expect knowledge bases and databases to migrate towards providing similar capabilities. Database will be enhanced by the addition of more meaning about the data, and knowledge bases will be enhanced by the development of more formal knowledge-base structures.

A forerunner of the future hybrid database/knowledge-base system is The Fact System developed by a small UK company, Deductive Systems. The Fact System allows users to exploit knowledge-base techniques in manipulating relationally based data and enables them to set up 'fuzzy' information about the structure of the data in the database and to use that information in manipulating the data. The Fact System can handle both expert-system and database constructs within the same environment. Expertise (in the form of rules) is organised into rule sets. These are held within, and operate on, the database.

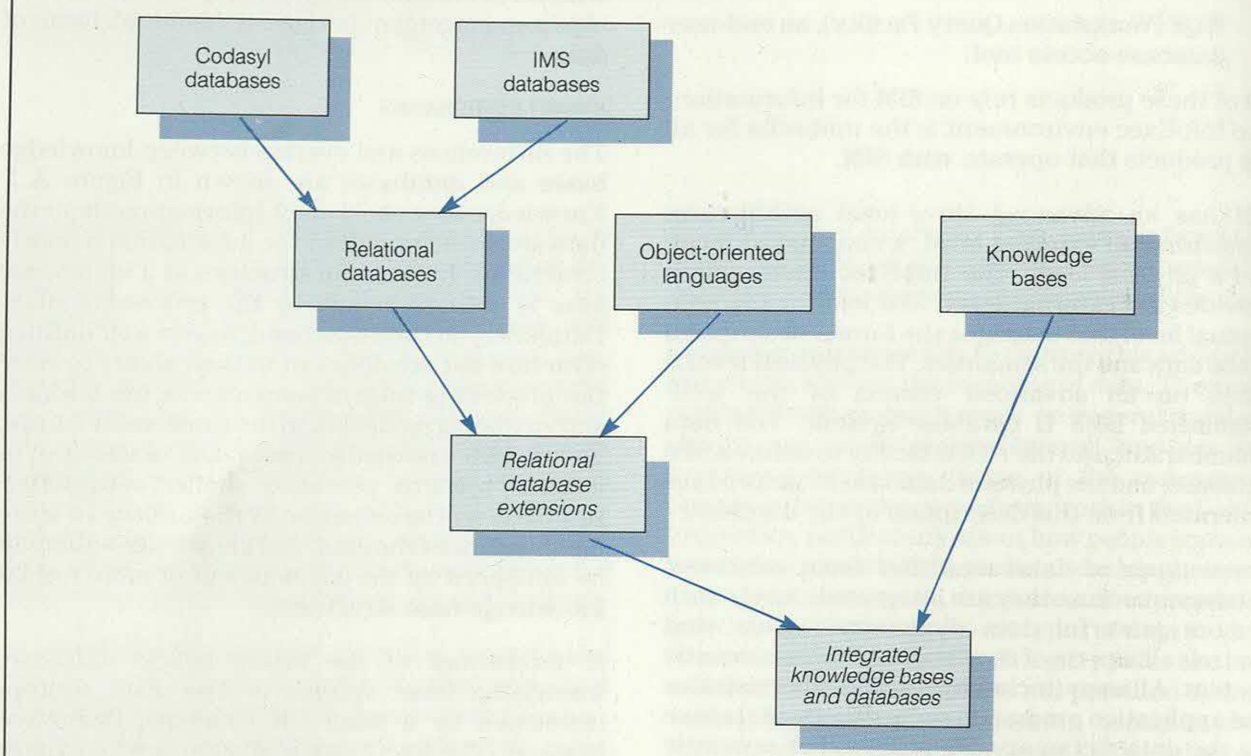
Users of the Fact System can embed highly complex rules within the database either as inference rules that describe the structure of the database, or as action rules that are triggered by changes to values within the database or by other external events

Figure A.1 The differences and overlap between knowledge bases and databases



A knowledge base is primarily concerned with rules and facts. A database is concerned with data and the structure of data.

Figure A.2 Databases, object-oriented concepts, and knowledge bases will ultimately converge to create a single store of knowledge and data



such as the current date and time. Thus, the database can support active models of the database environment rather than act only as a passive store of information. For example, a database that contains action rules could automatically notify a user when a credit balance exceeds its limit or when a payment becomes overdue. It is no longer necessary to write special programs to provide these types of report.

OBJECT-ORIENTED SYSTEMS

Object-oriented systems handle all types of objects in a consistent way because the procedures used to manipulate an object of a particular type are encapsulated with the object. For example, a map of Europe could be held as an object and interrogated with the query "Which rivers flow into the Mediterranean?" Producing the answer requires very complex manipulation of graphically stored information, yet the answer is a set of simple names.

A well-ordered hierarchy of objects can be constructed where sets of objects of the same type can inherit some or all of the characteristics and procedures of a parent class of objects. These concepts have been used to produce powerful and easy-to-use systems.

The ideas of encapsulation and of inheritance in object-oriented systems closely match the concepts of data classes in semantic databases. In the next 10 years, we expect the use of these concepts to develop as multimedia database systems, expert systems, and user access to databases become more common. Figure A.2 shows how these technologies are likely to converge to produce integrated knowledge-base and database management systems.

Glossary of terms

This glossary describes as simply as possible some of the basic concepts of data management.

Attributes and attribute types
An attribute is described in the Concise Oxford Dictionary as "a quality ascribed to anything." In data modelling, an attribute is used to describe the characteristics of the entities. For a building, an attribute may be an address — 12 Bloomsbury Square, for example. An attribute type is the set of all attributes of a particular type. In the above example, the attribute type would be 'address'. An attribute appears in a database as a field or data item, and the attribute type appears as the field name or data-item name.

Currency
The currency of a database is a measure of the degree to which information held in the database reflects the current situation in the real world. No database can exactly match the real world, and the extent to which the database is out of step with the real world needs to be controlled carefully.

Data analysis
The activity of identifying the entities that are of interest to an organisation and describing them, their attributes, and their relationships in a data model.

Data architecture
A data architecture comprises the high-level corporate enterprise model, which is subdivided into subject-area data models corresponding to important business activities. The data models are then further refined to produce the logical design of the databases that will be implemented. There may well be more than one logical database for each subject-area data model.

Data dictionary
At the highest level, a data dictionary holds information about entities (and their attributes) and the processes that use them. At the programming level, a dictionary stores the names of data items, their formats, and the rules governing their content. At the database level, it stores the descriptions of the logical and physical database structure.

Data or entity model
A data (or entity) model provides a description of the physical, financial, and human resources of the organisation. It describes the most important relationships between the major entities of interest to the organisation. It is important to note that data modelling is concerned with building an accurate picture of the real world that can be used to develop information systems that track the status of real-world events and answer a range of questions about them. Thus, two analysts independently addressing the same area of business activity must arrive at the same data model. A data model is also known as a Bachman diagram or as an entity model.

Database
A structured store of data; that is, a store containing data held in a structure that allows it to be retrieved using the contents of the data and its relationships.

Database management system
A set of software facilities to create, access, and control a database.

Database system
An application system comprising clerical procedures, application programs, and databases.

Data-management systems
Software facilities for the management of data. These includes data dictionaries, database management systems, and distributed database management systems.

Enterprise model
The high-level data model of the major business entities in an organisation.

Entities and entity types
An entity is described in the Concise Oxford Dictionary as "a thing's existence as opposed to its qualities or relations.; a thing that has real existence." An entity is represented by data, but this is only a label given to the entity in the data model. For example, Butler Cox exists as an organisation but its name, address, and telephone number are just 'qualities' that may be used to identify the organisation. Sets or classes of entities are called entity types. An entity type covers all the entities relevant to the enterprise that fit into a

given definition. Thus, the entity 'Butler Cox' is an instance of the entity type 'company'. In a database, entities usually appear as individual records or relations, and entity types appear as the record name.

Integrity
The term integrity is used to describe the measures taken to ensure the correctness of a database. These measures ensure that the database structure always reflects the structure of the data model used to design it, and that the data it contains conforms to the descriptions in the data dictionary.

Normalisation
A data model and a relational database should contain data that has been 'normalised'. The process of normalisation occurs during data analysis and is used to rationalise data into its simplest form. In this form, each data item contains only one fact, and each data item is directly related to the record it is in. Normalised data can be handled in a much more flexible way than unnormalised data.

Relational database
The theory of relational databases was invented by Ted Codd. He subsequently felt there was a need to define a relational database more explicitly and published 12 rules that define the functions and capabilities of a relational database. The most important is rule 0, which states that a relational database should be entirely managed through its relational capabilities. The other rules are rather technical and they describe all aspects of a relational database, including data independence, database integrity, database-cataloguing facilities, and database-manipulation facilities.

Security
Security measures ensure that the database is protected from unauthorised access and accidental or intentional harm.

SQL
SQL is the acronym for structured query language. It is a misleading term because SQL is not a structured language as is, for example, Algol. SQL is not only used for queries but also for updating and manipulating databases, and for other functions. It is, in fact, only a sublanguage because it cannot be used in isolation to develop programs. SQL is always used in conjunction with other languages such as Cobol.

Bibliography

Boisot M. Information and organisations: the manager as anthropologist. Fontana, 1987.

Codd E F. Extending the database relational model to capture more meaning. ACM Transactions on Database Systems, Volume 4, Number 4, December 1979.

Date C J. A guide to SQL standard. Addison-Wesley, 1987.

Hammer M, Mcleod D. Database description with SDM: a semantic database model. ACM Transactions on Database Systems, Volume 6, Number 3, September 1981.

Hammer M, Mcleod D. On the architecture of database management systems. Infotech state-of-the-art report on data design. Pergamon Infotech, 1980.

Martin J. End user's guide to database. Prentice Hall, 1981.

Rock-Evans, R. Data Analysis. IPC Business Press, 1981.

Stonebraker M, Rowe L A (editors). The Postgres papers. Memo Number UCB/ERL M86/85. University of California, Berkeley, 5 November 1986.

BUTLER COX FOUNDATION

Butler Cox
Butler Cox is an independent management consultancy and research organisation, specialising in the application of information technology within commerce, government, and industry. The company offers a wide range of services both to suppliers and users of this technology. The Butler Cox Foundation is a service operated by Butler Cox on behalf of subscribing members.

Objectives of the Foundation
The Butler Cox Foundation sets out to study on behalf of subscribing members the opportunities and possible threats arising from developments in the field of information systems.

The Foundation not only provides access to an extensive and coherent programme of continuous research, it also provides an opportunity for widespread exchange of experience and views between its members.

Membership of the Foundation
The majority of organisations participating in the Butler Cox Foundation are large organisations seeking to exploit to the full the most recent developments in information systems technology. An important minority of the membership is formed by suppliers of the technology. The membership is international, with participants from Australia, Belgium, France, Germany, Italy, the Netherlands, Sweden, Switzerland, the United Kingdom, and elsewhere.

The Foundation research programme
The research programme is planned jointly by Butler Cox and by the member organisations. Half of the research topics are selected by Butler Cox and half by preferences expressed by the membership. Each year a shortlist of topics is circulated for consideration by the members. Member organisations rank the topics according to their own requirements and as a result of this process, members' preferences are determined.

Before each research project starts there is a further opportunity for members to influence the direction of the research. A detailed description of the project defining its scope and the issues to be addressed is sent to all members for comment.

The report series
The Foundation publishes six reports each year. The reports are intended to be read primarily by senior and middle managers who are concerned with the planning of information systems. They are, however, written in a style that makes them suitable to be read both by line managers and functional managers. The reports concentrate on defining key management issues and on offering advice and guidance on how and when to address those issues.

Selected reports
8 Project Management
20 The Interface Between People and Equipment
21 Corporate Communications Networks
22 Applications Packages
23 Communicating Terminals
24 Investment in Systems
25 System Development Methods
26 Trends in Voice Communication Systems
27 Developments in Videotex
28 User Experience with Data Networks
29 Implementing Office Systems
30 End-User Computing
31 A Director's Guide to Information Technology
32 Data Management
33 Managing Operational Computer Services
34 Strategic Systems Planning
35 Multifunction Equipment
36 Cost-effective Systems Development and Maintenance
37 Expert Systems
38 Selecting Local Network Facilities
39 Trends in Information Technology
40 Presenting Information to Managers
41 Managing the Human Aspects of Change
42 Value Added Network Services
43 Managing the Microcomputer in Business
44 Office Systems: Applications and Organisational Impact
45 Building Quality Systems
46 Network Architectures for Interconnecting Systems
47 The Effective Use of System Building Tools
48 Measuring the Performance of the Information Systems Function
49 Developing and Implementing a Systems Strategy
50 Unlocking the Corporate Data Resource
51 Threats to Computer Systems
52 Organising the Systems Department
53 Using Information Technology to Improve Decision Making
54 Integrated Networks
55 Planning the Corporate Data Centre
56 The Impact of Information Technology on Corporate Organisation Structure
57 Using System Development Methods
58 Senior Management IT Education
59 Electronic Data Interchange
60 Expert Systems in Business
61 Competitive-Edge Applications: Myths and Reality
62 Communications Infrastructure for Buildings
63 The Future of the Personal Workstation

Forthcoming reports
Network Management
The Marketing of the Systems Function
Computer-Aided Systems Engineering (Case)
Mobile Communications
Software Strategy

Availability of reports
Members of the Butler Cox Foundation receive three copies of each report upon publication; additional copies and copies of earlier reports may be purchased by members from Butler Cox.

BUTLER COX FOUNDATION

Butler Cox & Partners Limited
Butler Cox House, 12 Bloomsbury Square,
London WC1A 2LL, England
☎ (01) 831 0101, Telex 8813717 BUTCOX G
Fax (01) 831 6250

Belgium and the Netherlands
Butler Cox BV
Burg Hogguerstraat 791c,
1064 EB Amsterdam
☎ (020) 139955, Fax (020) 131157

France
Butler Cox SARL
Tour Akzo, 164 Rue Ambroise Croizat,
93204 St Denis-Cédex 1, France
☎ (1) 48.20.61.64, Télécopieur (1) 48.20.72.58

Germany (FR)
Butler Cox GmbH
Richard-Wagner-Str. 13,
8000 München 2
☎ (089) 5 23 40 01, Fax (089) 5 23 35 15

United States of America
Butler Cox Inc.
150 East 58th Street, New York, NY 10155, USA
☎ (212) 891 8188

Australia and New Zealand
Mr J Cooper
Butler Cox Foundation
3rd Floor, 275 George Street, Sydney 2000, Australia
☎ (02) 236 6161, Fax (02) 236 6199

Ireland
SD Consulting
72 Merrion Square, Dublin 2, Ireland
☎ (01) 766088/762501, Telex 31077 EI,
Fax (01) 767945

Italy
SISDO
20123 Milano, Via Caradosso 7, Italy
☎ (02) 498 4651, Telex 350309, Fax (02) 481 8842

The Nordic Region
Statskonsult AB
Stora Varvsgatan 1, 21120 Malmo, Sweden
☎ (040) 1030 40, Telex 12754 SINTABS

Spain
Associated Management Consultants Spain SA
Rosalía de Castro, 84-2°D, 28035 Madrid, Spain
☎ (91) 723 0995