1-02-80 Planning for Object-Oriented Systems

John E. Gessford

Payoff

A new generation of information systems is emerging that is shaped by an object-oriented approach to design. This article describes a managerial approach to planning business applications that share data. The nine-step systems development strategy is based on the establishment of a businesswide definition of information resources and an organizationwide set of standards for the new object-oriented environment.

Introduction

For many users, Macintosh icons were an introduction to object-oriented systems. Each icon had behavioral as well as visual characteristics. Users learned about an icon more often by interacting with it than by reading about it. More important, users found it easy to think of the computer as being composed of a collection of objects that came to life when the system was turned on.

The Windows System, introduced in 1990 by Microsoft Corp., brought the concept of an object, called a window, to the much larger group of IBM PC users. Because Windows is capable of managing multiple windows simultaneously, it can manage the transfer of an object from one window to another.

IBM's description of its Application Development Cycle (AD/Cycle) and Repository Management System, which is intended to automate applications software development, shows clear evidence of object orientation. The repository is designed to store not only definitions and descriptions of data but also a definition of the programs that can access that data. Applications programming that is organized by the data that it operates on is presumed.

This way of structuring software is the hallmark of Object-Oriented Programming. A major objective of the joint venture between Apple Computer, Inc., and IBM, announced in 1991, is to create an object-oriented operating system that will be an alternative to Microsoft's Windows System and its successor OS/2 3.0. This alliance will strongly influence the way a large portion of the computer industry's efforts and resources will be spent in the coming years. It seems certain that the dominant operating systems of the future will be object-oriented.

The Need for Object-Oriented Applications Planning

If the hardware, operating system, and applications software all incorporate object-oriented principles, the method of analyzing and designing business applications probably should also be object-oriented. The fact that program specifications, which are an end result of analysis and design, need to be object-oriented if the programming is to be object-oriented strongly suggests that a complete object-oriented approach is needed. Comparing a set of traditional program specifications with a set of object-oriented programs to determine whether the specifications are satisfied can prove to be a difficult task. Different parts of the traditional, procedurally oriented program specification are satisfied by different program modules (i.e., methods) in the object-oriented system.

This change in planning methodology represents a quantum leap. Small adjustments to the traditional method of systems analysis and design do not work. The legions trained in
structured analysis and design methods during the past two decades find this change difficult to accept. Even today, many large applications software development departments are attempting to accommodate object-oriented software specification demands by adopting hybrid planning systems.

Hybrid planning usually begins with a structured decomposition of the relevant business procedures and data flow diagramming of these procedures; objects are then defined in terms of the functions and data stores of these data flow diagrams. Identifying the objects in this way is a poorly defined and complex process that tends to yield abstract objects that are often only arbitrary groupings of program modules.

Fortunately, the majority of applications software developers are not constrained by structured methods because, for the most part, they have no formal method of analysis and design. They simply obtain some performance specifications from those who will use the system and then go off and write the application programs. The first version the developers come up with may be called a prototype; a subsequent version is implemented, and the system is then refined for the rest of its life by what is euphemistically called maintenance programming.

Those who take this ad hoc approach, however, risk creating even more low-quality, expensive software with AD/Cycle and other new Computer-Aided Software Engineering tools than they have in the past. This is because the traditional method of systems development—the applications software project—leads to distinct, incompatible systems unless the individual projects are governed by a more comprehensive plan. The object-oriented approach requires an investment in an infrastructure of businesswide standards to create a full and accurate computer simulation of the objects vital to the business. Management's understanding of this necessity is requisite to ensure that appropriate attention and resources will be devoted to establishing and maintaining this infrastructure of information systems standards.

The Essence of Object-Oriented Planning

The object is the atomic unit in an object-oriented approach. The planning activities of systems analysis and design should be based on this fact. Rather than treat the need for object-oriented software specifications as an exogenous condition to be accommodated by the planning method, the planning should focus on objects from the start.

The starting point in systems analysis should be object identification, not procedure analysis. The objects that system users need to know about and work with should be the focus of attention. Only after these objects are identified and classified should other considerations come into play.

The classification of objects of vital importance to a business is the purpose of entity-relationship (ER) analysis. The entity types identified by this analysis are object classes to be included in the object-oriented business information systems being planned. In addition, Entity Relationship analysis models the relationships between entity types that should be simulated by the system and the information about entity types and relationships that is needed to run the business effectively.

The basic purpose of systems design should be to plan a simulation of the classes of objects identified through Entity Relationship analysis. A real-time simulation system is needed to show management the condition of the business at any time. Ultimately, after a series of development projects, the entire business can be simulated as a collection of objects that are created, evolve, interact, and finally cease to exist as far as the business is concerned.
The object simulation viewpoint defines and organizes the procedures of the systems in an entirely different way than the traditional input-process-output view of information systems. The object-oriented approach is to define the procedures needed to simulate the objects of each class. In contrast, the input-process-output approach organizes the data (i.e., objects) according to the business procedures that use the data.

Because applications software projects are defined in terms of automating specific business functions, the input-process-output approach is more compatible with traditional systems development than the object-oriented approach to planning is. The object-oriented approach is not compatible with project-by-project systems development unless the objects have first been modeled from a businesswide perspective. If standard data object classes are not defined businesswide, incomplete, conflicting, and incompatible object simulations will give misleading information to management, customers, and others who use the system.

Object-oriented systems design should begin with the object classes and relationships identified as part of the system in the analysis phase. With rare exceptions, standardized processing procedures should be applied in order to automatically create the software needed to capture, process, display, and print object information of interest to systems users.

**Avoiding the Very Large Systems Syndrome**

The need for leadership in developing object-oriented business information systems is imperative. As explained, a lack of leadership usually results in disparate systems because a project-by-project approach to development is taken without businesswide information systems standards.

A businesswide set of information systems standards requires senior management involvement. Only senior management can create an environment in which individual systems development initiatives can be harmonized and share object simulation software. It is important to note, however, that too much high-level involvement is possible. If senior management tries to micromanage systems development, the very large systems syndrome is experienced. It is easy to fall into the trap of thinking that the solution to incompatible systems is one large system. In the case of information systems, there is considerable evidence that this apparent solution is faulty. What seems to be a straightforward solution to the problem of fragmented systems turns into a quagmire in many cases. Very large information systems lead to uncontrollable development expenses and defective systems.

One example of the very large systems approach is IBM’s Business Systems Planning (BSP) methodology, which was introduced during the early 1970s, just after the first data base management systems were developed (the most recently revised version of BSP was published in 1984). BSP is an elaborate methodology that has been used by many companies, and variations on it are used by many systems consulting organizations today.

BSP studies have been conducted in hundreds of organizations. One survey of organizations that had completed BSP projects indicated, however, that the results have been disappointing in terms of eliminating incompatible systems, though some positive benefits were noted. The following three conclusions about BSP can be drawn from this survey and other evidence:

- The BSP methodology requires more time and resources than the results justify. The multiyear systems development plan is too detailed in view of the uncertainties an organization faces over the time period encompassed by the plan.
The top-down approach is helpful to the IS manager in the organization. One reason is that senior management provides a clearer picture of the systems and data needed to effectively run the business than do people at lower levels in the organization.

The BSP approach results in a plan for data bases that is too vague to ensure compatibility among systems.

The first conclusion points to the main reason why the very large systems approach fails. The life of a modern organization is too dynamic. The systems plans made today are inappropriate for the organization in the future, when the resources to build the system become available. A more flexible and results-oriented way of defining systems is needed.

The second conclusion supports the proposition that senior management involvement in systems planning is needed to align that planning with the strategic needs of the company. This is one clear benefit of the top-down approach of BSP.

The last conclusion indicates that the methodology must be extended to include the establishment of a high-level data administration function that can develop data specifications that will ensure systems compatibility. The function must be executed from a high-level position to ensure complete cooperation.

**Nine Steps to Object-Oriented Systems**

An approach to managing information systems provides for senior-level management involvement yet avoids very large systems commitments. In this approach, different aspects of systems are managed at different organizational levels. Some aspects are managed organizationwide, others businesswide, and the others on an application-by-application basis. (The distinction between organizationwide and businesswide is based on the conception of a legal organization engaged in more than one line of business.)

The organizationwide and businesswide management of certain aspects creates the environment within which compatible applications systems can be developed. The systems aspects that are managed organizationwide and businesswide should determine whether systems are compatible or vital to the strategic plans of the organization. All other aspects are best managed on an applications project basis.

This approach is summarized in Exhibit 1 as nine steps, from clarifying strategic goals to implementing the information systems. The first four steps have a businesswide scope. An organizationwide scope is recommended for the fifth step. The last four steps are done on an application-by-application basis.

**Steps from Strategy to Information Systems**
<table>
<thead>
<tr>
<th>STEP</th>
<th>MANAGEMENT</th>
<th>SCOPE</th>
<th>DATASOURCES</th>
<th>SOFTWARE USED</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Clarify the strategy and goals.</td>
<td>Senior management</td>
<td>Businesswide</td>
<td>Investment and policy proposals, Organizational data.</td>
<td>Strategic planning support, system data.</td>
</tr>
<tr>
<td>3. Identify functional experts</td>
<td>Human resource management and data administrator</td>
<td>Businesswide</td>
<td>Organizational chart, Job classification Employee data.</td>
<td>Repository data.</td>
</tr>
<tr>
<td>4. Define the businesswide information structure.</td>
<td>Functional experts and data administrator</td>
<td>Businesswide</td>
<td>Reports Transactions, Existing information structure.</td>
<td>Repository data.</td>
</tr>
<tr>
<td>5. Establish the computing environment</td>
<td>Senior and IS management</td>
<td>Organization Wide</td>
<td>Platform, DBMS Operating system, and CASE tool information DBMS, CASE tools.</td>
<td>Repository data.</td>
</tr>
<tr>
<td>6. Design the application</td>
<td>Systems analyst and users</td>
<td>Application by application</td>
<td>Systems requirements Prototype screens</td>
<td>CASE design tool.</td>
</tr>
<tr>
<td>7. Establish the physical data base</td>
<td>Data base administrator</td>
<td>Businesswide</td>
<td>Conceptual Repository data base DMBS DBMS documentation.</td>
<td>CASE coding tool.</td>
</tr>
<tr>
<td>8. Generate software</td>
<td>Programmer by application</td>
<td>Application by application</td>
<td>Systems design CASE coding requirement</td>
<td>Applications software.</td>
</tr>
<tr>
<td>9. Implement the system</td>
<td>Testers, documentation writers, trainers and users.</td>
<td>Application by application</td>
<td>Systems design Applications software.</td>
<td>Applications software.</td>
</tr>
</tbody>
</table>


This approach is a variation on the information engineering (IE) methodology defined by James Martin. The first five steps accomplish many of the same results achieved by the planning and analysis phases of the IE methodology. These five steps work in a simpler, more flexible way, however. Two complicated techniques in the IE approach—affinity analysis and process diagramming—are omitted from the methodology described here.
Step 1: Clarifying the Strategy and Goals

Clarifying the strategy and goals of a business is an important first step in aligning the information systems of the business with its goals. A clear statement of purpose and explicit strategic plans for achieving that purpose may directly identify some information systems needs. Information systems may even be identified that play a vital role in a strategy.

Objectives and the strategies to achieve them also provide the rationale for the functional model developed in step 2, which is used to systematically identify information systems requirements. An understanding of corporate strategy is needed to set systems development priorities.

Step 1 can be taken only by the senior management of an organization. Only as these leaders adopt a strategy and specific goals should they be allowed to influence the way resources are used, including resources devoted to information systems. Others in the organization should provide strategic ideas through policy or investment proposals and provide suggestions for solving problems. It remains up to senior management, however, to accept these ideas for the organization.

The procedures and schedule for strategic planning may vary from one organization to another. Approaches that involve a broad cross section of the organization generally promote a participative style of management. An information system that facilitates the formulation, communication, and evaluation of goal suggestions from across the organization (labeled a strategic planning support system in Exhibit 1) can be helpful in making such approaches more effective.

Step 2: Defining a Functional Model

The functional model of a business defines what has to be done to achieve the strategic goals. It is a form of strategic planning that breaks down the overall activity of fulfilling the purposes of the organization into the required actions and decision making.

Because it is a part of strategic planning, a functional model is appropriately defined by senior managers who understand the organizational purposes and plans to be fulfilled. A data administrator can provide assistance in formulating the model in a way that is compatible with a type of software package and data base that the information engineering methodology terms a system encyclopedia and that IBM calls a repository system. This system can be used to record the functional model and link it to the other plans developed in steps 3 and 4. A repository allows information about the functions and their subfunctions to be entered into an information system and then displayed or printed in various formats.

When the major functions have been identified, further definition and subdivision of them can be done by experts in each function (step 4). For example, marketing is often a major function required by the strategy of the organization; the definition and subdivision of marketing is best left to marketing experts, or should at least be done in consultation with such experts.

Step 3: Identifying Functional Experts

The functional model is not an organization chart, but it can be related to an organization chart and the jobs and positions in the organization. An analysis of how the functional model relates to the organization must be made to identify who should
determine the data requirements of each function and subfunction. In most cases, someone who is executing the function is best qualified to serve as the functional expert. This analysis can provide some equally important side benefits to those responsible for human resource (i.e., personnel) management. By comparing job descriptions with the definitions of the functional model, misalignments with respect to the organization's strategic goals can be detected. Functions not provided for in the organizational structure (or being done redundantly) are identified.

Ideally, the repository system should provide the information needed to make this analysis. The fact that the information is useful to both personnel and information systems management points to a need for cooperation and sharing that has not been traditionally recognized by the management of either of these departments.

**Step 4: Defining the Businesswide Information Structure**

The term information structure is used to designate a plan that is not a complete conceptual data base but that can serve as the framework of the conceptual data base. (The term conceptual data base is used to denote a complete definition of the data needed by the organization.) The information structure defines the entity types about which the business needs data, but not all the specific types of data required. Relationships between entity types are also defined in an information structure.

The functional experts and data administrator work together to characterize data requirements in terms of an information structure. The functional expert describes transactions, queries, and reports that are important to the function. The data administrator abstracts from the descriptions the entity types and relationships involved.

The reports, queries, and transactions, as well as the entity types and relationships that they imply, should be defined in the repository system. To the extent that the report, query, and transaction descriptions are complete, attributes of entity types can be recorded as well; however, attribute specifications should not be considered final until step 7 is completed.

The execution of step 4 by the data administrator should be businesswide so that data definition standards can be set up to govern all references to data in all software purchased or developed. This is the prerequisite for automatic data sharing without special import-export programs. Exhibit 2 illustrates how the results of step 4 are used in the development of integrated systems. The data base concept, shown in the upper left-hand corner of Exhibit 2, guides the development of an integrated set of data bases that all information systems of the business can use. The development of the integrated data bases from the results of step 4 is the responsibility of the data administration function.

**Data Administration and the Development of Integrated Systems**

The objective in this fourth step is limited to information structure definition, rather than the conceptual data base definition, so that the step can be accomplished businesswide within a few months. To obtain a complete data base definition would require a far more detailed analysis and could take years. Identifying the entity types gives the data administrator a basis for evolving the conceptual data base definition over a period of years as applications development projects are undertaken. The businesswide information structure linked to the functional model provides an overview of who needs what types of data in the organization.
Step 5: Establishing the Computing Environment

To gain the full benefit of the first four steps, it is imperative that an automated approach be taken to the remaining steps. Otherwise, the businesswide plans will be obsolete before the systems to implement them can be developed. This is one reason why properly designed Computer-Aided Software Engineering tools are so important. They can automate the design of applications systems, the creation and testing of applications programs, and the preparation of training and reference materials.

A computer-aided software engineering (CASE) tool is designed to work with a certain operating system. It may generate code that makes calls to a certain type of data base management system (DBMS) and network control system as well. The operating system, DBMS, and network control system usually have certain access control systems and require specific hardware to function. Consequently, standards for computer-aided software engineering (CASE), DBMS, the operating system, data networking, security, and computer hardware must be established, roughly in that order. Hardware should not be selected first. This common mistake often excludes the best Computer-Aided Software Engineering, DBMS, and operating system alternatives from consideration.

These aspects of the information systems standards are needed to ensure integrated systems and businesswide conceptual data bases. Without these organizationwide standards, the systems developed will be more or less incompatible and training and maintenance expenses will be greater than necessary.

The setting of these standards is obviously of strategic importance and therefore needs to be done by senior management. Because there are technical as well as business issues involved, information systems professionals should participate in the decision-making process. The business and technical leaders of the organization must reach a consensus on these decisions.

Step 6: Designing the Application

The definition of an application and its feasibility, the formulation of system performance requirements, and the detailed design of the system are included in this step. Step 5 represents that part of applications development that front-end computer-aided software engineering (CASE) tools are commonly designed to automate. It does not include the businesswide or organizationwide planning of steps 1 through 5.

Usually a systems analyst works with potential users in accomplishing this step (sophisticated users may prefer to design applications themselves, however). Rapid Application Development techniques and tools, including prototyping, help to facilitate communication between developers and system users during this step.

This step is greatly simplified if an object-oriented approach is taken and a businesswide information structure is available as described in step 4. For each entity type (and certain relationships) involved in the application, a data object class can be defined. For each screen and report to be included in the system, a window object class or report object class can be defined. Then, for each object class, standard programs required to simulate objects of the class can be specified. Each computer program is dedicated to simulating a certain behavior of objects of a single class. The entity types and relationships identified in step 4 can therefore be directly translated into design objects in step 6. The part of the businesswide information structure that a given application uses should be fleshed out at this time so that it constitutes a conceptual data base. This process must be done before a physical data base can be developed. The process should take into account any attributes of the entity types involved in the application that other future applications will require.
Step 7: Establishing the Physical Data Base

The planning and creation of physical data base files should be done on a businesswide basis. It may be that every application should have its own physical data base, though Transaction Processing times and data communications costs can be significantly reduced by storing certain data in a different data base or more than one data base.

In managing physical data bases, it is also important to satisfy security requirements and to consider the cost of providing adequate technical support to ensure that files are not corrupted. A file containing thousands of transactions or data on thousands of customers represents an important asset of the business, even though it is not an asset in the accounting system. These considerations tend to favor a centralized approach to physical data base management.

Physical data bases should be established and managed by a data base administrator who is responsible for maintaining the organization's data resources. This individual works with the systems analyst in defining a view of the data associated with each entity type and relationship that is relevant to a particular application. The data base administrator then maps that view to the data definitions of one or more physical data bases.

Step 8: Generating Software

Using a computer-aided software engineering (CASE) approach provides the biggest payoff in code generation, traditionally the most expensive and least controllable step. The system designed with computer-aided software engineering (CASE) tools can be coded automatically using a code generator designed to work from the design specifications created by the Computer-Aided Software Engineering design tool. Consequently, the programmer is removed from direct applications programming, and the time and cost of coding are thereby cut dramatically. Higher-quality code is also obtained, which means the code testing and debugging cycle is brought under control.

Step 9: Implementing the System

The final step includes preparing user documentation, training the users, and pilot testing the new system in an operational environment. This is a difficult step to manage because each phase often reveals omissions and misunderstandings from previous phases and steps; therefore, iterative processes are involved.

The importance of performing this final step well is commonly underestimated. Failure to properly test and document a new system and train its users can be as fatal to the success of the system as poor design or an inadequate DBMS. If the system's users do not understand a feature of the system, they will not use it, and it may as well have been excluded. Poorly trained users are usually frustrated users, who soon blame all their troubles on the system.

Step 9 should be executed by system user experts, not programmer/analysts. Individuals familiar with the business function but not the computer system are most likely to detect operational deficiencies during testing and to provide thorough training and documentation. Programmers usually take too many details about the system for granted.
Benefits of Businesswide Management of Objects

The benefits of preparing for object-oriented systems using the method described are not merely technical and qualitative. In an increasingly fast-paced, electronic business world, object-oriented systems are quantitative and central to business survival.

The major reason for coordinating references to objects (i.e., data) across an organization is the same as for establishing standards for communication between army, navy, and air force units engaged in battle against a common enemy. It is not enough that different parts of the organization share a common goal. They must also work from a common base of information, or they will be uncoordinated and will end up blaming one another for failures.

The way in which goals, information, and resources interact to determine what an organization accomplishes needs to be better understood. In one sense, information is a resource because it enables workers to effectively execute the functions that need to be performed to achieve the goals of the organization. Yet information is different from other resources in the way it shapes goals and guides the use of other resources. Information is not just another resource.

Information shared and communicated is probably the best means for coordinating the daily work of different departments in an organization. The goals of the organization determine the departments that are needed, and they can be used to set up measures of performance and incentives for individual managers as well as for the organization as a whole. However, it takes shared information to coordinate hourly and daily the work of different parts of an organization.

The natural tendency of individual managers to develop information systems customized to the needs of their departments leads to an incompatible collection of systems that inhibit the sharing of information. Each department develops its own version of what needs to be done to achieve the goals of the business. The result is misunderstandings and uncoordinated responses to the needs of customers and suppliers.

To obtain fast, coordinated organizational responses to customer requests, information systems that transcend both interorganizational and intraorganizational boundaries are needed. Such systems can deliver the information each person in the organization needs to work with others in achieving the goals of the organization.

The underlying drive to develop information systems clearly should and does come from those who stand to benefit directly from the use of them. Too often, this drive comes from units far down in the organization. Such initiatives lead to many separate, narrowly defined systems suited to the needs of the individual department. The manager who is in a position to make an information system profitable and commits to doing so is usually the one who gets the funds.

Senior management must decide how it should exert a countervailing force for coordination and integration of information systems. In cases in which senior management has simply taken over systems development, the result has been very large systems, which have been defined by one prominent consultant as projects that are never completed. A more effective approach is one that creates an environment within which the proactive manager builds systems. This environment channels the primary creative forces in ways that create systems that transcend organizational boundaries.

A more coordinated and responsive organization is the major reason for creating businesswide guidelines for object-oriented systems development. There are, however, other advantages to be gained by this approach.

Access to More Data
In a shared data base, all the data is available to each user group that has a valid need (and the appropriate access codes) to access the data. This assumes an adequate data communications system is in place.

**Shared Programs**
Business functions performed at multiple locations can use the same applications software if they use the same data base design (or the same data base), so the cost of developing and maintaining programs can be spread over multiple sites.

**Use of Compatible Purchased Software**
Two or more software packages that use the same commands (e.g., Structured Query Language commands) to access a data base, and that are able to access data bases that have the same design, can share data. Purchased software that does not meet these two characteristics results in independent data bases that can be shared only by importing and exporting data, which is a complex and slow process.

**Reduced Data Capture Expense**
When data is shared, it needs to be keyed in only once. Product descriptions, for example, need not be entered separately into sales, purchasing, manufacturing, inventory, engineering, and other systems. If product descriptions are shared by these systems, they can be entered once and accessed by all these systems. Independent applications systems usually have independent data capture modules, so there is much duplication of both data entry and data storage.

**Reduced Data Expense**
To the extent that two or more applications are using the same physical data base, only one copy of the data (instead of two or more) needs to be held in online storage (usually disk storage).

**Avoidance of Inconsistent Data**
When two executives at a meeting present contradictory data on the same subject, a great deal of expensive professional time can be consumed in reconciling the data. The discrepancy is often the result of a technical difference in the way updates to the data bases are handled in the two applications systems.

**Improved Interface Between Systems**
The output of one system is often input for another system. For example, the output of a Computer-Aided Design system in engineering is a product definition that must be reviewed by the manufacturing engineers if the product is to be produced in-house. Unless design and manufacturing engineering have integrated data bases, a data capture (or data conversion) operation must be performed before production planning analysis can take place.

Information systems departments generally have not been able to develop systems that share a common data base, according to a survey conducted by the Sloan School of Management. What they commonly do is extract data from the files of separate systems and put it into an information center data base that is accessible to managers and analysts. This provides access to more data—one of the advantages from data sharing listed—but it fails to provide the other advantages. If maintaining a duplicate data base is worthwhile because of the access it provides, the greater benefits of real systems integration must certainly be worthwhile if a workable way to build such systems is available.
Conclusion

A new generation of information systems is emerging that is shaped by an object-oriented approach to design. Object-oriented systems are relatively inexpensive and easy to work with, and when combined with Computer-Aided Software Engineering tools that make customized systems easy to generate, they are likely to lead to a proliferation of microcomputer-based systems in most organizations.

The planning of business applications must change to be compatible with object-oriented systems. In implementing new planning methods, there is an opportunity to define organizationwide standards that will make all information systems that adhere to the standards coordinated and integrated with respect to data access. Establishing such standards, however, requires business leadership.

Business management has an opportunity to make better use of computers by exercising more leadership in the development of systems. More technical expertise is not the critical factor. Leadership in creating an environment that fosters systems that are compatible and that share data is the vital factor. Without this leadership, a patchwork of disparate information systems is inevitable.

The managerial approach taken to establish systems that share data must avoid the very large system syndrome. Difficulties arise if systems development efforts are too centralized. Experience provides ample evidence of the fallacy of trying to centrally plan and manage information resources as one big system. Business is simply too dynamic and multifaceted for that strategy to be practical.

The most promising systems development strategy is one in which senior management supports the establishment of a businesswide definition of data resources and an organizationwide set of standards for the computing environment. The development of individual applications within this environment can then be left to individual departmental managers and professional staff members.

Bibliography


Author Biographies

John E. Gessford

John E. Gessford is president of Modern Information Services in San Marino CA and a professor in the business school at California State University, Long Beach. He received his doctorate in industrial engineering from Stanford University and has 14 years' experience in the industry—first at International Paper Co. and then as a consultant at SRI International—in information systems planning, development, and management. Formerly with the faculty of the information science program at Claremont Graduate School, he developed courses in data base management and artificial intelligence as well as a practicum in information systems development.