THE ACCOUNTING REVIEW
Vol. LIV, No. 4
October 1979

An Entity-Relationship View of Accounting Models

William E. McCarthy

ABSTRACT: This paper is an extension of previous work aimed at integrating ideas in the field of database systems with accounting theory. Unlike others, however, it does not propose use of a particular type of database system. Instead, it concentrates on an overall design methodology—the entity-relationship model—which transcends specific database theoretical structures. A complete model is specified for a small retail enterprise, and some accounting uses for the derived system are shown. Finally, the use of an entity-relationship model in integrating aspects of measurement theory, causal double-entry, and "events" accounting with computerized information systems is discussed.

A NUMBER of recent studies have addressed the issue of integrating accounting information systems and computer database technology. Colantoni, Manes and Whinston [1971], Lieberman and Whinston [1975], and Haseman and Whinston [1976] used aspects of Sorter's [1969] "events" accounting theory and incorporated them into a hierarchical database model. More recently, Everest and Weber [1977] suggested the use of Codd's [1970] relational framework to overcome the hierarchical model's lack of data independence. Finally, Haseman and Whinston [1977] constructed an accounting system using the network or CODASYL [1971] approach.

This paper reviews the same process of integrating accounting with database models, but on a more general level that does not necessitate commitment to any particular system. In lieu of debating the comparative advantages of the hierarchical, relational, or network approaches, the contention will be made here that an accounting system is most naturally modeled in a database environment as a collection of (1) real world *entities* and (2) *relationships* among those entities. Using methodology conceived by Chen [1976], an entity-relationship model of an accounting system will be developed that will overcome some of the difficulties encountered by the authors mentioned above, such as the use of accounting "artifacts." Additionally, readers will see that this method of development will allow a system designer to incorporate explicitly into an accounting model aspects of measurement theory proposed recently by Mock [1976] and causal double-entry advocated by Ijiri [1975].

This paper is based on research originally conducted for my doctoral dissertation at the University of Massachusetts. I would like to thank the following faculty members there for their guidance and comments: Van Court Hare, Joseph Sardinas, Conrad Wogrin, David Stemple, and Morton Backer. Financial support for this work was provided by Coopers & Lybrand through a grant to Michigan State University.

William E. McCarthy is Assistant Professor of Accounting, Michigan State University.

Manuscript received June, 1977.
Revisions received December, 1977 and August, 1978.
Accepted October, 1978.

Database Abstraction Process

The data in any information system is an abstraction of some aspect of reality. The more recently developed *database* information systems require that this data representing a particular slice of reality be organized in a structured manner that will remain consistent with itself and maintain its integrity over time.

Incorporating methodologies suggested by Chen [1976], Sundgren [1974], Date [1977], and Will [1974], Figure 1 portrays this structured abstraction process as it applies to an accounting system.

Beginning with LEVEL 1, 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 [Yu, 1976, p. 245].

At LEVEL 2, the description of reality narrows to those aspects that are of interest to intended database users. Sundgren [1974, p. 61] refers to this slice of reality as the "object system," and for the accounting example, it includes 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 the abstraction process moves to LEVEL 3, it passes from the world of realities, or principals as they are called by Ijiri [1975, Ch. 3], 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 community of database users. It is a scheme that represents, with data, the organization of the conceptual world of interest.

In traditional accounting, it is here that the "artifacts" (such as the chart of accounts) deplored by Everest and Weber are encountered. Those authors characterize many traditional accounting structures as "useful taxonomies, classification schemes, or naming conventions rather than real entities" [Everest and Weber, 1977, p. 342] and consequently conclude that database systems are not able to accommodate them easily.

This paper proposes to depart from traditional accounting at LEVEL 3, and to use instead an entity-relationship view of an enterprise to construct its accounting data model. Additionally, the scheme to be developed will not be limited by the principles of double-entry and monetary measurement, but will be allowed instead to assume more of the multidimensional and disaggregated aspects proposed by "events" accounting theorists [Sorter, 1969].

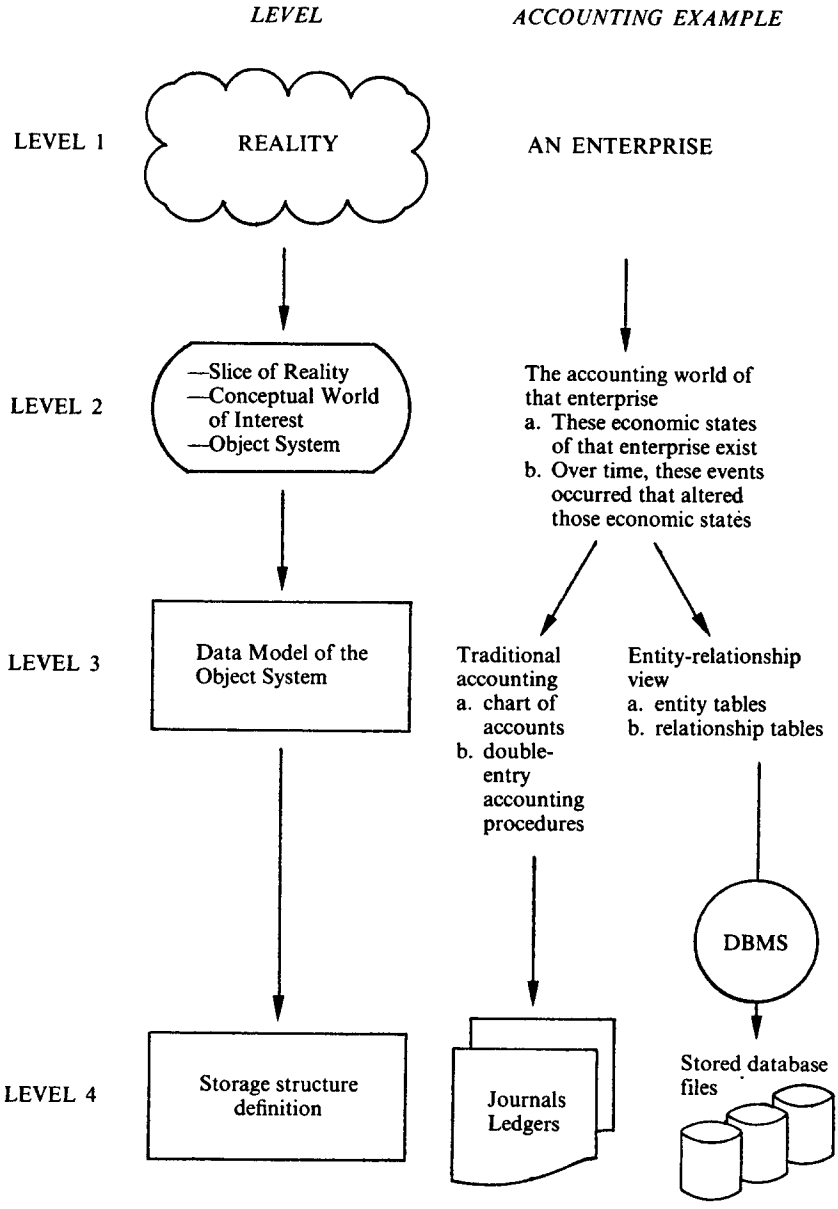
LEVEL 4, the storage structure definition, will not be of interest in the treatment here. However, note that once an object system has been modeled on an entity-relationship basis, it is possible to translate the data model of LEVEL 3 to the storage structure definitions of LEVEL 4, using the definition language and facilities of most existing database management systems (DBMS).

LEVEL 3 of the database abstraction process—development of an accounting data model for an enterprise—will now be treated in detail.

An Accounting Data Model

A chart of accounts and its accompanying double-entry procedures might be viewed simply as a scheme for organizing, classifying, and aggregating financial data. Additionally, however, it represents the imposition upon an accountant of a particular mode of thinking about the economic affairs of an entity. For example, when queried about the "things" that accounting deals with, an accountant might list items such as "prepaid revenues," "retained earnings," or "liabili-

FIGURE 1
DATABASE ABSTRACTION PROCESS



ties" because these, among others, constitute the elements in his/her predefined world of interest. Collectively, these account names and double-entry procedures represent a data model of an enterprise's economic aspects.

This predisposition toward certain types of "things" of interest will be discarded here. Instead, this paper will view an object financial system without "traditional-accounting colored" glasses and use the following steps to construct an

accounting data model [Chen, 1976]:

1. Identify (a) the *entity* sets such as classes of objects, agents, and events that exist in the conceptual world and (b) the *relationship* sets that connect those entities;
2. Construct an *E*ntity-*R*elationship (E-R) diagram that will exhibit the semantic nature of 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; and
4. Organize the results of steps 1, 2, and 3 into entity/relationship tables and identify a key (unique characteristic) for each entity/relationship set.

As each of these steps is considered in the following sections, its application to accounting will be made more concrete by using a small retail enterprise as an example.

Identification of Entity/Relationship Sets

The process of viewing an object system and identifying its relevant entities and relationships cannot be described exactly, because it will normally include a wide variety of systems analysis techniques (such as those detailed by Taggart and Tharp [1977]). The particular list of entities and relationships that any one person produces might differ quite legitimately from another person's list, depending upon their differing backgrounds and perceived uses of information. A guiding principle here for accountants is that in enumerating entities they limit themselves to real phenomena that can be distinctly identified and remain clear of accounting "artifacts." Everest and Weber [1977, p. 356] supply direction in this process by warning designers to avoid the use of "naming tree" entities (that is, accounts used for presentation or accumulation purposes only).

For a small retail enterprise, the entity sets (groups of objects, agents, events) of interest in the accounting object system might include those 16 items shown in Figure 2 (a). For simplicity, only this limited list will be used in the rest of the discussion here. A modeling of a real enterprise would probably produce a larger and more exhaustive group of entities, including such additional objects as buildings, such additional agents as federal and state governments, and such additional events as equipment disposals and purchase orders.

Once the entity sets have been identified, system design proceeds to specification of sets of relevant relationships that may exist among them. A partial list (a more complete diagram is given later) for the retail enterprise is shown in Figure 2 (b).

The bases for explicitly recognizing 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 the rationale for connecting some of the *Event* entities with each other. With the exception of the "order" event which is not, traditionally speaking, an accounting event at all, the *Event* (- - -) *Event* links represent explicit manifestations of Ijiri's [1975, Ch. 5] causal double-entry conventions; 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" [1964, p. 450].

Again, there is no claim to absolute "truth" in the entity/relationship sets given in Figure 2. They are not intended to be exhaustive enumerations, only illustrations of Chen's initial design stage.

FIGURE 2

Objects		*Agents*
Equipment		Stockholder
Inventory		Employee
Cash		Customer
		Vendor
	Events	
Order	Cash Receipt	Capital Transaction
Sale	Cash Disbursement	General and Administrative Service
Purchase	Equipment Acquisition	Personnel Service

(a) Entity sets in an accounting data model

Event (- - - -) *Event*
Sale (fills) Order
Cash Receipt (payment for) Sale
Cash Disbursement (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 relationship sets (not complete listing)

Construction of an Entity-Relationship Diagram

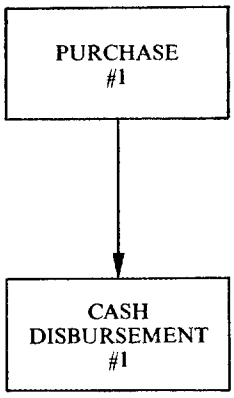
After identification of appropriate entity and relationship sets in an object system is completed, data modeling continues with a further analysis of how these concepts fit together. The semantic nature (or real world character) of a relationship is examined at this stage and is portrayed through the use of Entity-Relationship (E-R) diagrams. Again for simplicity, this paper will illustrate only a portion of the real world possibilities, in this case binary relationships. Such relationships concern associations between just two entity sets and have consequently three basic types: (1) one-to-one, (2) one-to-many, and (3) many-to-many. Each of these cases is illustrated below.

A one-to-one (1-to-1) relationship specifies a correspondence between a pair of entities, one from each of the two connected entity sets. Suppose, for example, that the small retail enterprise had an operational rule that all purchases were to be paid for, in full, exactly five days after receipt. Each purchase event would correspond then to only one cash disbursement event as exhibited in Figure 3 (a). Chen's E-R diagram would depict this 1-to-1 relationship in the manner shown in Figure 3 (b).

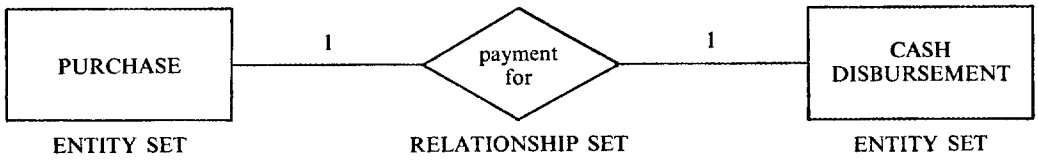
A *one-to-many* (1-to-*n*) relationship specifies a correspondence between just a single entity in one entity set and many entities in another entity set. Using the example again, suppose that the enterprise billed its customers once a month for all sales during the preceding month and that the customers pay in full shortly thereafter. In this case, each cash receipt event would correspond to multiple sale events as seen in Figure 4 (a) and (b).

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

FIGURE 3
A ONE-TO-ONE CORRESPONDENCE



(a) Two related event sets

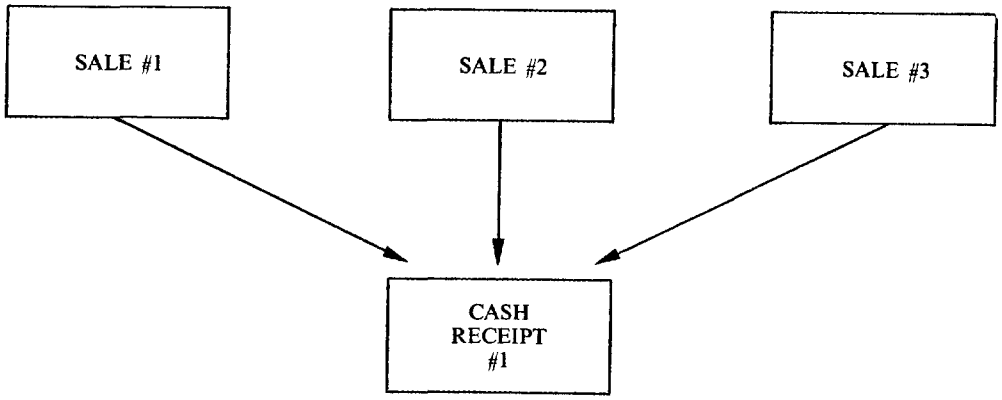


(b) E-R diagram

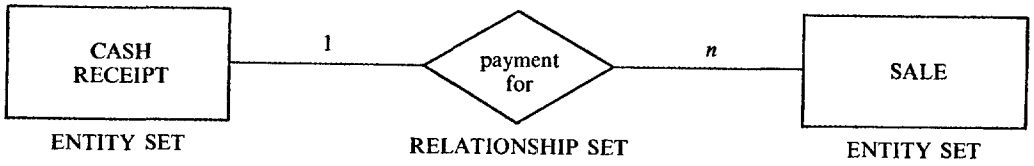
but also a possible correspondence between one entity in the second set and many entities in the first set. To illustrate in the case of a relationship between the *SALE* and *INVENTORY* entity sets from the retail example, suppose not only that each sale consist of many products, but also that each product participates in many sales. This bidirectional mapping is shown in Figure 5 (a) and (b).

Extending this same kind of analysis, an E-R diagram can be specified for the entire retail enterprise as shown in Figure 6. In this particular example, the use of a limited list of entities and binary relationships has produced a relatively simple and well integrated E-R diagram; however, more realistic (and hence, more complicated) situations can be represented in a similar fashion. For instance, if the additional event of equipment disposal were included for the enterprise, it would be possible to model the acquisition of an asset for cash and a trade-in by showing a three-way association among the entities equipment acquisition, equipment disposal, and cash disbursement.

FIGURE 4
A ONE-TO-MANY CORRESPONDENCE



(a) Two related sets of events



(b) E-R diagram

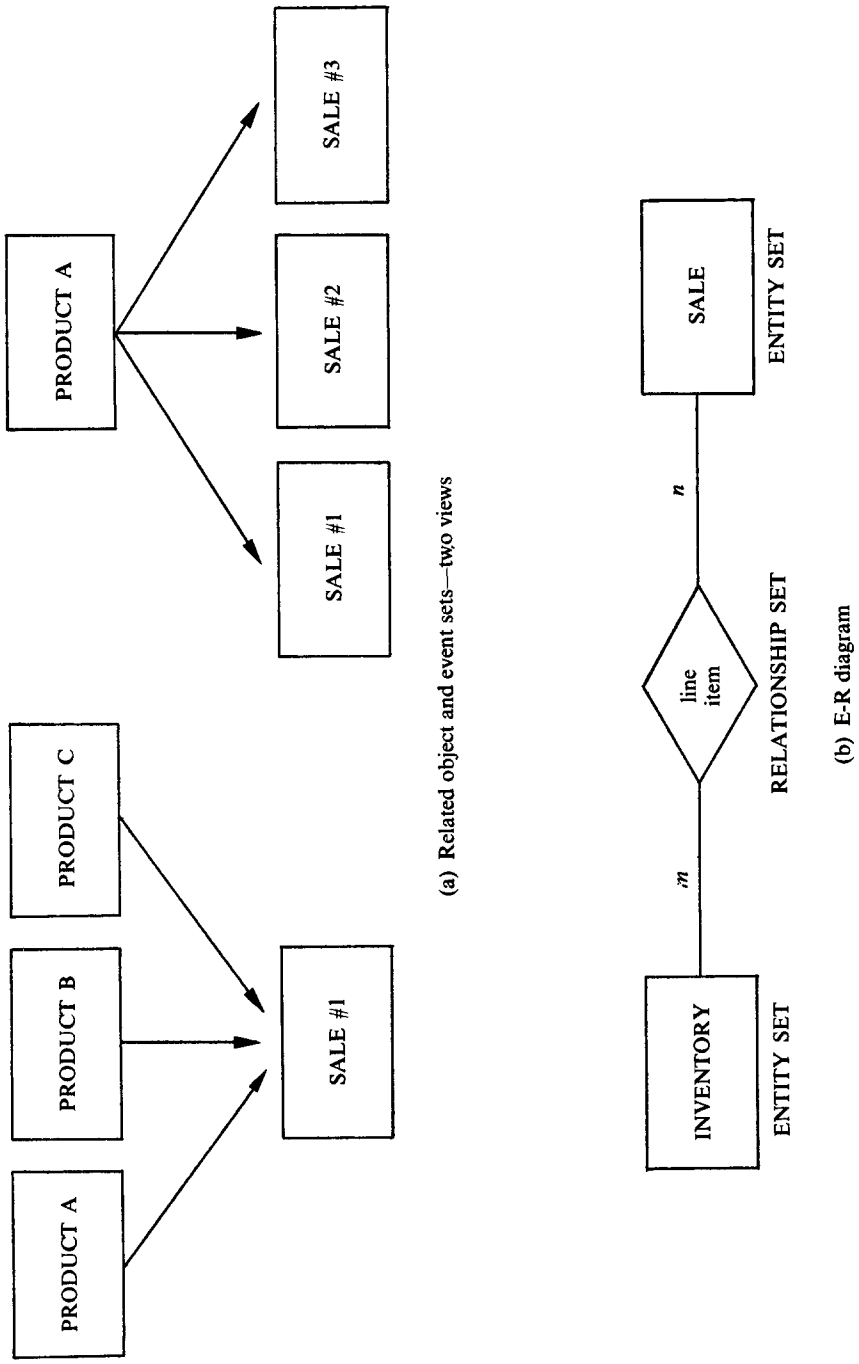
Identification of Characteristic Mappings

Once an E-R diagram has been constructed for the object system, the entity-relationship modeling process moves to a more detailed level of system design by starting to identify the relevant properties or characteristics of each entity/relationship set. Similar to the process of determining the entities and relationships themselves, this identification procedure can be described only in very general terms. The final listing of characteristics will be heavily dependent upon the designer's perception of the ultimate decision environment and information need. A framework proposed by Mock [1976, p. 88] addresses the needs of this design phase. His outline is based upon accepted principles of measurement theory and lists criteria certain to help in reducing the unstructured nature of this process.

Specifying all of the characteristics for each entity/relationship set given in Figure 6 would be a prohibitively long process. Therefore, this section will be limited to mappings for two entity sets—

FIGURE 5

A MANY-TO-MANY CORRESPONDENCE



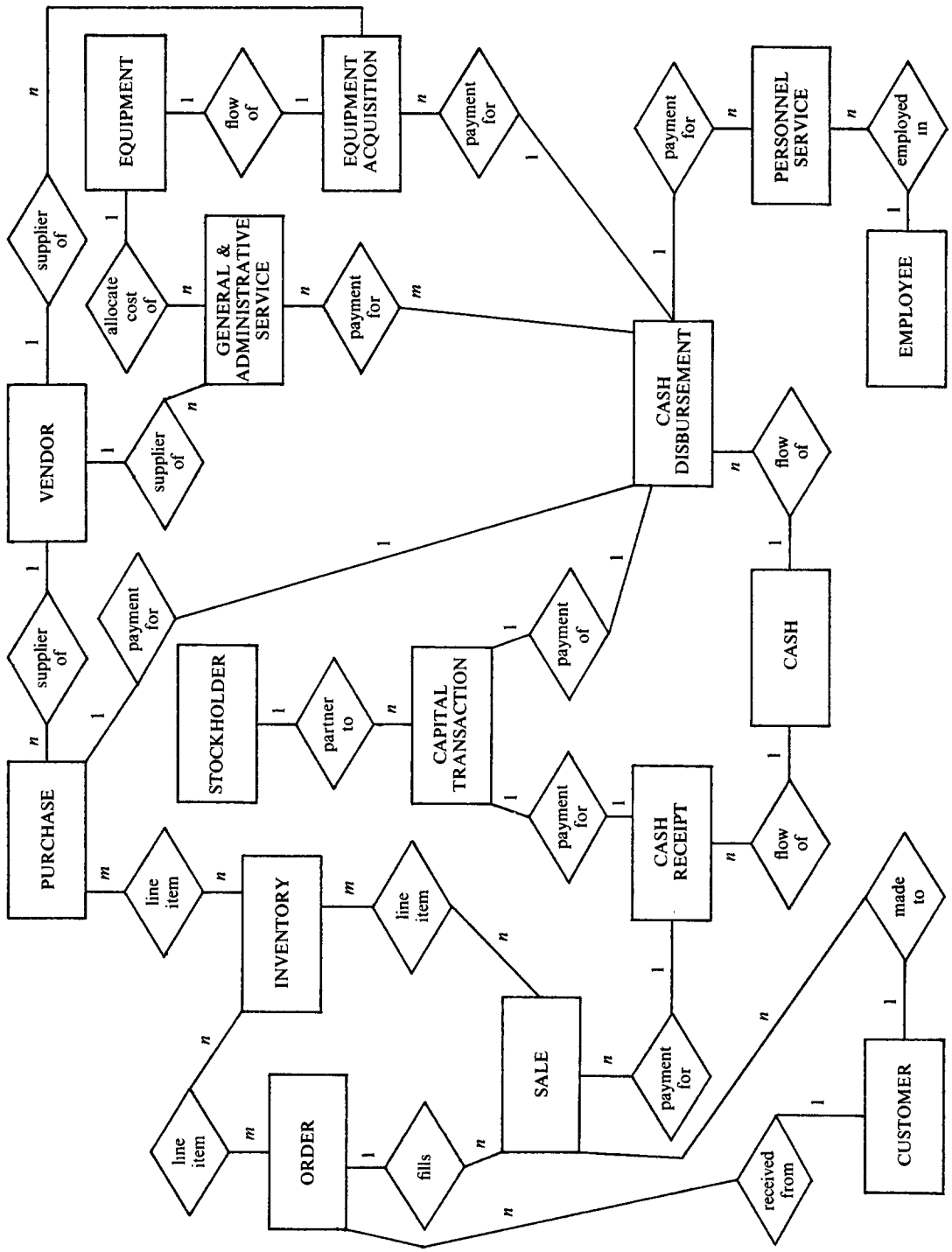
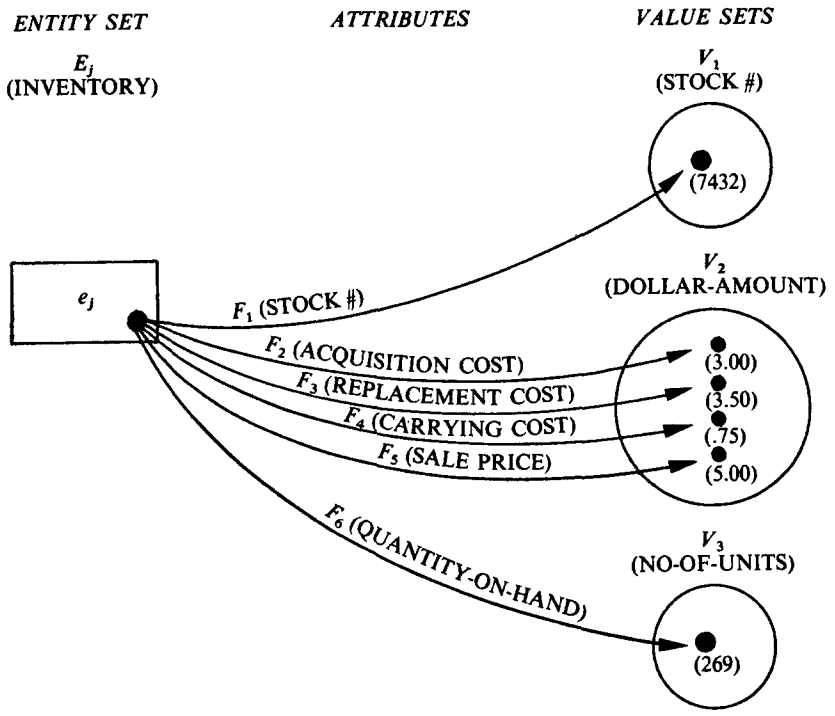


FIGURE 6
E-R DIAGRAM FOR THE ENTIRE RETAIL ENTERPRISE

FIGURE 7
CHARACTERISTIC MAPPINGS DEFINED ON ENTITY SET *INVENTORY*



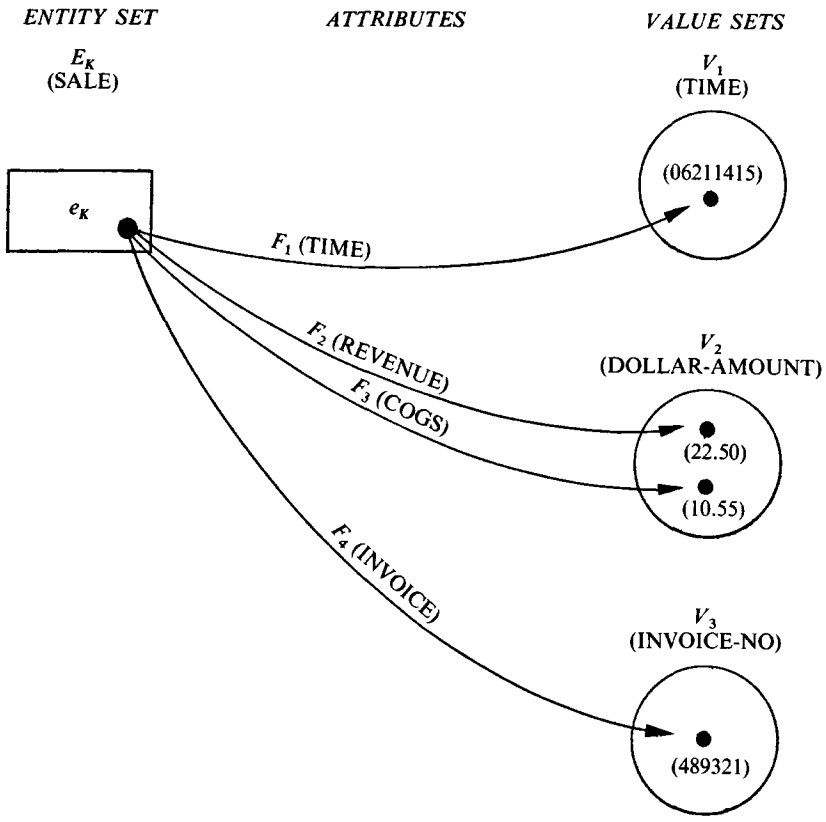
SALE and *INVENTORY*—and the relationship set that connects them—*SALE line item*. Additional mappings for the other sets, and possibly for these three sets as well, would have to be accomplished before the data model would be completely specified.

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," "acquisition cost/$3.00," and "quantity-on-hand/269."

The first item in each of the pairs above—the *attribute*—is formally defined by Chen [1976, p. 12] "as a function which maps from an entity set or relationship set into a value set." This conceptualization of an attribute as a function is important because it allows the E-R model constructs to develop in a form free of certain unwanted properties such as addition, deletion, and update anomalies [Date, 1977, Ch. 9].

The second item listed in each pair—the *value*—is taken from different *value sets* such as stock numbers, dollar-amounts, and number-of-units. These value sets are analogous to the relational database concept of "domain" described by Everest and Weber [1977] and, in a more limited sense, to the measurement concept of "numerical relational system" described by Mock [1976, Ch. 2]. Indeed, it can be seen that the entire characteristic identification process used in entity-relationship modeling fits in well with Mock's [1976, p. 107] call for em-

FIGURE 8
CHARACTERISTIC MAPPINGS DEFINED ON ENTITY SET *SALE*



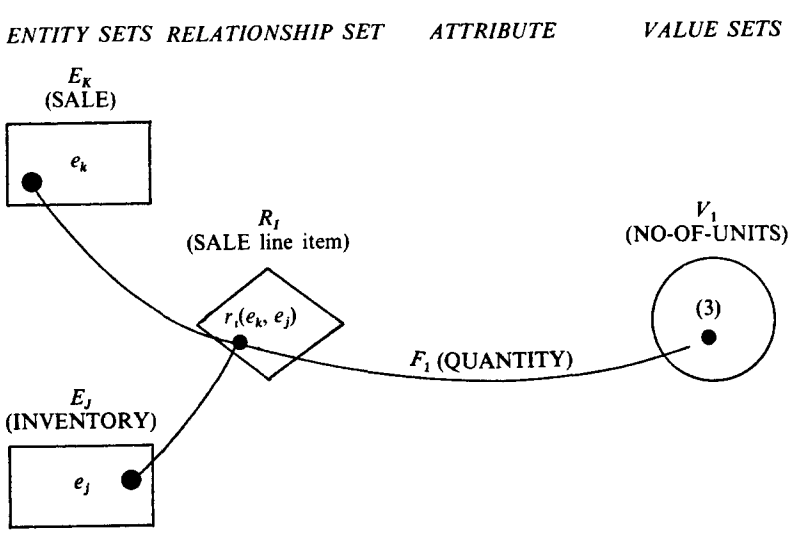
phasis on the measurement aspects of data to be used in accounting systems. For those characteristics whose values are numerical, Chen's attribute mappings and value sets correspond exactly to Mock's homomorphic mappings and numerical relational systems.

Figures 7 and 8 illustrate some of the characteristic mappings for the entity sets *INVENTORY* and *SALE*. The first function (F_1-STOCK#) in Figure 7 maps a particular element (e_J) in the set *INVENTORY* (that is, one particular product) to its stock number (7432). Likewise, the next four functions (F_2, F_3, F_4, F_5) all map that same element to its various unit costs and unit price. F_6 simply couples a product to its value for quantity-on-hand.

In Figure 8, the first function maps a particular sale event (e_K) that happened on June 21 at 2:15 p.m. to a coded representation of its time occurrence, while F_2, F_3, and F_4 determine other relevant characteristics. The actual mappings for entities in a real enterprise would be obtained via observation or measurement.

The attribute mappings for the relationship set *SALE line item* are given in Figure 9. Although this particular example illustrates a case where the relationship itself possesses a characteristic that cannot be fully identified with either of

FIGURE 9
CHARACTERISTIC MAPPINGS DEFINED ON RELATIONSHIP SET *SALE line item*



the participating entities, it is not necessary for all relationships to have properties of their own. In some cases (one of which will be seen later), the relationship simply specifies an association and possesses no further information content.

Before leaving this section, readers should note the close correspondence between the model as developed so far and the ideas of "events" accounting theorists such as Sorter [1969], Johnson [1970], and Colantoni, Manes, and Whinston [1971]. According to Sorter, "the purpose of accounting is to provide information about relevant economic events that might be useful in a variety of possible decision models" [1969, p. 13]. Even with the limited presentation given to this point, it can be seen that an entity-relationship model accommodates well the things implied by Sorter's statement, such as multidimensional measures (dollars and time), different valuation bases (acquisition cost and replacement cost), and aspects of accounting entities useful in management science modeling (unit carrying cost for economic order quantity models).

Organization of Data Into Entity/Relationship Tables

The final step in the enterprise data modeling process involves (1) designation of primary keys and (2) organization of the constructs identified above (entity sets, relationship sets, value sets, and attributes) into entity/relationship tables.

A primary key (PK) for an entity set (or a relationship set) is simply an identifying characteristic that maps one-to-one with the elements of that entity set, and thus is able to represent those elements in the database. To do this, the identifying characteristic must be both universal—every entity must have it as an attribute—and unique—each entity's value for that characteristic must be different. An example of a PK would be the student number in a university database or the Social Security number in the Internal Revenue Service database. The primary key is not always a single charac-

FIGURE 10

	← PRIMARY KEY →			
ATTRIBUTE	TIME	REVENUE	COGS	INVOICE
VALUE SET	TIME	DOLLAR-AMOUNT		INVOICE-NO
ENTITIES (one each row)	06211415	22.50	10.55	489321
	06211418	150.00	90.00	489322
	.	.	.	.
	.	.	.	.
	.	.	.	.

(a) *SALE* entity table

	← PRIMARY KEY →					
ATTRIBUTE	STOCK #	ACQUISITION COST	REPLACEMENT COST	CARRYING COST	SALE PRICE	QOH
VALUE SET	STOCK #	DOLLAR-AMOUNT				NO-OF-UNITS
ENTITIES (one each row)	7432	3.00	3.50	.75	5.00	269
	8519	.30	.32	.15	1.00	85
	6784	.05	.10	.06	.50	62
	.	.	.	.	.	.
	.	.	.	.	.	.
	.	.	.	.	.	.

(b) *INVENTORY* entity table

teristic; sometimes it is necessary to concatenate several attributes to identify something uniquely. Such is the case for all relationship sets in Chen's model, because he defines their PKs as the combined PKs of their involved entities.

The organization of data into *entity tables* is illustrated in Figure 10. As can be seen, the two tables correspond closely to the mappings shown in Figures 7 and 8. The only real difference is that now a primary key value is allowed to represent each element of the two entity sets, "STOCK#" for the *INVENTORY* entities and "TIME" for the *SALE* entities. This assumes that the company assigns each of its products a different stock number and that each sale event is mapped to a unique time identifier. (Methods for keying events in this manner are discussed in Lamport [1978].)

A relationship table for *SALE line item* is illustrated in Figure 11. Again, the table closely resembles the mappings derived earlier (Figure 9) except that a primary key value, the concatenated PKs of the involved entities, now represents an instance of the relationship. Chen's illustration of relationships in this manner resembles the use of case-grammar models by Roussopoulos and Mylopoulos [1975, p. 147], because it requires specification of "who plays the *roles* (or fills the *cases*)" associated with each relationship. In a complete data model, an entity could fill a number of widely varying roles depending upon the nature of the relationships it participated in.

Figure 12 illustrates two additional

FIGURE 11

RELATIONSHIP TABLE FOR *SALE line item*

	← PRIMARY KEY →			
ENTITY TABLE NAME	SALE	INVENTORY		
ROLE	flow	stock		
ENTITY ATTRIBUTE	TIME	STOCK #	QUANTITY	RELATIONSHIP ATTRIBUTE
VALUE SET	TIME	STOCK #	NO-OF-UNITS	
RELATIONSHIPS (one each row)	06211415	7432	3	
	06211415	8519	4	
	06211415	6784	7	
	06211418	7432	30	
	.	.	.	
	.	.	.	
	.	.	.	

FIGURE 12

	← PRIMARY KEY →				
ENTITY TABLE NAME	PURCHASE	INVENTORY			
ROLE	flow	stock			
ENTITY ATTRIBUTE	TIME	STOCK #	QUANTITY	PRICE	RELATIONSHIP ATTRIBUTE
VALUE SET	TIME	STOCK #	NO-OF-UNITS	DOLLAR-AMOUNT	
	.	.	.	.	
	.	.	.	.	
	.	.	.	.	

(a) RELATIONSHIP TABLE FOR *PURCHASE line item*

	← PRIMARY KEY →	
ENTITY TABLE NAME	SALE	CASH-RECEIPT
ROLE	decrement	increment
ENTITY ATTRIBUTE	TIME	TIME
VALUE SET	TIME	TIME
	.	.
	.	.
	.	.

(b) RELATIONSHIP TABLE FOR *SALE payment*

relationship tables to be used in later examples. The mappings for these two relationship sets were not shown previously, but the procedures involved would be identical to those used in obtaining the table for *SALE line item* (Figure 11). The table for the set *SALE payment* (Figure 12 (b)) illustrates two additional points. First, it portrays a relationship set which simply specifies a connection without any attributes of its own. Second, its role names are designated in accordance with the causal double-entry reasoning underlying the relationship itself. According to Ijiri [1975, Ch. 5], each occurrence of such a relationship requires specification of the increment and decrement involved. Additionally, the role names in this table would be used to qualify the attributes because the attribute names ("TIME" and "TIME") are not unique.

A total modeling effort for the retail enterprise would involve constructing a table for every set depicted in Figure 6. After these specifications were finished, the entity-relationship view of the enterprise's accounting system would be complete, and the data model would be ready for use with actual database systems. Discussion of the translation process involved in mapping E-R constructs into the specification language of a particular DBMS is beyond the scope of this paper. However, interested readers may consult Chen [1975, pp. 25–34] who describes such processing for relational and network systems.

Reconciliation of Traditional Accounting with the New Data Model

The data model developed to this point for a small retail enterprise certainly differs radically from the traditional accounting paradigm generated by using a chart of accounts and double-entry bookkeeping. The question now arises, "Can such a model serve the everyday needs of accountants?" The new system is missing certain familiar and financially important items such as accounts-receivable, and its lack of a debit-credit framework might leave some with the uneasy feeling that all comings and goings are not being properly accounted for.

This section will demonstrate that an information system modeled on an entity-relationship basis can indeed accommodate the conventions of traditional accounting. To show this, part of the data model outlined in Figure 6 will be used, specifically two entity sets—*SALE* and *INVENTORY*—and three relationship sets— *SALE line item*, *PURCHASE line item*, and *SALE payment*. Tables representing each of these five sets were presented earlier in Figures 10, 11, and 12.

In the paragraphs below, two types of accounting procedures for the entity-relationship (E-R) model will be described: (1) the derivation of accounts-receivable and (2) the maintenance of perpetual inventory and cost of goods sold. These descriptions will concentrate primarily on the *set* aspects of the E-R constructs and will be very general in nature. In actual practice, the calculations discussed would be performed using the processing language of a particular database system. Readers interested in more specific explanations may consult McCarthy [1978] where a number of detailed accounting computations are done with the relational language SEQUEL [Chamberlin *et al.*, 1976].

At any time during its operations, the total of outstanding receivables for the example enterprise modeled in Figure 6 is equal to the revenue of sale events not yet paid for. This set of sale events is not present explicitly in the system, but it can be obtained easily by subtracting the set of paid-for sales (the elements in the

relationship set *SALE payment*) from the set of all sales. After such a set difference operation is performed, accounts-receivable can be calculated by summing the revenue characteristic for all elements of the identified subset.

The relationships between object and event entities were characterized in the E-R modeling process as stock-flow interactions; that is, there existed a stock of some entity such as cash or equipment whose level was affected by flows of events such as cash receipts or equipment acquisitions. Yu [1976, p. 242] discusses the importance to financial reporting of properly accounting for these stock-flow interplays. In traditional systems, their connection is directly accounted for by an entry to the asset (object) account upon the occurrence of a transaction (event), but in an entity-relationship model the interaction must be effected in another way—by defining certain updates to the stock entities that will be invoked upon the occurrence of flow entities. In the case of perpetually maintained inventory for the retail enterprise, these invoked procedures would take place in the following manner. When a purchase occurs in the object system (LEVEL 2 of Figure 1), the data model (LEVEL 3) would have new elements added to the sets *PURCHASE* and *PURCHASE line item*. These additions would then "trigger" the following updates to characteristics of the appropriate element in the set *INVENTORY:*

(1) A new acquisition cost would be set (assuming that the cost scheme is one, such as the moving weighted average method, which can be updated on a perpetual basis);
(2) A new replacement cost would be set (equal to the latest transaction price); and
(3) A new quantity on hand would be calculated.

Mechanisms for implementing these triggered updates are described by Eswaran [1976]. The cost of goods sold characteristic of *SALE* elements would then be calculated at the time of sale using the perpetually maintained acquisition cost of the appropriate inventory entities.

The inventory and cost of goods sold figures calculated above are components of a perfectly consistent database where all elements of the data model (LEVEL 3 in Figure 1) immediately and accurately mirror changes in the object system (LEVEL 2). There are times, however, when certain information is used only at specified intervals, in which case the characteristics involved can be produced by calculation and aggregation only when needed. In the retail accounting model, this stepwise production of information would apply to accounts-receivable and to periodic inventory and cost of goods sold. The process of choosing among the various temporal alternatives in database maintenance depends not so much upon the nature of the characteristic itself but upon its expected decision use. The bases for making such choices involve the issue of "conclusion materialization" and are discussed extensively by Bubenko [1976]. In all cases, the final choice of a consistency level should be "subject to the tradeoff between applications requirements (benefit) and economic feasibility (cost)" [ANSI/X3/SPARC, 1975, p. II-1].

This paper has now considered the fundamental features of Chen's entity-relationship methodology as it applies to the construction and maintenance of an accounting model. Before moving on to a summary, there remains one matter to be considered in the next section: comparison of the E-R accounting model with the relational accounting models of Everest and Weber.

Comparison with Relational Model

It was mentioned previously that an entity-relationship modeling process could be used as an initial step in actual implementation of many database systems, among them the relational model of Codd [1970]. In fact, the final products of an E-R design process—the entity/relationship tables—are nearly identical in both appearance and mathematical form to Codd relations, and one can see on inspection that the mapping of a Chen model to a relational model is relatively straightforward. There are, however, some important features that differentiate the accounting system developed in this paper from the relational accounting systems developed by Everest and Weber [1977]. These differences stem from design considerations and are explained below.

The relational decomposition or transformation approach [Codd, 1972] to data modeling used by Everest and Weber starts with an arbitrary collection of relations (based upon a chart of accounts, for example) and proceeds, using the notion of functional dependencies, to recast them in a form free of certain undesirable properties such as addition, deletion, and update anomalies [Date, 1977]. In contrast, Chen's approach emphasizes the initial identification of entities and relationships and then proceeds by designating mappings that identify both relevant associations among entities and relevant characteristics of entities and relationships. Use in this paper of the second approach has led to an accounting model superior in two respects.

First, the E-R data model "fits" the object system better. This occurs because its design process begins with *a priori* identification of relevant aspects of the conceptual world rather than with use of a framework based on existing accounting methods. The effect on the Everest and Weber paradigm of using traditional accounting classification schemes is to introduce into the final (fully normalized) data model certain extraneous concepts such as "naming tree" relations and debit/credit account numbers while masking or omitting more essential features such as stock-flow interactions and causal double-entry. The two authors recognized that the use of accounting "artifacts" would cause such problems and suggested possible solutions. However, the use of entity-relationship modeling prevents the problems from occurring at all.

Second, the entity-relationship approach displays more clearly the organization of the object system. It concludes with a data model where each entity/relationship set is represented in a separate table (relation). Chen [1976, pp. 25–29] argues that such an arrangement leads to clearer semantic meaning and accommodates more easily changes in the nature of object system relationships. He also contends that his model clarifies better the nature of dependencies among data, because it differentiates between those mappings used to identify characteristics and those used to identify associations. Chen's modeling process also produces a convenient shorthand—the E-R diagram—that provides a clear overview of the object system and distinguishes explicitly to users the different 1-to-1, 1-to-*n*, and *m-to-n* associations.

Finally in the matter of comparing the two approaches, the issue of multivalued dependencies and fourth normal form arises. Recent research (which postdated Everest and Weber's article) by Fagin [1977b] has identified deficiencies in the relational decomposition process and

proposed an augmented approach to database design that uses a new semantic object called a "multivalued dependency" [1977b, p. 433]. The final product of this new design process is a "fourth normal form" [Fagin, 1977a] that eliminates from a data model further undesirable aspects of maintenance behavior. The question now is "How do Fagin's new concepts relate to entity-relationship modeling?," and the answer is, "They have already been accounted for." Multivalued dependencies (which are 1-to-*n* mappings) are identified clearly by Chen. More importantly, the storage problems caused by multivalued dependencies arise when a data model attempts to portray two independent relationships in one table [Fagin 1977a, p. 272]. As shown earlier, such a representation would not materialize in entity-relationship modeling.

In summary, the data model developed in this paper represents the accounting universe better than those developed by Everest and Weber. Its development is not encumbered with accounting artifacts, its concepts portray clearly the nature of the accounting object system, and its final form displays none of the undesirable anomalies identified in the normalization literature.

Summary of Model Concepts and Advantages

The presentation of an accounting information system based on entity and relationship sets is now complete. The steps necessary to model an object system have been outlined, and specific illustrations of each step for an example enterprise have been given. The paper will conclude by summarizing features of the entity-relationship approach relevant to accounting practice and research.

First, Chen's methodology represents one of the first attempts to construct an enterprise view of data [Chen, 1977], that is, a data model viewed from the perspective of the whole enterprise rather than from the perspective of individual users or particular database management systems. As such, it can be considered a "top-down" approach to system design and a definite aid to logical thought in a database environment. Accountants need a tool like this. With the use of database systems becoming more widespread, they need to be able to view many enterprise schemata in a clear, consistent, and somewhat standardized manner. In this sense, E-R diagrams are analogous to flowcharts; they represent visual aids to be used in understanding the full significance of the data processing operations.

Second, the entity-relationship approach to accounting-database models provides a design framework that can be integrated into any of the database models previously proposed in the literature such as those of Colantoni, Manes, and Whinston [1971], Lieberman and Whinston [1975], and Haseman and Whinston [1976; 1977]. It is especially appropriate for Codd's [1970] relational model, because the transfer from Chen's entity-relationship tables to Codd relations is straightforward. Use of an E-R approach also avoids the drawbacks of decomposition.

Third, the entity-relationship model provides a specific vehicle for the incorporation of measurement concepts and causal double-entry into a single-entry database system. Proponents of these concepts [Mock, 1976 and Ijiri, 1975] have presented them for use in accounting systems, but their ideas do not fit well with journals and ledgers. As seen earlier in the paper, they do adapt well to an E-R model.

Fourth, entity-relationship modeling provides a sound theoretical basis for the expansion of the "events" approach to

accounting [Sorter, 1969]. Consideration of a wide range of decision models using multidimensional measures, such as those advocated in two recent AAA reports [1969; 1971], is accomplished easily. Aggregation problems are reduced also because the information in an E-R based system can be stored in disaggregated form and summarized only according to the intent of a particular user.

On the whole, therefore, this paper has shown that the entity-relationship approach to accounting models represents a logical extension to work in the area of accounting information systems. It provides an overview of much of the work done in database systems, and it also offers practical aid to accountants who plan to do work in a database environment.

REFERENCES

American Accounting Association, "Report of Committee on Managerial Decision Models," THE ACCOUNTING REVIEW (Supplement 1969), pp. 43–76.

———, "Report of the Committee on Non-Financial Measures of Effectiveness," THE ACCOUNTING REVIEW (Supplement 1971), pp. 164–211.

ANSI/X3/SPARC Study Group on Data Base Management Systems, "Interim Report," *ACM-SIGMOD FDT* (February 1975).

Bubenko, J., "The Temporal Dimension in Information Modeling," Research Report RC 6187 (IBM Research Laboratories, Yorktown Heights, NY, November, 1976).

Chamberlin, D. D., M. M. Astrahan, K. P. Eswaran, P. P. Griffiths, R. A. Lorie, J. W. Mehl, P. Reisner, and B. W. Wade, "SEQUEL 2: A Unified Approach to Data Definition, Manipulation, and Control," *IBM JOURNAL OF RESEARCH AND DEVELOPMENT* (November 1976), pp. 560–75.

Chen, P. P., "The Entity-Relationship Model—Toward a Unified View of Data," *ACM TRANSACTIONS ON DATABASE SYSTEMS* (March 1976), pp. 9–36.

———, "The Entity-Relationship Model: A Basis for the Enterprise View of Data, "*Proceedings of the National Computer Conference*, 1977 (AFIPS, 1977), pp. 77–84.

CODASYL Programming Language Committee, *Data Base Task Group Report* (Association for Computing Machinery, 1971).

Codd, E. F., "A Relational Model of Data for Large Shared Data Banks," *COMMUNICATIONS OF THE ACM* (June 1970), pp. 377–87.

———, "Further Normalization of the Data Base Relational Model," in R. Rustin, ed., *Data Base Systems* (Prentice-Hall, 1972), pp. 33–64.

Colantoni, C. S., R. P. Manes, and A. B. Whinston, "A Unified Approach to the Theory of Accounting and Information Systems," THE ACCOUNTING REVIEW (January 1971), pp. 90–102.

Date, C. J., *An Introduction to Database Systems*, 2nd ed. (Addison-Wesley, 1977).

Eswaran, K. P., "Specifications, Implementations and Interactions of a Trigger Subsystem in an Integrated Database System," Research Report RJ 1820 (IBM Research Laboratories, San Jose, CA, August, 1976).

Everest, G. C., and R. Weber, "A Relational Approach to Accounting Models," THE ACCOUNTING REVIEW (April 1977), pp. 340–59.

Fagin, R. (1977a), "Multivalued Dependencies and a New Normal Form for Relational Databases," *ACM TRANSACTIONS ON DATABASE SYSTEMS* (September 1977), pp. 262–78.

———, (1977b), "The Decomposition Versus the Synthetic Approach to Relational Database Design," *Proceedings of the Third International Conference on Very Large Data Bases*, 1977 (ACM, 1977), pp. 441–46.

Haseman, W. D., and A. B. Whinston, "Design of a Multidimensional Accounting System," THE ACCOUNTING REVIEW (January 1976), pp. 65–79.

———, *Introduction to Data Management* (Richard D. Irwin, 1977).

Ijiri, Y., *Theory of Accounting Measurement* (American Accounting Association, 1975).

Johnson, O., "Toward an 'Events' Theory of Accounting," THE ACCOUNTING REVIEW (October 1970), pp. 641–53.

Lamport, L., "Time, Clocks, and the Ordering of Events in a Distributed System," *COMMUNICATIONS OF THE ACM* (July 1978), pp. 558–65.
Lieberman, A. Z., and A. B. Whinston," A Structuring of an Events-Accounting Information System," THE ACCOUNTING REVIEW (April 1975), pp. 246–58.
Mattessich, R., *Accounting and Analytical Methods* (Richard D. Irwin, 1964).
McCarthy, W. E., "A Relational Model For Events-Based Accounting Systems," (doctoral dissertation, University of Massachusetts, 1978).
Mock, T. J., *Measurement and Accounting Information Criteria* (American Accounting Association, 1976).
Roussopoulos, N., and J. Mylopoulos, "Using Sematic Networks for Data Base Management," *Proceedings of the First International Conference on Very Large Data Base*, 1975 (ACM, 1975), pp. 144–72.
Sorter, G. H., "An 'Events' Approach to Basic Accounting Theory," THE ACCOUNTING REVIEW (January 1969), pp. 12–19.
Sundgren, B., "Conceptual Foundation of the Infological Approach to Data Bases," in J. W. Klimbie and K. L. Koffeman, eds., *Data Base Management* (North Holland Publishing Company, 1974), pp. 61–96.
Taggart, W. M., and M. O. Tharp, "A Survey of Information Requirements Analysis Techniques," *COMPUTING SURVEYS* (December 1977), pp. 273–90.
Will, H. J., "Auditing in Systems Perspective," THE ACCOUNTING REVIEW (October 1974), pp. 690–706.
Yu, S. C., *The Structure of Accounting Theory* (The University Presses of Florida, 1976).