1-1-1977

A relational model for events-based accounting systems.

William Ernest McCarthy
University of Massachusetts Amherst

A RELATIONAL MODEL FOR EVENTS-BASED ACCOUNTING SYSTEMS

A Dissertation Presented

By

WILLIAM ERNEST McCARTHY

Submitted to the Graduate School of the
University of Massachusetts in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 1977

Business Administration

A RELATIONAL MODEL FOR EVENTS-BASED ACCOUNTING SYSTEMS

A Dissertation Presented

By

William E. McCarthy

Approved as to style and content by:

Van Court Hare, Jr.

Van Court Hare, Jr., Chairman of Committee

Joseph L. Sardinas, Jr.

Joseph L. Sardinas, Jr., Member

Conrad A. Wogrin

Conrad A. Wogrin, Member

Donald G. Frederick

Donald G. Frederick
Ph.D. Program Director
School of Business Administration

To Jane, Meghan, and Molly

ACKNOWLEDGEMENTS

I wish first to thank the members of my dissertation committee--Van Court Hare, Joseph Sardinas, and Conrad Wogrin--for their time and guidance during the last year. Their support was invaluable in finishing this work.

I am also grateful to Bob Taylor who introduced me to the ideas which led to this dissertation and to David Stemple who provided much needed knowledge at critical points.

I am indebted too to John Burch who served as my doctoral advisor, and who is primarily responsible for my teaching and research career in accounting.

Finally to my wife Jane, who helped type, read, and draw the many versions of this paper, goes my sincerest love and gratitude. She has supported me in every possible way during my studies and given me two beautiful daughters along the way. With her, I hope to spend many more years as happy as these past four.

ABSTRACT

A RELATIONAL MODEL FOR EVENTS-BASED
ACCOUNTING SYSTEMS

(February 1978)

William Ernest McCarthy, A.B., Boston College
M.B.A., Southern Illinois University
Ph.D., University of Massachusetts

Directed by: Professor Van Court Hare, Jr.
School of Business Administration

This research is an extension of previous work aimed at integrating ideas in the field of database systems with accounting theory. Specifically, its purpose is to integrate the type of disaggregated and multidimensional accounting systems implied by Sorter's "events" theory with the relational database model of Codd. The result of this synthesis is to be an integrated and highly flexible information system able to meet the needs of both accountants and non-accountants without use by one party rendering the data unusable by the other. Most of the prior research in this area had concentrated on development of hierarchical database models.

The events-based system developed differed from the conventional accounting framework in two major respects:

1. It did not adhere to the conventions of the monetary principle and double-entry in structuring the economic events to be allowed into the system from the environment

2. If did not use the traditional system of classification, summation, and aggregation in maintaining the data and preparing it for output; instead it maintained information concerning both economic entities and the relationships between them as relations (tables) and used an appropriate set-oriented language to process them

The model was first developed in the context of a simple accounting example dealing with operations of a small retail enterprise over one month. During this development, data constructs based on "debit-credit" and "chart of accounts" processing were deemed to be inadequate. The entity-relationship modeling approach of Chen was substituted and resulted in a data model that fit well the specifications of a normalized relational system.

After the database model of the enterprise was presented, its processing needs were addressed. The relational language SEQUEL 2 was proposed for this purpose, and its use was exhibited in a variety of retrieval contexts. It was shown that the model and language constructs could be used to meet the needs of traditional accounting, "events" accounting, and a variety of other decision making processes. Additionally, the use of SEQUEL 2 conventions insuring maintenance of system auditability and internal control was illustrated.

Following exposition of the retail enterprise example, normative aspects of an events-based relational system were discussed. A set of normative guidelines based upon the accounting theories of Ijiri, Mattessich, and Mock were developed, and their use in a stage approach to model construction was recommended. This stage approach would consider provision of accountability standards as a minimum goal for an events system; after which the needs of other decision makers would be considered.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF EXHIBITS

CHAPTER I

INTRODUCTION

Double-Entry Accounting and Computer Systems

While the computer revolution of the past three decades has radically transformed many of the information processing and communication functions of business organizations, it has not yet changed the fundamental methodology of financial reporting--double-entry accounting. This conclusion is inescapable and was noted recently by two distinguished computer auditors, Elise Jancura and Fred Lilly:

> Computers and their related support equipment have not in any significant way altered established accounting theory as it relates to the kind of data to be collected or the manner in which such data should be organized for reporting purposes. (Italics mine)[1]

Clearly, computers have had some effect. They have made the accounting recording and transmission process faster and better able to handle large volumes of data, and they have provided new classes of decision models which use accounting information as their primary input, but they have not altered the essence of the traditional system which mandates the following:

1. That exchange transactions be recorded in monetary terms

2. That double-entry be maintained (debits=credits)

3. That costs be matched by some process to associated benefits and periods

4. That the effects of similar transactions be aggregated before reporting

---

[1]Elise G. Jancura and Fred L. Lilly, "SAS No. 3 and the Evaluation of Internal Control," The Journal of Accountancy 143 (March 1977):69.

In information systems use, these accounting tenets can be very restraining as we shall see in the following example which illustrates the treatment of machine depreciation.

Let us suppose that a certain group of machines costing $10,000 has been used for 300 hours during a month-long production process with an output of 6500 units. To recognize the decrease in the machines' value or service potential, an accountant would record:

Debit	Depreciation Expense-Machine	$XXX	
Credit	Accumulated Depreciation-Machine		$XXX

Note the following with regard to the accounting principles stated above:

1. Only the dollar cost dimension of the transaction is recorded; hours run and units produced are not inputted to the accounting system

2. The decrease in service potential (accumulated depreciation) is paired with and set equal to an expense of running the company (depreciation expense)

3. The effect of the economic event that has occurred (machine service potential decrease) will not be recognized until the particular accounting period in which it happened ends or the product it produced is sold

4. This recording of machine depreciation will be aggregated with other events from different time periods and departments before being reported, thus obscuring its unique qualities

We see that, in this computerized age, the long enduring double-entry system remains essentially intact. This is not a state of affairs that pleases all accounting theorists, and many have called for a reevaluation of the accounting framework with a mind toward better integrating its operations with the capabilities of modern systems. Among these theorists are John Wheeler who has said: "There is no reason to believe that the methods appropriate to the quill pen in the fifteenth century will be equally applicable to on-line computer systems of the last third of the twentieth century."[2], and R.L. Mathews who has contended that, "The principal reason for the lag in the application of computer technology to the accounting information system would seem to be incompatability between the form of the double entry recording system, which it must be remembered has slowly evolved during the last five hundred years or more, and the data processing qualities of a computer."[3]

Thesis

We agree with Wheeler and Mathews, and it is to be the primary thesis of this research work that the basis for the extended accounting methods they call for can be found in the rapidly developing area of computer database systems. More specifically, we will contend that the theory of relational databases developed by E. F. Codd[4] will provide a sound mathematical and analytic basis for the design and construction of an expanded accounting recording and transmitting process. We believe that Codd's data management framework, when combined with George Sorter's "events" system[5] of transaction data capture, will provide the

---

[2] John T. Wheeler, "Accounting Theory and Research in Perspective," The Accounting Review 45 (January 1970):7.

[3] R.L. Mathews, "A Computer Programming Approach to the Design of Accounting Systems," ABACUS 3 (December 1967):133.

[4] E.F. Codd, "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM 13 (June 1970):377-87.

[5] George H. Sorter, "An 'Events' Approach to Basic Accounting Theory," The Accounting Review 44 (January 1969):12-19.

capabilities to record economic events in the environment of the firm in a manner that will provide the following benefits:

1. It will allow transactions to be recorded along dimensions other than money thus providing input to a larger class of models

2. It will provide a high degree of flexibility in reporting and retrieval techniques which will allow accountants to implement their own procedures (such as matching) without rendering the information unusable by others

3. It will allow information to be recorded and stored at a very low level of aggregation thus reducing the amount of information loss normally caused in the classification and transmission of such data

4. It will provide, without disbenefit to present users, an inquiry capability to decision makers wherein they can pose unstructured and unanticipated questions to the organization's database

5. It will provide a proven theoretical basis for the integration of accounting data with modern computer capabilities and a sound methodology for the maintenance of very large volumes of accounting data over time and changing circumstances

In summary, we believe that flexible accounting systems which feature both disaggregated and multidimensional information will be able to provide service to a much larger class of decision makers both internal and external to a firm than is presently the case. Given the desirability of such systems, we believe that Codd's database theories provide a logical avenue for their development and maintenance.

In the next chapter, we will review the theories and research work supporting our thesis. Material to be examined will include both recent advances in database theory and publications supporting expanded accounting systems.

CHAPTER II

BACKGROUND AND RELATIONSHIP OF PROPOSED RESEARCH TO WORK OF OTHERS

Introduction

To understand completely why we believe that the integration of Codd's relational database theories into an accounting framework represents a logical step forward, it will be necessary to review certain topics in the published literature of both accounting and computer science. We will begin by examining the rationale for viewing accounting as an organizational information system rather than simply as a methodology to be used in the preparation of financial statements and then proceed to an investigation of those theorists who champion the extension of these accounting information systems in terms of more dimensions and lower aggregation levels. This will lead us to a consideration of the work of the "events" accounting theorists beginning with the initial hypothetical ideas of George Sorter and proceeding through research dealing with the actual integration of "events" concepts into working database systems. At this point, we will then turn our study to the field of computer science and examine briefly recent developments in database theory, especially relational database theory. Finally, we will summarize by restating our primary thesis and enumerating again the benefits we hope to produce with our proposed system.

Accounting as a Unified Information System

Throughout this research, we will adhere to the notion that accounting should be viewed as an integral part of a total organizational information system rather than as a simple framework for cost accumulation and preparation of published financial statements. This is decidedly a broad or loose constructionist view, but it is one that is supported by many accountants as will be seen below. Much of the logic for these supporting arguments is taken from the work of Gerlad Feltham.[1]

Most often, accountants tend to view their discipline not only as divorced from other information collecting activities of an organization but also as split into two parts: (1) financial accounting which deals with providing information for external reporting purposes and (2) managerial accounting which deals with providing information for internal decision makers. The former of these is governed by long standing rules of public reporting (usually referred to as "generally accepted accounting principles") such as conservatism, consistency, and verifiability, while the latter is usually governed more on an ad hoc basis depending upon the perceived needs of particular decision makers. This dichotomy, although commonly sanctioned, is deplored by many theorists among them Sidney Davidsion:

> The notion of managerial analysis and financial reporting as separated, fragmented, and even opposing activities should, and I am confident will, be soon supplanted by the view which emphasizes the basic unity of the accounting function. Accounting is an information system which provides significant, meaningful financial information about the firm--both for internal management use and for external financial reporting.[2]

---

[1] Gerald A. Feltham, Information Evaluation (Sarasota: American Accounting Association, 1972) pp. 1-5.

[2] Sidney Davidsion, "The Day of Reckoning-Managerial Analysis and Accounting Theory," Journal of Accounting Research 1 (Autumn 1963):117.

Davidson's call for a unified view of accounting information systems was further extended by two committees of the American Accounting Association, both of them noting the major roles that accountants should play in the development and use of an overall organizational information system:

> Essentially, accounting is an information system. More precisely, it is an application of a general theory of information to the problem of efficient economic operations. It also makes up a large part of the general information systems which provide decision-making information expressed in quantitative terms. In this context accounting is both a part of the general information system of an operating entity and a part of a basic field bounded by the concept of information. (Italics mine)[3]

> The accounting function is one of the most important information systems in an organization. Clearly, it is not the whole of even the formal information system given (a) the diversity of data collected, processed, and distributed for the many different functions within the organization and (b) the expanded range of disciplines bearing on the information function. It is, however, difficult to conceive of accounting not being an integral part of the formal information system or accounting personnel not being a major force in information management. (Italics mine)[4]

This line of reasoning was advanced one step further by Churchill and Stedry who advocated not only that accounting data be integrated into the total organizational information system, but also that accountants expand their data collection and measuring procedures in an effort to provide more reliable input to a variety of newer management decision models:

---

[3]American Accounting Association, A Statement of Basic Accounting Theory (Chicago: American Accounting Association, 1966) p. 64.

[4]American Accounting Association, "Report of Committee on Accounting and Information Systems," The Accounting Review 46 (Supplement, 1971):344.

> In most business firms, accounting data are frequently the only data available whose collection proceeds periodically, without missing observations and, generally, with recurrences of the same event recorded consistently in the same category. It is this expertise that we suggest be adapted to the information requirements presented by new management techniques through expansion of the scope of the definition of accounting and its methodology. The alternative would seem to be development of an entirely new measurement theory and practice divorced from the data collection and verification traditions of accounting and the discipline of the accounting profession.[5]

It is this last view of accounting (Churchill and Stedry's) that we will use in our work from this point on. Accounting will be viewed as an integral part of an enterprise-wide information system, and accountants in this scheme will be tasked with measuring, collecting, and storing not only data useful for traditional purposes (external reporting and internal cost accumulation) but also data useful for any of the various decision making processes of the firm. Certainly, this is a much larger audience toward whom an accountant would have to address his work, and it will necessitate an expanded type of accounting system. We will now turn our attention to such a system and attempt to ascertain some of its desirable features in terms of meeting the information needs of its larger user population.

User-Oriented Features of Accounting Information Systems

Model Accounting System

As a starting point for discussion of these expanded accounting

---

[5]Neil C. Churchill and Andrew C. Stedry, "Some Developments in Management Science and Information Systems with Respect to Measurement in Accounting," in Research in Accounting Measurement, ed. Robert K. Jaedicke, Yuji Ijiri, and Oswald Nielsen (Sarasota: American Accounting Association, 1966), p. 30.

schemes, the view of a financial information system proposed by Buckley and Lightner is used (see Figure 1). Their model embodies the traditional accounting framework, and their rules for input, transformation, and output are largely dictated by the existing set of generally accepted accounting principles. We can see how information flows through this system by following (through each box and along the arrows) the depreciation example mentioned earlier.

INPUT. An economic event is recognized in the environment of the firm and inputted to the information system depending upon its compliance with the given rules (entity, economic event, etc.). For the depreciation case, this would involve a recognition that the use of machines and their subsequent loss of service potential was an event of significance.

TRANSFORM-1 and TRANSFORM-2. The captured information is first journalized and then posted to ledger accounts. The depreciation would be journalized by assigning it a dollar value and by pairing it with another entry (in this case a debit to depreciation expense). It would then be posted to a ledger and in the process combined with depreciation of other departments and other time periods.

OUTPUT. Depending upon the particular reporting period and the materiality (relative size) of the aggregated ledger totals, the information would be extracted from the system and relayed to its potential users. For the depreciation, this would involve definitely its listing as an asset adjustment on the balance sheet and possibly its use in certain decision models (such as computation of Return-on-Investment).

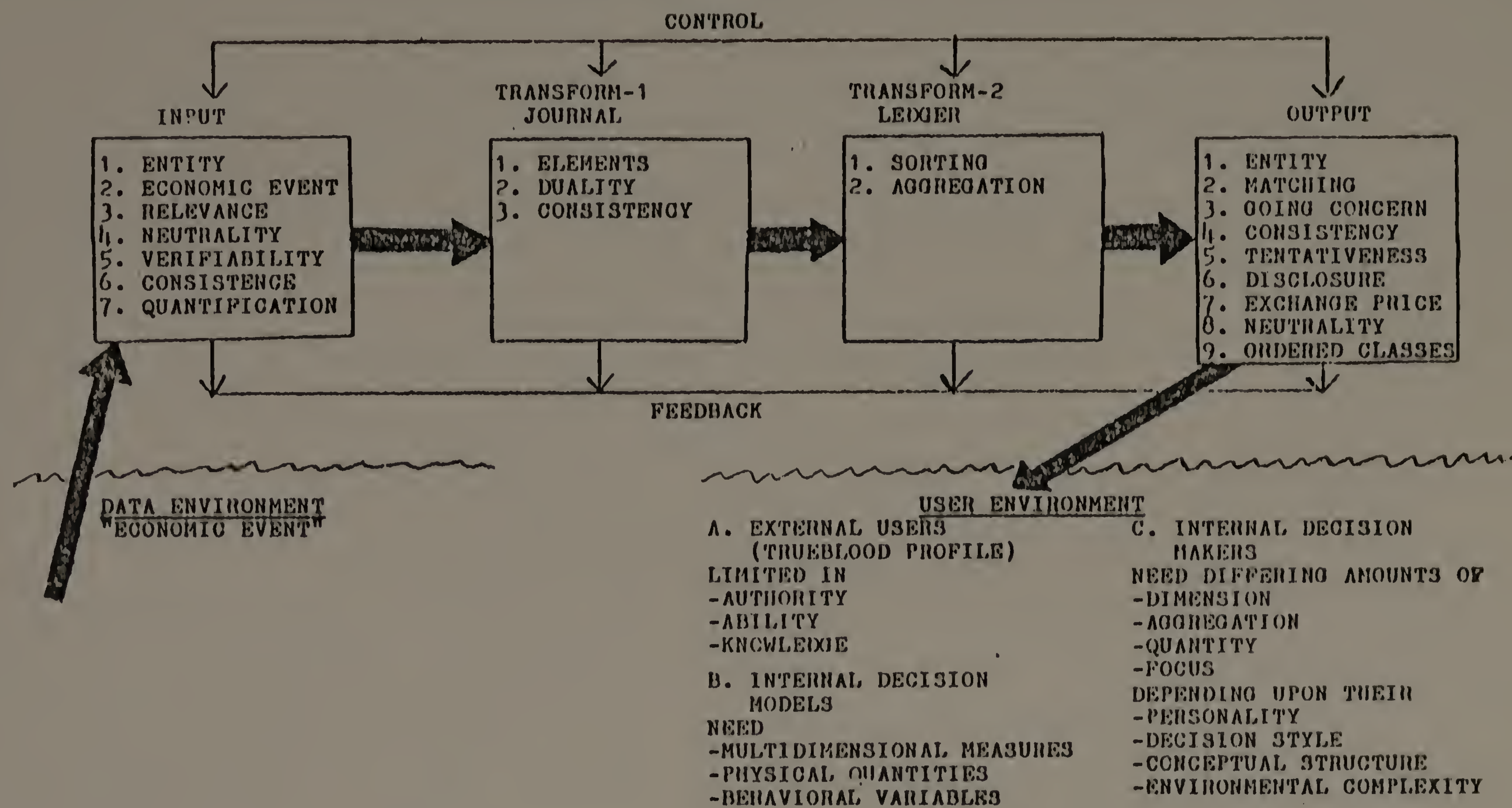


FIGURE 1--Accounting information system

User Environment

In view of our decision to assign accounting a broader role in providing decision-relevant information to a wider range of users, we now proceed to a more detailed analysis of the user environment (pictured in the bottom right section of Figure 1). We will partition this environment into three sections as detailed below and use recent published work to characterize some of the needs of each division. These are not meant to be mutually exclusive or exhaustive listings, but they do include the majority of user applications.

A. External Users. This group would primarily consist of potential creditors and stockholders who are typed in the Trueblood Report[6] as having limited authority, ability, and knowledge to obtain information from company records. An accounting system designed to optimally serve these people would obviously need casual user interaction facilities and extensive ad hoc question-answering capabilities. Research has also indicated that this group needs: (1) multidimensional information to assess corporate effectiveness in non-financial areas such as social reporting[7] and (2) disaggregated information to allow them to use their own classification schemes to assess given information.[8]

---

[6] Report of the Accounting Objectives Study Group: Objectives of Financial Statements, by Robert M. Trueblood, Chairman (New York: American Institute of Certified Public Accountants, 1973), p. 17.

[7] American Accounting Association, "Report of the Committee on Non-Financial Measures of Effectiveness," The Accounting Review 46 (Supplement 1971):181.

[8] Irwin Bernhardt and Ronald M. Copeland, "Some Problems in Applying an Information Theory Approach to Accounting Aggregation," Journal of Accounting Research 8 (Spring 1970):95-98.

B. Internal Decision Models. This group comprises the newer management science techniques such as linear programming, queuing models, and the use of learning curves. Churchill and Stedry conclude that these methods are best served by input data expressed in physical or multidimensional terms.[9] Also, Corcoran lists a number of cost accounting models where nonmonetary measures provide both better indices of effectiveness and better optimization criteria.[10]

C. Internal Decision Makers. These are the managers of the firm, and their information needs have been researched extensively by theoreticians in the fields of accounting, human information processing, and others. This research has indicated the following:

1. That decision makers need differing amounts of aggregation, quantity, and focus of information depending upon their own personalities, decision styles, conceptual structures, and environmental complexities[11,12]

2. That managers need multidimensional measures to assess effectiveness in areas such as marketing, productivity, personnel, and internal communication[13]

---

[9] Churchill and Stedry, pp. 28-48.

[10] A. Wayne Corcoran, "Costs: Accounting Analysis & Control," Unpublished Manuscript, 1976.

[11] Michael J. Driver and Theodore J. Mock, "Human Information Processing, Decision Style Theory, and Accounting Information Systems," The Accounting Review 50 (July 1975):490-508.

[12] Henry Miller, "Environmental Complexity and Financial Reports," The Accounting Review 47 (January 1972):31-7.

[13] American Accounting Association, "Report of the Committee on Non-Financial Measures of Effectiveness," p. 178.

3. That decision makers perform better with disaggregated data[14]

4. That accounting information system designers should provide as much flexibility as possible in their systems in order to accommodate the wide range of question-answering capabilities needed by successful decision makers[15]

Ranging across the different partitions of the user environment listed above, we can see certain information needs that will be applicable to a majority of potential accounting system users. In particular, we note that these systems should be designed to accommodate (1) a great deal of flexibility, (2) an ability to capture and store multidimensional data, and (3) a capability for storing and retrieving data at very low levels of aggregation. These are precisely the major characteristics we hope to build into our proposed system later in this paper, but to do so, we will have to expand the range of data input far beyond what traditional accounting principles now allow. For example, instead of recording our machine depreciation as:

Debit	Depreciation Expense-Machine	$XXX	
Credit	Accumulated Depreciation-Machine		$XX

we might want to record the same event like this:

START TIME	END TIME	MACH NO.	OPERATOR NO.	PRODUCT NO.	% DEFECT	POWER USED	JOB NO.
66-0200	66-1700	42B	7763	A516	15	400	M16

[14] Russell M. Barefield, "The Effect of Aggregation on Decision Making Success: A Laboratory Study," Journal of Accounting Research 10 (Autumn 1972):229-42.

[15] Henry C. Lucas, Jr., "The Use of an Accounting Information System, Action and Organizational Performance," The Accounting Review 50 (October 1975):735-46.

This type of expanded transaction is a feature of the events accounting concept first proposed in 1969 by George Sorter,[16] and we will next turn our attention to the literature in that area.

Review of "Events" Accounting

Introduction

To build an information system which will provide disaggregated and multidimensional output, we will have to utilize a process that will accept input in the same form. Normally, the set of economic events to be absorbed by an accounting system is well defined by convention (as is evidenced by the given input rules in Figure 1), but this set must be extended if we wish to expand in scope beyond traditional accounting. What Sorter proposed with his "events" theory was (1) that all of the relevant variables surrounding an economic event be measured and recorded and (2) that this information be left in a company's records in atomic (disaggregated) form to be classified and aggregated by the eventual users. Sorter likened the reporting process that would follow to that of a television weatherman giving his viewers the various instrument readings and regional weather data and then letting them decide for themselves the accuracy of forecasted weather patterns. In contrast, his conception of a traditional accountant who simply reports net income and summarized financial statements would be similar to the idea of a weatherman who gives a single forecast without any of the supporting data.

---

[16] George H. Sorter, "An 'Events' Approach to Basic Accounting Theory," The Accounting Review 44 (January 1969):12-19.

A summary of the published work by Sorter and others championing the events approach is given in Figure 2, and we will briefly review the major points of its development.

Sorter and Johnson

The first two of the events theorists listed in Figure 2 produced work that was more speculative than empirical in nature. Sorter first defined the term events accounting and postulated its basic features while Johnson simply refined some of Sorter's arguments and defined more rigorously some of his terms. Our discussion below of the theoretical aspect of the events model depends primarily on the pioneering work of Sorter.

George Sorter coined the term "events accounting" in describing his alternative approach to traditional transaction recording. He termed the opposing view the "value theory" wherein "accountants assumed that users' needs are known and sufficiently well specified so that accounting theory can deductively arrive at and produce optimal input values for used and useful decision models."[17] This scheme, he contended, tasks accountants with arriving at near optimal measures of income and capital values (presumed to be the principal input to investment or managerial decisions) and gives rise to many of the strained matching rules that try to optimally pair costs and revenues. Specifically, Sorter criticized the value theory on these counts:

> "1. There are many and varied uses of accounting data and it is therefore impossible to specify input values that are optimal for the wide range of possible uses.

---

[17]Ibid., p. 12.

FIGURE 2

Review of events accounting literature

YEAR	TITLE	AUTHOR	IDEAS
1969	An "Events" Approach to Basic Accounting Theory	Sorter	- "Events" Accounting - Disadvantages of Value Theory - Operational Rules
1970	Toward an "Events" Theory of Accounting	Johnson	- Forecast and Observational Verification Criteria - Definition of Permissible Aggregation - Mathematical Model
1971	A Unified Approach to the Theory of Accounting and Information Systems	Colantoni, Manes and Whinston	- Introduction of Database Concepts - Event Coding - Key Algebra
1975	A Structuring of an Events-Accounting Information System	Lieberman and Whinston	- Three Part Structure - User-Defined Database Characteristics - Self-Organizing Database Capabilities
1976	Design of a Multidimensional Accounting System	Haseman and Whinston	- Hierarchical Organization of Events Database - Definition of Restructuring Functions

"2. For each specified use different users utilize a wide range of different decision models, that they have so far been unable to describe, define, or specify . . .

"3. The value theory is unnecessarily restrictive. Thus, events such as leases and committments have, until recently, tended to be excluded from the accounting universe, . . .

"4. The value theory is not useful in explaining many current developments in accounting. . . .[18]

As a solution to these difficulties wrought by the value theorists, Sorter proposed his events system which would feature the following:

1. It would ask accountants to neither discern the ultimate users of their recorded data nor specify the appropriate user decision models

2. It would prevent accountants from valuing economic events by relieving them of the requirement to uniformly peg a dollar measurement and a time period designation (based on matching) upon a transaction

3. It would simply task accountants with recording the dimensions of an economic event and relax the recognition criteria for a transaction

In conclusion, Sorter clarified his definition of the events approach with these two operational rules:

> A balance sheet should be so constructed so as to maximize the reconstructability of the events being aggregated.
> . . . .
> Each event should be described in a manner facilitating the forecasting of that same event in a future time period given exogenous changes.[19]

A year after Sorter first proposed his theory, Orace Johnson[20] more rigidly defined many of its concepts in an article published in 1970. Among his principal clarifications were these:

---

[18] Ibid., p. 13. [19] Ibid., p. 16.

[20] Orace Johnson, "Toward an 'Events' Theory of Accounting," The Accounting Review 45 (October 1970):641-52.

1. He illustrated different kinds of <u>event</u> definitions and indicated that these terms "must derive their meaning from the possibility of forecast and observational verification"[21]

2. He differentiated among various types of data summation and specified the type of aggregation that would be acceptable for an events theorist because it would allow users to reconstruct the underlying events

Johnson concluded his work with a large mathematical model of the accounting environment, but he, like Sorter, did not specify the actual operational forms of the "events accounting" conventions. That type of investigation was first taken up by the GPLAN database researchers at Purdue University. We will look now at their work.

Work on the GPLAN Database at Purdue

While Sorter and Johnson were advancing their own ideas about the events approach, both of them recognized the existence of tremendous practical difficulties that would be encountered in implementing such a new approach to accounting. Neither of them really addressed this issue, but the 1971, 1975, and 1976 publications cited in Figure 2 above did. Those works are sample reports of the GPLAN project, a large scale database system, at Purdue University. Each group of authors, then at Purdue, recognized the utility of integrating events accounting concepts into database systems. Many of the models they constructed will be very similar to the system we will synthesize in our research.

The first people to attempt to construct a working events model were Colantoni, Manes, and Whinston who also expressly set out to tailor their system to fit the optimal data retrieval capabilities being first used around 1971. They devised an event coding scheme based on binary

---

[21]Ibid., p. 644.

numbers and used it to develop a searching algorithm predicated upon the presence or absence of a particular recordable dimension in a captured event. To see how their system was oriented toward the type of expanded transaction we mentioned earlier, we can look at one of their examples involving cash handling:

> An economic event can be described in terms of a (K+1)-tuple of properties where the first element of the (K+1)-tuple is the event code and the remaining K positions are values of the characteristics for the given event. Events can then be classified according to the characteristics used to describe them. For example, suppose there are six characteristics which are used in describing events. Each event code would then correspond to a 6 position binary number in which the position of the ones would determine the relevant characteristics. A particular type of event, say cash disbursements, might be described by characteristics 1, 2, and 4 (and hence an event code of $11=1\cdot2^0+1\cdot2^1+0\cdot2^2+1\cdot2^3+0\cdot2^4+0\cdot2^5$). To each event involving a disbursement of cash we can associate the sixtuple (1, 1, 0, 1, 0, 0), which is the "inverted" binary representation of the number 11. . . . Likewise, for cash receipts, the characteristics 1, 2, and 5 might be the relevant descriptors. To each event involving cash receipts we can associate the 6-tuple (1, 1, 0, 0, 1, 0) (and the event code $19=1\cdot2^0+1\cdot2^1+0\cdot2^2+0\cdot2^3+1\cdot2^4+0\cdot2^5$).[22]

The GPLAN researchers used their event classifications in the development of a user-oriented retrieval language, called a key algebra, which would allow managers to request information from their database with a minimum amount of knowledge of its storage structure. Their examples of language use and queries were limited, but at least they recognized the importance of this casual user interface.

At the end of their work, Colantoni, Manes, and Whinston raised two issues: (1) the nature of the cost-benefit decisions in an events-

---

[22] Clause S. Colantoni, Rene P. Manes, and Andrew B. Whinston, "A Unified Approach to the Theory of Accounting and Information Systems," The Accounting Review 46 (January 1971):90-102.

based system and (2) the need for criteria to be used in recognizing a reportable event. However, they did not try to address these questions fully concluding instead that both issues could be dealt with only on an individual firm basis.

The last two of the events articles, coauthored by Whinston with Lieberman[23] and Haseman,[24] were published only recently and incorporated many new database techniques into their accounting systems. Specifically, they did the following:

1. They introduced the concept of a three-part database structure (mass database, user-defined structures, and user-defined function) very similar in form to the three language model (schema definition, subschema definition, and data manipulation) proposed by the CODASYL Committee[25] in 1971

2. They outlined the logical framework of a hierarchically structured events accounting system

3. They described the processes involved in self-organizing databases involving the transformation, based upon a stream of user inquiries, of unstructured data files into logical data banks

In both articles, the accounting examples given were brief, and neither publication addressed at length the issue of developing criteria for reportable events.

It is difficult to summarize the contributions of the GPLAN researchers in terms of our proposed work because their articles do not

---

[23] Arthur Z. Lieberman and Andrew B. Whinston, "A Structuring of an Events-Accounting Information System," The Accounting Review 50 (April 1975):246-58.

[24] William D. Haseman and Andrew B. Whinston, "Design of a Multidimensional Accounting System," The Accounting Review 51 (January 1976): 65-79.

[25] CODASYL Programming Language Committee, Data Base Task Group Report (New York: Association for Computing Machinery, 1971), pp. 13-22.

go into sufficient detail on operational forms and actual working systems. The important fact to note, however, is that all of their work involves the use of hierarchical or tree-like logical database organization. As will be seen in the next section, this is in marked contrast to our proposed model which will use the relational framework of data organization.

Summary of "Events" Accounting Proposals

The above review of the GPLAN researchers concludes our examination of the background literature in the field of accounting. For our purposes, the major contribution of the events accounting theorists was their suggestion and later incorporation of the <u>expanded transaction concept</u> (more dimensions and detail in event recognition) into accounting models. This wide-ranging view of accounting information systems will facilitate our transformation of the traditional double-entry framework into a database type of model in the next chapter.

We now move to the field of computer science and a review of the theories underlying the second major part of our proposed new system: database technology and the relational systems of E. F. Codd.

Review of Database Concepts

Introduction

The concept of a computer database has its origins in man's wish to build a machine with thought capabilities paralleling his own. In particular, computer scientists and developers of management information systems have been disappointed with data processing systems that use files of information for one purpose only and are unable to integrate all the data received into a total information system. Their ideal in this respect would be a memory that is totally associative or able to link through all of its processed data to answer any question given it. To perform these tasks requires data memories that are content-adressable or able to be retrieved based on their data values rather than their storage locations. Efforts to produce such systems have resulted in the database processors of today. The past ten years has seen a proliferation in the number and type of these database systems and a concomitant increase in database technologies, and it now seems probable that most organizations will eventually switch their data processing configurations to these newer and more integrated systems. Richard Nolan mentions two key aspects of a companywide database:

> "1. The data that computer programs use are considered an independent resource in themselves, separate from the computer programs.
>
> "2. There is an art and an approach to managing and structuring a company's computer-readable data as a whole, so that they constitute a resource available to the organization for broadrange applications--especially on an ad hoc basis."[26]

We will not review here extensively the advantages and disadvantages of the database approach or the specific database management systems (DBMS) currently available. Instead we will restrict our discussion to the logical properties of databases and to a review of one of the most important objectives of a database system--data independence.

C. J. Date defines data independence as "the immunity of applica-

---

[26] Richard L. Nolan, "Computer Data Bases: The Future is Now," Harvard Business Review 51 (September-October 1973):101.

tions to change in storage structure and access strategy,"[27] but we can define it more simply as the separation, to the maximum extent possible, of the physical view or storage structure of data from the logical view or user perception of data. As can be seen in Figure 3, the architecture of most database systems reflects the importance of data independence. The portion of the system below the dotted line (physical structure) is concerned with matters such as character representation, record blocking, and access method while the portion above (logical structure) is concerned with the user's view of the named data elements and their relationships with each other. Our work with events accounting will concern the logical view of data only, and we will turn now to a discussion of the three alternatives in this area: (1) hierarchical structures, (2) network structures, and (3) relational structures. To facilitate comparison of the three methods, we will use the small conceptual model outlined below.[28]

Our small conceptual model or database is to contain information about two kinds of entities: (1) employees of an accounting firm and (2) their skills. At a given time, each of the employees listed below possesses the indicated skills:

<u>EMPLOYEES</u>	<u>SKILLS</u>
Dick	Auditing
Cathy	Computers and Statistics
Ann	Auditing and Tax
Bob	Auditing, Computers, and Statistics

---

[27] C. J. Date, <u>An Introduction to Database Systems</u> (Reading, Massachusetts: Addison-Wesley Publishing Company, 1975), p. 8.

[28] This sample database is a modification of one used by Date, pp. 58-9.

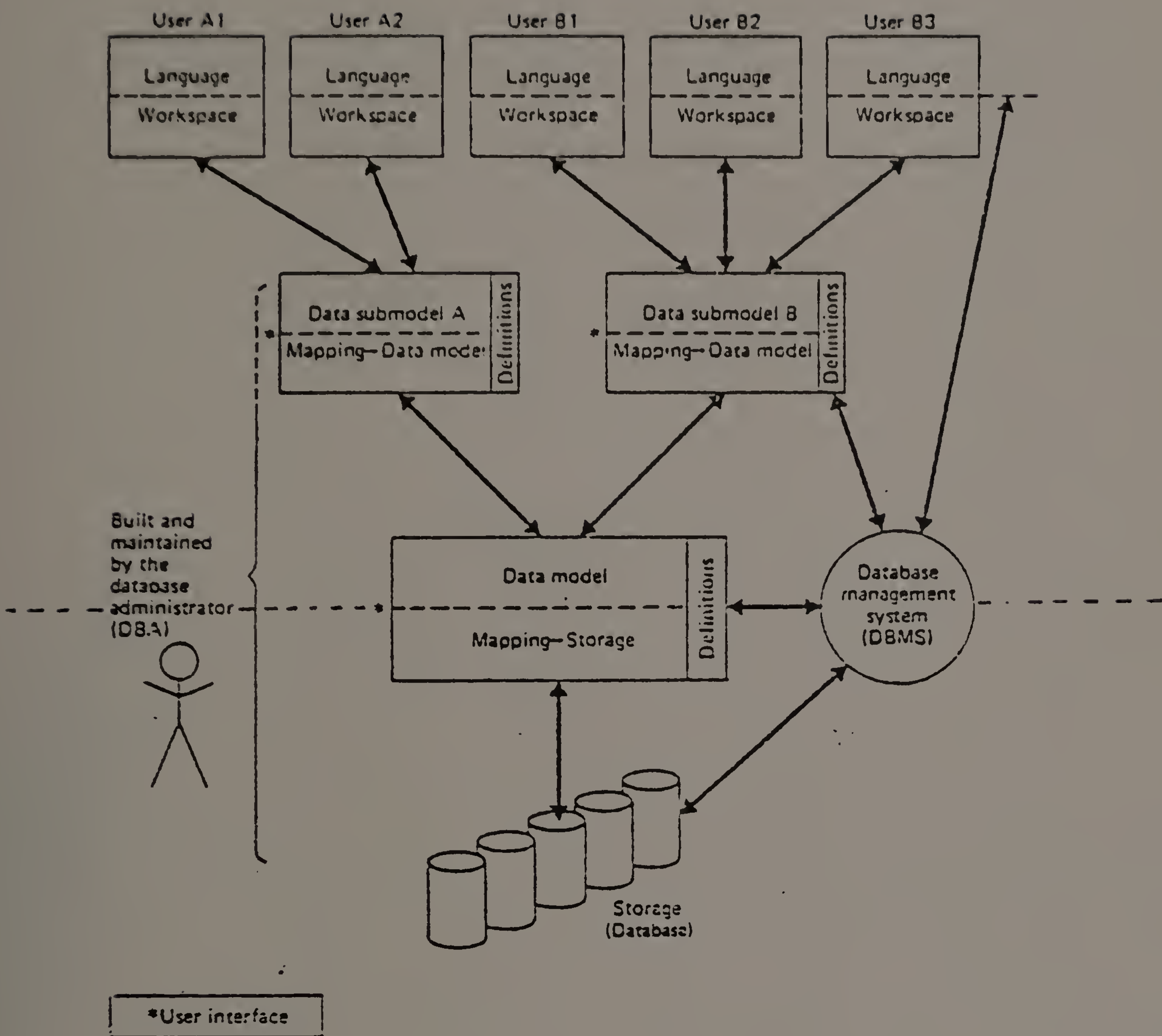


FIGURE 3--Database system architecture

Source: Date, p. 12.

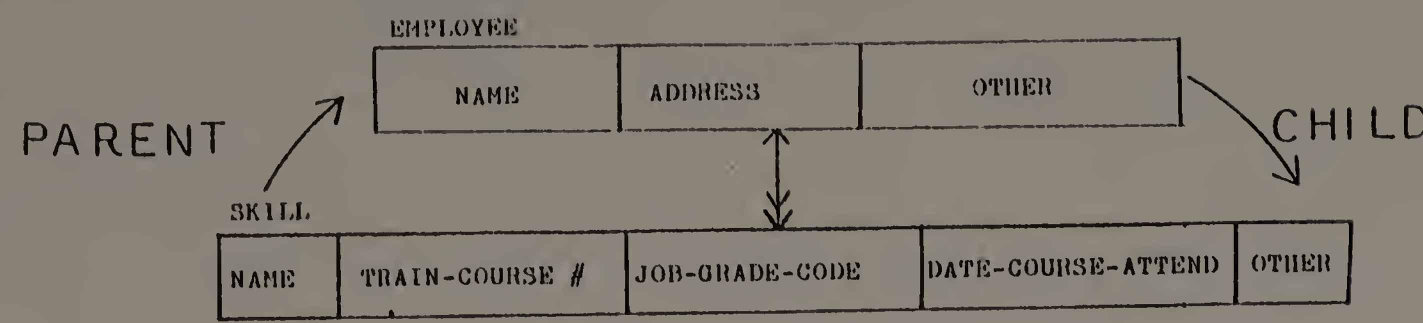
For each employee, the database contains various personal details such as address. For each skill, it contains an identification of the appropriate training course, an associated job grade code, and other information. The database also contains the date each employee attended each course where applicable . (The assumption is that attendance at the course is essential before the skill can be said to have been acquired.)

We now move to an examination of our first alternative logical view: data arranged in hierarchies.

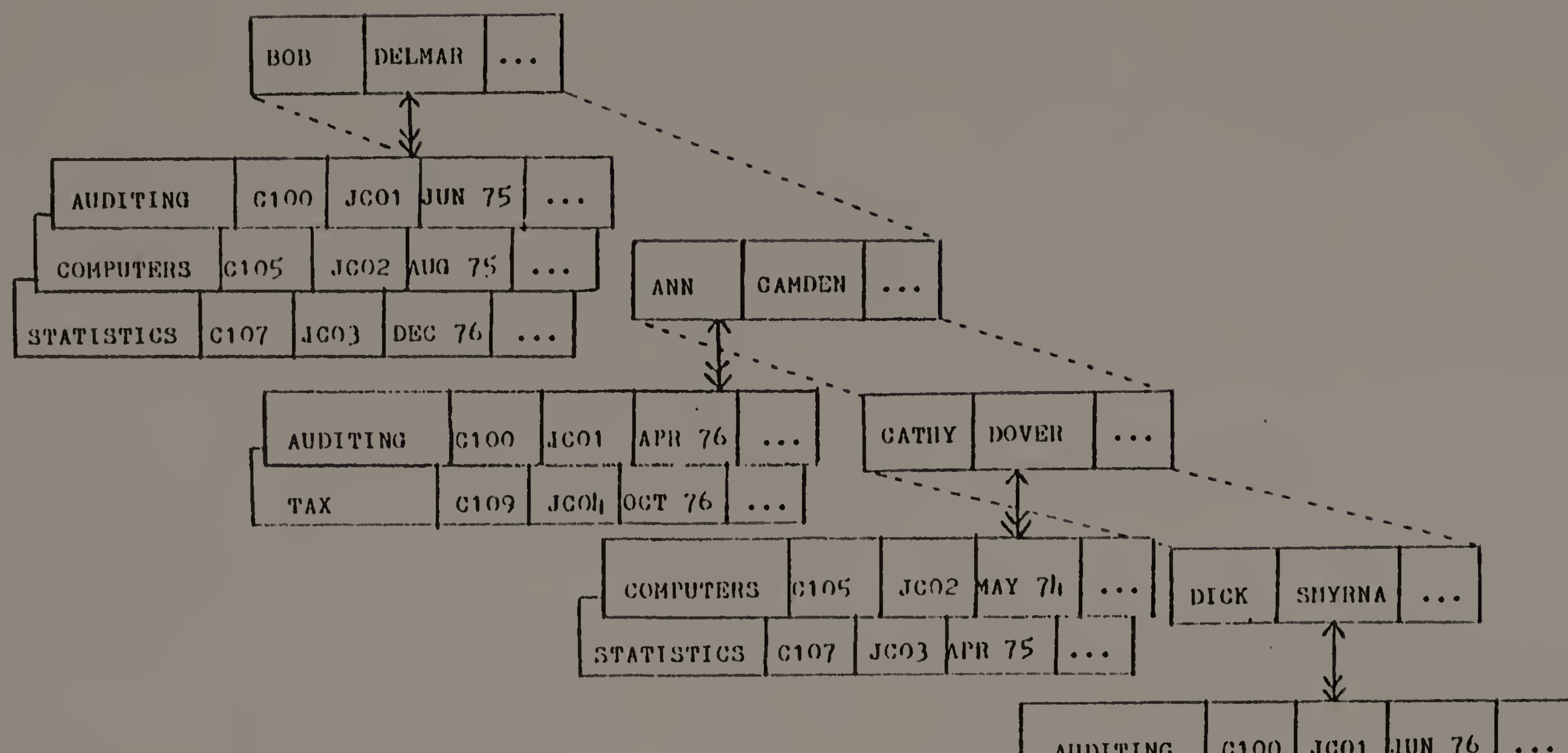
Hierarchical Structures

When data is presented in a tree-like manner reflecting a one-to-many (1-to-n) type of hierarchical relationship, the kind of logical structure seen in Figures 4 and 5 results (the top portion of each figure simply dispays the record layout for each entity while the bottom portion reflects the actual values taken from our small conceptual model). Note that there are two possible hierarchical choices for our sample database because the relationship between employees and skills is actually many-to-many (m-to-n) instead of one-to-many. Another way of saying this is that if we combined the two records, we could have either SKILLS being a repeating group in the EMPLOYEE record or EMPLOYEES being a repeating group in the SKILL record.

Using the analogue of a family tree, the relative positions in a hierarchical model are commonly designated as <u>parents</u> and <u>children</u> to indicate location above or below other positions, and the data elements in a hierarchical scheme take on their full logical significance only when they are viewed within the context of this relative position. For example in Figure 4, the data element "Tax" is only understood to be

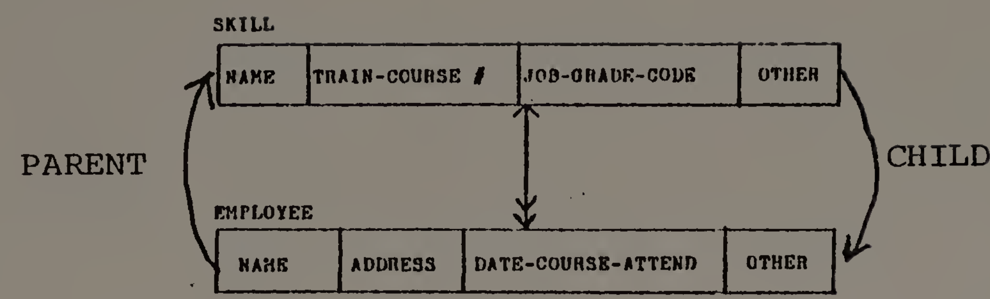


(a) Record layout for hierarchical model (EMPLOYEE superior)

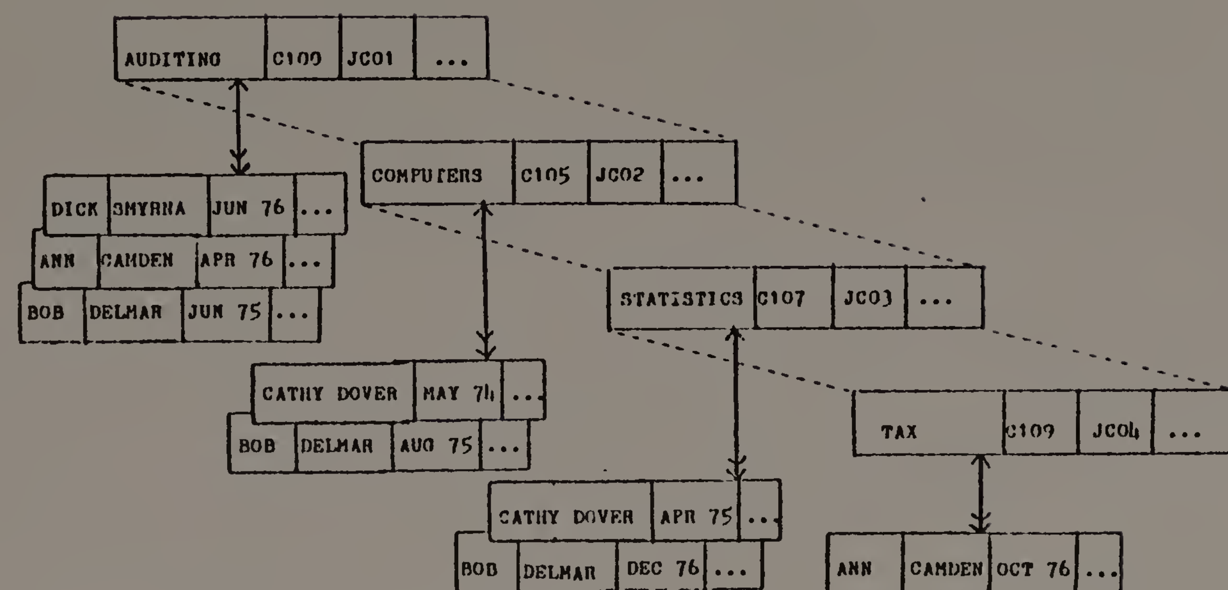


(b) Values for sample database

FIGURE 4--Hierarchy with EMPLOYEE superior



(a) Record layout for hierarchical model (SKILL superior)



(b) Values for sample database

FIGURE 5--Hierarchy with SKILL superior

the name of a skill possessed by Ann (rather than by Cathy, Bob or Dick) when it is determined to be the logical child of one particular parent.

An advantage claimed for hierarchical structure is that it mirrors many of the logical properties seen in organizations, such as management frameworks and charts of accounts, and is thus a very natural way for a user to picture data. However, it also has the following major disadvantages:

1. Because it is meant to model 1-to-n type of relationships, a choice must be made regarding the direction of the hierarchy when it is used to model m-to-n relationships such as our employee-skill example. The resulting choice may not be the one that is most natural for all users

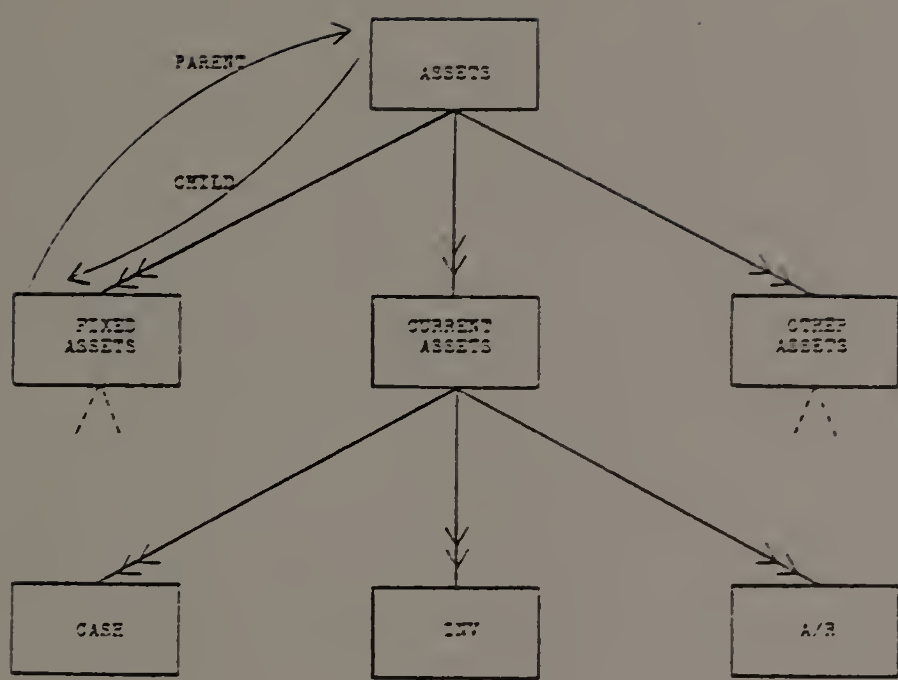
2. Because much of the database information is dependent upon position within the hierarchy, adding and deleting problems will occur when logical children have no logical parents to correspond to

In conclusion, we note again that it was this hierarchical type of framework that was used in the GPLAN research at Purdue University. Two of their user-defined hierarchical structures are pictured in Figure 6.

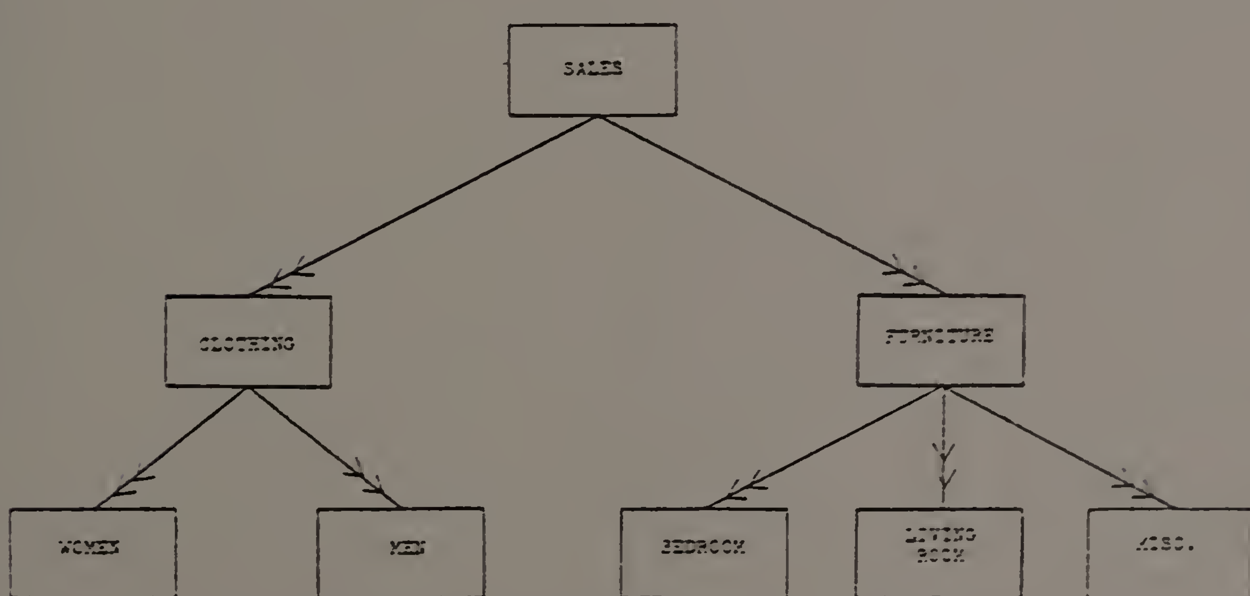
Network Structures

In the hierarchical models discussed above, each logical child is restricted to just one logical parent. If we relax this restriction and allow multiple parents, a more general type of logical model--the network or plex structure--can be formed.

For example purposes, we can show instances of a network model with our sample database of skills and employees if we expand it to include a third type of entity such as department of an accounting firm (to include Public Reporting and Management Services). As can be seen in the two plex occurrences in Figure 7, we are now allowed to let a child entity (EMPLOYEE) have two or more parent entities (SKILL and DEPARTMENT).



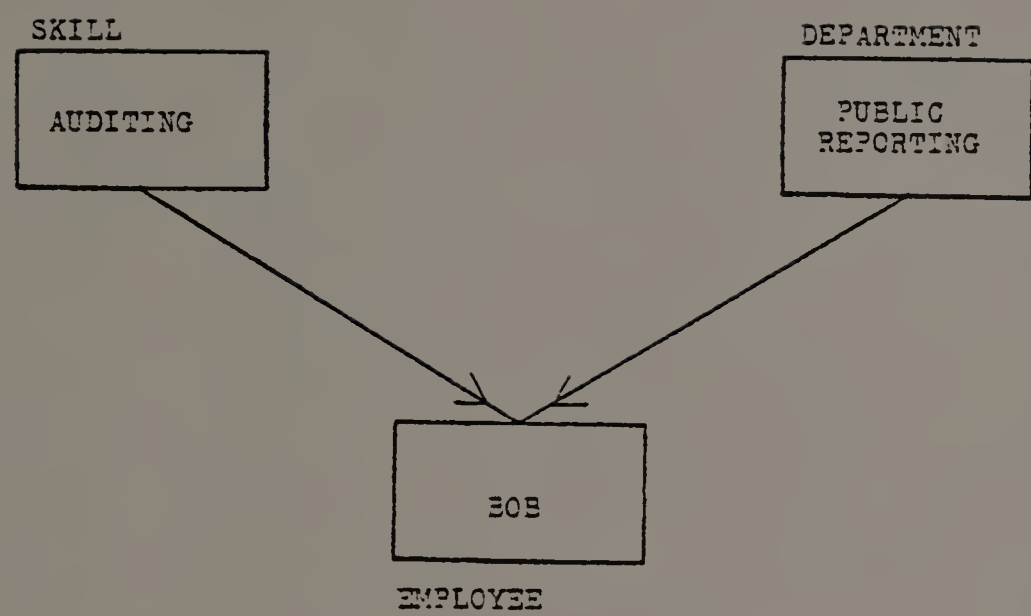
(a) User-defined hierarchy for an accountant



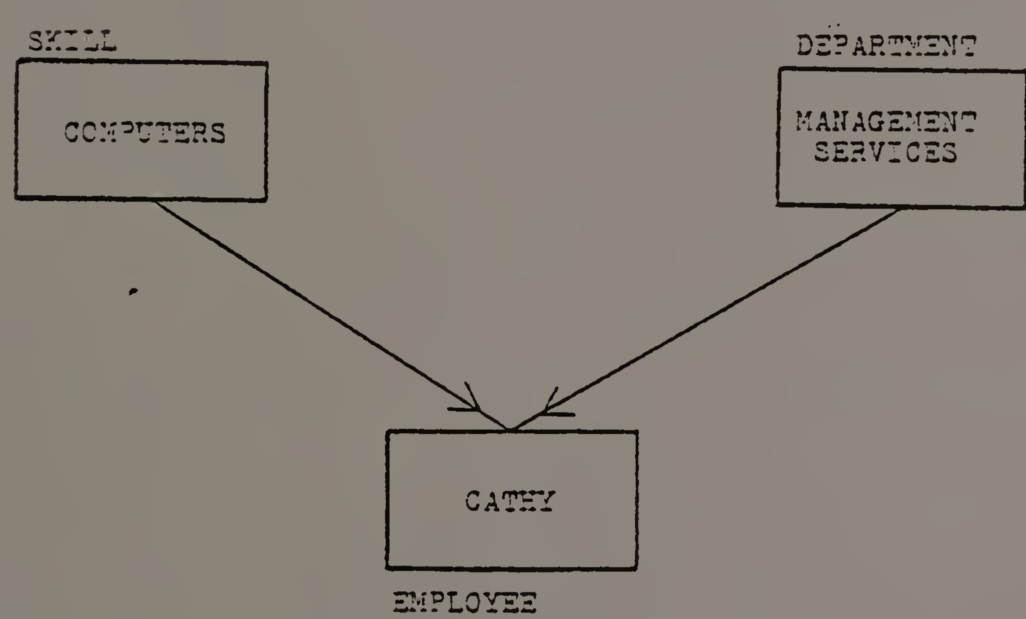
(b) User-defined hierarchy for a salesman

FIGURE 6--GPLAN hierarchical structures

Source: A. Z. Lieberman and A. B. Whinston, "A Structuring of an Events-Accounting System," The Accounting Review 50 (April 1975), 252.



(a) Occurence of multiple parents for employee Bob



(b) Occurrence of multiple parents for employee Cathy

FIGURE 7--Plex structures with multiple parents

Network database structures are commonly referred to as DBTG (Data Base Task Group) models because they are the logical framework advocated by the CODASYL Committee.[29] DBTG terminology replaces the ideas of parent and child elements with the concept of a set with one owner and multiple member elements. For example in Figure 8(a), we show the set structure having the entity EMPLOYEE as owner and the entities SKILLS as members (this set structure would correspond to the hierarchy pictured in Figure 4). In Figure 8(b) we show the opposite, the set structure having the entity SKILL as owner of the set and the entities EMPLOYEES as members (this second set structure corresponds to the Figure 5 hierarchy).

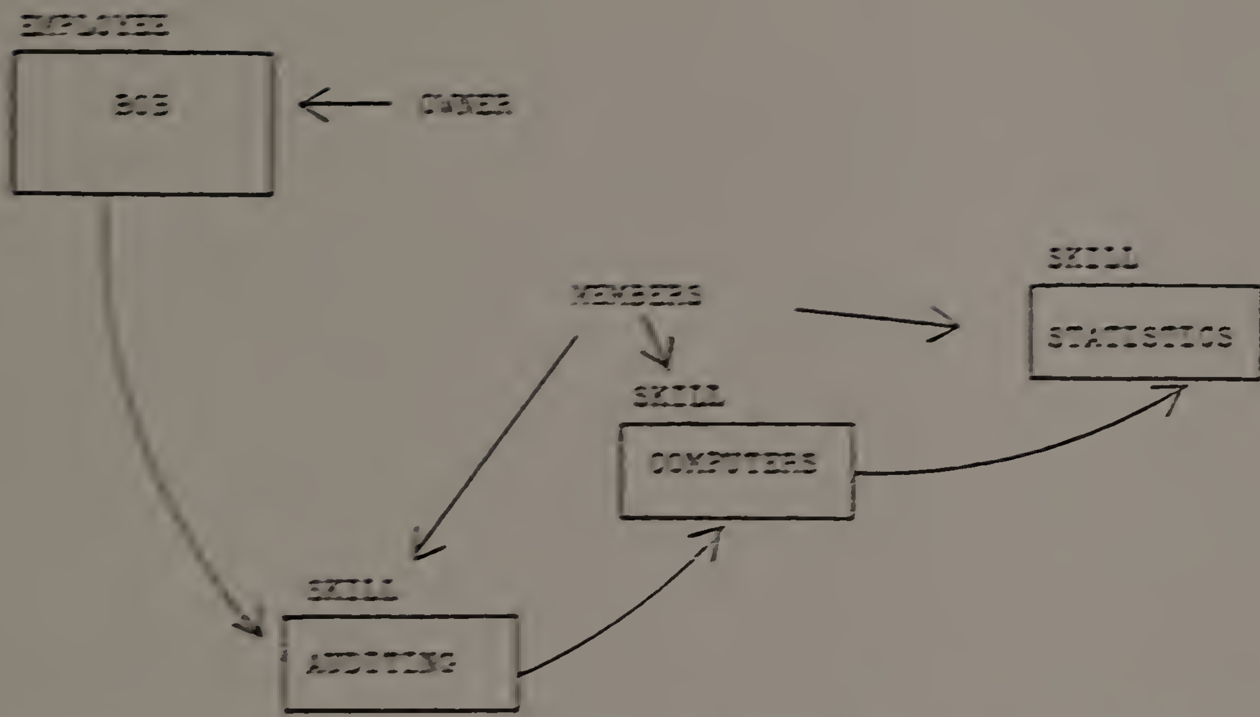
At this point, we will emphasize again that our discussion of database theory has been restricted to logical views only. The terms set, owner, and member are logical concepts that can be implemented on actual systems in a multitude of ways. For illustrative purposes, we show in Figure 9 four physical implementations of logical sets recommended by the DBTG.

Network database systems can be used to model concurrently the type of m-to-n relationships between entity records (such as our SKILL-EMPLOYEE example) that could not be accommodated by hierarchical models. To establish such a relationship for our sample database, we must do the following:

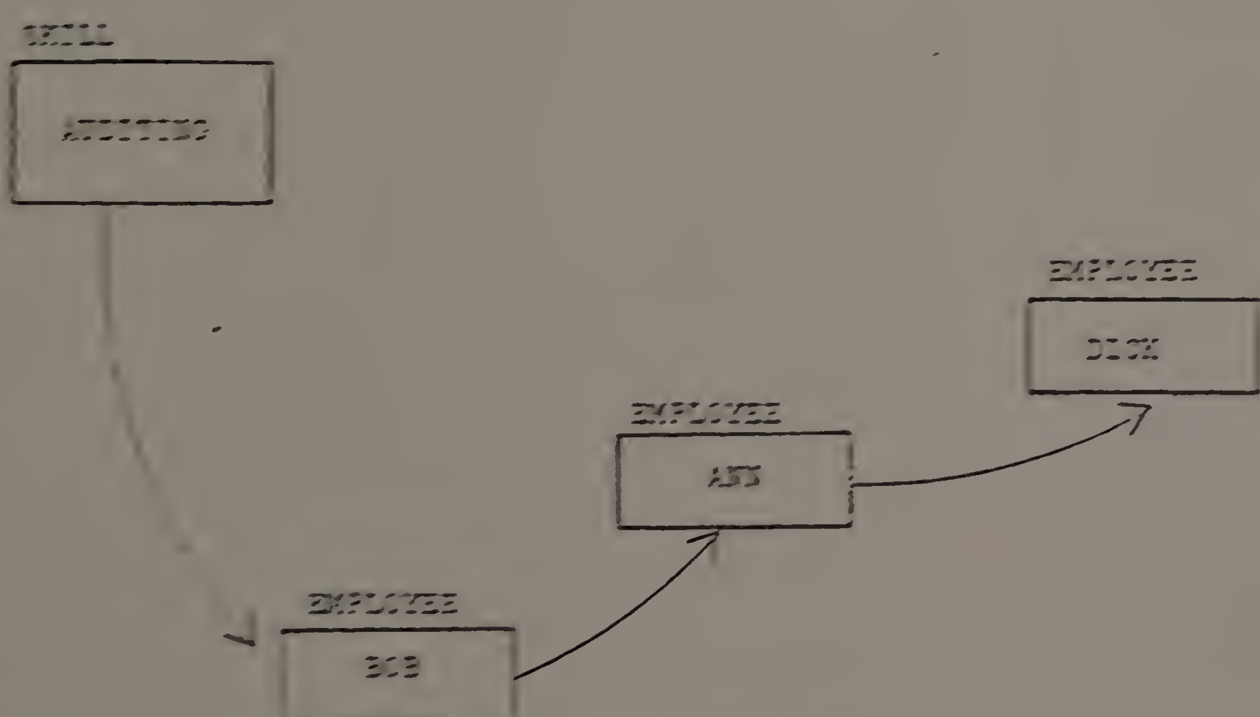
1. Introduce a new type of record called a link or relationship record which may or may not contain information about the relationship (our model contains the date of course attendance).

---

[29] CODASYL Programming Language Committee, pp. 67-148.



(a) Occurrence of set with EMPLOYEE as owner and SKILL as member



(b) Occurrence of set with SKILL as owner and EMPLOYEE as member

FIGURE 8--Set structure of DBTG networks

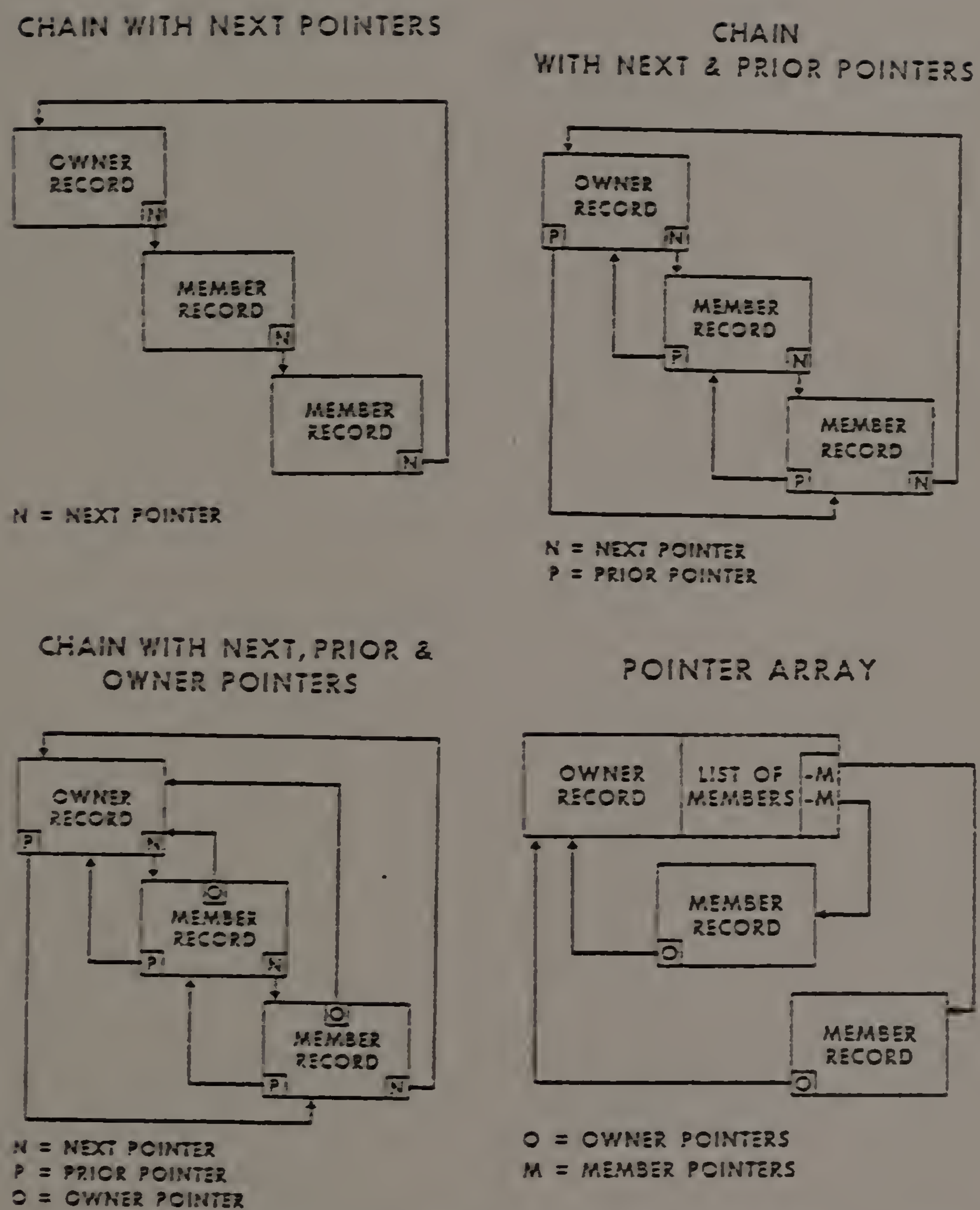


FIGURE 9--Physical database implementations of CODASYL "Set" concept

Source: CODASYL Programming Language Committee, Data Base Task Group Report (New York: Association for Computing Machinery, 1971), pp. 48-52.

2. Declare two sets for the model, each of which will use the corresponding entity records as _owners_ and the link records as _members_ (our sample would have the first set with EMPLOYEE as _owner_ and LINK as _member_ and the second set with SKILL as _owner_ and LINK as _member_)

The resulting network structure for our sample is pictured in Figure 10. We can see that by traversing the pointers, it is possible to show that not only can one skill (AUDITING) be possessed by multiple employees (BOB, ANN, DICK), but also that one employee (BOB) can possess multiple skills (AUDITING, COMPUTERS, STATISTICS). Figure 10 is in essence a combination of Figures 4 and 5.

Note that, in contrast to the hierarchical framework where relationship information is based on position within the hierarchy, network structures specify explicitly the relationships between entities through the use of chains or some kind of pointer scheme. Although this leads to more flexibility, it also compromises this structure's ability to support data independence. This fact is noted by Date:

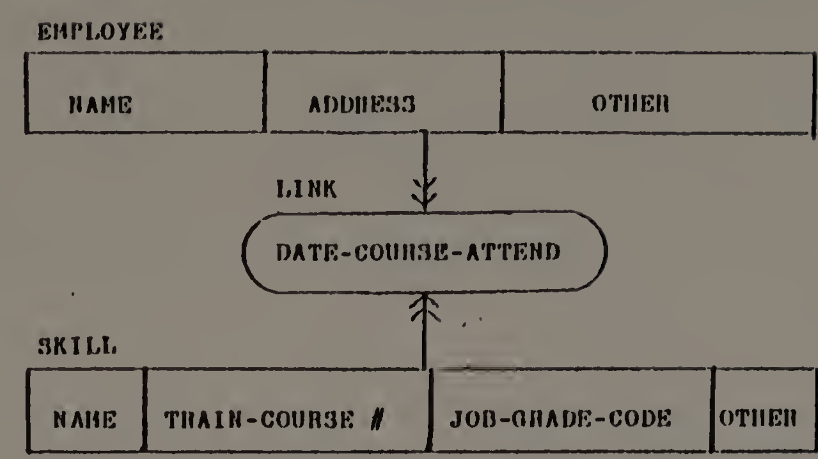
> The major disadvantage of the network model is simply that it is too close to a storage structure. The user has to be thoroughly aware of which chains do and do not exist, and his . . . programming rapidly becomes extremely complex . . . More significantly, the chains are directly visible to the user and hence must be directly represented in storage somehow. . . . There is thus a risk that the user will become locked into a particular storage structure, contrary the aim of data independence.[30]

Relational Structures of Codd

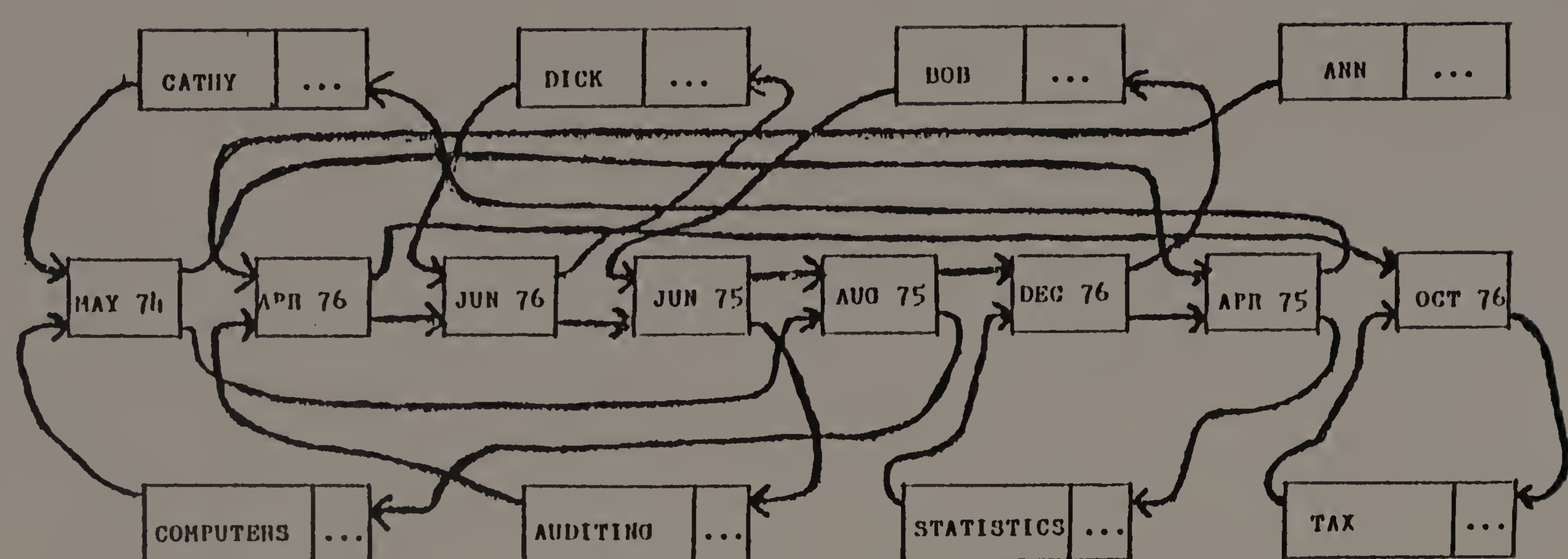
In contrast to the hierarchical and network models illustrated above where representing relationships by chains or relative positions for even two entities can rapidly become complex, the relational model of E. F. Codd structures both data entities and relationships between

---

[30] Date, pp. 52-3.



(a) Record layout for network model



(b) Values for sample database

Figure 10--Network structure of sample database

data entities in a relatively straightforward manner--as two-dimensional tables. An essential aspect of such a model is its simplicity as was noted in an introduction to relational systems by James Martin:

> Throughout the history of engineering a principle seems to emerge: Great engineering is simple engineering. Ideas which become too cumbersome, inflexible, and problematic tend to be replaced with newer, conceptually cleaner ideas which, compared to the old, are esthetic in their simplicity. When a programmer's block diagram looks like tangled cobweb the time has come to rethink the entire program.
>
> Data-base systems run the danger of becoming cumbersome, inflexible, and problematic. The logical linkages tend to multiply as new applications are added and as users request that new forms of query be answerable with the data. A high level of complexity will build up in many data-base systems. Unless the designers have conceptual clarity they will weave a tangled web.[31]

A key phase of structuring a relational database is the process of reducing all data records to two dimensions. This reduction is called normalization and was first outlined by Codd as a three-step process progressing through first normal form (1NF), second normal form (2NF), and third normal form (3NF). We will not outline these procedures in detail here (the interested reader may consult Codd[32]); instead we will take an abbreviated look at overall normalization by deriving the tables for our sample employee-skill database.

Starting with the hierarchy of Figure 5 (a start with Figure 4 will achieve the same end), we can consider that we have a SKILL record with EMPLOYEE information as a repeating group as represented in the top table of Figure 11. The normalization process simply breaks down the informa-

---

[31] James Martin, Computer Data-Base Organization (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975), p. 149.

[32] E. F. Codd, "Further Normalization of the Data Base Relational Model," in Data Base Systems, ed. R. Rustin (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972), pp. 33-64.

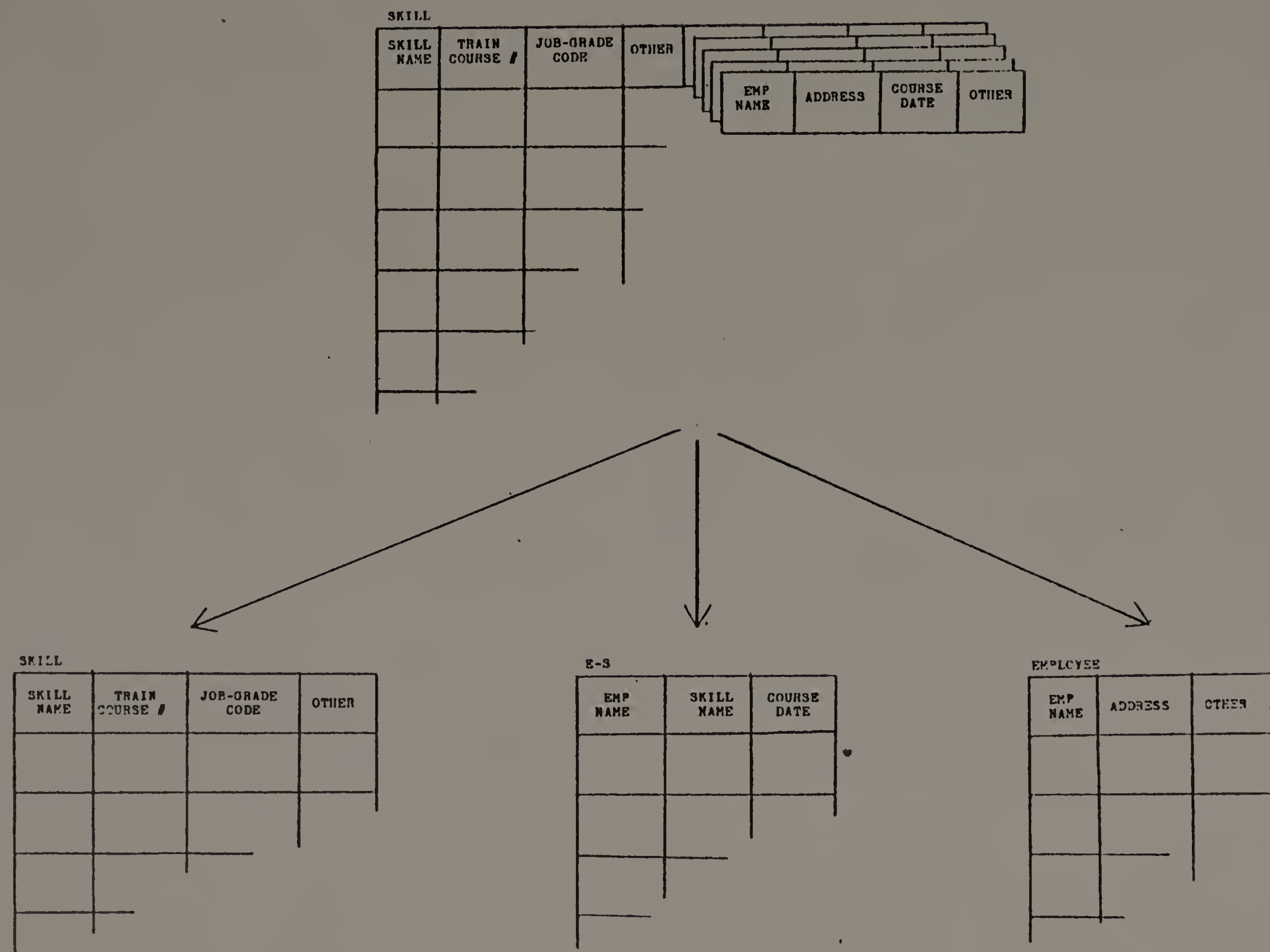


FIGURE 11-Normalization of sample database

tion in this elevated or three-dimensional structure into the flat tables pictured below the arrows. The result is that we have three tables with each describing a <u>separate concept</u>: one (SKILL) describing the skill entities, one (EMPLOYEE) describing the employee entities, and one (E-S) describing the relationship between skills and employees. In addition to its simplicity, this normalization process gives the resulting database structure a number of very desirable properties in terms of update, insertion, and deletion of entities over time-varying circumstances.

The full relational structure for our sample database is illustrated in the three tables pictured in Figure 12. Each instance of an entity or relationship is represented by a <u>row</u> (also called <u>tuple</u>) in a <u>table</u> (also called <u>relation</u>) and the names of the data fields are represented by <u>columns</u> (also called <u>domains</u>). We note that, in terms of information content, Figure 12 corresponds exactly to Figures 4, 5, and 10 but does so with an obvious lack of complexity. The kind of m-to-n relationships that were difficult to portray before have become relatively straightforward in form.

The relational model has a number of other advantages, among them the following:

1. Because Codd based his original work on the mathematical theory of sets and relations, a normalized relational model is a mathematically sound system; a trait which gives it advantages in maintaining the integrity of its application program interfaces over time

2. Its foundation and development have been unbiased by a priori ideas relating to the storage structures of early computer systems, and it eliminates consideration of both these storage structures and access strategy from the user interface

3. The view of the data is very clear, and users are not burdened with following pointers or traversing trees to discern possible information about data relationships

EMPLOYEE

EMP-NAME	ADDRESS	OTHER
DICK	SMYRNA	. . .
CATHY	DOVER	. . .
ANN	CAMDEN	. . .
BOB	DELMAR	. . .

(a) Table representing employees

E-S

EMP NAME	SKILL-NAME	COURSE DATE
DICK	AUDITING	JUN 76
CATHY	COMPUTERS	MAY 74
CATHY	STATISTICS	APR 75
ANN	AUDITING	APR 76
ANN	TAX	OCT 76
BOB	AUDITING	JUN 75
BOB	COMPUTERS	AUG 75
BOB	STATISTICS	DEC 76

(b) Table representing relationship between employees and skills

SKILL

SKILL NAME	TRAIN-COURSE #.	JOB-GRADE-CODE	OTHER
AUDITING	C100	JC01	. . .
COMPUTERS	C105	JC02	. . .
STATISTICS	C107	JC03	. . .
TAX	C109	JC04	. . .

(c) Table representing skills

FIGURE 12--Relational structure of sample database

4. It possesses two types of well defined query languages, the relational calculus and the relational algebra, which are simple, complete, and non-procedural (no knowledge of database organization needed to use them)[33]

5. It offers a high degree of flexibility with its query language's ability to "cut and paste" the various tables to better fit a particular user's needs

6. It offers advantages in the implementation of security controls[34]

In summary, we note that a significant portion of database theorists favor Codd's model and consider it the best candidate for further development of a data independent system. Codd himself is now working on a new casual interface language called "RENDEZ-VOUS"[35] which if successful would make his systems even more desirable for our purposes because it would expand considerably the database's flexibility and the ad hoc user population. We also note that, unlike the network and hierarchical models, wide scale use of actual relational systems has not yet taken place.

The above review of Codd's work concludes our examination of database systems. We now move to a synthesis of the relational model with the <u>events accounting</u> concepts reviewed earlier in this chapter.

---

[33] E. F. Codd, "Relational Completeness of Data Base Sublanguages," in <u>Data Base Systems</u>, ed. R. Rustin (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972), pp. 65-98.

[34] E. F. Codd, "Access Control for Relational Data Base Systems," Presented at BCS Symposium on Relational Database Concepts, April 1973, British Computer Society, London, 1973.

[35] E. F. Codd, "Seven Steps to RENDEZVOUS with the Casual User," <u>Proceedings of IFIP TC-2 Working Conference on Data Base Management in Systems</u> (Amsterdam: North-Holland Publishing Company, 1974), pp. 179-200.

Proposed Synthesis of Relational Databases and Events Accounting

We have considered selected publications in accounting and computer science that support (as mentioned in Chapter 1) our proposed combination of (1) Codd's relational framework with (2) an events-based accounting information system. Specifically, we have demonstrated the following:

1. That accounting is best viewed as a significantly large and integrated part of an enterprise's information system rather than as an insular methodology for accumulating and reporting financial information

2. That recent research has indicated some desirable characteristics that such integrated accounting systems ought to possess such as (1) low levels of data aggregation, (2) multidimensional measures, and (3) high degrees of flexiblility in responding to users

3. That the events accounting theories of George Sorter and others provide guidelines useful in the construction of accounting systems featuring the desirable characteristics mentioned above

4. That given the desirability of events accounting systems, a method of implementing them must be found that will accommodate their data capture, storage, transmission, output preparation and user interface needs. Logically these tasks fall to database systems, and the work of the GPLAN researchers with hierarchical models represents the first steps in this integration process

5. That there exists enough research supporting the relational model to warrant its consideration as a basis for an events based system[36]

Having thus provided the theoretical basis for our proposed synthesis, we proceed now to a more specific consideration of an events-based

---

[36]While this research was in progress, two authors at the University of Minnesota proposed introductory work combining the relational model with accounting concepts. Specifically, their research dealt with the following issues:

1. The hierarchical model's lack of data independence
2. Normalization of a chart of accounts
3. Relational examples of accounting data, but not a complete system

The approach they took differs substantially from the one we will use in the next two chapters, but they did offer some insights that we will incorporate into our system. See Gordon C. Everest and Ron Weber, "A Relational Approach to Accounting Models," The Accounting Review 22 (April 1977): 340-59.

relational system. In the next two chapters, we will construct and describe an accounting information system similar in form but differing in substance from that model given in Figure 1 (see page 10). In particular, we intend to differ from that framework in two major respects:

1. We will not adhere to the conventions of the monetary principle and double-entry in structuring the economic events to be allowed into the system from the environment

2. We will not use the traditional system of classification, summation, and aggregation in maintaining our data and preparing it for output; instead we will maintain information concerning both our economic entities and the relationships between them as a relational database system

In Chapter 3, we will build a limited system and contrast it with a traditional accounting model in order to demonstrate both how such a system would work and some of its possible advantages, while in Chapter 4 we will feature development of a normative theory. Our expressed intention in limiting the model to a small subsystem first is to provide a mechanism for exposition of our new conventions.

We proceed now to a limited extension of the traditional accounting model.

CHAPTER III

RELATIONAL MODEL OF A SMALL RETAIL ENTERPRISE

Introduction

This chapter will feature a two part presentation of opposing methodologies for recording the business transactions of a small retail enterprise. The first part will show the manual implementation of the conventional double-entry system while the second will exhibit a limited relational model incorporating the ideas of both Sorter and Codd. Normative aspects of the second model will be discussed in Chapter 4.

As we stated at the end of the last chapter, our purpose in presenting a concrete example at this point is to show the reader just what a relational accounting system might look like. After we have a full example presented, we will be better able to discuss its relative advantages. With this expositive purpose in mind, we will develop our chapter topics in the sequence indicated below.

First, we will discuss the changes inherent in moving from the traditional accounting model to a relational system. In particular, we will contend that most present accounting systems implemented on computers represent extensions of manual procedures and not changes in the fundamental nature of the recording process. We will also argue that certain basic accounting conventions normally included in a chart of accounts might not be appropriate for use in a database environment.

Second, we will present the underlying details of our example system in the form of one month's operating data for a small retail enterprise called Wilson Company. We will then use the Wilson company data to exhibit a manual double-entry system.

Third, we will begin the process of constructing a database model. Having shown the conventional framework, we will discard its conventions and begin anew the process of abstracting an events-based accounting system for Wilson Company. This section will rely heavily on the Entity-Relationship (E-R) modeling approach of Peter Chen.[1]

Fourth, we will present a complete relational system. Once we are finished with the E-R abstraction process mentioned above--our final product will be called a data model of the small enterprise--we will be able to map Wilson's operating data into a set of relations as defined by Codd.

Fifth, we will use the system constructed above to manifest our model's ability to support "events accounting" features mentioned in Chapter 2, that is: (1) multidimensional measures, (2) disaggregated data, and (3) flexibility in responding to ad hoc needs.

Finally, we will show the use of our new system in responding to accounting needs such as auditing of records and enforcement of internal control.

We proceed now to a comparison of manual and database methods.

Contrast of Conventional and Database Environments

Introduction

In Chapter I, we quoted two accounting theorists, John Wheeler and R.L. Mathews, who had called for fundamental changes in the accounting

---

[1]Peter P. Chen, "The Entity-Relationship Model--Toward a Unified View of Data," ACM Transactions on Database Systems 1 (March 1976):9-36.

process in order that the profession be better able to avail itself of advances in computer data processing. Their sentiments are echoed in a recent book by T. W. McGrae who voiced similar feelings while differentiating between accounting technology and accounting systems:

> We must be careful to differentiate between accounting technology and accounting systems.
>
> The technology of accounting is concerned with the physical artifacts which are employed to process accounting data. These artifacts range all the way from quill pens to remote controlled computer systems. Accounting systems are concerned with the classifying and structuring of accounting data. . . .
>
> It is possible to change the accounting technology without changing the accounting system and vice versa. This is illustrated by the . . . example set out later in this chapter. An unchanged system is processed on a changed technology.
>
> The clear distinction between system and technology is important since many accountants suffer from the delusion that because they have changed the accounting technology they must automatically have affected dramatic changes in the accounting system. This is not so. The new computer technology provides a dramatic improvement in the speed of data processing and automatic control but this potential cannot be realized without affecting major alterations in the accounting system. In other words the accountant must develop more sophisticated accounting models to benefit from computer technology.[2]

We present this quote from McGrae here because it will be especially helpful in pointing out the differences between the two accounting models to be illustrated in the chapter: (1) the manual conventional system and (2) the computerized relational system. Each of them is at opposite ends of a continuum that illustrates the processing evolution of accounting models as shown in Figure 13. In the sections below, we discuss the changes inherent in moving through the continuum shown in Figure 13.

---

[2] T. W. McGrae, Computers and Accounting (London: John Wiley & Sons Ltd., 1976), p. 39.

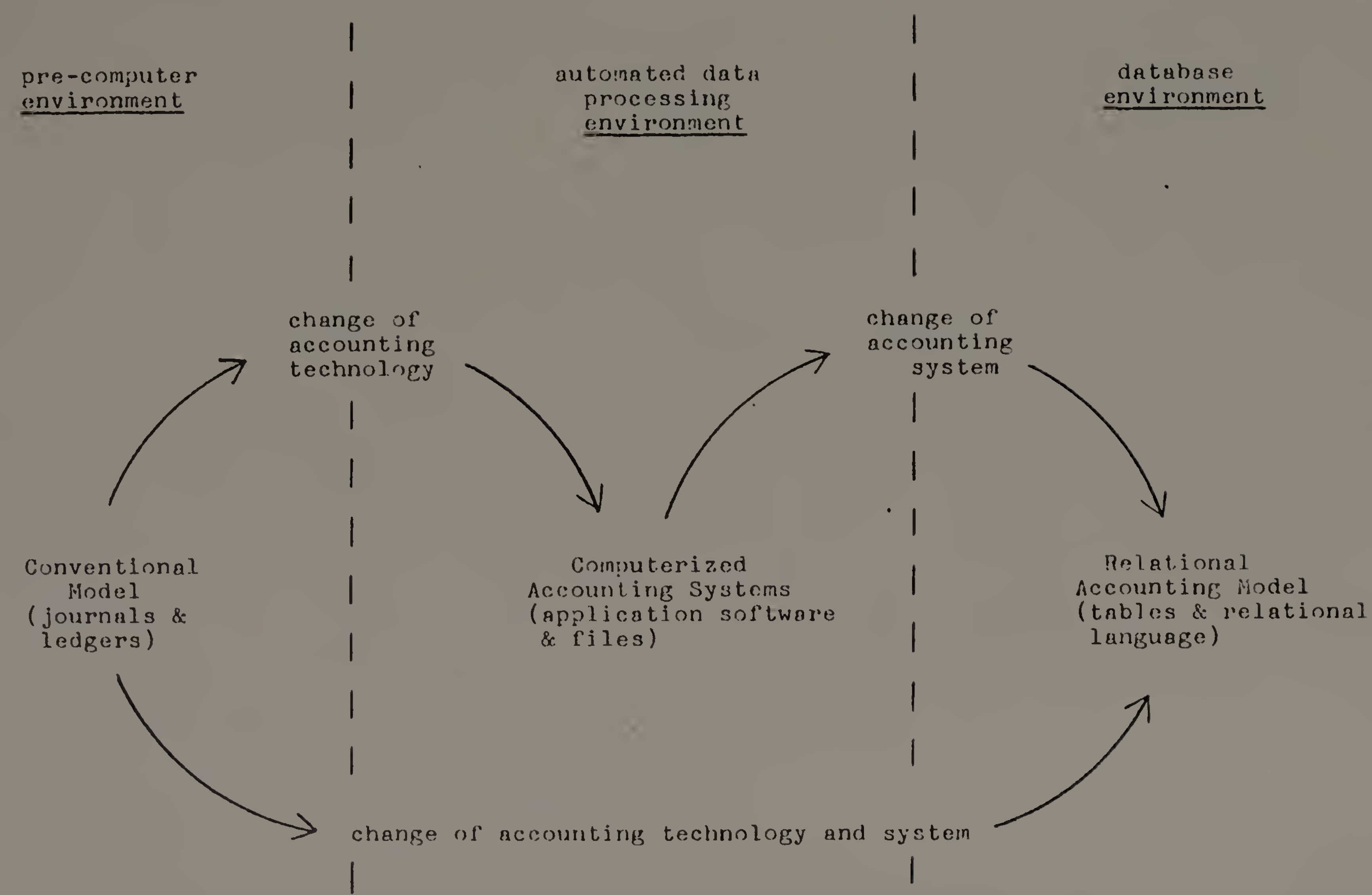


FIGURE 13--Processing continuum of accounting models

Pre-Computer Environment

The model on the far left corresponds to the accounting information system shown in Figure 1 of Chapter 2 (see page 10). In such a scheme, data is captured from source documents, is adapted to conform with the principles of double-entry and monetary valuation, and is entered into some kind of journal (special journals for transactions that occur often such as sales, purchases, cash disbursements and cash receipts, and general journals for less frequent transactions). After being journalized, the data is posted to the ledger accounts in a process that sorts and aggregates similar transactions. The set of allowable accounts or categories to be used for the journal and ledger is called the chart of accounts and is largely determined by historical practice for a particular industry. After adjusting and closing, the ledger accounts produce the financial statements. This entire conventional process is exhibited for a sample company (not the Wilson Company) in Figure 14.

Automated Data Processing Environment

Referring back to Figure 13, let us discuss what happens to this conventional model when it enters an automated data processing environment.

Most modern accounting software, such as general ledger programs, accounts receivable and payable programs, and inventory accounting programs, are simply computerized versions of the journalizing and posting methods discussed in the previous section. They use double-entry principles to define procedures and the chart of accounts to classify the meaning of economic events.[3]

On the positive side in this phase, accounting procedures have started to become more integrated into the entire information system of an organization. For example, general ledger software must use output generated from a variety of other programmed sources, and it is sometimes possible to link journal and ledger accounts with the type of multidimensional data we mentioned in Chapter 2. However, the basic application of the manual system has not changed. What we have, in McGrae's terms, is a change of technology but not of system.

At this point we might ask: why not be satisfied with simple computerization of the old system?, or in other words: why not use programs that imitate journalizing and posting procedures and that produce files that resemble journals and ledgers for our proposed model? The answer to this question is that this kind of application-oriented approach produces a host of disjoint, redundant, and single purpose file systems which are incompatible with the database approach to information systems management. This incompatability is illustrated graphically by Nolan in Figure 15.

The file system depicted in Figure 15(a) shows an application oriented approach to file structure. Nolan probably overemphasizes his

---

[3] An appreciation of this fact can be gained by examining the accounting program and file descriptions published by various software vendors or by reviewing the systems development effort involved in a computerization of the corporate accounting function. Two recent articles illustrating changes to an automated data processing environment are: (1) William R. Hindman and Floyd F. Kettering, "Integrated MIS: A Case Study," Management Accounting 55 (August 1973):20-27, and (2) Daniel P. Lubas, "Developing a Computerized General Ledger System," Management Accounting 57 (May 1976):53-56.

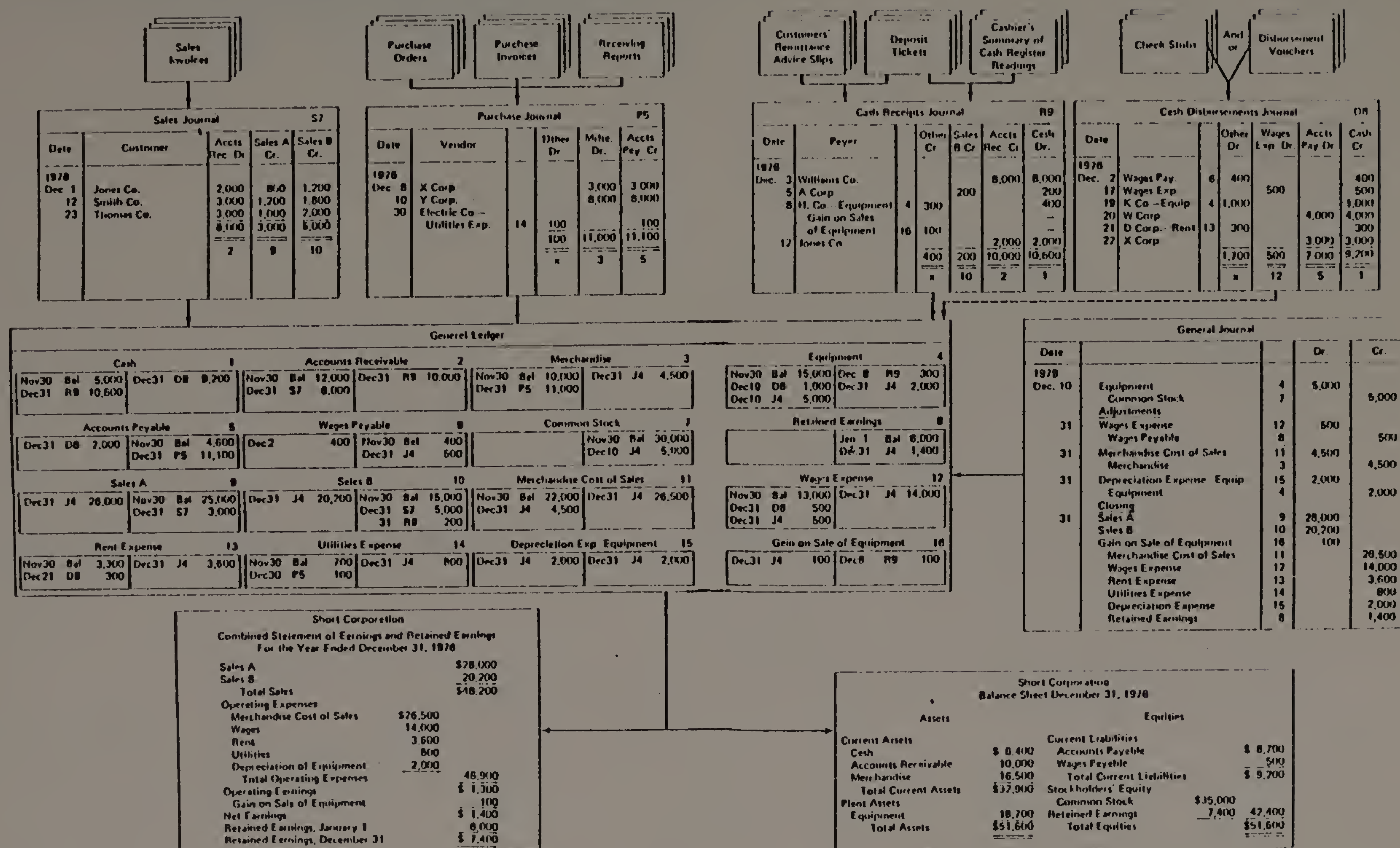
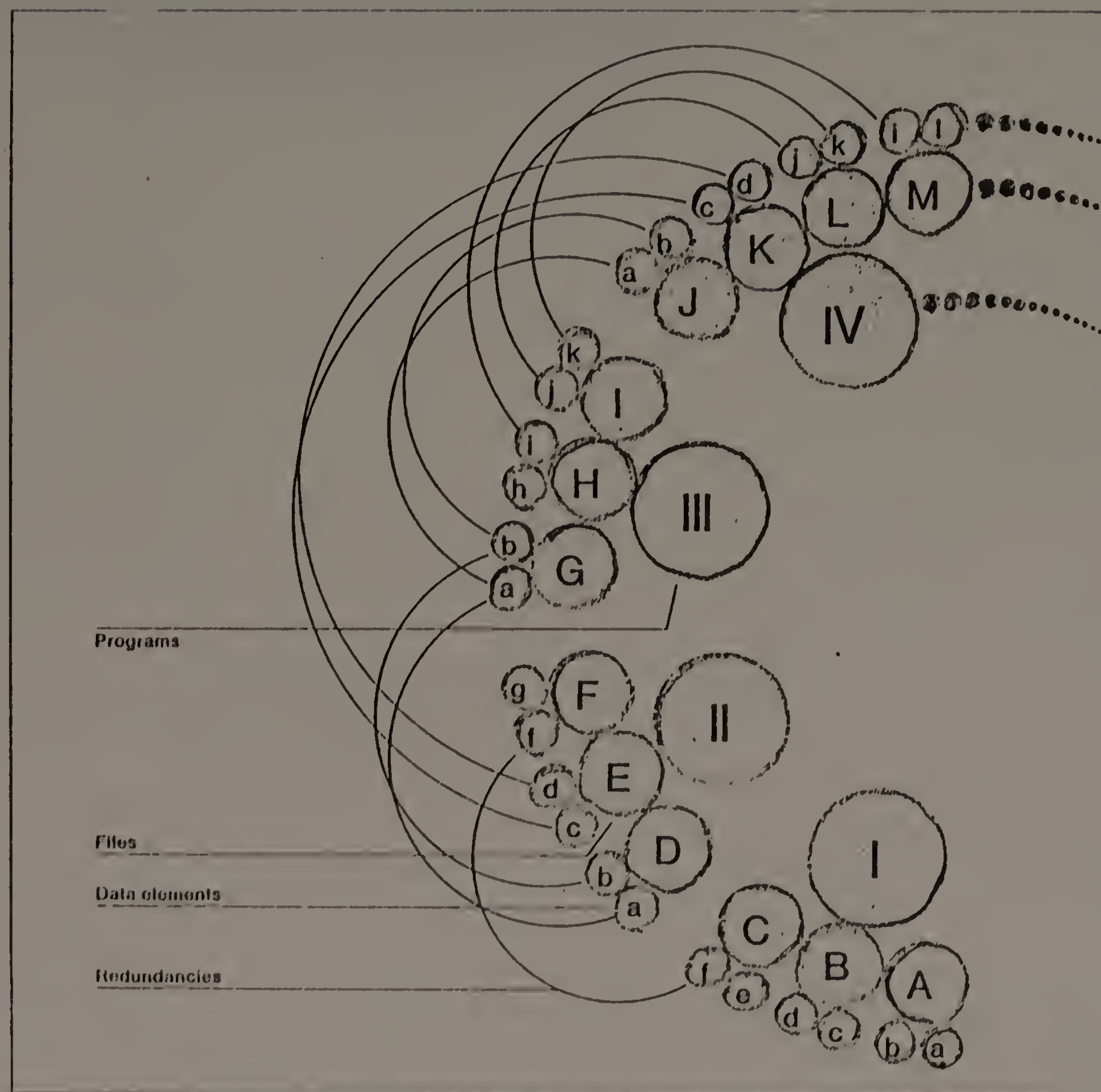


FIGURE 14--Conventional accounting process

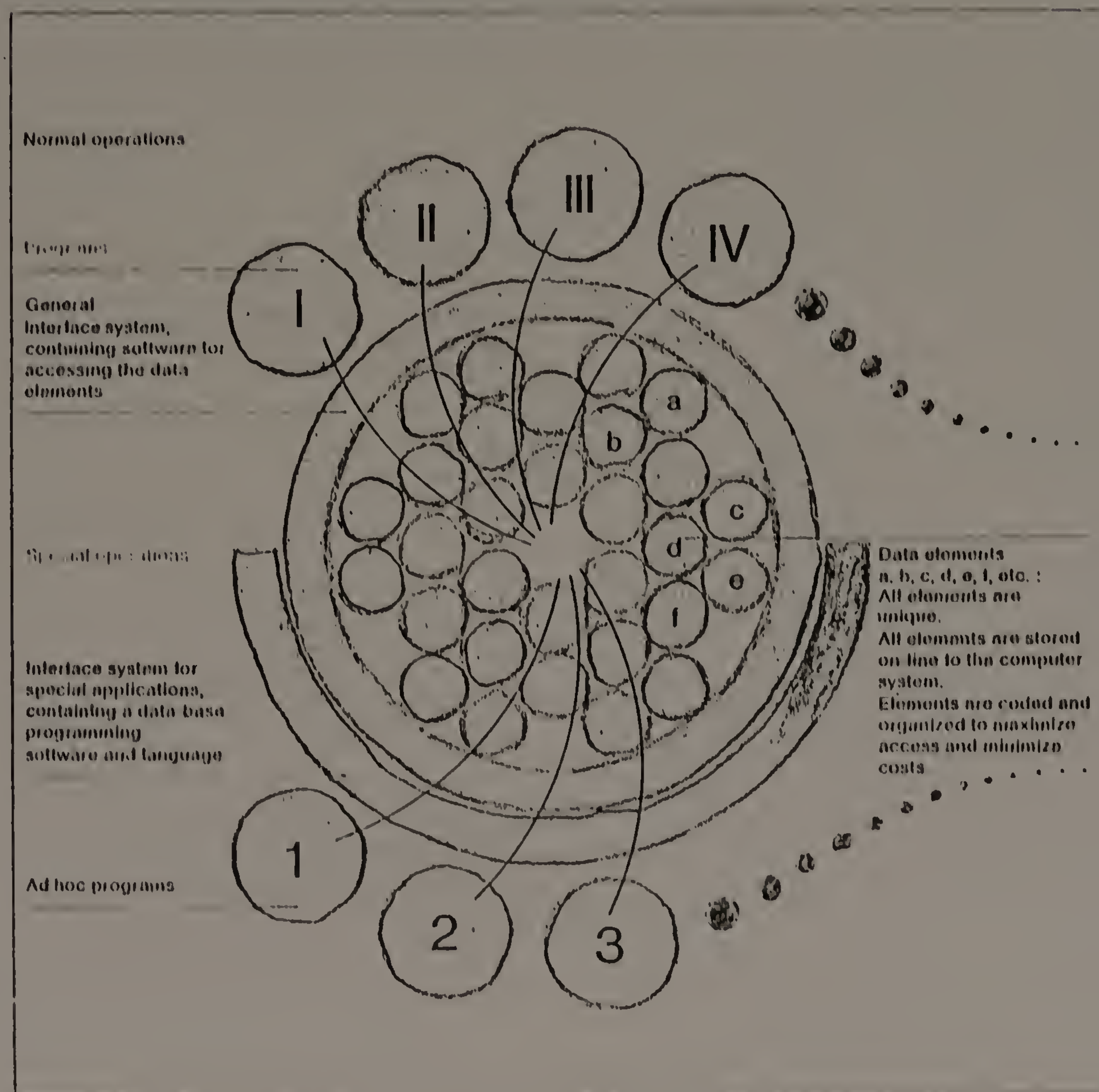
Source: L. Morrison and K. Cooper, <u>Financial Accounting</u> (Hinsdale, Illinois: The Dryden Press, 1975), pp. 76-77.



(a) Traditional applications-oriented system

FIGURE 15--Nolan's contrasting systems

Source: Richard L. Nolan, "Computer Data Bases: The Future is Now," Harvard Business Review 51 (September-October 1973) 98-114.



(b) Database system

FIGURE 15--Continued

point by indicating that every application system uses disjoint files but his basic premise is correct, that is when you organize your file structures around the needs of individual applications, you produce many redundant and single purpose data fields. In an accounting context, the type of redundancies that might exist would not be hard to imagine in a sample like the following:

Program	Files	Data elements
General Ledger Processing	Journal Transactions	journal account #'s
A/R Processing	Ledger Balances	discount %
Order Entry	Accounts Receivable	customer city address
Invoice Operation	Customer File	customer balance
Inventory Updating	Inventory and Price Master	sales quantity
Sales Forecasting		inventory prices

The type of system we hope to achieve with our model is depicted in Figure 15(b). We next discuss the changes in accounting system (as opposed to accounting technology) that will be needed to build it.

Database Environment

The transformation in data management techniques needed to move from the applications oriented system of Figure 15(a) to the database system of Figure 15(b) involves more than a simple centralization of files and elimination of redundant elements. It involves viewing an organization's store of management information from an entirely new perspective as shown in Figure 16 and explained below.

The applications view of management information concentrates on the various functions of an enterprise. To the extent that application programs represent tasks to be carried out in each of these functional areas, the views of data tend to become restricted and evolve toward the redundant file structures we described previously.

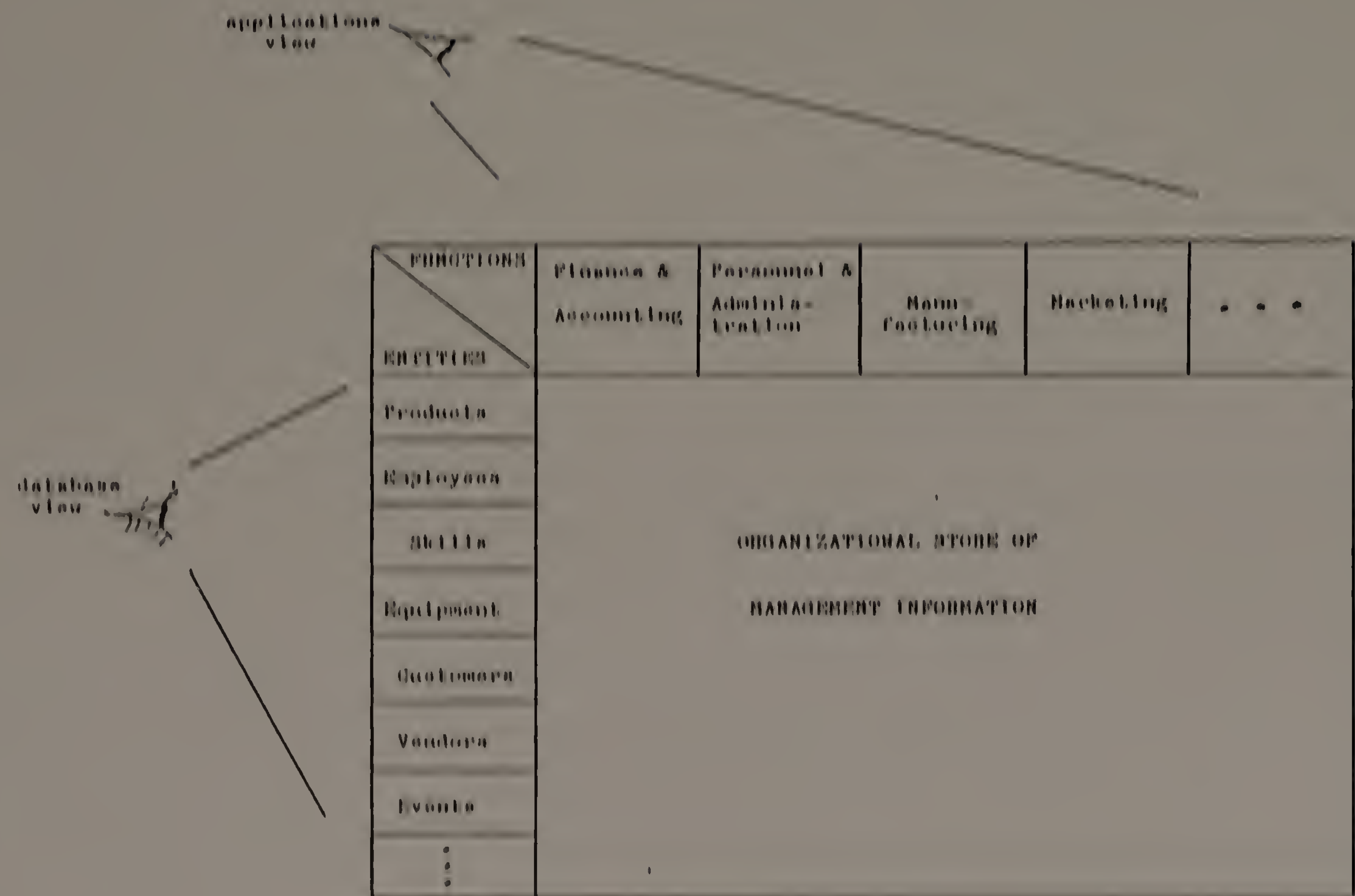


FIGURE 16--Two views of management information

Source: Neil C. Churchill, John H. Kempster, and Myron Uretsky, Computer-Based Information Systems for Management: A Survey (New York, National Association of Accountants, 1969), p. 37.

On the other hand, the database view of management information concentrates on describing the various entities involved in an organization's operations. The primary data management task from this perspective involves not the creation and maintenance of files for separate programs but instead the creation and maintenance of a data model (also called a schema) of the organization. The employee-skills database we described in Chapter 2 was an example of a data model as seen with three different logical orientations.

This change in emphasis from describing functions to describing entities will be important to us in constructing an accounting database model. In particular, we will not be able to rely on a traditional chart of accounts for classification because some of the accounts such as "current assets," "accounts-receivable," and "cost of goods sold" are not entities at all but are simply conventions used in performing accounting procedures. Everest and Weber[4] recently reached this same conclusion saying that "naming" and "presentation" accounts might best be left out of a database description.

Summary of Changes in Accounting Technology and System

Our purpose in this section was to describe the changes inherent in moving from a manual double-entry accounting system to a relational database. We stated that such a movement would involve changes in both the accounting technology and accounting system and that the definition of our database would therefore involve more than simple computerization

---

[4] Gordon C. Everest and Ron Weber, "A Relational Approach to Accounting Models," The Accounting Review 22 (April 1977):340-59.

of a chart of accounts and bookkeeping rules. It would entail (1) definition of a data model based upon description of the basic entities of an organization and (2) specification of procedures needed to make this database system operational.

Having described these systems on a very general level, we move now to specific examples of their operations. Our first presentation will feature the conventional accounting model.

Sample Operating Data and Conventional System

Introduction

In this section we will illustrate the conventional accounting model which "considers past data as the basic input, the chart of accounts as the basic classification model, and the financial statements as the basic output."[5] To do so however, we will first have to provide a concrete set of transactions to work with. This concrete set will be presented in the form of one month's operating data for a fictitious retail enterprise called Wilson Company (see Exhibit 1).

We will presume that this fictitious company organized itself and gathered the relevant information it needed to operate on the first day of June. The example will include description of its operations for the entire first month up to and including the calculation of income and the payment of some dividends. For simplicity reasons, we have not included any dealings with government agencies including corporate and personal income tax payments.

---

[5] American Accounting Association, A Statement of Basic Accounting Theory (Chicago: American Accounting Association, 1966), p. 56.

Wilson Company Operating Data for June

The Wilson Company is a small retail enterprise that was incorporated on 1 June when Fred Wilson sold 500 shares in his company to various investors. The company personnel force consists of Wilson plus 5 newly hired employees, and its product line consists of 5 products: A, B, C, D, & E. Wilson Company's operations consist simply of purchasing these products from various suppliers at wholesale prices and then distributing them at retail to customers. The following business events occurred during Wilson's first month of operation.

Number	Date	Event
1	June 1	Fred Wilson sold 500 shares of stock at $50 to each of six investors: Conley, Brewer, Sullivan, Wall, Delock, and Schwall (total capital: 3000 shares @ $50 = $150000)
2	1	Wilson bought on account from Horvath: 6 sets of desks and chairs @ $400 - $2400 2 typewriters @ $1200 - 2400 packaging machine - 3500 $8300
3	1	Bought for cash from Shore, a truck for $4800
4	1	Paid June rent for office and storage space to Green, $1600
5	1	Bought on account merchandise from Oliver: 6000 of A @ $2 - $12000 2000 of B @ $4 - 8000 $20000
6	2	Bought on account merchandise from Williams: 20000 of E @ $1 - $20000 3000 of C @ $9 - 27000 $47000
7	3	Bought on account merchandise from Smith: 600 of D @ $10 - $6000

EXHIBIT 1. Wilson data

Number	Date	Event
8	June 3	Received an order from White: 2000 of A 1000 of B 700 of C 1000 of D
9	4	Paid Horvath for #2, $8300
10	4	Received an order from Nelson: 1000 of A
11	5	Sold to White on account: 2000 of A @ $ 3 - $ 6000 1000 of B @ $ 5 - 5000 700 of C @ $12 - 8400 $19400
12	6	Paid Oliver for #5, $20000
13	6	Sold to Nelson on account: 1000 of A @ $3 - $3000
14	6	Received an order from Scott: 2000 of A 1000 of E
15	7	Paid Williams for #6, $47000
16	8	Paid McKenzie $500 for advertising
17	8	Sold to Scott on account: 2000 of A @ $3 - $6000 1000 of E @ $1.50 - 1500 $7500
18	8	Paid Smith for #7, $6000
19	9	Purchased on account from Smith: 600 of D @ $11 - $6600
20	10	Sold to White on account: 1000 of D @ $15 - $15000
21	10	Bought on account merchandise from Oliver: 2000 of B @ $4.30 - $ 8600 2000 of A @ $2 - 4000 $12600

EXHIBIT 1. Continued

Number	Date	Event
22	June 11	Received an order from White: 2000 of E 2000 of A
23	13	Sold to White on account: 2000 of E @ $1.50 - $3000 2000 of A @ $3 - 6000 $9000
24	13	Received a bill from Hodge for $3,000 transportation and paid half
25	14	Paid Smith for #19, $6600
26	14	Paid $3280 to employees, rates listed below: Wilson: 80 hours @ $10 - $ 800 Jackson: 80 hours @ $8 - 640 Johnson: 80 hours @ $4 - 320 Harrison: 80 hours @ $5 - 400 Taylor: 80 hours @ $8 - 640 Adams: 80 hours @ $6 - 480 $3280
27	15	Paid Oliver for #21, $12600
28	15	Received a bill from Simmons for cleaning, $300
29	16	Received an order from Jones: 1000 of A 300 of D
30	17	Purchased merchandise on account from Oliver: 4000 of A @ $2.30 - $ 9200 2000 of B @ $4.30 - 8600 $17800
31	17	Received an order from Russell: 4000 of B 1500 of C 3000 of E
32	18	Sold to Jones on account: 1000 of A @ $3 - $3000
33	18	Purchased from Williams on account: 1000 of C @ $10 - $10000

EXHIBIT 1. Continued

Number	Date	Event
34	June 19	Sold to Russell on account: 4000 of B @ $5 - $20000 1500 of C @ $12 - 18000 3000 of E @ $ 1.50 - 4500 $42500
35	19	Received an order from Howell: 2000 of A 500 of D
36	20	Received another cleaning bill from Simmons for $350 and paid for this one and #28, total=$650
37	21	White paid for sales: #11 - $19400 #20 - 15000 #23 - 9000 $43400
38	21	Nelson paid for sale: #13 - $3000
39	21	Sold on account to Howell: 2000 of A @ $3 - $6000
40	22	Paid Oliver for #30, $17800
41	23	Scott paid for sale: #17 - $7500
42	23	Paid Williams for #33, $10000
43	24	Received an order from Nelson: 3000 of A 800 of C 7000 of E
44	25	Paid Hodge other half of transportation bill (#24) $1500
45	26	Sold to Nelson on account: 800 of C @ $12 - $ 9600 7000 of E @ $ 1.50 - 10500 $20100
46	27	Received advertising bill from McKenzie, $500

EXHIBIT 1. Continued

Number	Date	Event
47	June 28	Paid all employees for 80 hours, $3280
48	28	Purchased on account from Williams: 500 of C @ $9.25 - $4625
49	29	Received a transportation bill from Hodge, $4000
50	29	Received an order from White: 500 of C 4000 of E
51-55	30	Accounting adjustments and closings made
56	30	Paid a dividend of $2 to each of the 3000 outstanding shares of common stock

EXHIBIT 1. Continued

Following presentation of the Wilson data, we will proceed immediately to a double-entry accounting of its operations.

Conventional Double-Entry Presentation of Wilson's June Operations

Introduction

In previous sections, we outlined the workings of the conventional double-entry accounting model (for example, see Figure 14 on page 49. We noted at that time that the recognition and classification of input data in the form of transactions consisted of (1) making a double entry into an appropriate journal and (2) posting the journals to a set of ledger accounts. For simplicity's sake, we will combine these two steps in the following section by using T-accounts. This is a convention widely used in accounting textbooks to ease presentation and save considerable space. To further simplify matters, we will also not use subsidiary accounts to record the various subclassifications of items such as individual pieces of machinery or accounts payable and receivable.

In the next two sections we will present: (1) the T-accounts for Wilson's operations and (2) the financial statements prepared from those accounts.

Wilson transactions

Wilson's June operations are recorded and classified in double-entry form below (see Figure 17). Each of the entries is keyed to the number (not the date) given for each event. For example, the first two events would result in the following journal transactions:

```
#1
Debit                   CASH                  $150000
  Credit                  COMMON STOCK           $150000
    to record the
    sale of stock

#2
Debit                   MACHINE & EQUIPMENT   $83000
  Credit                  ACCOUNTS PAYABLE         $8300
    to record pur-
    chase of new
    equipment
```

Each time an account is ruled with a single line, the entries on a particular side are totaled. When they are ruled with a double line, the two sides are netted against each other.

In our description of Wilson's operations, we did not elaborate on events numbered 51 through 55. These events consist of adjusting and closing entries, and their proper inclusion in an "events" accounting model will be discussed in later sections. They are, however, part of the traditional process. For completeness sake, we outline them in general journal form below:

```
#51
DEPRECIATION EXPENSE                          145
  ACCUMULATED DEPRECIATION-M&E                  145
    to record depreciation on
    various machines

#52
WAGE EXPENSE                                  656
  ACCOUNTS PAYABLE                              656
    to record accrued wages
    payable to employees

#53
COST OF GOODS SOLD                            93250
INVENTORY                                     31375
  PURCHASES                                     124625
    to record counted ending
    inventory @ weighted
    average cost
```

#54		
SALES	125500	
EXPENSE AND REVENUE SUMMARY		125500
to close revenue accounts		
EXPENSE AND REVENUE SUMMARY	110861	
RENT EXPENSE	1600	
ADVERTISING EXPENSE	1000	
CLEANING EXPENSE	650	
WAGE EXPENSE	7216	
TRANSPORTATION EXPENSE	7000	
COST OF GOODS SOLD	93250	
DEPRECIATION EXPENSE	145	
to close expense accounts		
#55		
EXPENSE AND REVENUE SUMMARY	14639	
RETAINED EARNINGS		14639

We can also see in the accounts below that none of the "order" events are recognized. This is in accordance with the realization principle of accounting. Their proper inclusion in an events-based model will also be discussed later. We now present the Wilson Company accounts in Figure 17.

Wilson financial statements

As can be seen in Figure 14 (page 49), the output from the conventional model consists of the company's financial statements. These are prepared from the general ledger accounts (or in our case, the T-accounts) and in many cases provide the only public disclosure of enterprise operations. The two most common of these financial statements--the statement of earnings and retained earnings, and the balance sheet--are presented below for Wilson Company in Figure 18 and 19.

Conclusion of traditional accounting cycle

The preparation of financial statements marks the conclusion of our

CASH

(1)	150000	(3)	4800
(37)	43400	(4)	1600
(38)	3000	(9)	8300
(41)	7500	(12)	20000
		(15)	47000
		(16)	500
		(18)	6000
		(24)	1500
		(25)	6600
		(26)	3280
		(27)	12600
		(36)	650
		(40)	17800
		(42)	10000
		(44)	1500
		(47)	3280
		(56)	6000
	203900		151410
	52490		

ACCOUNTS RECEIVABLE

(11)	19400	(37)	43400
(13)	3000	(38)	3000
(17)	7500	(41)	7500
(20)	15000		
(23)	9000		
(32)	3000		
(34)	42500		
(39)	6000		
(45)	20100		
	125500		53900
	71600		

ACCOUNTS PAYABLE

(9)	8300	(2)	8300
(12)	20000	(5)	20000
(15)	47000	(6)	47000
(18)	6000	(7)	6000
(25)	6600	(19)	6600
(27)	12600	(21)	12600
(36)	300	(24)	1500
(40)	17800	(28)	300
(42)	10000	(30)	17800
(44)	1500	(33)	10000
		(46)	500
		(48)	4625
		(49)	4000
		(52)	656
	130100		139881
			9781

COMMON STOCK

		(1)	150000

MACHINE AND EQUIPMENT

(2)	8300		
(3)	3800		
	13100		

ACCUMULATED DEPRECIATION-M & E

		(51)	145

FIGURE 17--Double-entry presentation of Wilson data

PURCHASES

(5) 20000	
(6) 47000	
(7) 6000	
(19) 6600	
(21) 12600	
(30) 17800	
(33) 10000	
(48) 4625	
124625	(53) 124625
0	

SALES

	(11) 19400
	(13) 3000
	(17) 7500
	(20) 15000
	(23) 9000
	(32) 3000
	(34) 42500
	(39) 6000
	(45) 20100
(54) 125500	125500
0	

RENT EXPENSE

(4) 1600	
1600	(54) 1600
0	

ADVERTISING EXPENSE

(16) 500	
(46) 500	
1000	(54) 1000
0	

TRANSPORTATION EXPENSE

(24) 3000	
(49) 4000	
7000	(54) 7000
0	

WAGE EXPENSE

(26) 3280	
(47) 3280	
(52) 656	
7216	(54) 7216
0	

CLEANING EXPENSE

(28) 300	
(36) 350	
650	(54) 650
0	

INVENTORY

(53) 31375	

COST OF GOODS SOLD

(53) 93250	(54) 93250
0	

EXPENSE AND REVENUE SUMMARY

(54) 110861	(54) 125500
(55) 14639	14639
0	

DEPRECIATION EXPENSE

(51) 145	(54) 145
0	

RETAINED EARNINGS

(56) 6000	(55) 14639
	8639

FIGURE 17--Continued

WILSON COMPANY
Combined Statement of Earnings and Retained Earnings
For the Month Ended June 30

Sales Revenue		$125,500
Less Cost of Goods Sold		93,250
Gross Margin.		$ 32,250
Less Expenses:		
Rent.	$1,600	
Advertising	1,000	
Cleaning.	650	
Wages	7,216	
Transportation.	7,000	
Depreciation.	145	
Total		17,611
Net Income.		$ 14,639
Less Dividends.		6,000
Retained Earnings, 30 June		$ 8,639

FIGURE 18--Wilson statement of earnings and retained earnings

WILSON COMPANY
Balance Sheet
30 June

ASSETS

Current Assets:		
Cash		$ 52,490
Accounts Receivable		71,600
Inventory		31,375
Total Current Assets		$155,465
Noncurrent Assets:		
Machine and Equipment	$13,100	
Less Accumulated Depreciation	145	
Total Noncurrent Assets		12,955
Total Assets		$168,420

LIABILITIES AND STOCKHOLDERS' EQUITY

Current Liabilities:		
Accounts Payable		$ 9,781
Stockholders' Equity:		
Common Stock	$150,000	
Retained Earnings	8,639	
Total Stockholders' Equity		158,639
Total Liabilities and Stockholders' Equity		$168,420

FIGURE 19--Wilson balance sheet

presentation of the conventional accounting model. In the next section, we will commence the work of transforming this framework into a system based upon the ideas of Sorter and Codd.

As we mentioned previously, the design of our new model will not involve simple computerization of manual methods but instead will concentrate on an entity-oriented or data modeling approach. We proceed now to this different way of viewing proposed information systems.

Data Model Development

Introduction

We have seen Wilson Company's data structured in conventional form. Our task now is to transform this description into an events-based relational system. In this section, we will begin this process by developing a data model or schema of Wilson's retail enterprise. Such a data model will simplify the transformation to a relational system.

Generally speaking, this schema input development can be approached in two ways. The first, called the "bottom-up approach," starts with existing files and attempts to restructure their individual data elements around basic data aggregates in order to better suit the needs of a database environment. Bottom-up approaches have been described and formalized by Mitoma,[6] Kahn,[7] and Shepard.[8] The second method, called

---

[6]Michael F. Mitoma, "Optimal Data Base Schema Design" (Ph.D. dissertation, The University of Michigan, 1975).

[7]Beverly K. Kahn, "A Method for Describing the Information Required by the Data Base Design Process," in Proceedings of ACM-SIGMOD Conference on Management of Data (New York: Association for Computing

the "top-down approach," starts with a close examination of an enterprise's operations and attempts to identify a priori the important principals within those operations. Top-down approaches have been described and formalized by Chen,[9] and Smith and Smith.[10]

For this paper, the top-down approach appears to suit our purpose better. We are tasked with constructing a framework that will differ fundamentally from existing models; therefore, a priori designation of system elements seems more appropriate. In particular, the Entity-Relationship (E-R) approach of Peter Chen suits our needs because (1) its final product--a set of entity and relationship tables--maps easily into a Codd type of database and (2) its emphasis on a priori designation of separate entities and relationships allows us to avoid the normalization transformations that many relational systems must undergo (for instance, see our employee-skill example of Chapter 2).[11]

In the rest of this section then, we will concentrate on develop-

---

Machinery, 1976), pp. 53-64.

[8]Donna L. Sheppard, "Data Base: A Business Approach to Systems Design" (Cincom Systems, Inc., 1974).

[9]Chen, pp. 9-36.

[10]John M. Smith and Diane C. P. Smith, "Database Abstractions: Aggregation and Generalization," ACM Transactions on Database Systems 2 (June 1977):105-33.

[11]We should emphasize that, although Chen's methodology fits our relational system well, it was designed for use with any type of database. In fact, his model could be characterized as a database approach to a system definition language, and in this regard it resembles earlier simulation-based approaches such as SIMSCRIPT. For further examples, see Van Court Hare, Jr., Systems Analysis: A Diagnostic Approach (New York: Harcourt, Brace & World, Inc., 1967), chap. 4.

ment of a data model. We will begin by describing the steps involved in the database abstraction process and then proceed to a detailed explanation of part of that abstraction process using Chen's E-R methodology.

Database Abstraction Process

The data in any information system is an abstraction of some aspect of reality. Database information systems require that this data representing a particular slice of reality be organized in a structured manner that will both remain consistent and maintain its integrity over time.

Incorporating methodologies suggested by Chen[12] and Sundgren,[13] we may view this structured abstraction process applied to an accounting system (see Figure 20).

Beginning with LEVEL 1, we note that a database system is intended to model some part of the real world or some reality. In an accounting context, this reality is an economic enterprise defined by the business entity principle.

At LEVEL 2, we start to narrow down the description of reality to those aspects that exist in the conceptual world of interest of the intended database users. Sundgren[14] refers to this slice of reality as the "object system," and for the accounting example, we restrict our conceptual view of the enterprise to its economic aspects. This means

---

[12] Chen, pp. 9-36.

[13] Bo Sundgren, "Conceptual Foundations of the Infological Approach to Data Bases," in Data Base Management, ed. J. W. Klembie and K. L. Koffeman (Amsterdam: North Holland Publishing Company, 1974), pp. 61-96.

[14] Ibid., p. 61.

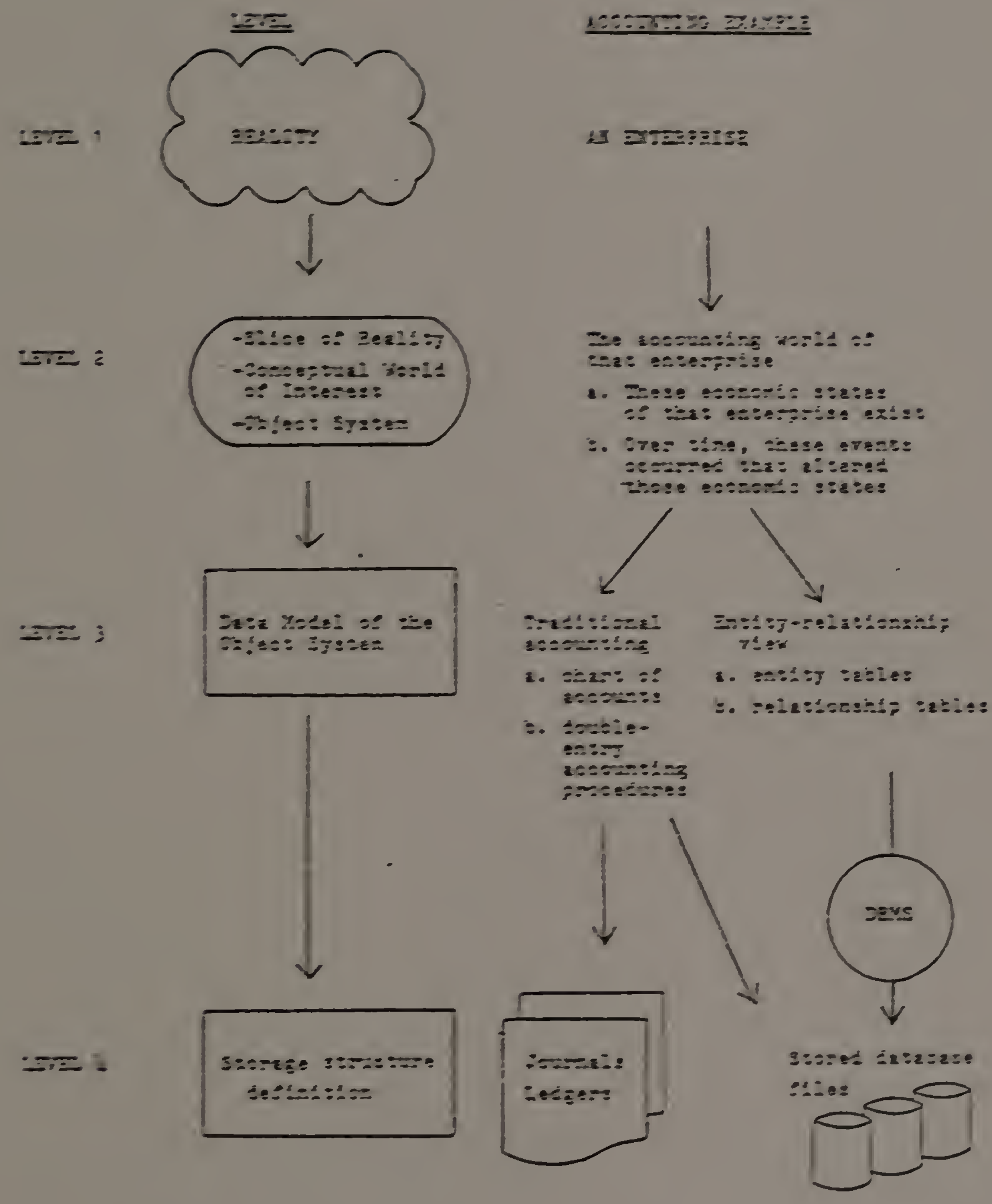


FIGURE 20--Database abstraction process

maintaining information about two matters of substance: (1) the economic states of the enterprise and (2) the events occurring over time that alter those economic states.

As we descend to LEVEL 3, we pass from the world of realities or principals as they are called by Ijiri[15] to the world of data models or surrogates. A data model is intended to be a description of the logical structure of the object system as seen by the database users or a scheme that represents with data, the organization of the conceptual world of interest.

In the traditional model we presented previously, it is here that we encounter problems in the form of accounting "artifacts" (a term used by Everest and Weber). Artifacts are devices used in performing manual accounting procedures, and they include "taxonomies, classification schemes, and naming conventions."[16] Because they are procedural devices and not essential aspects of enterprise operations, they do not fit into a data modeling process well.

We will depart from traditional accounting at LEVEL 3, and use instead an entity-relationship view of an enterprise to construct its accounting data model. Additionally, we will not limit our scheme with the principles of double-entry and monetary measurement, but allow it instead to assume more of the multidimensional and disaggregated aspects proposed by Sorter.

---

[15] Yuji Ijiri, Theory of Accounting Measurement (Sarasota: American Accounting Association, 1975), chap. 3.

[16] Everest and Weber, p. 342.

Level 4, the storage and access media to which data are allocated, will not be of interest in our treatment. As we noted in Chapter 2, there are many ways to implement physically a given logical database view. We also note in Figure 20 that, in line with our previous discussions, it is possible to computerize accounting systems without changing their basic structures. The result is simply a change in technology without a change in system.

We proceed now to a detailed treatment of LEVEL 3 of the database abstraction process--development of an accounting data model for our sample enterprise.

An Accounting Data Model

Introduction

When queried about the "things" that accounting deals with, a traditional accountant might list items such as "prepaid revenues," "retained earnings," or "liabilities" because these, among others, constitute the elements in his/her predefined world of interest. What has happened is that charts of accounts and accompanying double-entry routines have become more than schemes for organizing, classifying, and aggregating financial data. They have become the principals of accounting instead of surrogates used to represent real world phenomena. Their use imposes upon an accountant a particular mode of thinking about economic aspects of an operation.

In this section, we will discard this predisposition toward certain types of "things" of interest. Instead, we will view our object financial system without traditional-accounting-colored glasses and use the following steps outlined by Chen to construct its data model:

1. Identify (a) the entity sets such as classes of objects, agents, and events that exist in our conceptual world and (b) the relationships among those entities.

2. Construct an Entity-Relationship (E-R) diagram that will exhibit the semantic nature of our identified relationships.

3. Define the characteristics of entity and relationship sets that will be of interest to particular system users, and specify mappings that will identify those characteristics.

4. Organize the data gathered above into entity/relationship tables and identify a key--unique characteristic--for each entity/relationship set.

5. Map the entity-relationship model to a relational database.

As we consider each of these steps in the following sections, we will make their application more concrete by using aspects of Wilson's operations as examples.

Identification of entity/relationship sets

The process of viewing an object system and identifying its relevant entities and relationships can only be inexactly specified. The particular list that any one person produces might differ quite legitimately from another person's depending upon their differing backgrounds and perceived uses for the database. The only caveat for accountants is that in enumerating entities we limit ourselves to real phenomena that can be distinctly identified and remain clear of accounting artifacts. Everest and Weber supply direction in this process by warning us to avoid use of "naming tree" entities, that is accounts used for presentation or accumulation purposes only.[17]

---

[17] Ibid., p. 356.

For Wilson Company, the entities (objects, agents, events) of interest in the accounting object system might include those shown in Figure 21 (a).

Once the entities have been identified, we specify the relevant relationships that may exist among them. A partial list (a more complete diagram is given later) for our Wilson Company is shown in Figure 21 (b).

The bases for recognizing explicitly relationships between the various entities will again be peculiar to the person constructing the data model, but it is of interest to note at this point our rationale for connecting some of the <u>event</u> entities with each other. With the exception of the "order" event which is not traditionally speaking an accounting event at all, our <u>event (---) event</u> links represent explicit manifestations of Ijiri's causal double-entry conventions,[18] that is each change in the resource set of the enterprise is linked explicitly to another change by means of a causal relationship. In Mattessich's terms, these events would be called "a pair of requited transactions where . . . one transaction is the legal or economic consideration of the other."[19] This is a point we will return to in Chapter 4.

Again, there is no claim to absolute truth in our entity/relationship enumerations. The lists are not intended to be exhaustive. Indeed, for most object systems, there will be no need to discern every aspect of the conceptual realm of interest.

---

[18] Ijiri, chap. 5.

[19] Richard Mattessich, <u>Accounting and Analytical Methods</u> (Homewood, Illinois: Richard D. Irwin, 1964), p. 450.

Objects	Agents
Equipment	Stockholder
Inventory	Employee
Cash	Customer
	Vendor

Events

Order	Cash Receipts	Capital Transaction
Sale	Cash Dispersal	General & Administrative Service
Purchase	Equipment Acquisition	Personnel Service

(a) Entities in an accounting data model

event (---) event

Sale (fills) Order
Cash Receipt (payment for) Sale
Cash Dispersal (payment for) Personnel Service

agent (---) event

Employee (employed in) Personnel Service
Vendor (supplier of) General and Administrative Service
Customer (made to) Sale

object (---) event

Cash (flow of) Cash Receipt
Inventory (line item) Sale
General and Administrative Service (allocate cost of) Equipment

(b) Some relationships between entities (not complete listing)

FIGURE 21--Wilson entities and relationships

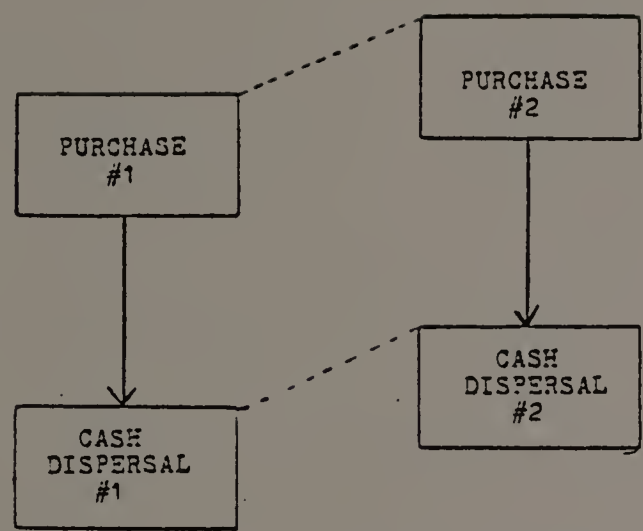
Construction of an Entity-Relationship diagram

Once we have identified our relevant entities and relationships, we exhibit the semantic nature of the relationships through the use of Chen's Entity-Relationship (E-R) diagrams. In this presentation, we will restrict ourselves for simplicity to binary relationships (between only two entities) and be consequently left with three possible semantic mappings: (1) one-to-one, (2) one-to-many, and (3) many-to-many. These were discussed briefly in Chapter 2 and are reiterated here with examples from Wilson operations.

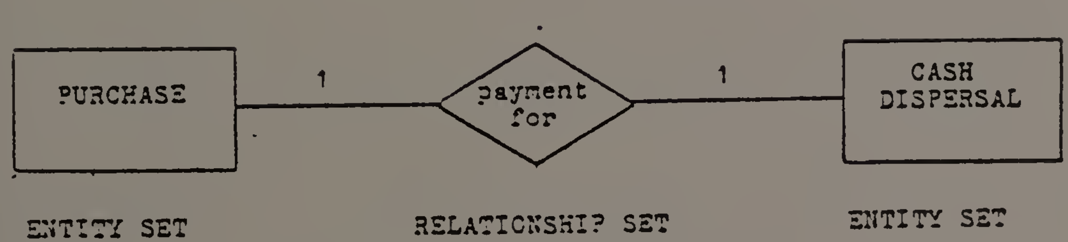
A one-to-one (1-to-1) relationship specifies a correspondence between just one entity from each of the two connected entity sets. Suppose, for example, that Wilson Company had an operational rule that all purchases were to be paid for exactly five days after receipt. Each purchase event would correspond then to only one cash dispersal event as exhibited in Figure 22 (a). Chen's E-R diagram would depict this 1-to-1 relationship in the manner shown in Figure 22 (b).

A one-to-many (1-to-n) relationship specifies a correspondence between just a single entity in one entity set and many entities in another entity set. Using our example again, let us suppose that the Wilson Company bills its customers twice a month, on the fifteenth and thirtieth, for all sales; after which they pay in full immediately. In this case, each cash receipt event would correspond to many sale events as seen in Figure 23 (a) and 23 (b).

Finally, a many-to-many (m-to-n) relationship specifies not only a possible correspondence between one entity in the first set and many entities in the second set, but also a possible correspondence between

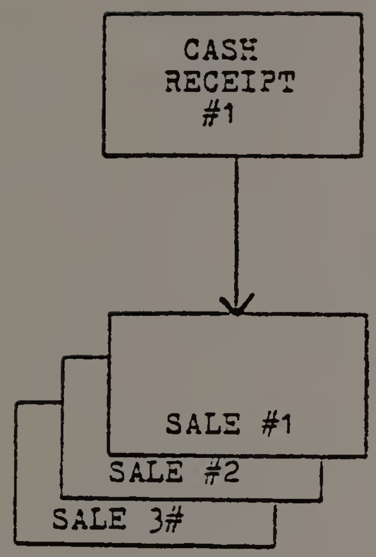


(a) Two related events

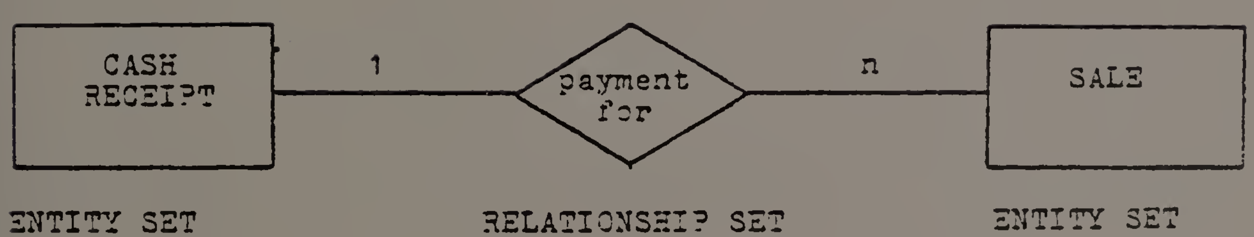


(b) E-R diagram

FIGURE 22--A one-to-one correspondence



(a) Two related events



(b) E-R diagram

FIGURE 23--A one-to-many correspondence

one entity in the second set and many entities in the first set. To illustrate, we use the SALE and INVENTORY entity sets from our example and show that not only does each sale consist of many products, but also that each product participates in many sales. This bidirectional mapping is shown in Figure 24 (a) and 24 (b).

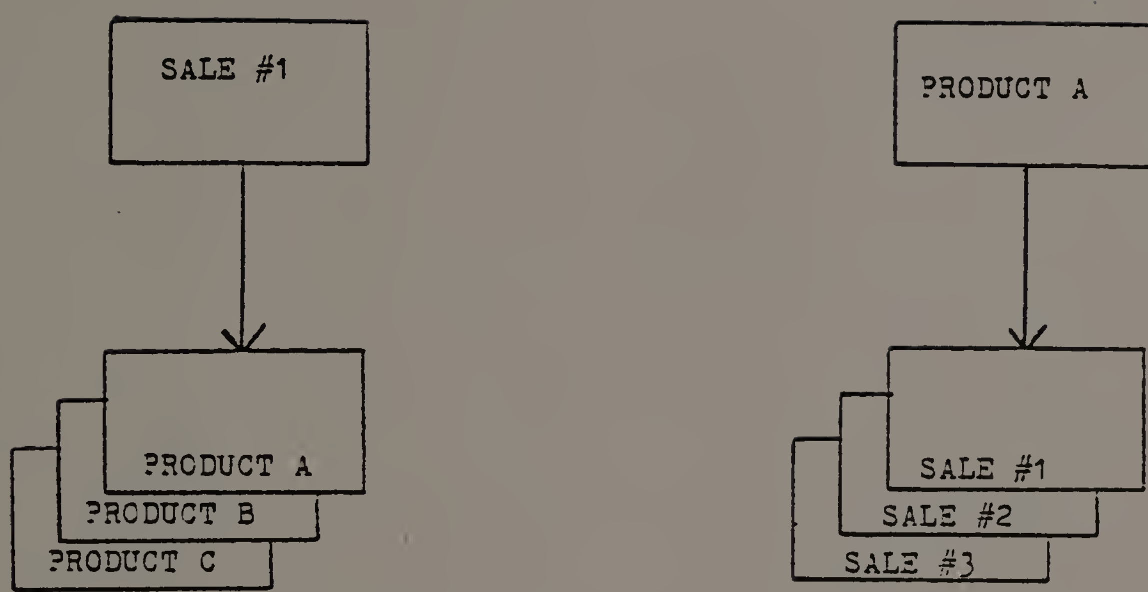
Extending this same kind of analysis for each of the entities identified in Figure 21 (a), we can specify the entity sets (boxes) and relationship sets (diamonds) that fit together in an overall view of the enterprise. The piecing together of each sub-diagram (such as Figures 22, 23, and 24) into one large diagram is an iterative process that we will discuss in Chapter 4. For now, we can assume that this analysis has been extended to include all relevant relationships for Wilson. Its final product--a complete enterprise E-R diagram--is pictured in Figure 25.

Identification of characteristic mappings

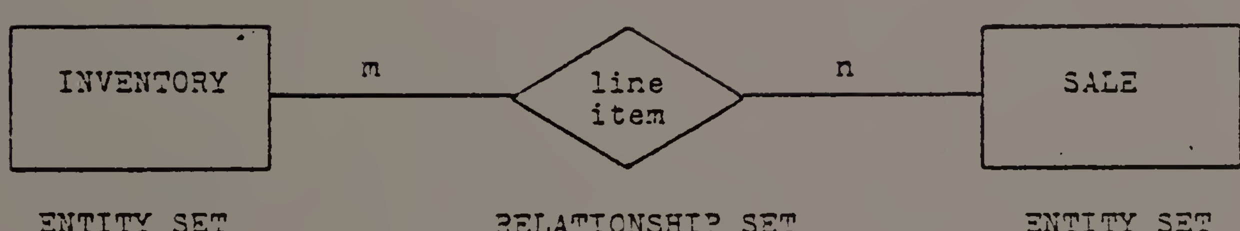
After identifying our object system's sets of entities and relationships, and their semantic correspondences, we start to describe the relevant properties or characteristics of each entity/relationship that will be included in the data model. Like the process of delineating the entities and relationships themselves, this identification procedure can only be described imprecisely. The final listing of characteristics will be heavily dependent upon the data model designer's perception of the database users' decision environment and information needs. A framework proposed by Mock[20] addresses the needs of this design phase. His

---

[20] T. J. Mock, Measurement and Accounting Information Criteria Sarasota: American Accounting Association, 1976), p. 88.



(a) A related object and event--two views



(b) E-R diagram

FIGURE 24--A many-to-many correspondence

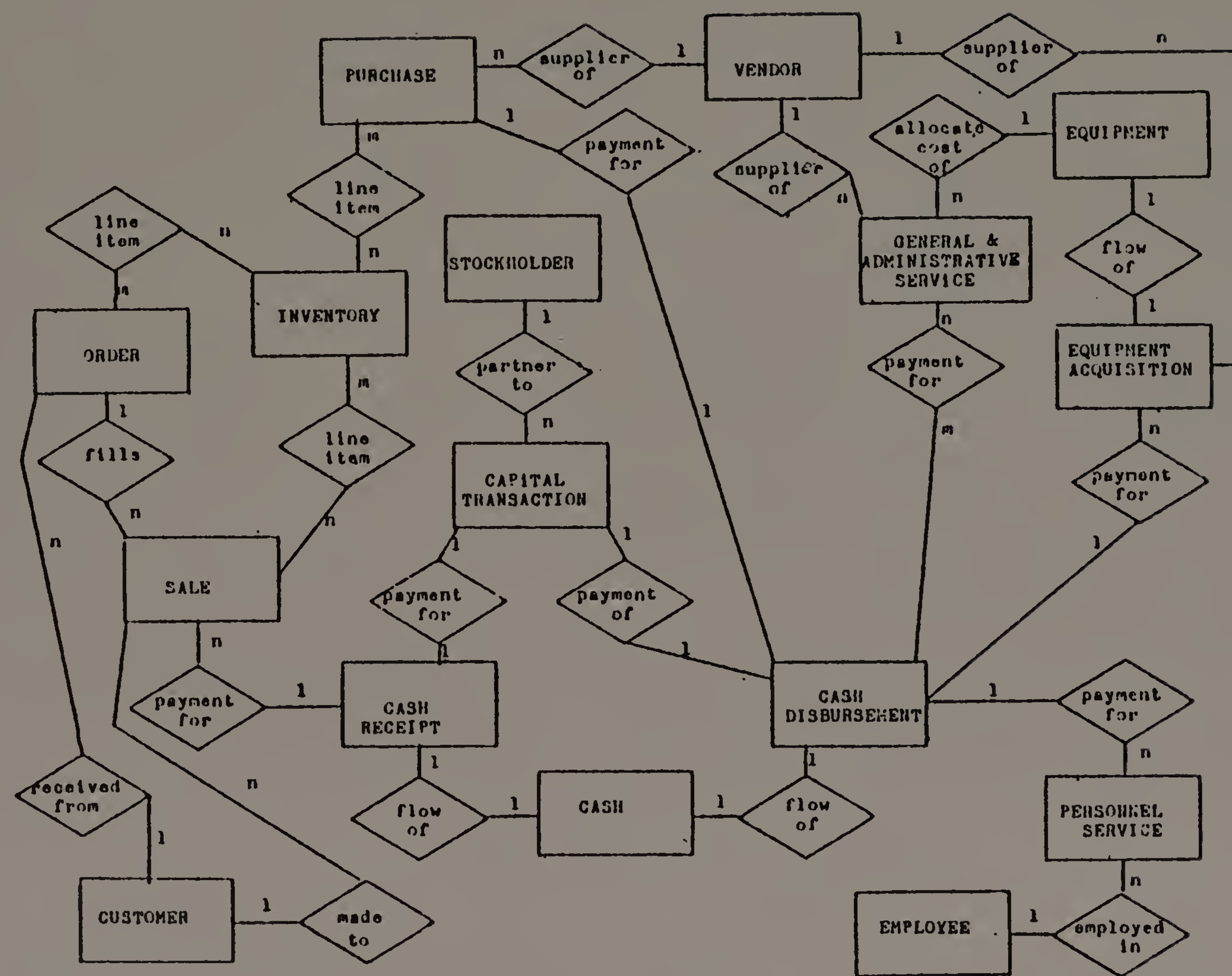


FIGURE 25--Complete enterprise E-R diagram

outline is based upon accepted principles of measurement theory and lists criteria certain to help in reducing the unstructured nature of this process (this is another topic we will return to in Chapter 4).

Specifying all of the characteristics for each entity/relationship set given in Figure 25 would be a prohibitively long process. Therefore, we will limit our mappings to two entity sets--SALE and INVENTORY--and the relationship set that connects them--SALE--line-item. Additional mappings for the other sets, and possible these three sets as well, would have to be accomplished before the data model would be completely specified.

The characteristics of a particular member of an entity/relationship set can be expressed by a listing of attribute/value pairs. Examples of these pairs for a specified product in the entity set INVENTORY might be "stock#/7432," "cost/$2.10," and "quantity-on-hand/2000."

The second item listed in each of the pairs above--the value-- is taken from different value sets such as stock #'s, dollar-amount, and no-of-units. These value sets are analogous to the relational database concept of "domain." Additionally, for those characteristics whose values are numerical, the value sets correspond to the definition of a Numerical Relational System (NRS) described by Mock.[21]

The first item in each pair--the attribute--is defined by Chen formally "as a function which maps from an entity set or relationship set into a value set."[22] This conceptualization of an attribute as a function is important to us for two reasons. First, it helps us, as we

---

[21] Mock, chap. 2. [22] Chen, p. 12.

shall later see, to avoid the normalization problems that other database systems encounter. Second, it fits in well with calls by theorists, such as Mock, for emphasis on the measurement aspects of data to be used in accounting systems. For those value sets characterized as Numerical Relational Systems (NRS), Chen's attribute mappings are identical to Mock's homorphic mappings.[23]

Figure 26 and Figure 27 illustrate some of the characteristic mappings for our entity sets INVENTORY and SALE. Note that it is possible for multiple functions (F_2-COST and F_3-REPLACEMENT COST) to map into a single value set (DOLLAR-AMOUNT). It is also possible, although we do not show it here, to have a single function such as NAME map an entity into the Cartesian product of multiple value sets such as FIRST-NAME and LAST-NAME.

The attribute mappings for the relationship set SALE-line-item are given in Figure 28. Although this particular example illustrates a case where the relationship itself possesses a characteristic that cannot be fully identified with either of the participating entities, it is not necessary for all relationships to have properties of their own. In some cases the relationship will simply specify a connection and possess no further information content.

Before leaving this section, we should note the close correspondence between our model as developed so far and the ideas of "events" accounting theorists. According to Sorter, "the purpose of accounting is to provide information about relevant economic events that might be

---

[23] Mock, p. 13.

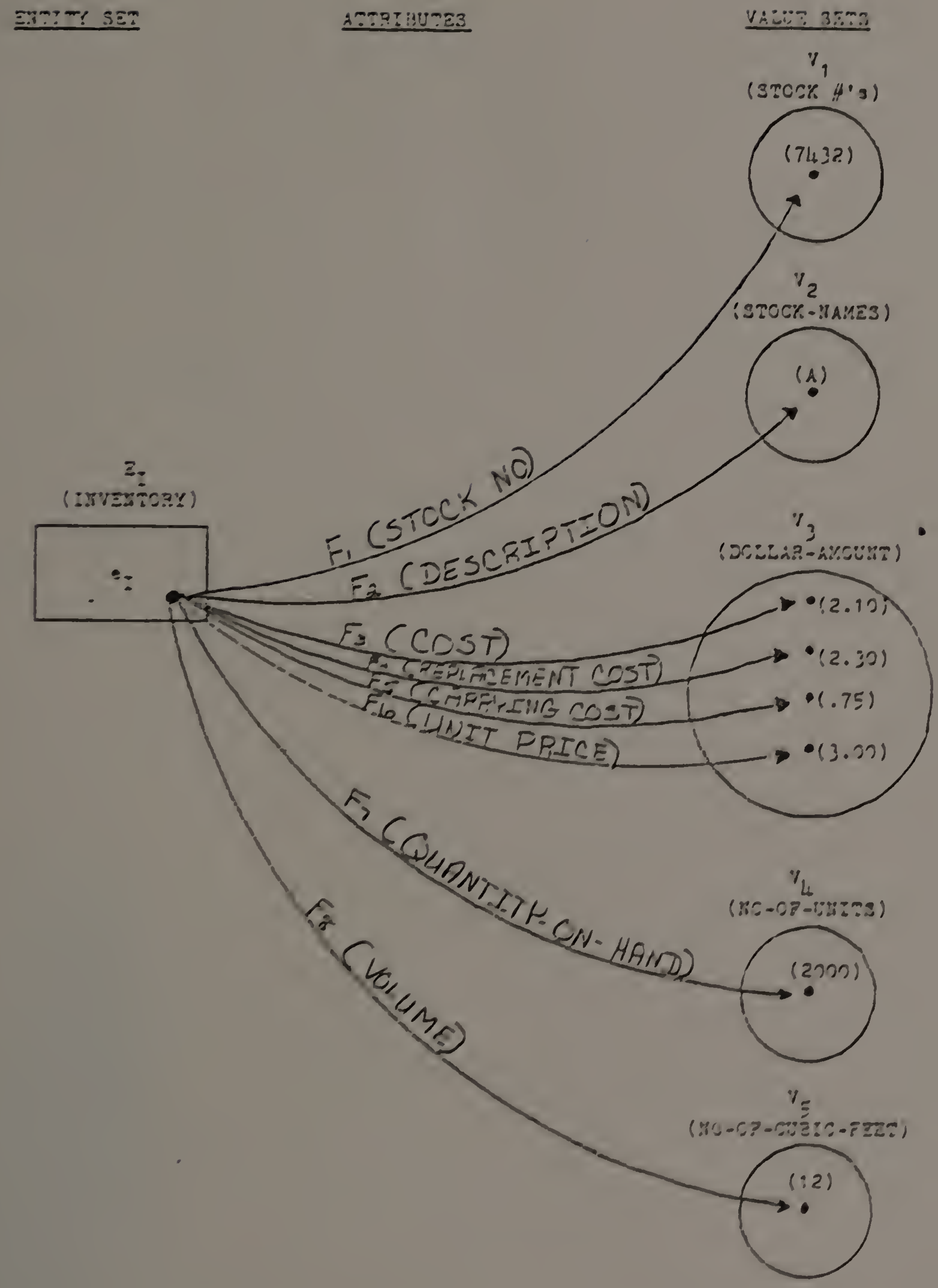


FIGURE 26--Characteristic mappings defined on entity set INVENTORY

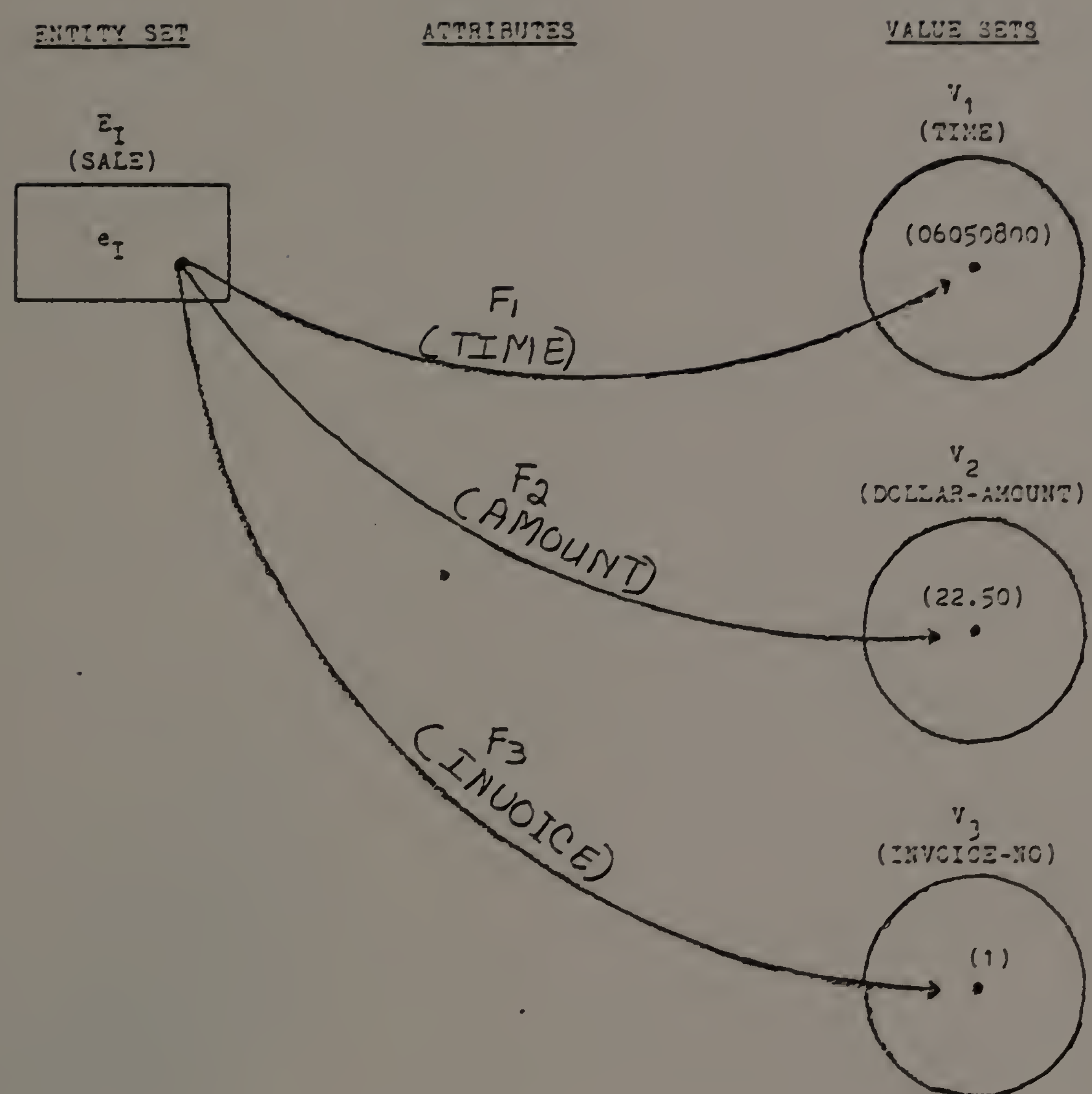


FIGURE 27--Characteristic mappings defined on entity set SALE

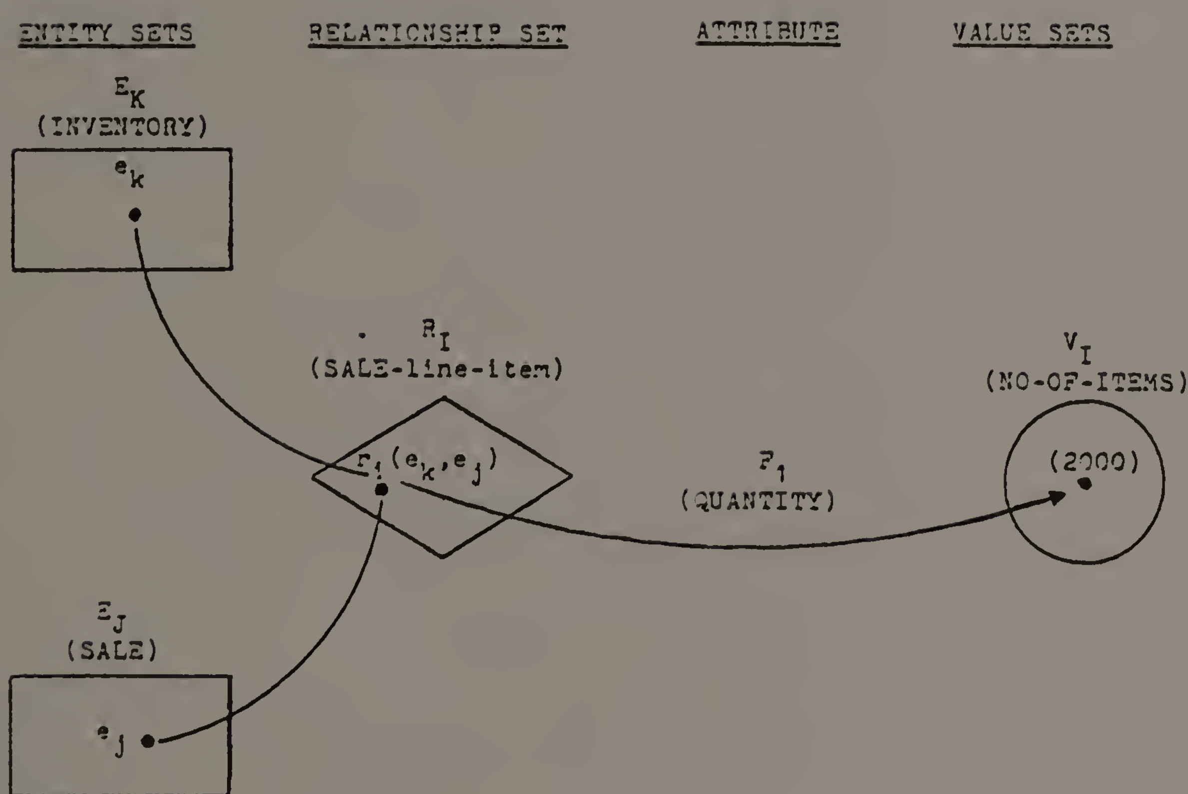


FIGURE 28--Characteristic mappings defined on relationship set <u>SALE-line-item</u>

useful in a variety of possible decision models."[24] Even with our limited presentation so far, we can see that an entity-relationship model accommodates well things implied by Sorter's statement such as multi-dimensional measures (dollars vs. cubic feet), different valuation bases (cost vs. replacement cost) and aspects of accounting entities useful in management science modeling (carrying cost or volume in EOQ models).

Organization of data into entity/relationship tables

The final step in preparing the data model for translation into a relational database involves organization of the model constructs into entity/relationship tables and identification of primary keys (PK).

A primary key (PK) for an entity (or a relationship) is simply an identifying characteristic that maps one-to-one with the entity itself, and thus is able to represent that entity in the database. To do this, this identifying characteristic must be both (1) universal--every entity must have it as an attribute--and (2) unique--each entity's value for that characteristic must be different. An example of a PK would be a student number in a university database or a social security number in the IRS database. The primary key is not always a single characteristic; sometimes it is necessary to combine or concatenate several attributes to uniquely identify something. Such is the case for all relationship sets in Chen's model because he defines their PK's as the combined PK's of their involved entities.

---

[24] George H. Sorter, "An 'Events' Approach to Basic Accounting Theory," The Accounting Review 44 (January 1969):13.

The organization of our data into entity tables is illustrated in Figure 29. As can be seen, the two tables correspond closely to the mappings shown in Figures 26 and 27. The only real difference is that we now let the primary key represent the entity itself, "TIME" for the SALE entities and "STOCK NO" for the INVENTORY entities.

A relationship table for SALE-line-item is illustrated in Figure 30. Again, the table closely resembles the mappings derived earlier (Figure 28) except that a primary key, the concatenated PK's of the involved entities, represents the relationship. The row in the table labeled "ROLE" simply lists the role that each entity plays in the relationship.

A complete data modeling effort would involve constructing tables for every entity/relationship depicted in Figure 25. After these specifications were finished, the data model would be ready for conversion to a particular DBMS.

Mapping into a relational database

Retracing briefly our steps in the data model design process, we have thus far:

1. Identified the relevant entity and relationship sets for Wilson

2. Constructed the E-R diagrams exhibiting the semantic nature of identified relationships

3. Defined characteristic mappings for each entity/relationship set

4. Organized the data gathered above into entity/relationship tables

The last step in our modeling process--mapping to a relational system--

	← PRIMARY KEY →		
ATTRIBUTE	TIME	AMOUNT	INVOICE
VALUE SET	TIME	DOLLAR-AMOUNT	INVOICE-NO
ENTITIES (one each row)	06050800	194.00	1
	06060900	30.00	2
	.	.	.
	.	.	.
	.	.	.

	← PRIMARY KEY →							
ATTRIBUTE	STOCK NO	DESCRIP-TION	COST	REPLACEMENT COST	CARRYING COST	UNIT PRICE	QOH	VOLUME
VALUE SET	STOCK #'s	STOCK NAMES	DOLLAR-AMOUNT				NO-OF-UNITS	NO-OF-CUBIC-FEET
ENTITIES	7432	A	2.10	2.30	.75	3.00	2000	12.0
	8519	B	4.20	4.30	.15	5.00	1000	3.6
	6784	C	9.25	9.25	.06	12.00	1500	1.5
	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.

FIGURE 29--Entity tables

	← PRIMARY KEY	→	
ENTITY TABLE NAME	INVENTORY	SALE	
ROLE	line	master	
ENTITY ATTRIBUTE	STOCK NO	TIME	QUANTITY (RELATIONSHIP ATTRIBUTE)
VALUE SET	STOCK #'s	TIME	NO-OF-ITEMS
RELATIONSHIPS	7432	06050800	2000
	8519	06050800	1000
	6784	06050800	700
	7432	06060900	1000
	⋮	⋮	⋮

FIGURE 30--Relationship table for <u>SALE-line-item</u>

is one that we will save for the next section. Excepting minor differences, Codd tables and Chen entity/relationship tables are identical: a fact which makes the mapping of an E-R data model into a relational system fairly straightforward. However, when comparing our system with the ones that most relational theorists synthesize, it becomes apparent that a major design difference exists, that is we avoid the normalization transformations that others, such as Codd,[25] and Everest and Weber,[26] encounter. We discussed these transformations briefly in Chapter 2 in the context of our employee-skill database, and we also mentioned in this chapter that Chen's methodology would allow us to avoid them. There are two reasons why his model accomplishes this, and each is discussed below.

First, by designating attributes as functional mappings and by identifying all 1-to-1 and 1-to-n relationships in the design process, we have already pinpointed the functional dependencies present in our system. With careful designation of primary keys (some relationship sets will use only part of their PK), we are given a data model that already meets the requirements for Codd's second normal form (2NF).

Second, careful application of Chen's emphasis on a priori identification of entities and relationships precludes violation of the mutual independence constraint for third normal form (3NF). To put it another way, Chamberlin says that "each relation [table] should describe

---

[25]E. F. Codd, "Further Normalization of the Data Base Relational Model," in Data Base Systems, ed. R. Rustin (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972), pp. 65-98.

[26]Everest and Weber, pp. 347-51.

a single 'concept' and if more than one 'concept' is found in a relation the relation should be split into smaller relations [tables]."[27] The E-R design process simply does not allow this mixing up of multiple concepts within one table.

We will now continue with a complete relational specification of Wilson Company.

Relational Presentation of Wilson's June Operations

Introduction

As we present our events-based model, we will presume that it has been implemented on a relational system similar to the System R[28] prototype being used at the IBM San Jose Research Laboratories. This is the most advanced relational system in development in 1977. We will not lose significant generality in using its conventions because our discussion will center primarily on the model's logical aspects which are theoretically similar to any relational implementation.

The language used in System R to define, manipulate, control, and query the database is called SEQUEL 2.[29] SEQUEL 2 has been developed

---

[27] Donald D. Chamberlin, "Relational Data-Base Management Systems," Computing Surveys 8 (March 1976):48.

[28] M. M. Astrahan et al., "System R: A Relational Approach to Data Base Management," ACM Transactions on Database Systems 1 (June 1976): 97-137.

[29] D. D. Chamberlin et al., "SEQUEL 2: A Unified Approach to Data Definition, Manipulation and Control," IBM Journal of Research and Development 20 (November 1976):560-75.

using Codd's original relational languages[30]--the relational calculus and the relational algebra--as starting points, and it has proven to be relationally complete.[31] Additionally, the language has been refined based upon a series of human factor experiments,[32,33] and it is now designated for use primarily by non-programming professionals (specifically including accountants). As we move through our relational model, many of the system features will be displayed using SEQUEL 2.

In the next section, we will present a complete set of relational tables for Wilson's operations and then move on to specific implementation features.

Relational Tables for Wilson

The events-accounting relational system for Wilson's June operations (outlined in Exhibit 1) is displayed in Figure 32. Figure 31 is simply our E-R diagram of the previous section (Figure 25) coded with numbers for help in locating specific tables. In constructing the tables, we have followed a number of conventions which a reader must

---

[30] E. F. Codd, "Relational Completeness of Data Base Sublanguages," in Data Base Systems, ed. R. Rustin (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972), pp. 65-98.

[31] R. F. Boyce et al., "Specifying Queries as Relational Expressions," in Data Base Management, ed. J. W. Klimbie and K. L. Koffeman (Amsterdam: North Holland Publishing Company, 1974), pp. 169-178.

[32] Phyllis Reisner, Raymond F. Boyce, and Donald D. Chamberlin, "Human Factors Evaluation of Two Data Base Query Languages--SQUARE and SEQUEL, in Proceedings of the National Computer Conference (Anaheim, California: AFIPS Press, 1975), pp. 447-52.

[33] Phyllis Reisner, "Use of Psychological Experimentation as an Aid to Development of a Query Language," Research Report RJ 1707 (IBM Research Laboratoes, San Jose, California, January 1976).

understand in order to follow the system's operations. These conventions are given here.

First, the set of tables has been developed directly from the Entity-Relationship diagram with minor modifications of names where ambiguity might result. This presumes that the entire system has been modeled using Chen's methodology (for brevity, we did not display this entire process in the previous section). Additionally, we have mapped each entity/relationship set into one corresponding relational table to make coded presentation easier. In actual practice, some of the relationship sets (those with 1-to-1 and 1-to-n mappings) could be combined with one of their entities giving fewer tables.

Second, all of the <u>event</u> entities have been keyed on time using the following rules:

1. Each time field consists of month-day-time; thus 06010800 signifies an event that happend on June (06) first (01) at 8 o'clock (0800)

2. The first event of each day is presumed to have happened at 8 o'clock, the second at 9 o'clock, etc.

3. When an event involves several subevents (such as making cash dispersals to multiple stockholders), they are presumed to have happened a minute apart

4. In keeping with our causal double-entry reasoning, we have designated the time columns in <u>event --- event</u> relationships as I-time (initiation) and T-time (termination)[34]

Third, the database as given represents a "snapshot" taken on the last day of June. This means that the specific insertion and update effects of all events cannot be seen. However, we will explain some of

---

[34] Ijiri, p. 72.

these specific operations in later sections.

Using these conventions and the data supplied in Exhibit 1, the reader may now follow Wilson's operations through a relational model. For illustrative purposes, we will narrate the effects of the first event given (the selling of stock).

1. This event was the first to happen on the first of June, thus its initial time occurrence is 06010800

2. This one description actually includes 12 events: 6 capital transactions (one with each stockholder) and 6 cash receipts (again with each stockholder)

3. Looking at the E-R diagram, we can see that a capital transaction will cause an insertion into both the CAPITAL-TRANSACTION table (#18) and the CAPITAL-PARTNER table (#20); it will also "trigger" an update in the SHARES column of the STOCKHOLDER table (#19)

4. Again looking at the E-R diagram, we can see that a cash receipt (in payment for a stock sale) will cause an insertion in the CASH-RECEIPT table (#13), the STOCK-PAYMENT table (#16), and the CASH-IN table (#14); it will also "trigger" an update in the AMOUNT column of the CASH table (#17).

If these operations seem numerous and involved, the reader is reminded that all of this specification, insertion and updating is done by the computing system using SEQUEL 2. We will explain in detail later the <u>triggering</u> actions involved above.

The full set of tables and its accompanying E-R diagram are followed by detailed explanations in the next section.

Details of Specific Operations

<u>Introduction</u>

Now that we have shown the relational tables for Wilson Company, we must begin a series of explanations detailing how they might be created and maintained in an operational sense. These explanations will

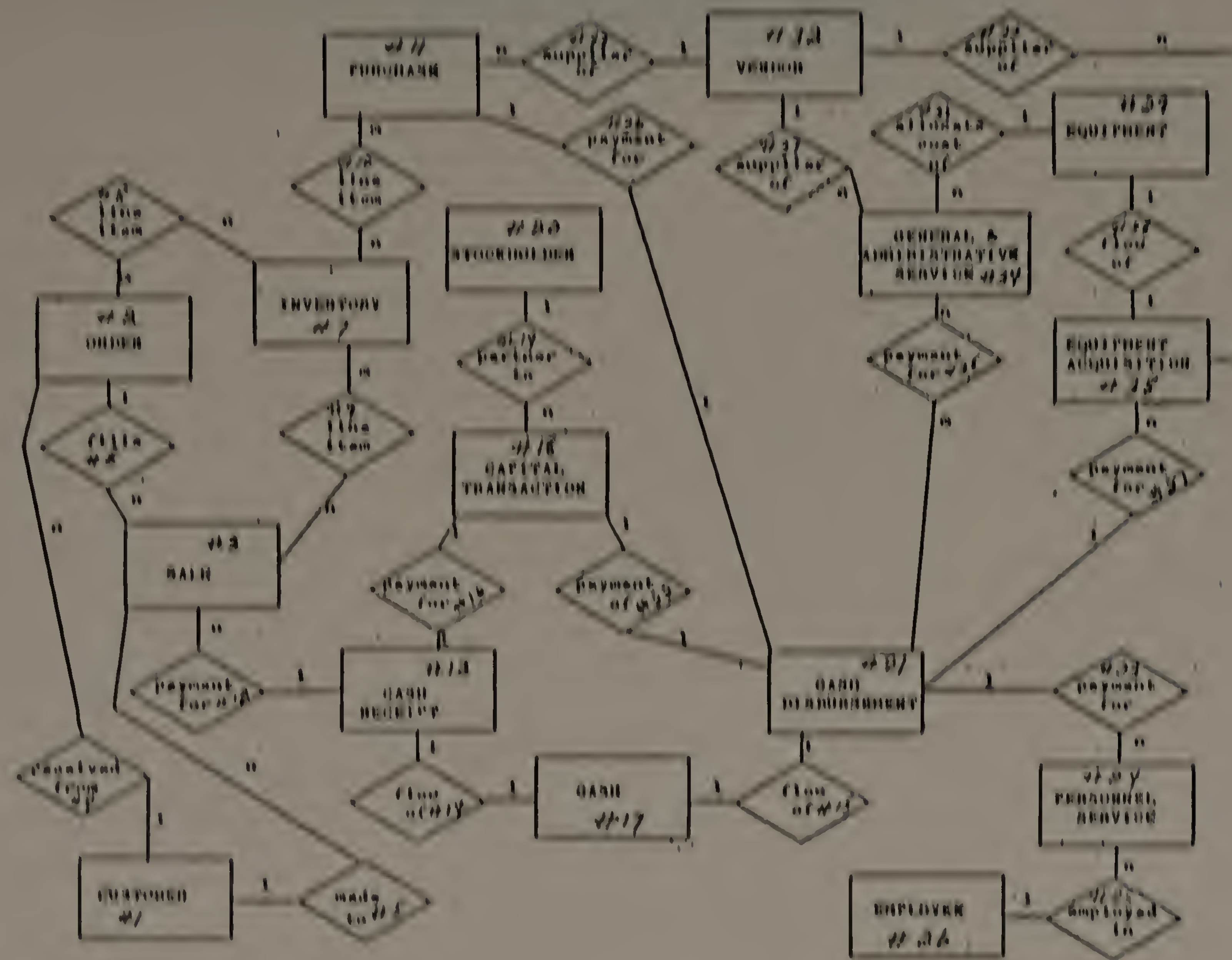


FIGURE 11--Coded E-R diagram

CUSTOMER (#1)

CUSNO	LAST NAME	FIRST NAME	CREDIT	STREET ADDRESS	CITY ADDRESS
100	White	Joe	A1	First	Medford
101	Nelson	Don	A2	Main	Malden
102	Scott	Charlie	B1	North	Revere
103	Jones	Sam	B1	Fourth	Chelsea
104	Howell	Bailey	A3	Pelham	Lynn
105	Russell	Bill	A1	Charles	Cambridge

ORDER (#2)

TIME	ORDNO	CUS ORDNO	SHIP TO STREET	SHIP TO CITY
06030900	1	67	Second	Medford
06040900	2	82	null	null
06061000	3	12	South	Revere
06110800	4	74	null	null
06160800	5	112	High	Chelsea
06170900	6	728	null	null
06190900	7	52	Prospect	Lynn
06240800	8	96	null	null
06290900	9	89	null	null

SALE (#3)

TIME	AMOUNT	INVOICE
06050800	19400	1
06060900	3000	2
06080900	7500	3
06100800	15000	4
06130800	9000	5
06180800	3000	6
06190800	42500	7
06211000	6000	8
06260800	20100	9

FIGURE 32--Wilson relational tables

CUS-ORD (#4)

CUSNO	TIME
100	06030900
101	06040900
102	06061000
100	06110800
103	06160800
105	06170900
104	06190900
101	06240800
100	06290900

CUS-SALE (#5)

CUSNO	TIME
100	06050800
101	06060900
102	06080900
100	06100800
100	06130800
103	06180800
105	06190800
104	06211000
101	06260800

ORDER-FILL (#6)

ITIME	TTIME
06030900	06050800
06040900	06060900
06061000	06080900
06030900	06100800
06110800	06130800
06160800	06180800
06170900	06190800
06190900	06211000
06240800	06260800

INVENTORY (#7)

STOCKNO	DESCR.	COST	REPLACE COST	CARRY COST	UNIT PRICE	QOH	VOLUME
7432	A	2.10	2.30	.75	3.00	2000	12
8519	B	4.20	4.30	.15	5.00	1000	3.6
6784	C	9.25	9.25	.06	12.00	1500	1.5
5862	D	10.50	11.00	.18	15.00	200	11
4888	E	1.00	1.00	.03	1.50	7000	20

FIGURE 32--Continued

ORD-LINE (#8)

STOCKNO	TIME	QUANTITY
7432	06030900	2000
8519	06030900	1000
6784	06030900	700
5862	06030900	1000
7432	06040900	1000
7432	06061000	2000
4888	06061000	1000
4888	06110800	2000
7432	06110800	2000
7432	06130800	1000
5862	06130800	300
8519	06170900	4000
6784	06170900	1500
4888	06170900	3000
7432	06190900	2000
5862	06190900	500
7432	06240800	3000
6784	06240800	800
4888	06240800	7000
6784	06290900	500
4888	06290900	4000

SALE-LINE (#9)

STOCKNO	TIME	QUANTITY
7432	06050800	2000
8519	06050800	1000
6784	06050800	700
7432	06060900	1000
7432	06080900	2000
4888	06080900	1000
5862	06100800	1000
4888	06130800	2000
7432	06130800	2000
7432	06180800	1000
8519	06190800	4000
6784	06190800	1500
4888	06190800	3000
7432	06121100	2000
6784	06260800	800
4888	06260800	7000

FIGURE 32--Continued

PURCHASE-LINE (#10)

TIME	STOCKNO	QUANTITY	PRICE
06011200	7432	6000	2.00
06011200	8519	2000	4.00
06020800	4888	20000	1.00
06020800	6784	3000	9.00
06030800	5862	600	10.00
06090800	5862	600	11.00
06100900	8519	2000	4.30
06100900	7432	2000	2.00
06170800	7432	4000	2.30
06170800	8519	2000	4.30
06180800	6784	1000	10.00
06280900	6784	500	9.25

SALE-PAY (#12)

ITIME	TTIME
06050800	06210800
06100800	06210800
06130800	06210800
06060900	06210900
06080900	06230800

CASH-RECEIPT (#13)

TIME	AMOUNT
06010800	25000
06010801	25000
06010802	25000
06010803	25000
06010804	25000
06010805	25000
06210800	43400
06210900	3000
06230800	7500

FIGURE 32--Continued

PURCHASE (#11)

TIME	PURORD	DISCOUNT	DIS-DATE	QUALITY	TRANS
06011200	1	null	null	9	null
06020800	2	null	null	9	null
06030800	3	null	null	6	null
06090800	4	null	null	8	null
06100900	5	null	null	9	null
06170800	6	null	null	9	null
06180800	7	null	null	8	null
06280900	8	null	null	8	null

CASH-IN (#14)

ACCNO	TIME
7848	06010800
7848	06010801
7848	06010802
6767	06010803
6767	06010804
8271	06010805
7848	06210800
6767	06210900
6767	06230800

CASH-OUT (#15)

ACCNO	TIME
6767	06011000
7848	06011100
7848	06040800
6767	06060800
7848	06070800
6767	06080800
6767	06081000
6767	06130900
6767	06140800
7848	06140905
7848	06150800
6767	06200800
7848	06220800
7848	06230900
6767	06250800
7848	06280805
7848	06301006

FIGURE 32--Continued

STOCK-PAY (#16)

ITIME	TTIME
06010800	06010800
06010801	06010801
06010802	06010802
06010803	06010803
06010804	06010804
06010805	06010805

CAPITAL-TRANSACTION (#18)

TIME	TYPE	AMOUNT	SHARES
06010800	Stocksell	25000	500
06010801	Stocksell	25000	500
06010802	Stocksell	25000	500
06010803	Stocksell	25000	500
06010804	Stocksell	25000	500
06010805	Stocksell	25000	500
06301000	Dividend	1000	500
06301001	Dividend	1000	500
06301002	Dividend	1000	500
06301003	Dividend	1000	500
06301004	Dividend	1000	500
06301005	Dividend	1000	500

CASH (#17)

ACCNO	TYPE	LOCATION	AMOUNT	INTEREST COST	WITHDRAW COST
0001	Petty	Office	0	.05	0
0002	Onhand	Office	0	.05	0
7848	Check	Firstnat	8540	.05	0
6767	Check	Farmer	18950	.05	0
8271	COD	Firstsav	25000	0	250

FIGURE 32--Continued

STOCKHOLDER (#19)

SSAN	LAST NAME	FIRST NAME	STREET ADDRESS	CITY ADDRESS	SHARES
400	Conley	Gene	Yawkey	Boston	500
401	Brewer	Tom	Copley	Boston	500
402	Sullivan	Frank	Kenmore	Boston	500
403	Wall	Murray	Scollay	Boston	500
404	Delock	Ike	Beacon	Boston	500
405	Schwall	Don	Fenway	Boston	500

CAPITAL-PART (#20)

SSAN	TIME
400	06010800
401	06010801
402	06010802
403	06010803
404	06010804
405	06010805
400	06301000
401	06301001
402	06301002
403	06301003
404	06301004
405	06301005

CAPITAL-PAY (#22)

ITIME	TTIME
06301000	06301006
06301001	06301006
06301002	06301006
06301003	06301006
06301004	06301006
06301005	06301006

FIGURE 32--Continued

CASH-DISPERSAL (#21)

TIME	AMOUNT	VOUCHER	CHECK
06011000	4800	1	1
06011100	1600	2	1
06040800	8300	3	2
06060800	20000	4	2
06070800	47000	5	3
06080800	500	6	3
06081000	6000	7	4
06130900	1500	8	5
06140800	6600	9	6
06140905	3280	10	4
06150800	12600	11	5
06200800	650	12	7
06220800	17800	13	6
06230900	10000	14	7
06250800	1500	15	8
06280805	3280	16	8
06301006	6000	17	9

PERSON-PAY (#23)

ITIME	TTIME
06140900	06140905
06140901	06140905
06140902	06140905
06140903	06140905
06140904	06140905
06140905	06140905
06280800	06280805
06280801	06280805
06280802	06280805
06280803	06280805
06280804	06280805
06280805	06280805

EQUIP-PAY (#27)

ITIME	TTIME
06011000	06011000
06010900	06040800
06010901	06040800
06010902	06040800
06010903	06040800

FIGURE 32--Continued

PERSONNEL EVENT (#24)

TIME	HOURS	OUTPUT	TASK	GROSS-PAY	INCTAX
06140900	80	null	null	800	null
06140901	80	null	null	640	null
06140902	80	null	null	320	null
06140903	80	null	null	400	null
06140904	80	null	null	640	null
06140905	80	null	null	480	null
06280800	80	null	null	800	null
06280801	80	null	null	640	null
06280802	80	null	null	320	null
06280803	80	null	null	400	null
06280804	80	null	null	640	null
06280805	80	null	null	480	null
06300900	16	null	null	160	null
06300901	16	null	null	128	null
06300902	16	null	null	64	null
06300903	16	null	null	80	null
06300904	16	null	null	128	null
06300905	16	null	null	96	null

EMP-SERVICE (#25)

EMPNO	TIME
300	06140900
301	06040901
302	06140902
303	06140903
304	06140904
305	06140905
300	06280800
301	06280801
302	06280802
303	06280803
304	06280804
305	06280805
300	06300900
301	06300901
302	06300902
303	06300903
304	06300904
305	06300905

FIGURE 32--Continued

EMPLOYEE (#26)

EMPNO	FIRST NAME	LAST NAME	CITY ADDRESS	STREET ADDRESS	PAY RATE	DOB	TRAIN COST	REPLACE COST	PSYCH MEAS
300	Fred	Wilson	Boston	Ashmont	10.00	1946	null	null	null
301	Bob	Jackson	Mattapan	Berkley	8.00	1936	null	null	null
302	Bill	Johnson	Leverett	Crescent	4.00	1951	null	null	null
303	Kathy	Harrison	Boston	State	5.00	1938	null	null	null
304	Ann	Taylor	Dover	Briar	8.00	1946	null	null	null
305	Karen	Adams	Hadley	River	6.00	1947	null	null	null

MACHINE & EQUIPMENT (#29)

SERIALNO	DESC	NO OF PIECES	ACC DEP	VALUE	UNIT CAPACITY	SALVAGE	LIFE	DEP METHOD
88	Furniture	6	20	2400	null	0	10	SL
87	Typewriter	1	25	1200	null	50	8	DDB
86	Typewriter	1	25	1200	null	50	8	DDB
85	Packager	1	25	3500	18	500	9	SYD
84	Truck	1	50	4800	null	600	7	SL

FIGURE 32--Continued

EQUIP-FLOW (#28)

SERIALNO	TIME
88	06010900
87	06010901
86	06010902
85	06010903
84	06011000

EQUIP-SUPPLY (#30)

VENNO	TIME
207	06010900
207	06010901
207	06010902
207	06010903
208	06011000

EQUIP-ALLOCATE (#31)

SERIALNO	TIME
88	06300800
87	06300801
86	06300802
85	06300803
84	06300804

VENDOR (#32)

VENNO	NAME	STREET ADDRESS	CITY ADDRESS	BUSINESS
200	Oliver	Main	Everett	Wholesaler
201	Williams	Fourth	Medford	Wholesaler
202	Smith	Oak	Woburn	Wholesaler
203	Hodge	Briar	Dover	Trucker
204	Simmons	Amity	Amherst	Cleaner
205	Green	Pleasant	Chelsea	Realtor
206	McKenzie	Maple	Winthrop	Pub-Rel
207	Horvath	Evergreen	Dover	Off-Equip
208	Shore	Dogwood	Saugus	Car-Dealer

PUR-SUPPLY (#33)

VENNO	TIME
200	06011200
201	06020800
202	06030800
202	06090800
200	06100900
200	06170800
201	06180800
201	06280900

FIGURE 32--Continued

GENERAL AND ADMIN SERVICE (#34)

TIME	TYPE	AMOUNT
06011100	Rent	1600
06080800	Advertise	500
06130900	Trans	3000
06150900	Clean	300
06200800	Clean	350
06270800	Advertise	500
06290800	Trans	4000
06300800	Deprec	20
06300801	Deprec	25
06300802	Deprec	25
06300803	Deprec	25
06300804	Deprec	50

G & AS-PAY (#35)

ITIME	TTIME
06011100	06011100
06080800	06080800
06130900	06130900
06150900	06200800
06200800	06290800
06130900	06250800

G & AS-SUPPLY (#37)

VENNO	TIME
205	06011100
206	06080800
203	06130900
204	06150900
204	06200800
206	06270800
203	06290800

EQUIP-ACQUISITION (#38)

TIME	AMOUNT
06010900	2400
06010901	1200
06010902	1200
06010903	3500
06011000	4800

PUR-PAY (#36)

ITIME	TTIME	DISCOUNT LOST
06011200	06060800	0
06020800	06070800	0
06030800	06081000	0
06090800	06140800	0
06100900	06150800	0
06170800	06220800	0
06180800	06230900	0

FIGURE 32--Continued

be couched primarily in System R and SEQUEL 2 (hereafter called simply SEQUEL) terminology and will follow in this sequence:

1. A group of retrievals on the Wilson database will be given to illustrate the mapping operations of SEQUEL and to familiarize the reader with the "set-at-a-time" orientation of relational languages

2. A sample of the operations needed to define and maintain Wilson's database will be shown

3. Procedures used to maintain database integrity will be exhibited

4. Facilities used to support parametric users will be discussed

Following these explanations, the last two sections of this chapter will concentrate on illustrating the given system's ability to support both traditional and "events" accounting operations and to support maintenance of internal control and audit programs.

Throughout the remainder of this chapter, we will assume that our users (Wilson employees or a database administrator) have access to the database via a display terminal. Additionally, in accordance with our purpose mentioned in Chapter 2, we will not treat machine implementation issues, but concentrate on logical operations. Readers interested in internal or physical storage aspects are referred to Martin[35] (databases in general) and Astrahan et al.[36] (System R implementation).

We proceed now to relational operations on the Wilson database.

Introduction to SEQUEL operations

SEQUEL is a language designed to work on normalized Codd relations

---

[35] James Martin, Computer Data-Base Organization (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975), chaps. 17-36.

[36] Astrahan et al., pp. 110-129.

or tables. It is a free form linear language but is commonly formatted in blocks to illustrate the logical structure of its statements. SEQUEL statements consist of keywords (given in Chamberlin et al.[37]) plus table and column names from a given database, (Figure 32, the "Wilson" tables just presented, for our purposes). Where it might be ambiguous, column names must be qualified by their tables in the following fashion: SALE.TIME or ORDER.TIME.

The primary operation in SEQUEL is a "mapping"[38]--a procedure that uses a known set of facts (called the predicate) to find an unknown set via the use of the database tables. Most commonly, the mapping are expressed in the form of a query block using the key words SELECT, FROM, and WHERE as illustrated in the two retrievals shown in Exhibit 2.

Simple Retrieval 1 shows a mapping designed to find Wilson employees who live in Boston and who were born before 1940. The reader may verify the answer to this question by consulting the EMPLOYEE table (#26) in Figure 32. Note that the result of this query is itself a table (relation) and could be further operated on if desired. This very important property of SEQUEL and other relational languages is called "closure"[39] and allows a user to validly specify any number of "cutting

---

[37]Chamberlin et al., pp. 572-74.

[38]The reader will notice that these "mapping" procedures are totally associative, that is they retrieve on data values rather than on keys or position. SEQUEL does not use explicitly the concept of primary key (PK): our specification of PK's earlier was based on their use in developing fully normalized relations.

[39]C. J. Date, An Introduction to Database Systems, 2nd ed. (Reading, Massachusetts: Addison-Wesley Publishing Company, 1977), p. 448.

SIMPLE RETRIEVAL 1--List our employees who live in Boston and who were born before 1940

```
SELECT   LAST-NAME                  →  name of column (Col. 3)
FROM     EMPLOYEE                   →  name of table (Table #26)
WHERE    CITY-ADDRESS = 'BOSTON'    →  predicate (Col. 4 and
AND      DOB < 1940                           Col. 7)
```

(Col. 3)

LAST-NAME
Harrison

→ result (Row 4 only met test requirement)

SIMPLE RETRIEVAL 2--Same as above using set operator INTERSECT

```
(SELECT  LAST-NAME
FROM     EMPLOYEE
WHERE    CITY-ADDRESS = 'BOSTON')
      INTERSECT
(SELECT  LAST-NAME
FROM     EMPLOYEE
WHERE    DOB < 1940)
```

LAST-NAME
Wilson
Harrison

LAST-NAME
Jackson
Harrison

→

LAST-NAME
Harrison

EXHIBIT 2--Simple SEQUEL retrievals

and pasting" operations to collect data.

Simple Retrieval 2 illustrates explicitly the closure property by using the results of the two query blocks enclosed in parentheses (that is, the two SELECT clauses) as operands in a set intersection to produce a final result. This second retrieval is simply an alternate way of answering the same question as the first retrieval, but with using nested operations (operations within operations).

The two retrievals shown in Exhibit 2 illustrated the use of SEQUEL on one table only, but it is often necessary to retrieve information contained in multiple tables as shown in Exhibit 3 where we attempt to answer the paraphrased question, "Who are our vendors who supply products #8519 or #5862?"

Looking first at our E-R diagram in Figure 31 and then at the appropriate tables, we see that this retrieval can be performed in the three steps below operating on tables PURCHASE-LINE (#10), PUR-SUPPLY (#33), and VENDOR (#32):

1. Find the set of purchase events involving either product #8519 or #5862 (innermost block of Exhibit 3)

2. Find the set (where set definition does not allow duplicates) of vendors who participated in those events (second block of Exhibit 3)

3. Find the names of the identified vendors (outermost or top block of Exhibit 3)

Note that the result of each query block again is a table illustrating the "set-at-a-time" orientation of relational systems.

For our final illustration in this introductory section, we will use a retrieval that will allow the reader to see how the designers of SEQUEL have incorporated fairly complex operators of the relational cal-

SIMPLE RETRIEVAL 3--List the vendors who supply products #8519 or #5862

```
SELECT    NAME
FROM      VENDOR
WHERE     VENNO IS IN

     SELECT    VENNO
     FROM      PUR-SUPPLY
     WHERE     TIME IS IN

          SELECT    TIME
          FROM      PURCHASE-LINE
          WHERE     STOCKNO = '8519'
          OR        STOCKNO = '5862'
```

NAME
Oliver
Smith

VENNO
200
202
~~202~~
~~200~~
~~200~~

TIME
06011200
06030800
06090800
06100900
06170800

EXHIBIT 3--SEQUEL set retrieval

culus (universal quantification) and the relational algebra (join, division) into a relatively straightforward format. This illustration is shown in Exhibit 4 where both the SEQUEL statements and the resulting tables have been coded to allow easier explanation. Each of the coded sections is described below:

(a) This innermost block simply retrieves the set of all stock numbers (column STOCKNO) from the set of inventory items (table INVENTORY (#7)) giving the table shown

(b) These two statements conceptually form the Cartesian product of the tables given in the FROM clause and then filter the result by the predicate given in the WHERE clause

(c) This statement takes the filtered table, or intermediate result, and arranges the rows into subgroups based upon the value of a given column (it clusters the rows based upon customer number in our case)

(c) This operation compares for equality the set of stock numbers in each subgroup of the table obtained in (c) above with the set of stock numbers contained in the table obtained in (a)

(e) The result of this compare between the original table #7 and the intermediate result table of step (c) is the set of customers (identified by customer number) who have bought at least some of every product in our inventory

(f) This block gives the names of the identified customers by using the set equality operator (IS IN), that is we look up customer 100 in table #1 thus completing the query

We have used dots in Exhibit 4 to indicate missing rows; obviously the number involved is much greater than those shown. The results of the final query are complete however: White is the only customer to buy all products during June.

Having illustrated some of the fundamental aspects and set orientation of SEQUEL, we will now continue with illustrations of operations needed to define and maintain the Wilson database.

COMPLEX RETRIEVAL 1--List our customers who buy all products.

```
f   SELECT    LAST-NAME
    FROM      CUSTOMER
    WHERE     CUSNO IS IN
e      SELECT    CUSNO
b      FROM      CUS-SALE, SALE-LINE
       WHERE     CUS-SALE.TIME = SALE-LINE.TIME
c      GROUP BY  CUSNO
d      HAVING    SET(STOCKNO) =
a         SELECT    STOCKNO
          FROM      INVENTORY
```

(From Table #7)

a

STOCKNO
7432
8519
6784
5862
4888

(Table #4 combined with Table #8)

b

CUSNO	TIME	STOCKNO	TIME	QUANTITY
100	06050800	7432	06050800	2000
100	06050800	8519	06050800	1000
⋮	⋮	⋮	⋮	⋮
102	06080900	7432	06080900	2000
⋮	⋮	⋮	⋮	⋮
101	06260800	4888	06260800	7000

d

(Intermediate result)

c

CUSNO	TIME	STOCKNO	TIME	QUANTITY
100	06050800	7432	06050800	2000
100	06050800	8519	06050800	1000
100	06050800	6784	06050800	700
100	06100800	5862	06100800	1000
100	06130800	4888	06130800	2000
101	06060900	7432	06060900	1000
⋮	⋮	⋮	⋮	⋮

(Result of compare)

e

CUSNO
100

(From Table #1)

f

LAST-NAME
WHITE

EXHIBIT 4--Complex SEQUEL retrieval

Definition and maintenance operations

The relational data model of Wilson's June operations given in Figure 32 assumed that the entire database structure had been defined a priori and that certain of the rows had been inserted beforehand (those dealing with employees or vendors for instance). In this section, we will look quickly at the SEQUEL operations needed to accomplish these tasks. All of the illustrations we will use are simple and require no more than a few paragraphs of explanation. The list of examples is given in Exhibit 5.

DEFINITION 1 and DEFINITION 2 (shown in Exhibit 5) illustrate the relative ease with which a SEQUEL database structure is defined and expanded. Obviously for the Wilson example, these definitions would have to be accomplished for every table before the system could become operational. Additionally, the use of CREATE and EXPAND commands would normally be limited to a specified enterprise employee, the database administrator.

Explanations of these definitions is the closest we will venture to the realm of physical database operations. Their ease of implementation, however, illustrates a critical point about the logical capabilities of a database system, that is its ability to support changing logical structures with a minimum of manpower effort and reprogramming. For example, the simple change illustrated in DEFINITION 2 would entail extensive restructuring and effort in a traditional file processing system and, in all probability, would simply not be done.

The statements needed to insert a new row or tuple into a table

DEFINITION 1--Create the stockholder table.

```
CREATE TABLE STOCKHOLDER
   (SSAN(CHAR(3),NONULL),
   LAST-NAME(CHAR(15)VAR),
   FIRST-NAME(CHAR(15)),
   STREET-ADDRESS(CHAR(15)VAR),
   CITY-ADDRESS(CHAR(15)VAR),
   SHARES(INTEGER))
```

DEFINITION 2--Add a new column called type to the stockholder table.

```
EXPAND TABLE STOCKHOLDER
   ADD COLUMN TYPE(CHAR(10)VAR)
```

INSERTION 1--Insert a new employee named Karen Adams with employee number 305, living on River Street in Hadley, making $6.00 an hour, born in 1947, and having other attributes null.

```
INSERT INTO EMPLOYEE (EMPNO, FIRST-NAME, LAST-NAME, CITY-ADDRESS,
   STREET-ADDRESS, PAY-RATE, DOB);

<'305', 'KAREN', 'ADAMS', 'HADLEY', 'RIVER', '6.00', '1947'>
```

UPDATE 1--Change the street address of stockholder 405 to 'Scollay'.

```
UPDATE      STOCKHOLDER
SET         STREET-ADDRESS = 'SCOLLAY'
WHERE       SSAN = 405
```

UPDATE 2--Declare a stock dividend of 10% on common shares.

```
UPDATE      STOCKHOLDER
SET         SHARES = SHARES*1.1
WHERE       TYPE = 'COMMON'
```

EXHIBIT 5--Definition and maintenance operations

are shown in INSERTION 1. For the Wilson database, such stand-alone SEQUEL operations would probably serve well the needs of tables with a relatively stable number of rows such as EMPLOYEE or CASH. However, for the more dynamic tables--those representing relationships or entities dealing with events--employing an INSERT command for every row addition would involve considerable duplication and operator tedium. Accordingly, we will suggest an alternate method of input for those types of tables in a later discussion.

The last operation we will exhibit under the heading of definition and maintenance will be the UPDATE command. Two types of updates are shown at the bottom of Exhibit 5, one dealing with a single member of an entity set and another dealing with multiple members. The form of both UPDATE 1 and UPDATE 2 is the same because both are conceptually dealing with a set of appropriate entities. Assuring that UPDATE 1 applies to a set with only one element is done by using the table's primary key in the predicate clause.

Maintenance of database integrity

Introduction

"The problem of integrity is the problem of ensuring--insofar as it can be ensured--that the data in the database is accurate at all times."[40] In traditional systems, integrity normally comes under the heading of input and processing control of accuracy, but in database systems, especially relational systems, we can take a much more central-

---

[40] Ibid., p. 395.

ized and dynamic approach to maintenance of integrity. SEQUEL integrity maintenance consists of two types of operations:

1. General assertions that describe required states of consistency and accuracy for a database

2. Triggered procedures that work on certain tables whenever the database undergoes specified actions

We will treat the triggered procedures first here because they are essential to the viable operation of the Wilson system. The general assertions, which are more control rather than operational actions, will be discussed second.

Triggers

When we first specified our data model using Chen's E-R methodology, we characterized some of the entities (cash, inventory, etc.) as object entities and others (sale, purchase, etc.) as event entities. As a general rule, the relationships between these object (---) event entities can be characterized as stock-flow associations where some stock object exists (such as inventory), and it is updated by flows of events (such as purchases and sales). To maintain the accurate representation of the stock entity in the database, we need to define update procedures that will be "triggered" upon the occurrence of a relevant event. The SEQUEL mechanism for accomplishing this is called a database trigger. Two of the triggers needed to make the Wilson database operational are shown in Exhibit 6 and explained below.

TRIGGER 1 sets its procedures in motion to update attributes of an inventory entity whenever a purchase of that inventory item occurs. The fact that this one trigger causes three separate attribute updates

TRIGGER 1--This trigger updates the INVENTORY table in 3 ways upon the occurrence of a purchase: (1) it calculates a new cost (based upon the weighted average method), (2) it sets a new replacement cost (equal to the latest market price), and (3) it updates the quantity on hand.

```
DEFINE TRIGGER T1

ON INSERT OF PURCHASE-LINE:

    (UPDATE     INVENTORY
    SET         COST = (QOH*COST + QUANTITY*PRICE)/(QOH + QUANTITY)
    SET         REPLACE-COST = PRICE
    SET         QOH = QOH + QUANTITY
    WHERE       PURCHASE-LINE.STOCKNO = INVENTORY.STOCKNO)
```

TRIGGER 2--This trigger updates the accumulated depreciation charged to a piece of equipment, but it does not change its value (ad hoc updates based on market conditions will be used for vaulation).

```
DEFINE TRIGGER T2

ON INSERTION OF EQUIP-ALLOCATE:

    (UPDATE     MACHINE-&-EQUIPMENT
    SET         ACC-DEP = ACC-DEP + GENERAL-AND-ADMIN-SERVICE. AMOUNT
    WHERE       EQUIP-ALLOCATE.SERIALNO = MACHINE-&-EQUIPMENT.SERIALNO
    AND         EQUIP-ALLOCATE.TIME = GENERAL-AND-ADMIN-SERVICE.TIME)
```

EXHIBIT 6--Triggered updates

shows the wide range of uses that triggers may have in a system and their potential for facilitating use of accounting information based upon different valuation bases.

TRIGGER 2 shows how the update of a stock entity attribute--accumulated depreciation--would work upon the occurrence of a depreciation expense event. Like the trigger above, it illustrates the system's potential for accommodating various valuation bases. The only difference is that both of the valuation procedures cannot be handled by triggered actions; the measurement procedure not based on transactions must rely on ad hoc updates.

This concludes our discussion of trigger-maintained integrity. We now move to the more static aspect of integrity control--assertions.

Assertions

We have mentioned already that assertions were SEQUEL operations enforcing required states of consistency and accuracy for a database. For an accountant however, they can be described in another way: they are the embodiment of the internal control procedures used to insure accuracy and completeness of accounting data and encourage adherence to prescribed managerial policies. Internal control procedures for the Wilson Company consist of (1) database assertions plus (2) other database procedures to be defined later that will enforce segregation of functional responsibilities. In this section, we will concentrate on assertions only and attempt to show the wide range of uses they have in enforcing policies and controls. The assertions to be described are given in Exhibit 7.

ASSERTION 1--Assert that no purchases are to be paid for until two days later.

```
ASSERT A1 on PUR-PAY:

ITIME + 20000 <= TTIME
```

ASSERTION 2--Assert that no payments be made to vendors unless some service has been rendered.

```
ASSERT A2:  (SELECT ITIME FROM G-&-AS-PAY)
                      IS IN
            (SELECT TIME FROM GENERAL-AND-ADMIN-SERVICE)
```

ASSERTION 3--Assert that no accounts may be deleted until they have a zero balance.

```
ASSERT A3 ON DELETION OF CASH:  AMOUNT = 0
```

ASSERTION 4--Assert that all wage checks to employees must be drawn on a specific account (#7848 for instance).

```
ASSERT A4:

   (SELECT    TTIME FROM PERSON-PAY)

              IS IN

   (SELECT    TIME

   FROM       CASH-OUT

   WHERE      ACCNO = '7848')
```

EXHIBIT 7--SEQUEL assertions

ASSERTION 5--Assert that no sales over $10,000 be made to customers not possessing an A1 or A2 credit rating.

```
ASSERT A5 ON SALE:
IF AMOUNT > 10000 THEN
(SELECT   TIME FROM SALE)
          IS IN
(SELECT   TIME
FROM      CUS-SALE
WHERE     CUSNO IS IN
   SELECT    CUSNO
   FROM      CUSTOMER
   WHERE     CREDIT IN ('A1','A2'))
```

ASSERTION 6--Assert that all cash dispersals be accompanied by consecutively numbered vouchers.

```
ASSERT A6 ON INSERTION OF CASH DISPERSAL:
   VOUCHER = (SELECT COUNT(*) FROM CASH-DISPERSAL)
```

ASSERTION 7--Assert that when a new value is determined for a piece of machinery or equipment, it must be less than the old value.

```
ASSERT A7 ON UPDATE OF MACHINE-&-EQUIPMENT.VALUE:
   NEW VALUE < OLD VALUE
```

EXHIBIT 7--<u>Continued</u>

ASSERTION 1[41] presents the control procedure used to enforce a given managerial policy. Assertions are statements that must always remain true after any database actions, such as insertions or updates, are performed. When an action that will cause an assertion to be false is performed on the system, that action is rejected and returned to the originator with an appropriate error code.

ASSERTION 2 presents a control that is a bit more complicated because it involves set operations. As we have mentioned before, the applicability of relational systems to such set-oriented processing gives them decided advantages over traditional systems.

ASSERTION 3 is an example of a more dynamic type of control that can be enforced upon certain database actions such as insertions and deletions rather than on the static table states.

ASSERTION 4 illustrates the utility of the closure property in specifying controls. The results of the bottom query-block are simply used as a superset of allowable values in specifying employee payments.

ASSERTION 5 is an example of a policy implementation that must normally be enforced by a program. It also illustrates the use of conditional statements (IF . . . THEN) in maintaining control.

---

[41]In this particular SEQUEL statement and others that will follow in this chapter, the reader will notice that we are performing arithmetic operations on the various "TIME" fields. Although the procedure works in this case, it is not always as valid. This is because the fields represent a concatenation of month-day-time with a resultant mixture of conversion moduli, for example, minute-hour conversion is modulus 60, hour-day conversion is modulus 24, etc. This problem can be solved by breaking down the "TIME" fields and specifying procedures to define the conversions, but for ease of presentation we have maintained the use of the simple forms in this chapter.

ASSERTION 6 shows the use of a specialized SEQUEL function COUNT which evaluates to the number of tuples or rows in a particular table. Since the rows in this case represent cash dispersal events, it becomes relatively simple to ensure consecutive numbering of vouchers.

Finally, ASSERTION 7 illustrates another dynamic aspect of SEQUEL's assertion capability. On the performance of update operations, it allows comparisons and operations on both the old and new field values.

The discussion of integrity maintenance--via triggers and assertions--concludes our presentation of stand-alone SEQUEL features to be used in operating the Wilson database. Before moving on to accounting uses however, we will mention one additional operational feature that could be included in a system such as Wilson's: that will be support of parametric users.

Parametric user facilities

In an earlier discussion concerning the insertion of rows into tables, we suggested that stand-alone SEQUEL use would probably suffice for insertion into tables representing entities or relationships with a fairly stable number of set elements. However for the more dynamic tables--those representing events--we concluded that separate use of an INSERT command for every addition would involve considerable duplication and operator tedium. For example to record an order event, it would be necessary to insert new entries into the ORDER (#2) table, the CUS-ORD (#14) table, and ORD-LINE (#18) table. Besides typing the INSERT command itself many times, recording an event might involve duplicate fields (such as TIME in this case) leading to obvious needless effort.

In addition to the problems mentioned above, there is another aspect of user interface to consider. Although relational languages such as SEQUEL can be designed to be "user-friendly," some of their more advanced features or some of the more complicated retrievals that they accomplish will still be beyond the comprehension of many potential users. What is needed to solve this problem is a system feature that will allow access to the relational language's capabilities without direct use of its statements.

Both of the problems mentioned above involve interaction with <u>parametric</u> users, that is users who usually interface with a system by simply supplying requested facts in predetermined form. Interaction with parametric users can be accomplished in SEQUEL by using it as a data sublanguage[42] embedded in a host general programming language such as PL/I. In this fashion, the general purpose language can be used to design the user interfaces (via display screens for example) and provide for manipulation of the database via calls to SEQUEL. To make application of this procedure more concrete, let us consider how such an interface might apply to Wilson's operations.

Suppose that Wilson Company has an Order-Sales Department whose primary responsibilities include: (1) entering customer orders, (2) recording sales, and (3) answering customer questions the most common of which are inquiries about open order status. The needs of such a department could be serviced by the set of display screens shown in

---

[42] Date, 2nd ed., p. 15.

Figure 33.

The screen shown on the left of Figure 33 is a header or menu screen that would be used to select appropriate further interactions. The two screens on the right indicate ways of solving both of the parametric interface problems we mentioned earlier. The top screen (ORDER ENTRY) would eliminate the multiple table insertions and duplications involved in entering orders. It would be faster and hopefully less tedious for an order-entry clerk. The bottom screen (OPEN ORDER INQUIRY) would allow a user to execute a fairly complicated SEQUEL query (the actual statements for this retrieval are shown later in the chapter) without direct use of the language.

We have now discussed the relational presentation of Wilson's June operations and the salient features of how such a system would be implemented. We next move to a discussion of the system's ability to support both traditional and "events" accounting operations.

Model's Ability to Support Accounting Features

Introduction

At the end of Chapter 2, we concluded that accounting was best viewed as an integral part of an enterprise's information system and further that recent research had indicated some desirable characteristics that such integrated systems ought to possess. These characteristics included (1) low levels of data aggregation, (2) multidimensional measures, and (3) high degrees of flexibility in responding to users. Additionally, although it was not stated as an explicit requirement in Chapter 2, we believe these integrated systems should also be able to

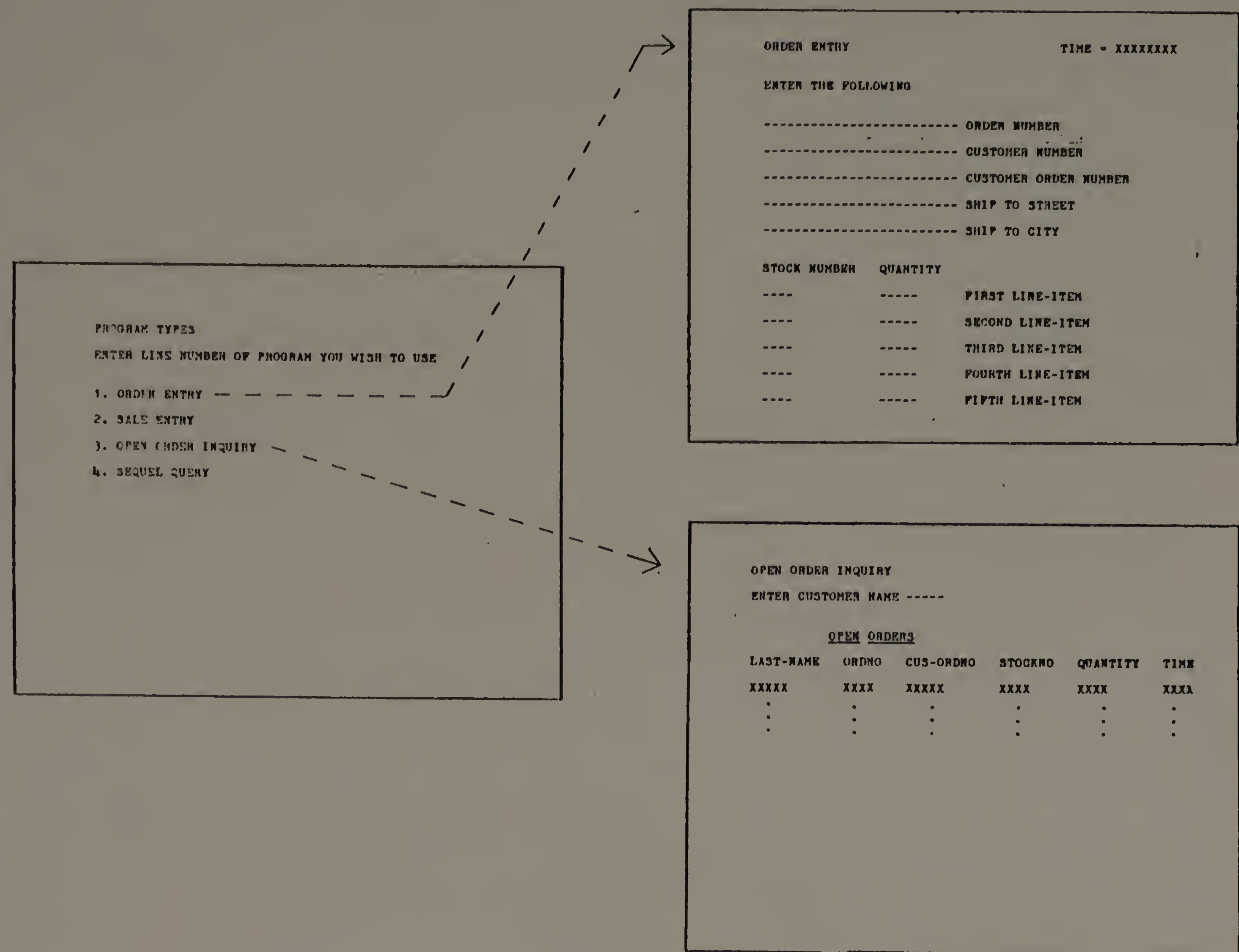


FIGURE 33--Parametric display screens

accommodate the needs of traditional accounting. Restated for the Wilson Company, this means that any of the data or reports that we can get from the double-entry T-accounts (see Figure 17) should also be obtainable from the relational system.

In this section, we intend to show our model's ability to support accounting features. This will be a twofold task. First, we will illustrate its capabilities in traditional accounting by choosing items from the financial reports (see Figure 18 and 19) and showing how they would be derived via SEQUEL. Second, we will exhibit the relational system's capability to support "events" accounting features including the desirable characteristics listed above.

Traditional Accounting Operations

The relational database that we have developed for Wilson Company certainly differs radically from the traditional accounting model generated using a chart of accounts and double-entry bookkeeping. The new system is devoid of certain familiar and financially important accounts such as accounts-receivable and cost of goods sold. Additionally, the omission of normal double-entry protocols might leave some with the uneasy feeling that all comings and goings are not being properly accounted for.

In this section we will argue that, despite the absence of the features mentioned above, the relational system can provide any information that the conventional system can. To make this point, we will show SEQUEL derivations of a balance sheet account--accounts-receivable--and a related group of income statement accounts--period expenses. The relevant statement sets are given in Exhibit 8.

ACCOUNTING-CALCULATION 1 illustrates the derivation of accounts-receivable. At any one time during the life of the enterprise, the total of outstanding receivables is the aggregate dollar amount of sale events that have not yet been paid for. This set of sale events is not present in the relational system, but it can be obtained easily by subtracting the set of paid-for sales (the elements in the SALE-PAY (#12) table) from the set of all sales (SALE (#3) table). This set difference operation is accomplished by the MINUS statement between the two indented query-blocks shown in ACCOUNTING-CALCULATION 1. The first three lines simply total the dollar amounts of the identified set elements by using a predefined function called SUM to give aggregate accounts-receivable.

ACCOUNTING-CALCULATION 2 shows the three steps necessary to obtain period expenses. To make their results clearer, we have shown the retrieval results from part (c) which may be compared to the June income statement (see Figure 18). Note that the relational system does not undergo any procedures comparable to "closing the books" in the conventional model. The events are simply described in disaggregate form and left in the database to be used or summarized by particular users according to their own needs.

"Events" Accounting Operations

The "events" orientation of our sample database should be clear. In the process of identifying relevant entities for our E-R framework, we used events as fundamental system concepts, and the database as pre-

ACCOUNTING-CALCULATION 1--List the amount of accounts-receivable.

```
SELECT      SUM(AMOUNT)
FROM        SALE
WHERE       TIME IS IN
   (SELECT     TIME
   FROM        SALE)
   MINUS
   (SELECT     ITIME
   FROM        SALE-PAY)
```

ACCOUNTING-CALCULATION 2--List the period expenses.

(a) Cost of Goods Sold

```
SELECT      SUM(QUANTITY*COST)
FROM        SALE-LINE, INVENTORY
WHERE       SALE-LINE.STOCKNO = INVENTORY.STOCKNO
AND         TIME BETWEEN '06010001' and '06302400'
```

(b) Wages

```
SELECT      SUM(GROSS-PAY)
FROM        PERSONNEL-EVENT
WHERE       TIME BETWEEN '06010001' AND '06302400'
```

(c) Other expenses

```
SELECT      TYPE, SUM(AMOUNT)
FROM        GENERAL-AND-ADMIN-SERVICE
WHERE       TIME BETWEEN '06010001' AND '06302400'
GROUP BY    TYPE
```

TYPE	AMOUNT
Rent	1600
Advertise	1000
Trans	7000
Clean	650
Deprec	145

EXHIBIT 8--Traditional accounting procedures

sented maintains their information structures in disaggregated form. Additionally, we have measured our event entities (and other entities as well) along both monetary and non-monetary dimensions and have presented SEQUEL retrievals which indicate ways of responding to ad hoc inquiries.

In this section, we will reiterate this events emphasis with a series of additional ad hoc retrievals. As we proceed through these examples (shown in Exhibits 9, 10, 11, and 12), we will also mention other system features available to support an "events" approach to accounting.

AD HOC QUERY 1 illustrates a use for a non-dollar dimension (quality rating) of an accounting event (purchase). The inclusion of such effectiveness indicators in accounting models has been recommended by a recent AAA Committee.[43] Also in the Wilson database, we can see below how other "multiple metrics"[44] of entities have been captured and could be used (many of these aspects have been included for presentation purposes only and have hence been designated as "null"):

1. The carrying cost and volume of inventory items for use in EOQ models[45]

---

[43]American Accounting Association, "Report of the Committee on Non-Financial Measures of Effectivenss," The Accounting Review 46 (Supplement 1971):164-211.

[44]A. Charnes, C. Colantoni, and W. W. Cooper, "A Futurological Justification for Historical Cost and Multi-Dimensional Accounting," Accounting, Organizations and Society 1 (November 1976):316.

[45]A. Wayne Corcoran, "Costs: Accounting Analysis & Control" Unpublished Manuscript, 1976, chap. 14.

AD HOC QUERY 1--List the average quality rating for our suppliers.

```
SELECT      NAME, AVG(QUALITY)
FROM        VENDOR, PUR-SUPPLY, PURCHASE
WHERE       VENDOR.VENNO = PUR-SUPPLY.VENNO
AND         PUR-SUPPLY.TIME = PURCHASE.TIME
GROUP BY    NAME
```

AD HOC QUERY 2--List June sales and payments by customer.

```
SELECT      CUSNO, ITIME, TTIME
FROM        CUS-SALE, SALE-PAY
WHERE       TIME = ITIME
AND         TIME BETWEEN '06010001' and '06302400'
ORDER BY    CUSNO ASC
```

EXHIBIT 9--Multidimensional and disaggregated features of system

2. The interest and withdrawals costs of cash accounts for use in money management models[46]

3. The replacement costs and psychological measures of employees for use in human resource accounting[47]

4. The unit capacities of machines for use in linear programming models[48]

In addition to capturing expanded aspects of previously recognized transactions and resources, our system could also be used to define and model new kinds of events heretofore unrecognized because of their non-market (not bought and sold) characteristics. A prime example of such an event class would be a "social" event which would periodically reflect an enterprise's positive (minority hiring, community service) and negative (pollution) contributions to society.[49]

AD HOC QUERY 2 illustrates the utility of leaving event descriptions in disaggregated form. The results of this retrieval can be used in a model designed to track customer collections:[50] a use which would not have been possible if the events had been totaled into aggregate accounts-receivable. Other decision models which require this same kind of disaggregated event description include cash flow[51] and finan-

---

[46]John W. Buckley and Kevin M. Lightner, Accounting: An Information Systems Approach (Belmont, California: Dickerson Publishing Company, 1973), pp. 412-17.

[47]Eric Flamholtz, Human Resource Accounting (Encino, California: Dickerson Publishing Company, 1974), chaps. 5-6.

[48]Corcoran, chap. 16.

[49]Charnes, Colantoni, and Cooper, pp. 315-37.

[50]Corcoran, chap. 4. [51]Buckley and Lightner, pp. 412-417.

cial planning models. Marketing management systems especially necessitate disaggregated event descriptions amenable to ad hoc retrieval as evidenced in the following quote:

> The "ideal" system would eliminate the need for [marketing] executives to explicate their models and criteria, allowing them to easily retrieve, screen, and manipulate any data they wish. These characteristics clearly suggest the need for a real-time system with extremely easy user operation, ...
>
> Care must be taken not to accumulate data in ways that appear reasonable at one time, but preclude analysis at a subsequent time. A disaggregated data file is an important feature of any MIS that hopes to respond to needs or problems that are identified only after the MIS is operational. Disaggregation refers to the maintenance of individual data in a detailed time sequence as they are generated, so that new data are not combined with existing data.[52]

AD HOC QUERY 3 shows the type of retrieval that might assist a manager in an unstructured and highly intuitive type of decision process. If a manager was thinking about reducing transportation costs by placing warehouses in locations where customers and suppliers were in close proximity, he might want some kind of preliminary indication of the extent of such possibilities. This is a type of retrieval that usually cannot be done in traditional systems because it involves an unanticipated question needing disaggregated information from files across different departmental boundaries.

AD HOC QUERY 3 also illustrates some different uses of tables in our database. Parts (a) and (b) both use ASSIGN statements to derive new tables which are then used in part (c). These derived tables are independent of their underlying source tables and can be stored or

---

[52] Lawrence D. Gibson et al., "An Evolutionary Approach to Marketing Information Systems," Journal of Marketing 37 (April 1973):3.

AD HOC QUERY 3--List the names of customers and their dollar sales who could have been supplied with products from our vendors located in the same city.

(a) Get list of products and cities they are supplied from

```
ASSIGN TO RESULT1(STOCKNO1, CITY1):
   SELECT   UNIQUE STOCKNO, CITY-ADDRESS
   FROM     PURCHASE-LINE, PUR-SUPPLY, VENDOR
   WHERE    PURCHASE-LINE.TIME = PUR-SUPPLY.TIME
   AND      PUR-SUPPLY.VENNO = VENDOR.VENNO
```

(b) Get customer names, city addresses, lists of products bought and dollar amounts for period sales

```
ASSIGN TO RESULT2(NAME, CITY2, STOCKNO2, AMOUNT):
   SELECT   LAST-NAME, CITY-ADDRESS, INVENTORY.STOCKNO, UNIT-
            PRICE*QUANTITY
   FROM     CUSTOMER, CUS-SALE, SALE-LINE, INVENTORY
   WHERE    CUSTOMER.CUSNO = CUS-SALE.CUSNO
   AND      CUS-SALE.TIME = SALE-LINE.TIME
   AND      SALE-LINE.STOCKNO = INVENTORY.STOCKNO
```

(c) Combine results to find out which sales and customers are applicable

```
SELECT      NAME, SUM(AMOUNT)
FROM        RESULT1, RESULT2
WHERE       CITY1 = CITY2
AND         STOCKNO1 = STOCKNO2
GROUP BY    NAME
```

EXHIBIT 10--SEQUEL retrieval of unanticipated information

dropped at the option of the creating user. SEQUEL also offers a mechanism to create derived tables that are linked dynamically to their underlying source tables, that is they are not independent and do reflect changes in their sources. This second type of derived table is called a "view" and will be used later.

AD HOC QUERY 4 shows a fairly complex retrieval that we have used already in our discussion of parametric users. At that time we mentioned that it might be possible to allow personnel to execute repetitive and complex queries, such as finding open orders for a customer, via the use of display screens rather than via the direct use of a query language. To be used in such a manner, AD HOC QUERY 4 would have to be embedded in a host programming language; it would also require an additional query block to specify the particular customer.

AD HOC QUERY 5 illustrates the gamut of aggregation choices for an events database. Corcoran[53] has advocated the use of "service level" as the determinant of inventory control policy instead of the use of traditional cost models, but "service level" could mean different things to different managers. Using AD HOC QUERY 5 and the derived table of AD HOC QUERY 4, we can cite the following instances of possible "service level" definitions for the Wilson Company (shown in increasing order of aggregation):

1. If a manager wanted to assess how well the company was serving its individual customers in terms of timing and amount of open orders, table OPENORDER (calculated by AD HOC QUERY 4) could be used

---

[53] Corcoran, chap. 14.

AD HOC QUERY 4--List open orders alphabetically by customer.

```
ASSIGN TO OPENORDER (NAME, ORDNO, CUS-ORDNO, STOCKNO, QUANTITY, TIME):
   SELECT    LAST-NAME, ORDNO, CUS-ORDNO, STOCKNO, QUANTITY, ORDER.
             TIME
   FROM      CUSTOMER, CUS-ORD, ORDER, ORD-LINE
   WHERE     CUSTOMER.CUSNO = CUS-ORD.CUSNO
   AND       CUS-ORD.TIME = ORDER.TIME
   AND       ORDER.TIME = ORD-LINE.TIME
   AND       <ORD-LINE.STOCKNO, ORD-LINE.TIME> DOES NOT CONTAIN
      (SELECT  ORD-LINE.STOCKNO, ORD-LINE.TIME
      FROM     ORD-LINE, ORDER-FILL, SALE-LINE
      WHERE    ORD-LINE.TIME = ITIME
      AND      SALE-LINE.TIME = TTIME
      AND      ORD-LINE.STOCKNO = SALE-LINE.STOCKNO)
   ORDER BY LAST-NAME ASC, ORDNO ASC
```

EXHIBIT 11--Open order retrieval

AD HOC QUERY 5--Find order shipment service level

(a) List open orders older than three days plus shipped orders that took more than three days to fill.

```
ASSIGN TO STOCKOUT(STOCKNO, TIME, WAIT, QUANTITY):
   (SELECT  STOCKNO, TIME, 0630000-TIME, QUANTITY)
   FROM     OPENORDER
   WHERE    TIME < 0627000)
      UNION
   (SELECT  ORD-LINE.STOCKNO, ORD-LIN.TIME, TTIME-ITIME, SALE-LINE.
            QUANTITY
   FROM     ORD-LINE, ORDER-FILL, SALE-LINE
   WHERE    ORD-LINE.TIME = ITIME
   AND      SALE-LINE.TIME = TTIME
   AND      ORD-LINE.STOCKNO = SALE.LINE.STOCKNO
   AND      (TTIME - ITIME) > 30000)
```

(b) List the number of units and transactions plus the average waiting time for each product in STOCKOUT.

```
ASSIGN TO SUM-OPENORDER(STOCKNO, UNITS, TRANSACTIONS, WAIT):
   SELECT   STOCKNO, SUM(QUANTITY), COUNT(*), AVG(WAIT)
   FROM     STOCKOUT
   GROUP BY STOCKNO
```

EXHIBIT 12--Gamut of aggregation choices

(c) List the total ordered number of units and transactions for each product in STOCKOUT.

```
ASSIGN TO SUM-ALLORDER(STOCKNO2, SUM2, COUNT2):
   SELECT   STOCKNO, SUM(QUANTITY), COUNT(*)
   FROM     ORD-LINE
   GROUP BY STOCKNO
   HAVING   STOCKNO IN STOCKOUT
```

(d) List the aggregate figures for each product suffering stockouts this month.

```
ASSIGN TO AGG-STOCKOUT(STOCKNO, PERCENT-UNITS, PERCENT-TRANSACTIONS,
AVG-WAIT):
   SELECT   STOCKNO, UNITS/SUM2, TRANSACTIONS/COUNT2, WAIT
   FROM     SUM-OPENORDER, SUM-ALLORDER
   WHERE    STOCKNO = STOCKNO2
```

EXHIBIT 12--<u>Continued</u>

2. If a manager wanted a list of all orders that had not been filled within a specified period (3 days for instance) according to company policy, table STOCKOUT (calculated by AD HOC QUERY 5 (a)) could be used

3. If a manager wanted to assess inventory policy for each product by reviewing quantities and transactions not filled, table SUM-OPENORDER (calculated in AD HOC QUERY 5 (b)) could be used

4. Finally, if a manager wanted to review relative frequencies of out-of-stock transactions for each product, table AGG-STOCKOUT (calculated in AD HOC QUERY 5 (d)) could be used

In each of the above cases, the only essential aspect of the database is the maintenance of the base relations--those tables that were described by the initial E-R design process. All of the derived relations are aggregations of those base concepts performed in accordance with individual decision models of particular managers. What is important about this process from an "events" accounting perspective is that no manager or accountant is allowed to introduce bias into the data model via aggregation, combination, or composition operations.[54]

We have now exhibited our model's ability to support normal enterprise operations and both traditional and "events" accounting. As a final example of the relational system's capabilities, we will discuss features of auditing and internal control in the next section.

Further Accounting Needs

Introduction

We have shown that our events-based system is able to handle management information and control needs for a wide range of operational and accounting requirements. In this last section dealing with

---

[54] Grace Johnson, "Towards an 'Events' Theory of Accounting," The Accounting Review 45 (October 1970):644-6.

facets of the model relational system, we will illustrate SEQUEL operations designed to support further accounting needs, specifically those of the external accountant or auditor. The two areas that we will concentrate on in this respect will be verification of internal control procedures and use of a relational language as an audit retrieval tool. The range of duties for an auditor is of much wider scope than just these areas of course, but these two subjects are the main topics of concern in an EDP environment.

Verification of Internal Control

In a previous section dealing with integrity maintenance, we discussed the issue of internal control. Internal control procedures were defined to be methods designed to encourage adherence to managerial policies and to insure the accuracy and completeness of accounting records. We also mentioned that in a relational system such as ours, internal control procedures for the database would consist primarily of two types: (1) assertions designed to enforce integrity maintenance and (2) other database procedures that would enforce segregation of functional responsibilities. Since we have already shown the use of assertions, our discussion of internal control procedures here will consist primarily of those features designed to enforce segregation of duties.

An auditor normally reviews the internal controls in an organization in an effort to determine the degree of actual record verification and testing that will be necessary for public attestation to the truthfulness of the accounting records. Generally speaking, the auditor should insure that employees be allowed access only to a subset of accounting records, never enough to conceal fraudulent recognition or omission of an entire group of related events. Additionally, with increasing federal regulation of privacy requirements in computerized databases, auditors may soon also be required to test for enterprise compliance with laws governing disclosure of employee and customer information.

The mechanism used in SEQUEL to enforce the types of access restrictions implied above is called a "view." We have mentioned this feature previously and stated at that time that views were derived relations that reflected dynamically changes in their underlying source relations.

Views can be used to look at database information in a myriad of different and restricted ways. In three SEQUEL statement blocks shown in Exhibits 13 and 14, we will show how to restrict database access to a specified group of users.

INTERNAL CONTROL 1 illustrates the VIEW definition and GRANT statements needed to allow each employee access only to his/her own personnel records. Note that, in addition to restricting a user's view of database information, these types of statements can be used also to prohibit sensitive database operations such as updating one's own PAYRATE field.

INTERNAL CONTROL 2 illustrates the use of closure in view definition. Since the result of any SEQUEL statement set is itself a valid table, that table can become an aggregated or connected view of under-

INTERNAL CONTROL 1--Allow employees to view only their own personnel records and to update their address when applicable.

(a) Define view

```
SELECT      VIEW OWN-INFO AS
   SELECT      *
   FROM        EMPLOYEE
   WHERE       EMPNO = USER
```

(b) Grant privilege to read all fields and to update addresses

```
GRANT       READ, UPDATE CITY-ADDRESS, STREET ADDRESS
ON          OWN-INFO
TO          PUBLIC
```

INTERNAL CONTROL 2--Allow the credit department to view accounts-receivable but not the underlying events.

```
SELECT      VIEW A/R-CONTROL(CUSNO, AMOUNT) AS
   SELECT      CUSNO, SUM(AMOUNT)
   FROM        CUS-SALE, SALE
   WHERE       CUS-SALE.TIME = SALE.TIME
   AND         SALE.TIME IS IN
      ((SELECT TIME
      FROM     SALE)
            MINUS
      (SELECT  ITIME
      FROM     SALE-PAY))
   GROUP BY    CUSNO
```

EXHIBIT 13--Simple control mechanisms

lying database entities.

Finally, INTERNAL CONTROL 3 shows a retrieval block that uses SEQUEL's view control features to enforce a very complex control mechanism. These view control features are described by Griffiths and Wade,[55] and use system-defined tables (such as SYSAUTH shown in INTERNAL CONTROL 3) to enforce user restrictions. This particular control feature is one that demonstrates dramatically the utility of a set-oriented system such as the one we have outlined.

We move now to another aspect of possible audit operations for the Wilson database: the use of SEQUEL procedures to perform audit retrievals.

Use of Relational Language as an Audit Retrieval Tool

> One real problem that auditors have had to face in regard to DBMS is the fact that the arsenal of computer audit software that has been developed over the years cannot, for the most part, cope with a data base file organization. Quite simply, computer audit software cannot read a data base.[56]

The statement shown above was given at a workshop during which auditors explored the impact of developing database technology upon their profession. In proposing possible solutions to the database and audit software interface problems, they decided that the most promising alternative for the future would be to "build audit functions into the

---

[55] Patricia P. Griffiths and Bradford W. Wade, "An Authorization Mechanism for a Relational Database System," ACM Transactions on Database Systems 1 (September 1976):242-255.

[56] John L. Berg, ed., DATA BASE DIRECTIONS The Next Steps (Washington, D.C.: National Bureau of Standards, 1976), p. 52.

INTERNAL CONTROL 3--Assert that no employee may be authorized to perform database actions on both of these event sets: (1) the events causing monetary liabilities of the firm to arise (purchases, general and administrative services, equipment acquisitions, and personnel services) and (2) the events designed to relieve those liabilities (cash dispersals).

```
ASSERT A8:

(SELECT     USERID
FROM        SYSAUTH
WHERE       TNAME = 'CASH-DISPERSAL')
   IS NOT IN
(SELECT     USERID
FROM        SYSAUTH
WHERE       TNAME IN ('PURCHASE', 'GENERAL-AND-ADMIN-SERVICE',
            'EQUIP-ACQUISITION', 'PERSONNEL-EVENT'))
```

EXHIBIT 14--Complex control mechanism

system."[57]

In a relational system such as Wilson's, these audit functions can be performed in most cases by the retrieval language, although some functions, especially those dealing with statistical manipulations, will need calls to the host programming language for computational processing. In this section, we will display SEQUEL's audit retrieval capability with sample audit questions. Our sample questions are shown in Exhibit 15 and are taken from literature describing the capabilities and functions of generalized audit software.

AUDIT RETRIEVAL 1 and AUDIT RETRIEVAL 2 are questions[58] designed to alert the auditor to possible out-of-control administration of corporate assets, in this case inventory and accounts-receivable. Note that the second query uses our previously defined view of outstanding accounts.

AUDIT RETRIEVAL 3 and AUDIT RETRIEVAL 4 are inquiries[59] used to assist the auditor in evaluating management purchasing policy and estimates of useful lives of assets. Note that both retrievals specify that a complete population is to be fetched. If a sample were desired (a very common audit use), SEQUEL would have to pass the population specification to a host language for sample selection.

In the discussion above, we have neglected one important aspect of using a built-in audit function for a system, that is the question

---

[57] Ibid.

[58] Ernst & Ernst, "Auditronic 32 System," Cleveland, 1976.

[59] Coopers & Lybrand, "Auditpak II," New York, 1975.

AUDIT RETRIEVAL 1--List possible obsolete inventory items by testing for quantities on hand in excess of units sold during a specified period.

```
SELECT      STOCKNO
FROM        INVENTORY
WHERE       QOH >
   SELECT      SUM(QUANTITY)
   FROM        SALE-LINE
   WHERE       STOCKNO = INVENTORY.STOCKNO
   AND         TIME BETWEEN '06010001' AND '06302400'
   GROUP BY    STOCKNO
```

AUDIT RETRIEVAL 2--Compare amounts due from individual customers with approved credit limits.

```
SELECT      CUSNO, AMOUNT, CREDIT
FROM        A/R-CONTROL, CUSTOMER
WHERE       A/R-CONTROL.CUSNO = CUSTOMER.CUSNO
```

AUDIT RETRIEVAL 3--Classify vendors by cumulative purchase volume.

```
(SELECT     VENNO, SUM(QUANTITY*PRICE)
FROM        PUR-SUPPLY, PURCHASE-LINE
WHERE       PUR-SUPPLY.TIME = PURCHASE-LINE.TIME
GROUP BY    VENNO)
ORDER BY    SUM(QUANTITY*PRICE)DESC
```

AUDIT RETRIEVAL 4--Identify fully depreciated assets.

```
SELECT      SERIALNO, DESC
FROM        MACHINE-&-EQUIPMENT
WHERE       ACC-DEP =
   SELECT      AMOUNT
   FROM        EQUIP-FLOW, EQUIP-ACQUISITION
   WHERE       EQUIP-FLOW.TIME = EQUIP-ACQUISITION.TIME
   AND         SERIALNO = MACHINE-&-EQUIPMENT.SERIALNO
```

EXHIBIT 15--Audit functions

of auditor independence. This is a very complicated issue that we will not discuss here, but it does have important implications for system design. Interest readers are referred to Weber and Jenkins,[60] and Rittenberg.[61]

Chapter Conclusion

We have now finished our presentation of a specific, events-based relational system. This presentation included:

1. Discussion of the changes needed in system orientation as we move toward a database environment

2. Illustration of the conventional accounting model

3. Exposition of Chen's E-R design methodology and its use in the development of an "events" accounting system

4. Illustration of an actual relational system and specification of relational language features needed to make the system operational

5. Evidence of the relational system's ability to support both conventional and "events" accounting

6. Use of relational language features to support internal control and audit retrieval

Our purpose in presenting a concrete example during this chapter was to show specific benefits of its use. Hopefully at this time, we have convinced readers that the integrated, set-oriented, and disaggregated approach of the relational model offers substantial advantages, especially in terms of criteria advocated by "events" accounting theorists.

---

[60] Ron Weber and A. Milton Jenkins, "Using DMBS Software as an Audit Tool: The Issue of Independence," The Journal of Accountancy 141 (April 1976), 67-9.

[61] L. E. Rittenberg, Auditor Independence and Systems Design (Altamonte Springs, Florida: The Institute of Internal Auditors, 1977).

This chapter answered the question of "How would we structure an events-based system?"; the next chapter addresses the question of "What (in terms of entities, relationships, and attributes) do we include in such a system?" Normative aspects of our model will be discussed in light of both accounting and database theory.

CHAPTER IV

NORMATIVE ASPECTS OF AN "EVENTS" DATABASE

Introduction

Normative and Positive Modes of Accounting Thought

In this chapter we will review normative aspects of an events-based relational system or, in other words, attempt to decide what entities and relationships should be included in a particular accounting model. By using the term normative, we imply value judgments, based upon specified goals, as to what "should" or "ought to be" included in our information system. Normative modes of accounting thought in this sense can be contrasted with positive modes wherein accounting is conceptualized as a basic framework for explaining certain phenomena (accounting events) free of value issues.[1]

In reality, there is rarely a clear-cut distinction between the normative and positive aspects of a particular accounting model. Yu characterizes this absence as typical of most social sciences and concludes that formulation of either type of accounting theory will necessarily be on a relative basis.[2]

We mention these distinctions here because Sorter's "events" theory is normally considered to be a positive approach, that is a way

---

[1] Detailed explanations (from which must of the logic of this section was obtained) of normative and positive modes of accounting thought are contained in S. C. Yu, The Structure of Accounting Theory (Gainesville: The University Presses of Florida, 1976), chap. 5.

[2] Ibid., pp. 130-31.

of doing accounting free of value judgments. The trouble with this characterization is that any information system, even one which purports to record transactions free of end-user bias, must decide ultimately on some basis what inputs to absorb. Sorter himself conceded this point,[3] but argued that his approach (emphasis on disaggregated event data) would lead to less valuation bias.

Our normative discussion in this chapter is concerned primarily with decisions on which basic entities, relationships, and attributes to admit to the database. The task of considering these decisions is twofold. First, we have to specify some goals we wish our events-based accounting model to possess, and second, we have to use these goals to delineate normative criteria or particular system characteristics that are desirable. For performing these tasks, neither accounting nor database theory provide definitive guidelines; in fact, the only thing they agree on is that all admission decisions are ultimately a matter of informed judgment.

We move now to the first of our tasks: discussion of normative accounting thought.

Normative Accounting Goals

In the preface to his recently published book Theory of Accounting Measurement, Ijiri identifies two major schools of normative accounting thought:

---

[3]George H. Sorter, "An 'Events' Approach to Basic Accounting Theory," The Accounting Review 44 (January 1969):14.

> This book has been written from the viewpoint that accounting is a system designed to facilitate the smooth functioning of accountability relationships among interested parties.
>
> This view is in contrast to the widespread idea that accounting is a system for providing information useful for economic decisions.[4]

These two approaches to accounting theory, namely the accountability approach (also called the historical communication approach) and the user decision model approach are the two major classes under which most normative accounting criteria can be grouped. The accountability class encompasses the traditional double-entry model used to keep financial records and to accumulate costs while the user decision model class encompasses the type of integrated accounting information systems that we mentioned in Chapter 2 (see Figure 1). A third class of normative accounting thought--the information evaluation approach--is advocated by other accounting theorists;[5] however, for reasons to be given later, we will not include discussion of that third approach at this time.

In this chapter, we will present a framework for incoporating both of the normative goals given above: provision of accountability and provision of information useful for economic decision making. Many theorists, like Ijiri, picture these two as somewhat incompatible, and such is the case in the conventional accounting model which makes no provision for disaggregation, multidimensional measurements, and different valuation bases. However, in the relational accounting system

---

[4]Yuji Ijiri, Theory of Accounting Measurement (Sarasota, Florida: American Accounting Association, 1975), p. ix.

[5]American Accounting Association, "Report of the Committee on Concpets and Standards-Internal Planning and Control," The Accounting Review 49 (Supplement 1974):81.

we have constructed, different aggregation levels, multiple valuation and recognition bases, and non-monetary metrics are accommodated easily. Thus, many forced choices between the different normative goals are avoided completely.

Our normative framework is outlined in Figure 34. Beginning with the innter circle (I), each successive ring represents an information system of increasing complexity that is able to accommodate different levels of normative goals including the goals of those systems inside of it. For instance, Level II (which represents our Wilson example) encompasses only accountability norms while Level III encompasses both accountability norms and certain types of user decision model norms.

In the next two sections, we will discuss in more detail our two major classes of normative considerations: accountability (Levels I & II) and user decision information (Levels III & IV).

Accountability

Introduction

The basis for viewing accountability as the prime normative criterion for accounting systems can be found in the enterprise theory of the firm as expressed in the following quote by Hendriksen:

> In the enterprise theory the corporation is a social institution operated for the benefit of many interested groups. In the broadest form, these groups include, in addition to the stockholders and creditors, the employees, customers, the government as a taxing authority and as a regulatory agency, and the general public.[6]

---

[6] Eldon S. Hendriksen, Accounting Theory (Homewood, Illinois: Richard D. Irwin, Inc., 1970), p. 502.

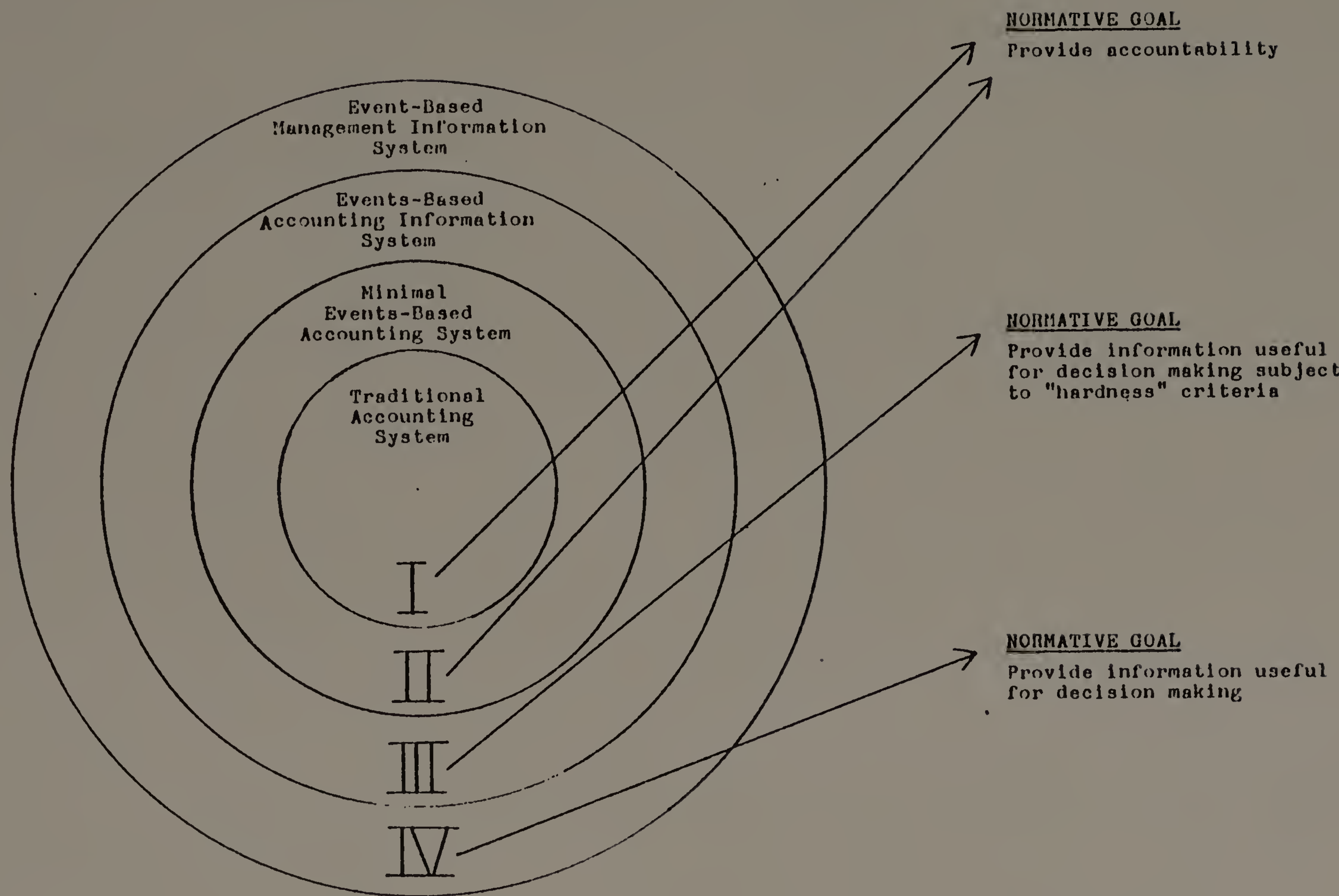


FIGURE 34--Normative framework

Ijiri states that "accountability has clearly been the social and organizational backbone of <u>accounting</u> for centuries"[7] and continues with a description of the type of system needed to support it:

> The accountability approach . . . emphasizes the assurance implicit in financial statements that figures on the statements can be accounted for by records and supporting documents of detailed transactions. Here, to <u>account for</u> means to explain a consequence (e.g. a cash balance) by providing a set of causes (e.g. cash receipts and disbursements) that have collectively produced the result. Thus, from the accountability standpoint, financial statements are merely the tip of the iceberg; what is important is the system behind the statements.[8]

The reader will notice that we have designated accountability as the normative goal for the two innermost systems shown in Figure 34. This is because we agree that it has been the prime determinant of the conventional model's operations, and we will require that our events-based system adopt as a <u>minimal goal</u> support of the accountability relationships that were provided by traditional systems. Only <u>after</u> these considerations have been sustained, will we allow our database system to expand in order to accommodate other information needs. This is actually a very realistic constraint. A business enterprise must direct its information system first toward tracking basic financial matters, such as keeping abreast of monies receivable and payable, insuring that payrolls are prepared accurately, and maintaining some notion of profitability, before it can start to use that system for support of marketing or policy decisions, etc. We can call this minimal accounting system an <u>accountability infrastructure</u> because, in a database sense, it will provide a basic framework of required entities and relationships upon which we may build as new information needs arise.

---

[7]Ijiri, p. 32. [8]Ibid., p. x.

In the next section, we will attempt to describe the set of entities and relationships needed to constitute a minimal accounting system. In addition to the obvious contributions of Ijiri (who outlined an axiomatic accounting system based on accountability), this work will be heavily influenced by the ideas of Richard Mattessich (who outlined a set of basic assumptions of accounting),[9] and S. C. Yu.[10]

Accountability Entities and Relationships

Entities

Introduction

When we first discussed the notion of a data model in Chapter 3, we illustrated the abstraction process that took place as we moved from an object system to a data representation of that object system. At that time, we mentioned that moving from an object system to a data model for an accounting enterprise meant abstracting two matters of substance: (1) that certain economic states of that enterprise existed and (2) that over time, some events occurred that altered those economic states. These two matters, namely economic states and events, make up two of the major entity set classifications, objects and events, in our Wilson model. The third major classification--agents--was in actuality a method of building accountability relationships into the model. The choice of each of these entity classifications was grounded in accounting theories, the details of which we explain in the paragraphs below.

---

[9] Richard Mattessich, Accounting and Analytical Methods (Homewood, Illinois: Richard D. Irwin, Inc., 1964), chap. 2.

[10] Yu, chap. 8.

Economic Objects

At first glance, our concept of economic object appears nearly synonymous with that of Mattessich[11] or the traditional accounting concept of asset/liability or Ijiri's concept of economic resource.[12] A major difference does exist, however, in our treatment of financial objects. With the exception of "cash" (which can be considered a real object), financial objects such as accounts-receivable or payable may be incorporated in the database system, but normalization theory suggests that their proper place would be as attributes rather than as separate tables in themsevles.

A guideline for database inclusion of economic objects would be to enumerate them first according to some rule (such as Ijiri's axiom of control)[13] and then to make sure that they are being accounted for in the data model either by (1) designation as a base entity, (2) designation as an attribute of a base entity, or (3) designation as a virtual attribute (SEQUEL view or new table assignment). In many cases, the choice between (2) or (3) would depend upon the expected frequency of use as noted by Bubenko.[14]

Economic Events

In a universe governed by accountability, economic events usually imply some swapping of the economic objects defined above. This fact is noted by both Ijiri and Mattessich:

---

[11] Mattessich, p. 36. [12] Ijiri, p. 66. [13] Ibid., p. 72.

[14] Janis Bubenko, "The Temporal Dimension in Information Modeling," Research Report RC 6187 (IBM Research Laboratories, Yorktown Heights, New York, November 1976) p. 15.

> An exchange [event] is an action whereby the entity [enterprise] foregoes control over some resources [economic objects] in order to obtain control over some other resources.[15]

> Economic transactions [events] are relations arising out of the actions of producing, holding, transferring, lending, and consuming economic objects.[16]

Yu mentions six criteria for recognizing economic events[17] while Ijiri limits himself to one (an axiom of exchange). In conventional accounting, the concepts of recognition, realization,[18] and periodicity[19] govern transaction definition.

A guideline for the inclusion of economic events in the database would have to account for the theoretical considerations listed above. Unlike the economic objects however, each recognized event class would become a separate database entity set (in Chen's terms[20]). We note here that each of the events used in the Wilson example were determined by such guidelines including the event set "order" (admitted under an extended axiom of exchange) and the group of adjusting entries (admitted under the periodicity concept).

Agents

The concept of an agent in an accounting system is best defined by Mattessich:

---

[15] Ijiri, p. 61. [16] Mattessich, p. 38. [17] Yu, p. 257.

[18] American Accounting Association, "Report of the Committee on Concpets and Standards - External Financial Reporting," The Accounting Review 49 (Supplement 1974):203-22.

[19] Yu, pp. 261-62.

[20] Peter P. Chen, "The Entity-Relationship Model--Toward a Unified View of Data," ACM Transactions on Database Systems 1 (March 1976):9-36.

> Economic agents (shortly: agents) are natural persons engaged in the economic activities (actions) of producing, owning, managing, storing, transferring, lending, borrowing, and consuming economic objects.[21]

Our notion of agent will be very closely linked to the idea of parties participating in accountability relationships.[22] For purposes of database inclusion, we can use the guideline that agents will be persons participating in recognized economic events. Thus defined, each of our agent classes will become an entity set to be represented by separate tables in the database.

Conclusion of entity definitions

We have now identified the bases in theory for each of our entity classes in the accountability infrastructure. These classes are: (1) economic objects, (2) economic events, and (3) agents. Before moving on to a discussion of the relationships connecting these entities, we should mention one more accounting principle that needs to be considered at this point in the data model specification. This principle is the *accounting* concept of an economic entity (not to be confused with the normal *database* concept of an entity).

To illustrate what we mean by an *accounting* economic entity, let us first look at definitions of Yú and Mattessich:

> Accounting entities are basic economic decision making units under which scarce resources are possessed and utilized. Of particular significance is that the accounting entity, whether it is a natural person or artificial one, is viewed as having its own identity; that is, it exists in its own right.[23]

---

[21] Mattessich, p. 37. [22] Ijiri, p. ix. [23] Yu, pp. 245-46.

> An entity is a social institution which may own or owe economic objects and which can (but need not) be owned by one or more agents or other entities. . . .
>
> An entity consists either of agents or objects or both of them. Thus, every agent can be regarded as an entity but not vice versa.[24]

In specifying the Wilson database, we did not have to deal explicitly with accounting entity definitions because we considered the whole company as one business entity without subentities such as departments, divisions, etc. In dealing with more complex enterprise structures, we would have to expand our Entity-Relationship methodology[25] to multiple levels of abstraction. This would mean identifying the relevant objects, events, and agents of each subentity at one level of abstraction and then dealing with them collectively as a single agent at a higher level of abstraction. An excellent methodology for dealing with these types of database formulations is contained in a recent work by Smith and Smith.[26]

This discussion of the accounting entity principle concludes our treatment of base entities (in the database sense) to be included in the minimal accounting system. We proceed now to a discussion of the appropriate system relationships.

Relationships

Introduction

In outlining the Wilson database in Chapter 3 and in discussing accountability in this chapter, we have already alluded to the theoretical

---

[24] Mattessich, p. 38. [25] Chen, pp. 9-36.

[26] John M. Smith and Diane C. P. Smith, "Database Abstractions: Aggregation and Generalization," ACM Transactions on Database Systems 2 (June 1977):105-33.

bases for inclusion of our three major relationship classes, namely accountability relationships, stock-flow relationships, and duality relationships. In this section, we will deal more specifically with each and decide where they fit best into the accounting model

Accountability relationships

Accountability relationships include items such as equity rights, debt claims and employment rights. Ijiri best defines both their rationale and the involved parties of different kinds:

> The accountability relationship may be created by a constitution, a law, a contract, an organizational rule, a custom, or even by an informal moral obligation. A corporation is accountable to its stockholders, creditors, employees, consumers, the government, or the public in general based on a variety of relationships created between them.[27]

In our minimal accounting system, we will specify accountability relationships as associations between agents and economic events; that is the enterprise becomes accountable to certain parties via their participation in certain activities. Examples of such associations would be stockholders participating in capital transactions or vendors supplying some service. In some cases, this may lead to some very small and specific event classes (such as a single annual tax payment to the government), but it should provide a mechanism for inclusion of all accountability factors.

Stock-flow relationships

In an accounting system, the economic objects are "stocks" measured at a particular time, while the economic events are "flows" that occur

---

[27] Ijiri, p. ix.

over time and update the "stock" objects. The stock-flow interplay between objects and events is recognized implicitly by Ijiri[28] and Mattessich[29] and the importance of its consequences is noted by Yu.[30]

In our relational system, we will connect "economic object stocks" and "economic event flows" together explicitly with relationship sets. Once linked, we may use a variety of implementation features (such as SEQUEL triggers) to maintain the currency of the "stock" variables. As mentioned previously, the choice of an update feature will depend upon the expected variable's frequency of use.

Duality relationships

The final relationship class needed in an accountability model is one that links two events together where one event "is the legal or economic 'consideration' of the other."[31] Mattessich terms this relationship the duality principle and contends that it constitutes the fundamental characteristic of the accounting model:

> Wherever--in our attempt to depict phases of the economic evironment--we explicitly adapt our model to this double aspect, we are confronted with an accounting system.[32]

Ijiri also recognizes the importance of this relationship but chooses instead to call it "causal double-entry."[33]

In our model, the duality between related event sets will be maintained with relationship sets. These associations will allow us to derive any of the accounting figures that the conventional model produces

---

[28]Ibid., p. 100. [29]Mattessich, p. 36. [30]Yu, p. 242.

[31]Mattessich, p. 38. [32]Ibid., p. 27. [33]Ijiri, p. 81.

while still maintaining disaggregated event descriptions. We saw an example of such a derivation in the Wilson database with our retrieval of accounts-receivable.

Duality is the last of our relationship specifications for the accountability system. We can now take an overview of how the entities (economic objects, economic events, and agents) and relationships (accountability, stock-flow, and duality) fit together in an Entity-Relationship (E-R) structure.[34]

Accountability E-R structure

A generalized version of Chen's E-R diagram for our accountability infrastructure is shown in Figure 35. We have not specified the types of mappings (such as m-to-n) between the entity sets because they will differ depending upon the circumstances and policies of a particular enterprise.

Figure 35 represents a tool to be applied in the step-by-step construction of an entire accountability data model. Rules and conventions, such as those of Ijiri and Mattessich, can be taken from conventional accounting and used to specify the relevant entities, relationships, and attributes. These specifications can then be combined with database ideas, such as Codd's theory of normal forms[35], in order to build a minimal events-based accounting system.

---

[34] Chen, pp. 9-36.

[35] E. F. Codd, "Further Normalization of the Data Base Relational Model," in Data Base Systems, ed. R. Rustin (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972), pp. 65-98.

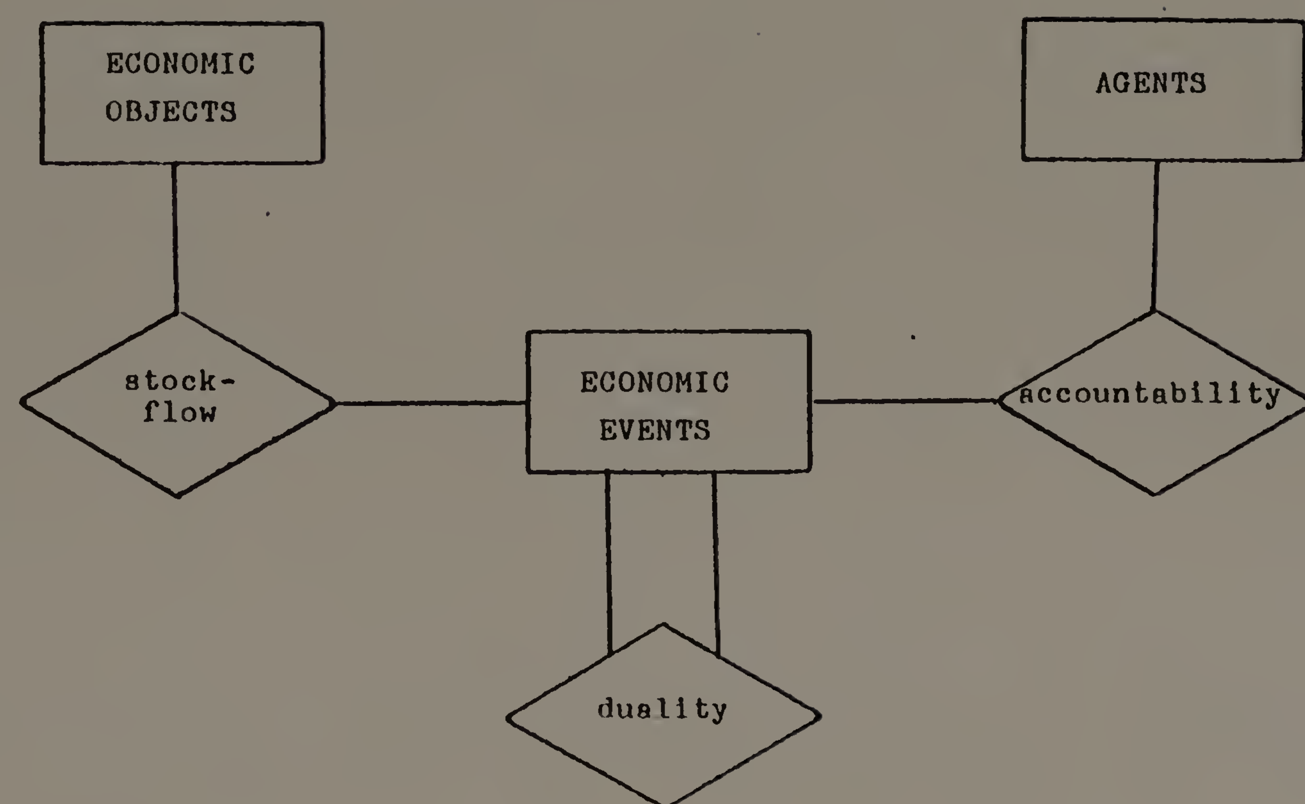


FIGURE 35--Accountability E-R structure

Accountability Conclusion

We have now completed discussion of our first normative goal--accountability. As mentioned previously, we have decided to accept support of an accountability structure as a minimum goal for an events-based information system. Once we have achieved this objective, we may start to build upon the database system in an effort to satisfy the information needs of decision makers outside of the accountability sphere.

We now turn to a discussion of criteria to be used in assessing database inclusion in this second area: provision of information useful for economic decision making.

Information Useful for Economic Decision Making

Introduction

During our normative discussion of accountability, we were able to use the relatively well defined models of accounting set forth by Ijiri and Mattessich. Each of their systems describes model elements in precise terms (in Ijiri's case by axioms, in Mattessich's by propositions) but do so by maintaining a fairly restricted notion of accounting. As we move into the area of user decision models, we will find that we are unable to remain as precise or outline our system entities in as definitive a fashion. This difficulty reflects the complexity of the expanded subject matter. Setting down normative standards for generalized, all-purpose information systems is an ambitious and amorphous task. In fact, one theorist recently concluded, after an exhaustive literature review of the area, that a generalized framework did not yet exist:

> Study reveals that though of late there is an awareness in the reported literature regarding the value aspects of information, a body of concepts needed for information evaluation in an organizational decision making context is yet to be developed.[36]

In an accounting sense, we can only partially agree with the statement above. Although there does not seem to be any all-encompassing frameworks for system evaluation, many accounting theorists have circumscribed the topic by outlining sets of desirable criteria for accounting information. Among these theorists are the following:

1. The Committee to Prepare a Statement of Basic Accounting Theory (the ASOBAT Committee) who listed a number of desirable traits in their report, chief among them the trait of usefulness[37]

2. Howard Snavely who constructed a hierarchy of criteria to be used in the selection of financial accounting information[38]

3. Daniel McDonald who promulgated feasibility criteria[39]

4. Beaver, Kennelly, and Voss who proposed predictive ability as the most desirable criterion to be used in the evaluation of accounting data[40]

5. George Staubus who consolidated the work of previous theorists (including the ASOBAT Committee, Snavely, and McDonald) in developing a multiple-criteria approach to making accounting decisions[41]

---

[36] R. Bandyopadhyay, "Information for Organizational Decisionmaking--A Literature Review," IEEE Transactions on Systems, Men and Cybernetics 7 (January 1977):1.

[37] American Accounting Association, A Statement of Basic Accounting Theory (Chicago: American Accounting Association, 1966) pp. 3-8.

[38] Howard L. Snavely, "Accounting Information Criteria," The Accounting Review 42 (April 1967):223-32.

[39] Daniel L. McDonald, "Feasibility Criteria for Accounting Measures," The Accounting Review 42 (October 1967):662-79.

[40] William H. Beaver, John W. Kennelly and William M. Voss, "Predictive Ability as a Criterion for the Evaluation of Accounting Data," The Accounting Review 43 (October 1968):675-83.

[41] George Staubus, "The Multiple-Criteria Approach to Making Accounting Decisions," Accounting and Business Research 6 (Autumn 1976):276-88.

6. Theodore Mock who developed a framework for information evaluation based upon tenets of measurement theory[42]

In our treatment of user-oriented aspects of an events system, we will choose to use the framework developed by Mock. Such a choice does not preclude consideration of the other work listed above; indeed, any accountant evaluating potential information systems should be cognizant of those theorists. However, Mock's model seems best suited to our purposes for the following reasons:

1. Its concepts have been operationalized and tested while many of the other theories give no implementation guidance and include overlapping and conflicting standards

2. It makes clear the difference between "relevancy" criteria and "hardness" criteria: a distinction which we consider important in delineating the scope of accounting information in a large database system

3. It specifically addresses the identification of relevant <u>events</u>, <u>objects</u>, and <u>attributes</u> in a user decision making context as opposed to some of the other theories which are more concerned with identification of relevant <u>procedures</u> (alternative accounting methods)

4. Many aspects of its methodology (based upon measurement theory) correspond well with Chen's E-R methodology

Having decided that Mock's framework fits our needs best, we now move to an examination of its application in a database system.

Mock's Framework

<u>Introduction</u>

Mock's basic framework of information criteria is shown in Figure 36, and the general system design steps implied in that framework are

---

[42]Theodore J. Mock, <u>Measurement and Accounting Information Criteria</u> (Sarasota: American Accounting Association, 1976), p. 88.

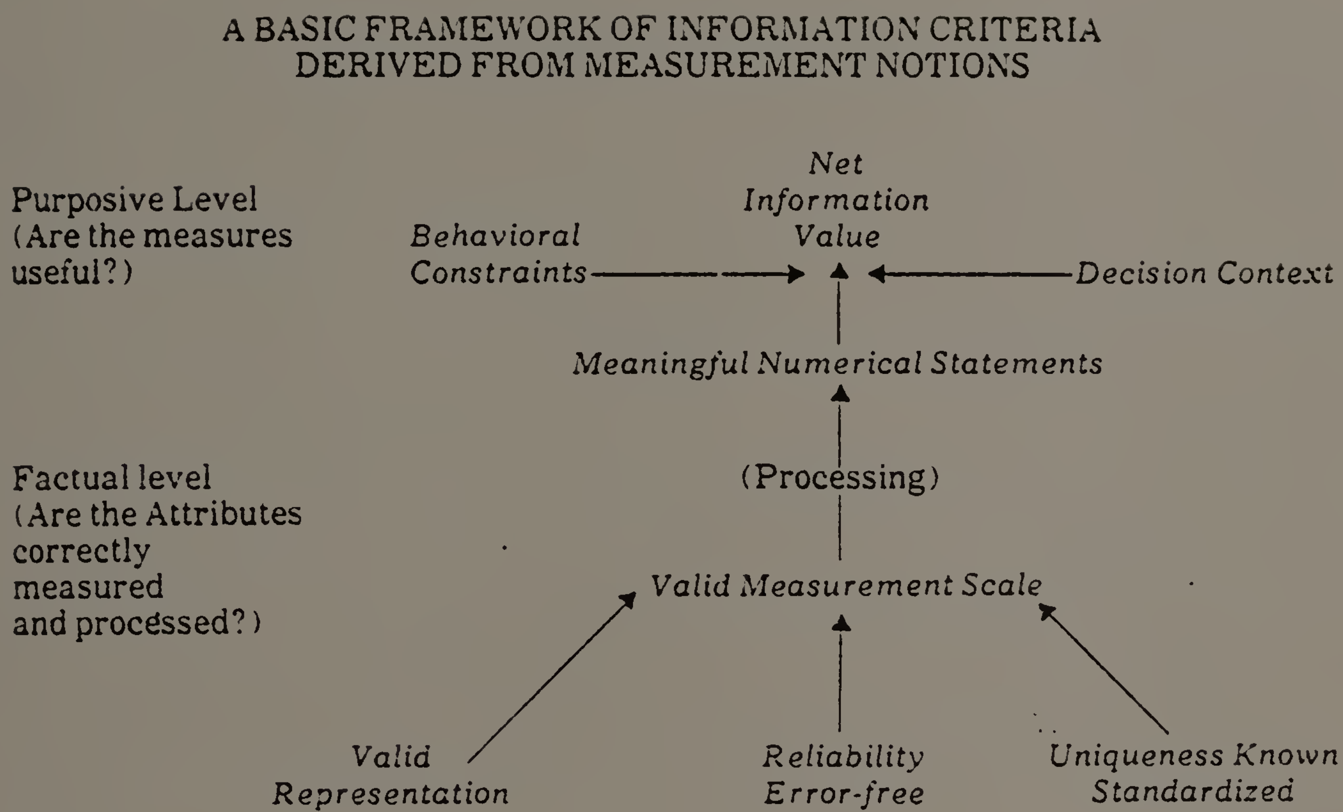


FIGURE 36--Mock's framework of information criteria

Source: Mock, p. 77.

illustrated in Exhibit 16. His system separates two levels of criteria, purposive and factual, and we will look at each in turn in following sections. First however, we will explain the rationale for the separation and its significance in our type of system.

Dual aspect of second normative goal

At the beginning of this chapter, we specified our normative outline in Figure 34. The reader will notice that the goals of both Level III (the events-based accounting information system) and Level IV (the events-based management information system) encompass user-oriented features: the difference between them is that the inner system (Level III) also requires that data be subjected to "hardness" criteria before admittance. This "hardness" standard is also an essential aspect of the accountability system (where it is commonly equated with objectivity or verifiability) and is defined by Ijiri in the following manner:

> Hard measurement may be stated as the processing of verifiable facts by justifiable rules in a rigid system which allows only a unique set of rules for a given situation.[43]

The dual aspect of desirable criteria sets at Level III, especially as it relates to inclusion in an events-based system, is reflected in the following statements taken from the accounting literature:

> An economic event is interpreted to be an occurrence in the environment of the firm, either external or internal, which is of economic significance to the decision makers of the firm.[44]

---

[43] Ijiri, p. 36.

[44] Claude S. Colantoni, Rene P. Manes, and Andrew Whinston, "A Unified Approach to the Theory of Accounting and Information Systems," The Accounting Review 46 (January 1971):91.

I. Initial Considerations at the Purposive Level

1. Specification of Decision Context
2. Determination of Needed Information
3. Determination of Available Decision Models
4. Determination of Potential Users
5. Consideration of Any Potential Behavioral Constraints

II. Construction of the Measurement System-Factual Level Considerations

1. Specification of the ERS in Theory (Step I should specify relevant objects, attributes, constructs and relations)

2. Research as to what is "known" in theory about the constructs and ERS of interest

3. Research in Existing Scales and Construction Methods for New Scales

4. Scale Choice and Evaluation (Factual Level)

   a. Existing Scales
      Evidence (prior and new) concerning
      i. Valid Representation
      ii. Reliability
      iii. Uniqueness (Standardization and implied meaningfulness)

   b. New Scales
      Evidence concerning
      i. Valid Representation
      ii. Reliability
      iii. Uniqueness

III. Final Considerations at the Purposive Level (for either new or existing scales)

1. Are the necessary conditions for relevant measurements met? (e.g., is the evidence concerning validity, reliability, etc. satisfactory)

2. What are the cost/benefit tradeoffs for alternative scales?

3. Are there any particular behavioral considerations that are evident?

EXHIBIT 16--Mock's design steps

Source: Mock, p. 88.

> Accounting theory defines a set of properties that an economic event must possess before it can be accepted as an input to the accounting system--before it can be recognized as a transaction. These properties include, for example, objectivity, quantifiability, verifiability and freedom from bias.[45]

In the first quote, the general idea of "relevance" (to economic decision makers) is the issue; in the second, "hardness" is. The distinction between these two is important in deciding at what point an events system goes beyond the realm of what we can call accounting. For instance, in Chapter 2, we referred to a suggestion by Churchill and Stedry[46] that the accounting expertise exhibited in periodic, systematic, and consistent data collection be extended to new areas of information management. Such expertise is obviously concerned with hardness criteria rather than relevance criteria, and accountants are to be considered a priori no more expert at deciding relevant information than any other class of information analysts.

In Mock's framework, purposive is synonymous with relevant and factual with hard. Elements of an accounting database will have to meet both standards. Drawing a clear distinction between Levels III and IV will be useful in accounting matters such as determination of model aspects subject to public attestation.

We will now review briefly each part of Mock's framework and

---

[45] Peter A. Firmin and James J. Linn, "Information Systems and Managerial Accounting," The Accounting Review 43 (January 1968):79.

[46] Neil C. Churchill and Andrew C. Stedry, "Some Developments in Management Science and Information Systems with Respect to Measurement in Accounting," in Research in Accounting Measurement, ed. Robert K. Jaedicke, Yuji Ijiri, and Oswald Nielsen (Sarasota: American Accounting Association, 1966), p. 30.

point out aspects especially pertinent to development of a system such as ours.

Factual criteria

The factual level of Mock's framework is undoubtedly its strongest component. Based upon established and operational concepts of measurement theory, the design steps have the additional benefit of fitting well with Chen's E-R methodology. The similarity of procedures is illustrated in Figure 37 and below.

Strictly speaking, only those Chen "value sets"[47] whose elements are numerical would correspond exactly to a Mock "numerical relational system,"[48] although other measurement theorists, such as Ackoff[49], would ease this numerical restriction to include all value sets. The purposes of the two mappings are different--Chen's attribute mapping establishes a functional dependency while Mock's homomorphic mapping preserves relationships among set elements--but compatible in the sense that Chen's procedure is a subset of Mock's. The two methodologies complement each other well in an overall design process. Mock's framework provides rigorous procedures for mapping relevant entity attributes into hard information measures. Chen's framework can take those measures and incorporate them into an integrated and disciplined set of database constructs.

---

[47]Chen, p. 12. [48]Mock, p. 11.

[49]Russell L. Ackoff, Scientific Method: Optimizing Applied Research Decisions (New York: John Wiley & Sons, Inc. 1962), pp. 178-79.

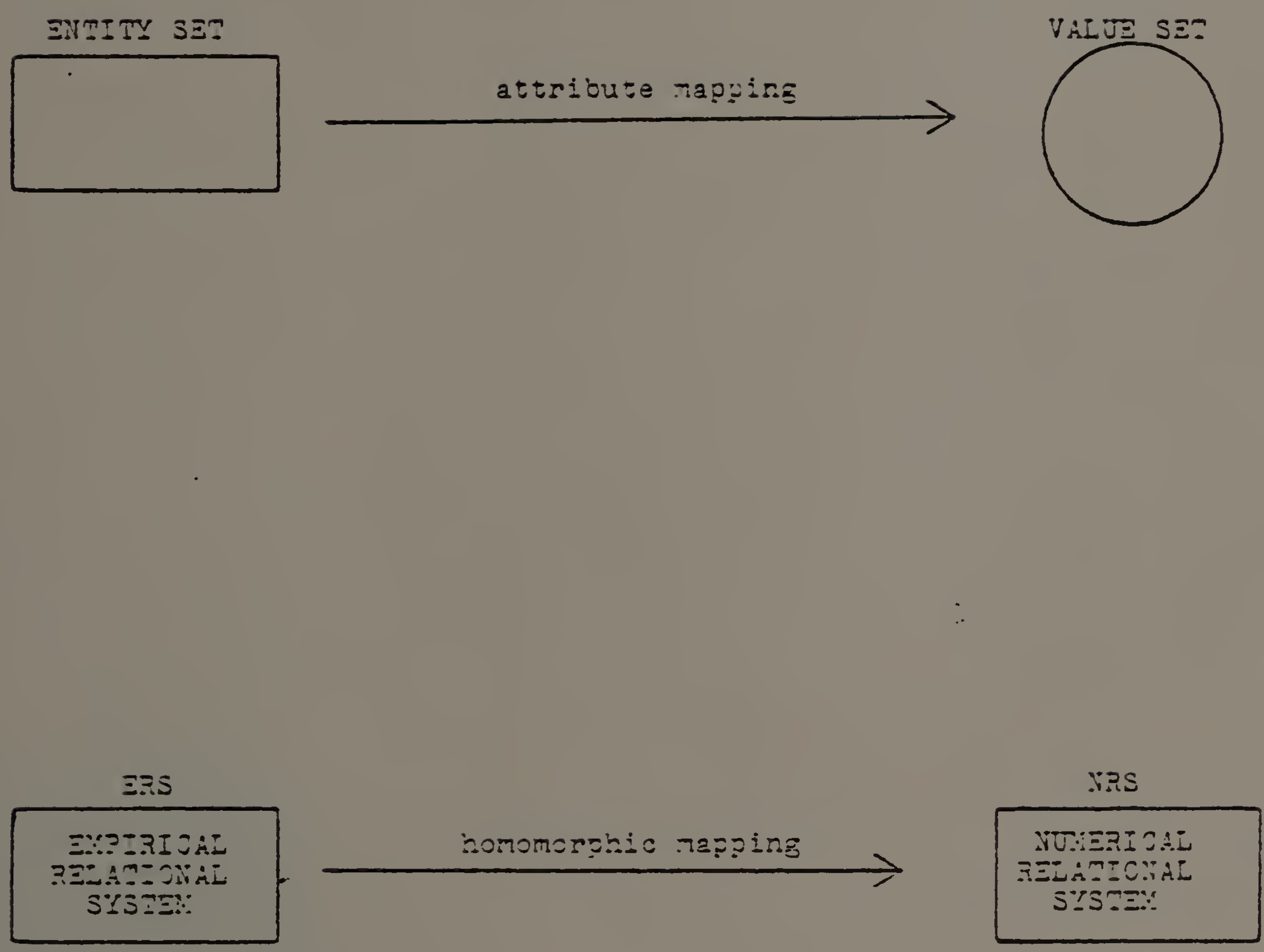


FIGURE 37--Similarity of measurement and E-R mappings

Relating Mock's work specifically to Sorter's "events"[50] theory, Mock had undertaken his inquiry into measurement concepts in order to determine if their application could result in proven information value without regard for a specified decision context. What he found in this respect was that measurement theory provided a necessary but not sufficient basis for information value. In terms of Sorter's original goal of having accountants simply record events (implying measurement of events characteristics) without any regard for ultimate users, Mock would conclude that such a value-free system was impossible.[51] We have already alluded to this fact earlier in this chapter where we noted the probable non-existence of a purely positive "events" theory.

We move now to the second aspect of Mock's framework: his purposive (relevant to economic decision making) level.

Purposive criteria

We stated earlier that there was no reason to believe that the traditional duties of accountants made them any more expert at deciding relevant information than any other class of information system user. We should keep this in mind when trying to construct an information system at the purposive level. Mock's methodology provides guidelines here but in not nearly as definitive a manner as his factual level. Database designers at this stage should avail themselves of tools in all areas of organizational decision making when trying to identify relevant entities, relationships, and attributes. These are very imprecise instructions,

---

[50] Sorter, pp. 12-19. [51] Mock, p. 83.

but as we commented earlier, their imprecision is a reflection of the uncertain nature of large, general-purpose system analysis.

We will not discuss in detail all of the purposive design steps illustrated in Exhibit 16; they are basically self-explanatory. However before leaving the topic of relevancy, we should expand briefly on the first step--specification of decision context--because of its general importance in the design of a database system.

Decision making contexts may be illustrated along two dimensions (for instance, see Figure 38). The first dimension deals with the relative structuring of the process, that is whether the decision making tends to be repetitive and programmed or highly intuitive and unique. The second dimension deals with the place in the organizational hierarchy of the process, that is whether the decisions tend to be made by low-level management with short time horizons or by corporate executives with longer horizons. For design purposes of an events database, these different contexts are important because they identify the areas most amenable to inclusion in the system. The database designer should begin with the <u>structured</u> and <u>operational</u> types of decisions, and identify relevant elements needed to support them. Subsequently, the data model can be expanded out along both dimensions, but there will usually be a point where it makes no sense to continue further. A large-scale, multiple-purpose system such as the one we built in Chapter 3 will most probably not be useful for many <u>unstructured</u> and <u>strategic</u> decisions even though it possesses extensive ad hoc capabilities. In most cases at strategic levels, the low volume and individual nature of the relevant environmental events would preclude their inclusion in a formal

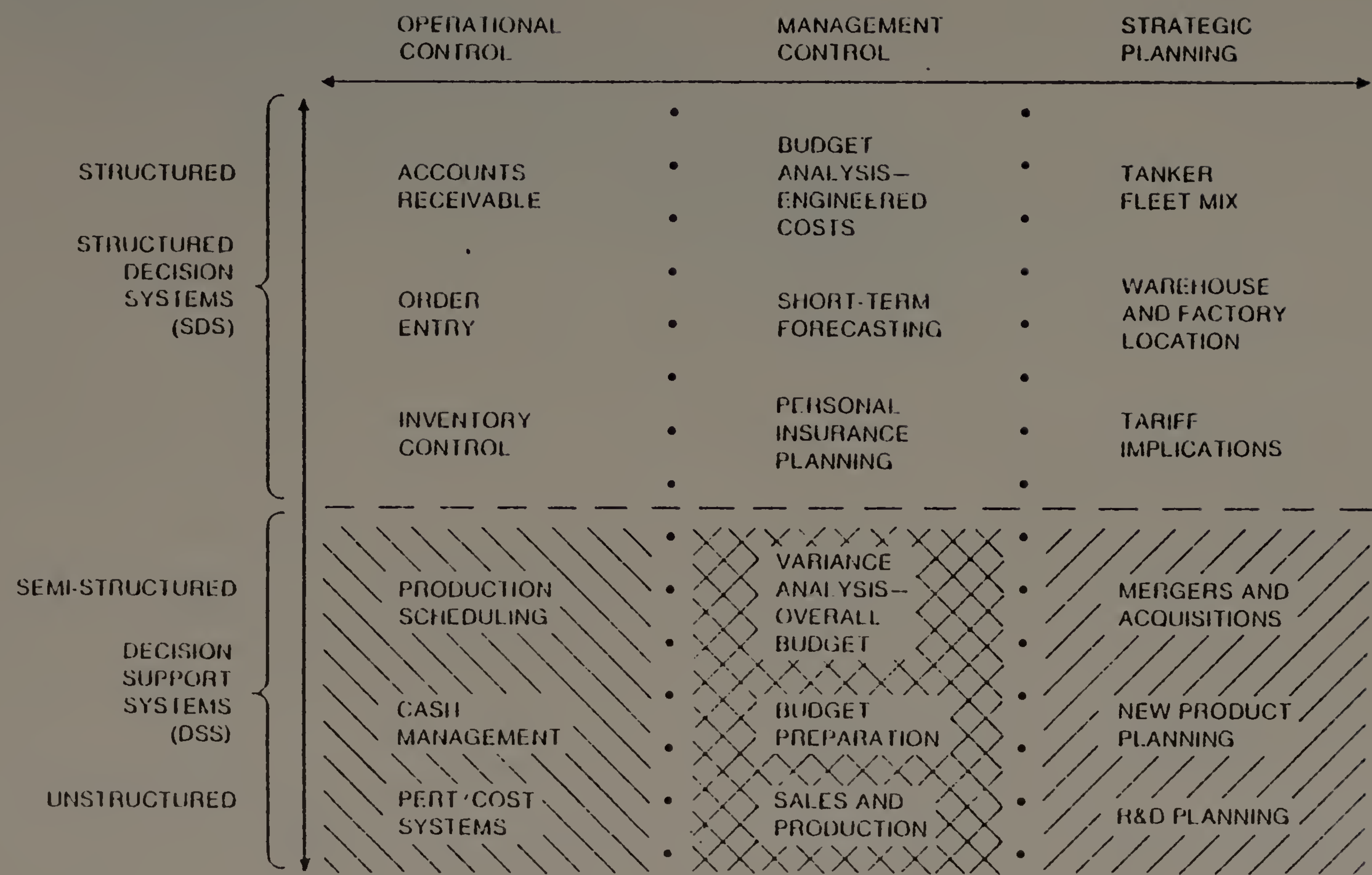


FIGURE 38--Decision making dimensions

Source: John J. Donovan and Stuart Madnick, "Institutional and Ad Hoc DSS and Their Effective Use," <u>Data Base</u> 8 (Winter 1977):80.

information system.

The discussion of Mock's framework concludes our treatment of user decision model considerations. Before moving on to a summary of our normative chapter, however, we will return briefly to a point mentioned earlier: use of information evaluation approaches in normative design.

Information Economics Considerations

When we outlined our normative goals for an events database earlier in this chapter, we included the accountability and user decision model approaches but not the information evaluation approach. We should emphasize at this point that we are not excluding the type of judgments implied by the use of information economics permanently from our database design process. However, based on a preliminary analysis of the literature in the area (chiefly that of Demski and Feltham[52]), it does not seem that information economics addresses our primary normative concern: identification of entities, relationships and attributes to be included in the database. Indeed, for an information economist, the goal of trying to accurately reflect in a data model some aspect of reality is supplanted by the goal of trying to induce some desired state partition in the utility function of a particular decision maker. Information evaluation is not concerned with the truthfulness of any data, only with the result that the use of certain data is likely to produce. As such, its use is also incompatible with our previously accepted goal of accountability.

---

[52] Joel S. Demski and Gerald A. Feltham, Cost Determination: A Conceptual Approach (Ames, Iowa: The Iowa State University Press, 1976).

In information evaluation, there is a very tight coupling of (1) information and (2) the use to which that information is put. Such an occurrence is contrary to the notion of data independence. Additionally, much of the analysis that takes place in information evaluation is concerned with different ways of doing accounting (such as different cost allocation schemes or different divisional reporting methods). These are procedural questions that are of minor concern to either a database theorist or an "events" accounting theorist.

Finally, this chapter has concentrated on an implementation plan, that is it has tried to identify portions of accounting theory that would be helpful in constructing an events-based relational system such as the one we designed in Chapter 3. Information economics has substantial implementation problems that would preclude its use in such a plan, chief among them an inability to specify methods for constructing consistent, measurable, and wide-ranging utility functions.[53]

We have now treated all of our normative topics. In the next section, we will conclude this chapter with a summary of our theoretical development plan.

Chapter Conclusion

In Chapter 3, we developed an accounting data model based upon methodology of Chen and then used Codd's concept of a relational database to build a limited operational model of an events-based information system. In this chapter, we explored the ideas of various accounting

---

[53]These practical difficulties have been noted by both Mock, p. 83 and Staubus, p. 276.

theorists in order to identify basic accounting considerations that would have to be built into such a new system. The procedures outlined in both chapters are intended to be used together in the development of an overall model that will fit the specifications we laid down in Chapter 2.

In developing our accounting database, we have recommended using the stage approach first outlined in Figure 34 and shown again in different fashion in Figure 39. Using this approach, an accountant or database designer would first examine the economic environment of an enterprise and then, using the accountablity guidelines of Ijiri and Mattessich, proceed to specification of an accountability infrastructure: a minimal accounting system composed of entities and relationships similar in generalized form to Figure 35. On top of this infrastructure could then be built additional entities, relationships, and attributes based upon the goal of providing "hard" and "relevant" information to decision makers. This Level III system could be constructed using both the factual and purposive aspects of Mock's framework, and its structure would comprise all of the model's accounting elements. Finally, the characteristics of certain entities and relationships identified as relevant to decision making but unable to meet hardness standards could be added on to the structure. We should emphasize that throughout this process, the designer should consider use of theories and tools from all areas of business administration and computer science. We propose the ideas of Ijiri, Mattessich, and Mock as guidelines, not as inflexible rules.

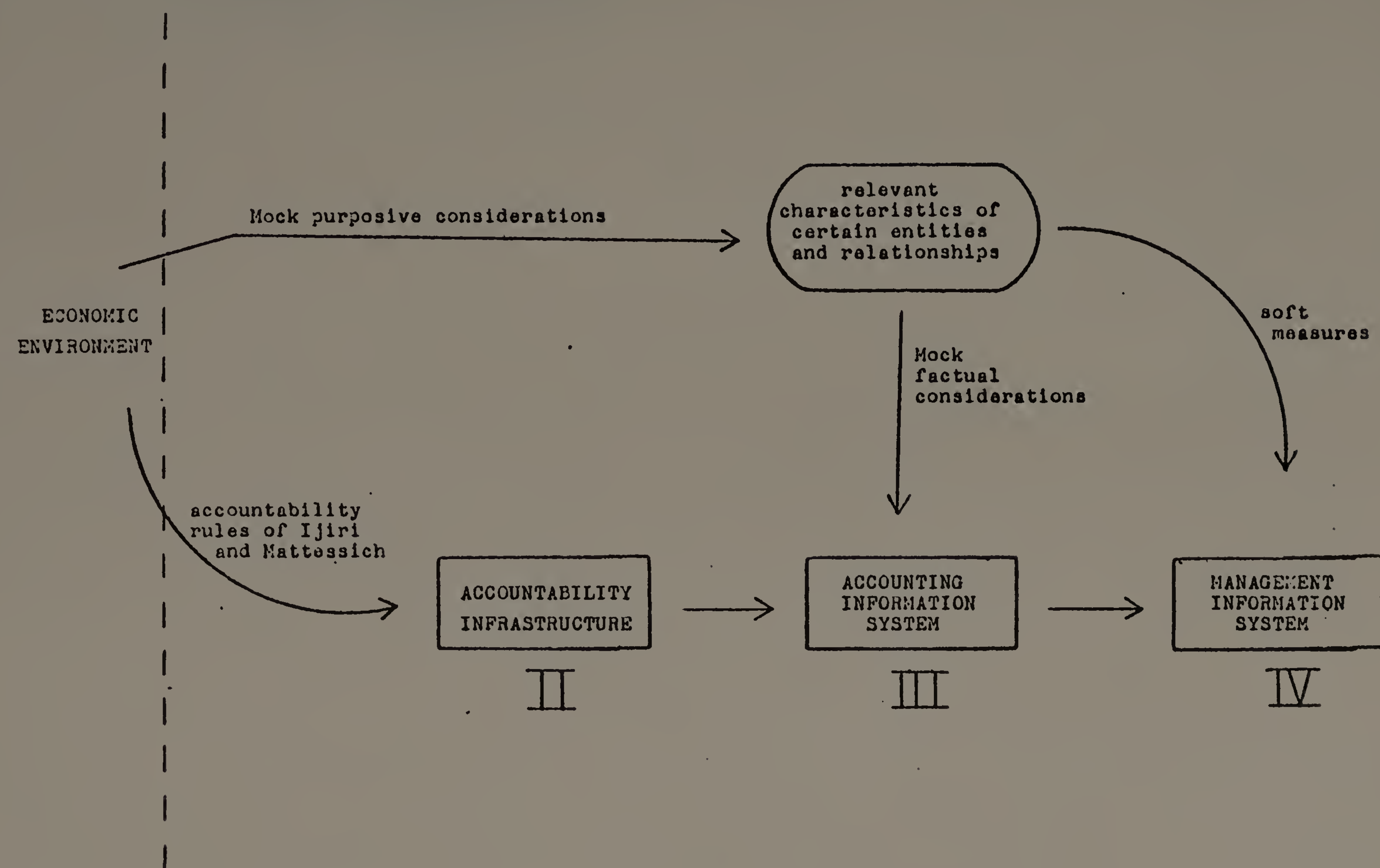
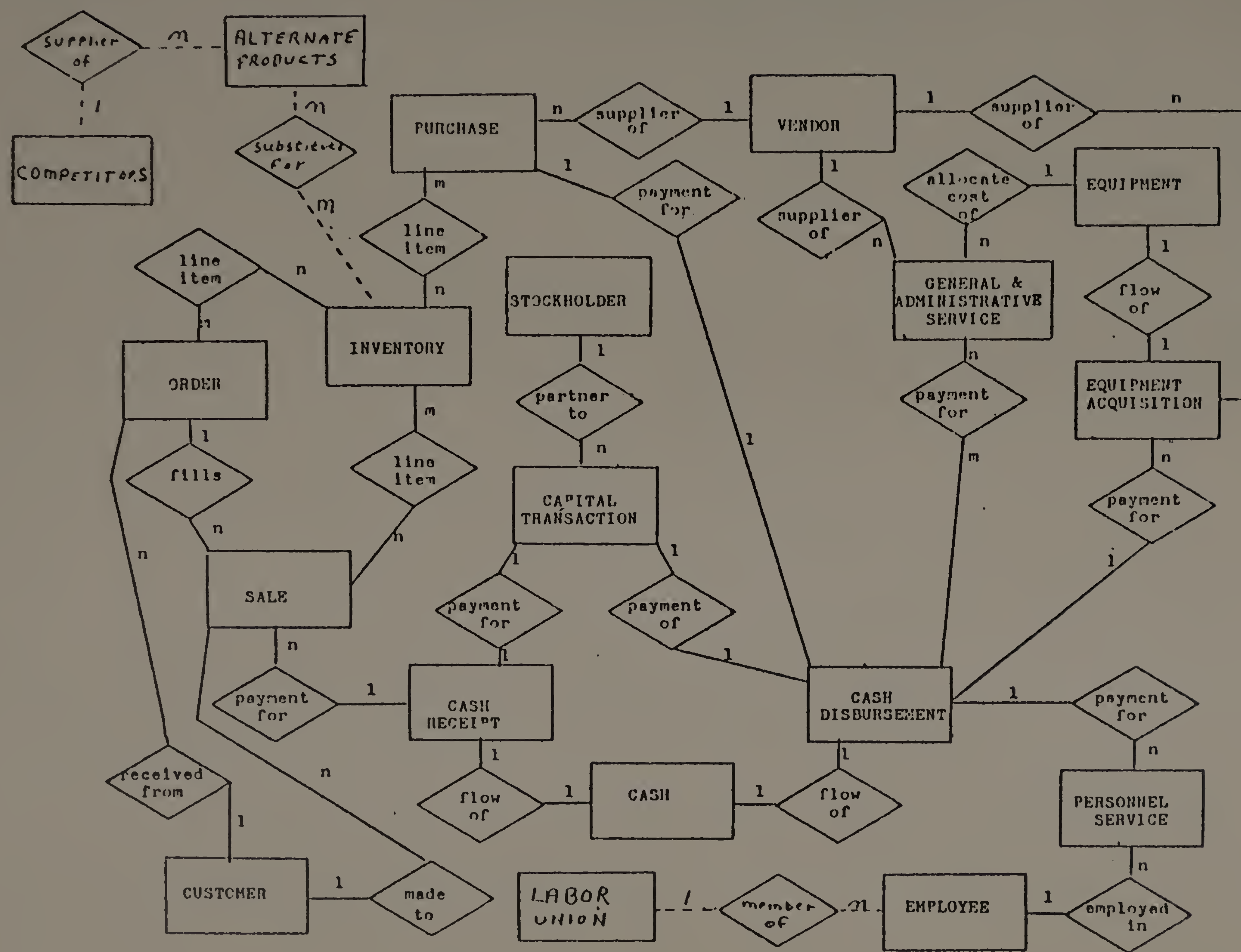


FIGURE 39--Normative implementation stages

To make this idea of a stage approach more concrete, let us consider how such a process might apply to the Wilson database of Chapter 3. In Figure 30, we illustrate with solid lines the E-R structure developed in Chapter 3, and consider it as an accountability infrastructure. Note that all of the specified entity and relationship sets fit the generalized structure of Figure 35. Now let us suppose that purposive analysis (in Mock's sense) indicates a need to consider the employees' union in the information system. It could be added on in the manner illustrated (shown at the bottom with dotted lines). In a like manner, if marketing decision makers wanted to incorporate rival products and competitors explicitly into the system, they could also be added on top of the infrastructure (shown at the upper left with dotted lines). This analysis is incomplete because we have not shown the characteristic mappings or subjected the data to hardness criteria, but the general direction of the modeling process is clear. As a final note, we add that there is certainly no rule against adding unconnected entities to the data model. Most probably in a realistic database, there will be at least a few distinct stores of information, but the designer should try for the maximum amount of integration possible.

We have now finished our proposed task. We have built an events-based relational system, we have indicated how it should be structured and used, and we have consulted both accounting and database theory to show what aspects of an enterprise's economic environment ought to be included in such a system. In our final chapter, we will examine the significance of our work and propose directions for further research.

FIGURE 40--Accountability infrastructure



CHAPTER V

SIGNIFICANCE AND FURTHER RESEARCH

Introduction

Our model building is now finished. We have considered both operational and normative aspects of an events-based relational system and shown that such a model is feasible within the limits of present technology. We do remind readers that this technology is currently limited to small experimental systems[1] (specifically System R in our case) and has not yet been extended with full capabilities to large operational environments. However, based upon recent advances in both relational theory and other types of computer technology, we can agree with Date's assessment that "the future will see the implementation of one or more large-scale systems based on the relational approach."[2]

Given our completed system, we intend to explore briefly in this chapter both its present significance and potential avenues of future research. Our treatment of significance follows next, and we will divide the discussion into three areas of potential import: (1) overall accounting theory, (2) "events" accounting theory, and (3) database theory.

---

[1]Discussion of present relational implementations can be found in C. J. Date, An Introduction to Database Systems, 2nd ed. (Reading, Massachusetts: Addison-Wesley Publishing Company, 1977), chap. 11.

[2]Ibid., p. xi.

Significance

Overall Accounting Theory

In discussing the possibility of a general theory of accounting and the feasibility of integrating it with management information theory, Richard Mattessich called for research in the following vein:

> A series of attempts should be made to construct and test alternative, comprehensive, theoretical structures amenable to management information systems, accounting systems and subsystems of them.[3]

And in the same paper he also addressed the issue of fundamental accounting research:

> The decisive matter about fundamental accounting research is not an immediate practical application, but conceptual and methodological clarification to attain new insights into the complex relations of information creation and evaluation.[4]

We view our work as falling within the purview of Mattessich's call. Certainly, the integration of Codd's and Chen's ideas into accounting theory and their use in the administration of an expanded and comprehensive information system represent collectively a step forward in the development of better methodology for the recording and transmission process. Our success in effecting this step forward is primarily attributable to the natural union of two very similar ideas: relational databases which concentrate on simple and independent data models free of user applications and "events" accounting systems which concentrate on reporting atomic events free of end-user bias.

In addition to this very general but important consideration, we

---

[3]Richard Mattessich, "Methodological Preconditions and Problems of a General Theory of Accounting," The Accounting Review 47 (July 1972): 484.

[4]Ibid., p. 481.

believe that our work relates to other areas of practice and theory in accounting.

First, we have shown that a full multidimensional model of accounting is feasible, that it can be structured to meet the needs of both accountants and non-accountants (without employment by one party rendering the data unusable by the other), and that it is able to meet the auditability and internal control constraints of traditional systems. These are all important facts for a system designer contemplating implementation of management science models and non-monetary systems of accounting such as human resource and social responsibility reporting.

Second, our work has provided a specific vehicle for incorporation of heretofore disjoint ideas such as Mock's measurement concepts and Ijiri's convention of causal double-entry into an accounting framework. Proponents of these concepts have presented them for use in accounting systems, but the ideas do not fit well with journals and ledgers. As we have seen, they do adapt well to our model.

Third, our development of Chen's Entity-Relationship (E-R) approach in an accounting environment provides guidelines that can be used to design any accounting database schema. We feel that Codd's system provides the best theoretical base, but if other considerations dictate use of an alternative database management system (DBMS), a designer can simply graft the E-R tables into the data constructs of that particular system.

Finally, our work has expanded significantly aspects of the "events" theory of accounting. This is an important topic by itself and deserves individual attention. In our next section, we will outline these expansions in more detail.

"Events" Accounting Theory

For purposes of explanation, the summary view of "events" theorists given in Chapter 2 is recreated in Figure 41 with the addition of the recent article by Everest and Weber. Given the different technological environments and different author emphases, a point by point comparison of our system with each of those shown would be a long if not impossible process. Therefore, the discussion here will relate to very general areas where we believe our work has made a significant contribution.

First, with the exception of the Everest and Weber work, our model is the only one to use relational systems. As we mentioned previously, such use provides an important theoretical foundation for the expansion of an "events" approach. Among its advantages (many of which were demonstrated in the Wilson example) are the following:

1. Increased data independence and a theory of normalization (both noted by Everest and Weber)

2. A rigorous mathematical foundation based on set theory and offering a system closed under the operations of the relational algebra

3. Relationally complete and tested user interface features such as SEQUEL and Query-by-Example[5]

4. A number of implementation advantages in the areas of integrity and security

Second, our model is the only one to be completely operationalized

---

[5]Moshe M. Zloof, "Query-By-Example," Proceedings AFIPS National Computer Conference (Montvale, New Jersey: AFIPS Press, 1975), pp. 431-38.

FIGURE 41--Review of events theorists

YEAR	TITLE	AUTHOR	IDEAS
1969	An "Events" Approach to Basic Accounting Theory	Sorter	- "Events" Accounting - Disadvantages of Value Theory - Operational Rules
1970	Toward an "Events" Theory of Accounting	Johnson	- Forecast and Observational Verification Criteria - Definition of Permissible Aggregation - Mathematical Model
1971	A Unified Approach to the Theory of Accounting and Information Systems	Colantoni, Manes and Whinston	- Introduction of Database Concepts - Event Coding - Key Algebra
1975	A Structuring of an Events-Accounting Information System	Lieberman and Whinston	- Three Part Structure - User-Defined Database Characteristics - Self-organizing Database Capabilities
1976	Design of a Multi-dimensional Accounting System	Haseman and Whinston	- Hierarchical Organization of Events Database - Definition of Restructuring Functions
1977	A Relational Approach to Accounting Models	Everest and Weber	- Data Independence - Normalization

in the sense that (1) it shows events for an entire accounting example and (2) it specifies the steps needed to account for normal business transactions and produce normal financial reports. The other systems show only fragments of an enterprise's operations and do not try to specify a complete data model.

Third, our work is the only one that tried to outline a comprehensive set of normative guidelines. Treatment of normative aspects in the other models is either (1) completely neglected or (2) done in very general (and therefore not useful) terms or (3) based on very narrow criteria sets (such as predictability).

Finally, and most importantly, our model is the only system which attempted the integration of accounting and information systems with a "top down" approach. This means that we did not take the existing "debit-credit" and "chart of accounts" framework as a given starting point and attempt to fit it into a database model. Instead, we used the Entity-Relationship approach of Peter Chen along with normative accounting theory to specify a data model unbiased by previous accounting frameworks. We believe that such an approach resulted in a much "cleaner" final product.

Consideration of our contribution to "events" theory finishes the accounting import of our work. Our final treatment of significance will be in database theory.

Database Theory

The area where our work will have the least significance is database theory. Most of the concepts which we used in our model are unknown in accounting theory but well developed and recognized here. However, there are two aspects of our system which might lead to clarification in data modeling procedures.

First, our applications area modeling is one that proceeds from theory rather than from practice. This is a restatement of a point made above but it is especially important to database theorists. It should remind them to search for the basic underlying structures of a problem area rather than being content with a user interpretation of that structure.

Second, our system is one of the few seen that considers explicitly temporal and dynamic aspects of data modeling,[6] that is it portrays an actual working enterprise and the transactions affecting it. Most database theory deals with static models uncomplicated by concepts such as our events. Our normative considerations would be especially important here because they provide guidelines (such as periodicity notions or exchange axioms) for recognition of events as database entities.

Our significance discussion is complete. We move now to our final topic--areas of further research.

Further Research

Now that we have illustrated our model in its entirety and considered some of its possible contributions to accounting and database theories, we will outline areas of further research. Some of the fol-

---

[6]The dearth of database knowledge in this particular area is noted by Janis Bubenko, "The Temporal Dimension in Information Modeling," Research Report RC 6187 (IBM Research Laboratories, Yorktown Heights, New York, November 1976) p. 19.

lowing topics are only vaguely defined, but all are based on our work here.

First, an attempt should be made to extend the accounting models developed to more complicated types of corporate structures. These should include manufacturing and budgeting environments and enterprises with subentities such as departments and divisions.

Second, user validation studies should be performed in an effort to find out if accountants can work with the novel types of models we developed here. Especially important would be any changes in their cognitive processes engendered by the lack of debit-credit conventions.

Third, work needs to be performed with Chen's E-R modeling process in order to incorporate explicitly the effects of dynamic entities such as events at the highest level of abstraction.

Fourth, the entire topic of aggregating accounting data should be reconsidered from the perspective of an events-based system with considerable ad hoc query and view definition facilities. Conclusions drawn on the basis of traditional accounting procedures might not prove valid with our different conventions.

Finally, the use of relational systems in the specification of audit retrieval and internal control needs to be researched further. We addressed these issues briefly in the Wilson database, but more comprehensive treatments are warranted.

SELECTED BIBLIOGRAPHY

Ackoff, Russell L. Scientific Method: Optimizing Applied Research Decisions. New York: John Wiley & Sons, Inc., 1962.

American Accounting Association. A Statement of Basic Accounting Theory. Chicago: American Accounting Association, 1966.

________. "Report of Committee on Managerial Decision Models." The Accounting Review 44 (Supplement 1969):43-76.

________. "Report of the Committee on Non-Financial Measures of Effectiveness." The Accounting Review 46 (Supplement 1971):164-211.

________. "Report of Committee on Accounting and Information Systems." The Accounting Review 46 (Supplement 1971):287-350.

________. "Report of the Committee on Concepts and Standards--Internal Planning and Control." The Accounting Review 49 (Supplement 1974): 75-96.

________. "Report of the Committee on Concepts and Standards--External Financial Reporting." The Accounting Review 49 (Supplement 1974): 203-22.

Astrahan, M. M.; Blasgen, M. W.; Chamberlin, D. D.; Eswaren, K.P.; Gray, J. N.; Griffiths, P. P.; King, W. F.; Lorie, R. A.; McJones, P. R.; Mehl, J. W.; Putzolu, G. R.; Traiger, I. L.; Wade, B. W.; and Watson, V. "System R: A Relational Approach to Database Management." ACM Transactions on Database Systems 1 (June 1976): 97-137.

Bandyopadhyay, R. "Information for Organizational Decisionmaking--A Literature Review." IEEE Transactions on Systems, Man and Cybernetics 7 (January 1977):1-15.

Barefield, Russell M. "The Effect of Aggregation on Decision Making Success: A Laboratory Study." Journal of Accounting Research 10 (Autumn 1972):229-42.

Beaver, William H.; Kennelly, John W.; and Voss, William M. "Predictive Ability as a Criterion for the Evaluation of Accounting Data." The Accounting Review 43 (October 1968):675-683.

Berg, John L., ed. Database Directions the Next Steps. Washington D.C.: National Bureau of Standards, 1976.

Bernhardt, Irwin and Copeland, Ronald M. "Some Problems in Applying an Information Theory Approach to Accounting Aggregation." Journal of Accounting Research 8 (Spring 1970):95-98.

Boyce, R. F.; Chamberlin, D. D.; King, W. F. III; and Hammer, M. M. "Specifying Queries as Relational Expressions." In Data Base Management, pp. 169-78. Edited by J. W. Klimbie and K. L. Koffman. Amsterdam: North Holland Publishing Company, 1974.

Bubenko, Janis. "The Temporal Dimension in Information Modeling." IBM Research Report RC 6187 (IBM Research Laboratory, Yorktown Heights, N.Y., November 1976).

Buckley, John W., and Lightner, Kevin M. Accounting: An Information Systems Approach. Belmont, California: Dickenson Publishing Company, 1973.

Butterworth, John E. "The Accounting System as an Information Function." Journal of Accounting Research 10 (Spring 1972):1-27.

Chamberlin, Donald D. "Relational Data-Base Management Systems." ACM Computing Surveys 8 (March 1976):43-66.

Chamberlin, D. D.; Astrahan, M. M.; Eswaran, K. P.; Griffiths, P. P.; Lorie, R. A.; Mehl, J. W.; Reisner, P.; and Wade, B. W. "SEQUEL 2: A Unified Approach to Data Definition, Manipulation, and Control." IBM Journal of Research and Development 20 (November 1976):560-75.

Charnes, A.; Colantoni, C.; and Cooper, W. W. "A Futurological Justification for Historical Cost and Multi-dimensional Accounting." Accounting, Organizations and Society 1 (November 1976):315-37.

Chen, Peter P. "The Entity-Relationship Model--Toward a Unified View of Data." ACM Transactions on Database Systems 1 (March 1976): 9-36.

________. "The Entity-Relationship Model: A Basis for the Enterprise View of Data." In Proceedings of the National Computer Conference, pp. 77-84. Montvale, N.J.: AFIPS Press, 1977.

Churchill, Neil C.; Kempster, John H.; and Uretsky, Myron. Computer-Based Information Systems for Management: A Survey. New York: National Association of Accountants, 1969.

Churchill, Neil C., and Stedry, Andrew C. "Some Developments in Management Science and Information Systems with Respect to Measurement in Accounting." In Research in Accounting Measurement, pp. 28-48. Edited by Robert K. Jaedicke, Yuji Ijiri and Oswald Nielsen. Sarasota, Florida: American Accounting Association, 1966.

Churchman, W. West. Prediction and Optimal Decision. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1961.

CODASYL Programming Language Committee. Data Base Task Group Report. New York: Association for Computing Machinery, 1971.

Codd, E. F. "A Relational Model of Data for Large Shared Data Banks." Communications of the ACM 13 (June 1970):377-387.

________. "A Data Base Sublanguage Founded on the Relational Calculus." In Proceedings 1971 ACM-SIGFIDET Workshop on Data Description Access and Control, pp. 35-68. New York: Association for Computing Machinery, 1971.

________. "Further Normalization of the Data Base Relational Model." In Data Base Systems, pp. 33-64. Edited by Randall Rustin. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972.

________. "Relational Completeness of Data Base Sublanguages." In Data Base Systems, pp. 65-98. Edited by Randall Rustin. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972.

________. "Access Control for Relational Data Base Systems." Presented at BCS Symposium on Relational Database Concepts, British Computer Society, London, April 1973.

________. "Seven Steps to RENDEZVOUS with the Casual User." In Proceedings IFIP TC-2 Working Conference on Data Base Management Systems, pp. 179-200. Amsterdam: North-Holland Publishing Company, 1974.

Colantoni, Claude S.; Manes, Rene P.; and Whinston, Andrew. "A Unified Approach to the Theory of Accounting and Information System." The Accounting Review 46 (January 1971):90-102.

Coopers & Lybrand. "Auditpak II." New York, 1975.

Corcoran, A. Wayne. "Costs: Accounting Analysis and Control." Unpublished Manuscript, 1976.

Crandall, Robert H. "Information Economics and Its Implications for the Further Development of Accounting Theory." The Accounting Review 44 (July 1969):457-466.

Date, C. J. An Introduction to Database Systems. Reading, Massachusetts: Addison-Wesley Publishing Company, 1975.

________. An Introduction to Database Systems, 2nd edition. Reading, Massachusetts: Addison-Wesley Publishing Company, 1977.

Davidson, Sidney. "The Day of Reckoning--Managerial Analysis and Accounting Theory." Journal of Accounting Research 1 (Autumn 1963):117-126.

Davis, Gordon B. Management Information Systems: Conceptual Foundations, Structure, and Development. New York: McGraw-Hill Book Company, 1974.

Demski, Joel S., and Feltham, Gerald A. Cost Determination: A Conceptual Approach. Ames, Iowa: The Iowa State University Press, 1976.

Devine, Carl T. "Some Conceptual Problems in Accounting Measurements." In Research in Accounting Measurement, pp. 13-27. Edited by Robert K. Jaedicke, Yuji Ijiri and Oswald Nielsen. Sarasota, Florida: American Accounting Association, 1966.

Donovan, John J., and Madnick, Stuart E. "Institutional and Ad Hoc DSS and Their Effective Use." Data Base 8 (Winter 1977):79-88.

Driver, Michael J., and Mock, Theodore J. "Human Information Processing Decision Style Theory, and Accounting Information Systems." The Accounting Review 50 (July 1975):490-508.

Eaves, B. Curtis. "Operational Axiomatic Accounting Mechanics." The Accounting Review 41 (July 1966):426-42.

Ernst & Ernst. "Auditronic 32 System." Cleveland, 1976.

Eswaran, Kapali P. "Aspects of a Trigger Subsystem in an Integrated Database System." In Proceedings of Second International Conference on Software Engineering, pp. 243-50. San Francisco, 1976.

Everest, Gordon C., and Weber, Ron. "A Relational Approach to Accounting Models." The Accounting Review 52 (April 1977):340-59.

Feltham, Gerald A. Information Evaluation. Sarasota, Florida: American Accounting Association, 1972.

Firmin, Peter A. "The Potential of Accounting as a Management Information System." Management International Review (February 1966): 45-55.

Firmin, Peter A., and Linn, James J. "Information Systems and Managerial Accounting." The Accounting Review 43 (January 1968):75-82.

Flamholtz, Eric. Human Resource Accounting. Encino, Calif.: Dickerson Publishing Co. 1974.

Fry, James B., and Sibley, Edgar H. "Evolution of Data-Base Management Systems." ACM Computing Surveys 8 (March 1976):7-42.

Gibson, Lawrence D.; Mayer, Charles S.; Nugent, Christopher E.; and Vollman, Thomas E. "An Evolutionary Approach to Marketing Information Systems." Journal of Marketing 37 (April 1973):2-6.

Griffiths, Patricia P. and Wade, Bradford W. "An Authorization Mechanism for a Relational Database System." ACM Transactions on Database Systems 1 (September 1976):242-55.

Hardgrave, W. T., and Sibley, E. H. "Problems of Basic and Applied Research in Database Systems." ACM-SIGMOD Record 8 (September 1976): 16-22.

Hare, Van Court, Jr. Systems Analysis: A Diagnostic Approach. New York: Harcourt, Brace and World, Inc., 1967.

Haseman, William D., and Whinston, Andrew B. "Design of a Multidimensional Accounting System." The Accounting Review 51 (January 1976):65-79.

Haseman, William D., and Whinston, Andrew B. Introduction to Data Management. Homewood, Illinois: Richard D. Irwin, 1977.

Hendriksen, Eldon S. Accounting Theory. Homewood, Illinois: Richard D. Irwin, Inc., 1970.

Hindman, William R., and Kettering, Floyd F. "Integrated MIS: A Case Study." Management Accounting 55 (August 1973):20-27.

Ijiri, Yuji. Theory of Accounting Measurement. Sarasota, Florida: American Accounting Association, 1975.

Jancura, Elise G., and Lilly, Fred L. "SAS No. 3 and the Evaluation of Internal Control." The Journal of Accountancy 143 (March 1977): 69-74.

Johnson, Orace. "Toward an 'Events' Theory of Accountong." The Accounting Review 45 (October 1970):641-653.

Kahn, Beverly K. "A Method for Describing the Information Required by the Data Base Design Process." In Proceedings of ACM-SIGMOD Conference on Management of Data, pp. 53-64. New York: Association for Computing Machinery, 1976.

Langefors, Borje and Samuelson, Kjell. Information and Data in Systems. Mason/Charter Publishers, Inc., 1976.

Lee, Lucy C., and Bedford, Norton M. "An Information Theory Analysis of the Accounting Process." The Accounting Review 44 (April 1969):256-275.

Lev, Baruch. Accounting and Information Theory. Sarasota, Florida: American Accounting Association, 1969.

Lieberman, Arthur Z., and Whinston, Andrew B. "A Structuring of an Events-Accounting Information System." The Accounting Review 50 (April 1975):246-258.

Lubas, Daniel P. "Developing a Computerized General Ledger System." Management Accounting 57 (May 1976):53-56.

Lucas, Henry C., Jr. "The Use of an Accounting Information System, Action and Organizational Performance." The Accounting Review 50 (October 1975):735-46.

Manna, Zohar. Mathematical Theory of Computation. New York: McGraw-Hill, Inc., 1974.

Martin, James. Design of Man-Computer Dialogues. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1973.

________. Computer Data-Base Organization. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975.

Mathews, R. L. "A Computer Programming Approach to the Design of Accounting Systmes." ABACUS 3 (December 1967):133-52.

Mattessich, Richard. Accounting and Analytical Methods. Homewood, Illinois: Richard D. Irwin, Inc., 1964.

________. "Methodological Preconditions and Problems of a General Theory of Accounting." The Accounting Review 47 (July 1972):469-87.

McDonald, Daniel L. "Feasibility Criteria for Accounting Measures." The Accounting Review 42 (October 1967):662-679.

McRae, T. W. Computers and Accounting. London: John Wiley and Sons, Ltd., 1976.

Miller, Henry. "Environmental Complexity and Financial Reports." The Accounting Review 47 (January 1972):31-7.

Mitoma, Michael F. "Optimal Data Base Schema Design." Ph.D. Dissertation, The University of Michigan, 1975.

Mock, Theodore J. Measurement and Accounting Information Criteria. Sarasota, Florida: American Accounting Association, 1976.

Morrison, Lloyd and Cooper, Kerry. Financial Accounting. Hinsdale, Illinois: The Dryden Press, 1975.

Nolan, Richard L. "Computer Data Bases: The Future is Now." Harvard Business Review 51 (September-October 1973):98-114.

Reisner, Phyllis. "Use of Psychological Experimentation as an Aid to Development of a Query Language." IBM Research Report RJ 1707 (IBM Research Laboratory, San Jose, January 1976).

Reisner, Phyllis, Boyce, Raymond F. and Chamberlin, Donald D. "Human Factors Evaluation of Two Data Base Query Languages--SQUARE and SEQUEL." In Proceedings of the National Computer Conference, pp. 447-52. Anaheim, California: AIFPS Press, 1975.

Report of the Accounting Objectives Study Group: Objectives of Financial Statements. By Robert M. Trueblood, Chairman. New York: American Institute of Certified Public Accountants, 1973.

Rittenberg, L. E. Auditor Independence and Systems Design. Altamonte Springs, Florida: The Institute of Internal Auditors, 1977.

Ronen, Joshua. "Nonaggregation Versus Disaggregation of Variances." The Accounting Review 49 (January 1974):50-60.

Sheppard, Donna L. "Data Base: A Business Approach to Systems Design." Cincom Systems, Inc., 1974.

Smith, John M. and Smith, Diane C. P. "Database Abstractions: Aggregation and Generalization." ACM Transactions on Database Systems 2 (June 1977): 105-33.

Snavely, Howard L. "Accounting Information Criteria." The Accounting Review 42 (April 1967): 223-232.

Sorter, George H. "An 'Events' Approach to Basic Accounting Theory." The Accounting Review 44 (January 1969):12-19.

Staubus, George. "The Multiple-Criteria Approach to Making Accounting Decisions." Accounting and Business Research 6 (Autumn 1976): 276-88.

Sterling, Robert R. "On Theory Construction and Verification." The Accounting Review 45 (July 1970):444-57.

Sterling, Robert R. "An Explication and Analysis of the Structure of Accounting--Part One." ABACUS 7 (December 1971):137-52.

________. "An Explication and Analysis of the Structure of Accounting--Part Two." ABACUS 8 (December 1972):145-62.

Sundgren, Bo. "Conceptual Foundations of the Infological Approach to Data Bases." In Data Base Management, pp. 61-96. Edited by J. W. Klimbie and K. L. Koffeman. Amsterdam: North Holland Publishing Company, 1974.

Weber, Ron, and Jenkins, A. Milton. "Using DBMS Software as an Audit Tool: The Issue of Independence." The Journal of Accountancy 141 (April 1976):67-9.

Wheeler, John T. "Accounting Theory and Research in Perspective." The Accounting Review 45 (January 1970):1-10.

Will, H. J. "Auditing in Systems Perspective." The Accounting Review 49 (October 1974):690-706.

________. "ACL: Audit Command Language." INFOR 13 (February 1975): 99-111.

Yu, S. C. The Structure of Accounting Theory. Gainesville: The University Presses of Florida, 1976.

Zloof, Moshe, M. "Query-By-Example." In Proceedings of the National Computer Conference, pp. 431-38. Anaheim, AFIPS Press, 1975.