**WILLIAM E. MCCARTHY**

*Associate Professor*
*Michigan State University*

William E. McCarthy is an associate professor of accounting at Michigan State University where he teaches classes at all degree levels dealing with computer-based information systems. He has published research papers in both the computer science and accounting literature, and he is currently a member of a number of editorial review boards including those of *The Accounting Review* and *Auditing: A Journal of Practice & Theory*. His current research interests center around knowledge representation issues in both database management and artificial intelligence. He has been the recipient of an excellence in teaching award from Beta Alpha Psi, and he is currently the national chairman of the Information Systems/Management Advisory Services section of the American Accounting Association.

# On the Future of Knowledge-Based Accounting Systems

In recent years, commercial advances in artificial intelligence (AI) in general and in expert systems (ES) in particular have received considerable attention in accounting. Accounting researchers have used ES methods to study a wide range of decision-making behavior in judgement areas such as going-concern opinions [Biggs and Selfridge, 1986], audit materiality assessments [Steinbart, 1984], internal control evaluations [Gal, 1985] [Meservy, 1985], and receivables writeoffs [Dungan and Chandler, 1985]. In each of these cases, the researchers involved built a working expert system with the express goal of capturing computationally the accumulated wisdom of specific accounting experts, usually partners in public accounting firms. Additionally, these public accounting firms themselves have begun to devote a considerable amount of time, money, and manpower resources to assorted AI projects with the hope of benefiting from the tremendous upsurge which has been predicted repeatedly for commercial activity in this arena. For example, projections of 15 year increases in AI revenues to 113 billion and in AI employment to 650 thousand have recently been made.

This paper will explore the implications of AI technology for accounting. However, it will do so in a manner unlike the ES projects above. That is, I intend not to discuss individual decision modeling by itself, but in the context of linking AI-type systems with traditional processing of accounting transaction cycles in a database (DB) environment. This linking of expert systems and database technology has received extensive research attention from computer scientists of late [Brodie, Mylopoulos, and Schmidt, 1984], and it has the potential I believe of extending the scope of accounting work within most companies considerably. Accounting transactions are the infrastructure on which all commercial data processing systems are built. Most knowledge-

based systems will benefit from accounting systems specifically tailored for shared use and inference, and such a possibility is my primary thesis in this presentation.

The rest of this paper is organized as follows. My first major section will discuss the differences between knowledge representation methods in the separate computer science disciplines of database (DB) systems and artificial intelligence (AI) systems. These differences were once significant, but they are eroding rapidly, and I will also suggest here where knowledge-based accounting systems might fit as AI and DB come together. The second major section of the paper will be its longest—an overview of knowledge representation methods. My examples here will be quite simple, and they should equip a non-computer oriented reader with a basic understanding of the issues involved in these newer accounting systems. The paper will end by discussing the implications of the demonstrated new systems and by discussing possible future directions.

## *Accounting Information Systems in the Worlds of Databases and Artificial Intelligence*

In trying to pinpoint where accounting information systems are now and where they might be in the future, I intend to use as reference points the world of database (DB) applications and the world of artificial intelligence (AI) applications (see Figure 1). The difference between these two worlds is admittedly arbitrary [Sowa, 1984], and its definition relies heavily on an equally arbitrary distinction between type and occurrence data. However, I think that readers will find both dichotomies useful even if the definitional edges become somewhat fuzzy at times.

Most database or data processing applications rely on just a few basic information groupings or record types, but their processing involves many, many instances of those

Fig. 1

| TYPES { | MERCHANDISE | STOCK # | DESCRIPTION | PRICE | QOH | COST |
|---|---|---|---|---|---|---|
| INSTANCES { | | 8555 | Cat tree | 5.00 | 86 | 4.85 |
| | | 5510 | Poodle clipper | 6.50 | 512 | 6.00 |
| | | 1187 | Budgerigar perch | 11.00 | 729 | 4.85 |
| | | 2751 | Ordinary mousetrap | .75 | 902 | .06 |
| | | 2752 | Improved mousetrap | 10.50 | 804 | 3.29 |
| | | 8400 | Rat cage | 50.00 | 165 | 11.04 |
| | | 4563 | Pet taxi | 42.00 | 231 | 10.00 |
| | | 8700 | Hampster swing | 26.00 | 231 | 25.00 |

**Database (DB) World**

1. Few types, many instances
2. Emphasizes performance, capacity, & security
3. Prime concern is "computational realism"
4. Examples:
   - Payroll
   - Order Entry
   - General Ledger

**Artificial Intelligence (AI) World**

1. Many types, few instances
2. Emphasizes functionality & flexibility
3. Prime concern is "epistemological adequacy"
4. Examples:
   - Chess
   - New product decision
   - Audit judgement

**CONTRASTING DB AND AI ORIENTATIONS**

structures. Figure 1 portrays the type/instance dichotomy for merchandise data in a hypothetical pet store supply business. The "type" data (also known as the database *intension*) would consist of a record name ("merchandise") and the various field names ("stock#," "QOH," etc.), and these intensional features will most often be defined in the data definition sections of a programming language like COBOL or in the schema definition language of a database management system or a data dictionary. The "instance" data (also known as the database *extension*) in the pet store supply example embraces the eight rows which individually describe the various kinds of merchandise (mousetraps, etc.) this company holds for sale. A name often used in AI research to denote an individual instance of a concept (for example, an individual row in the Figure 1 table) is a *token*.

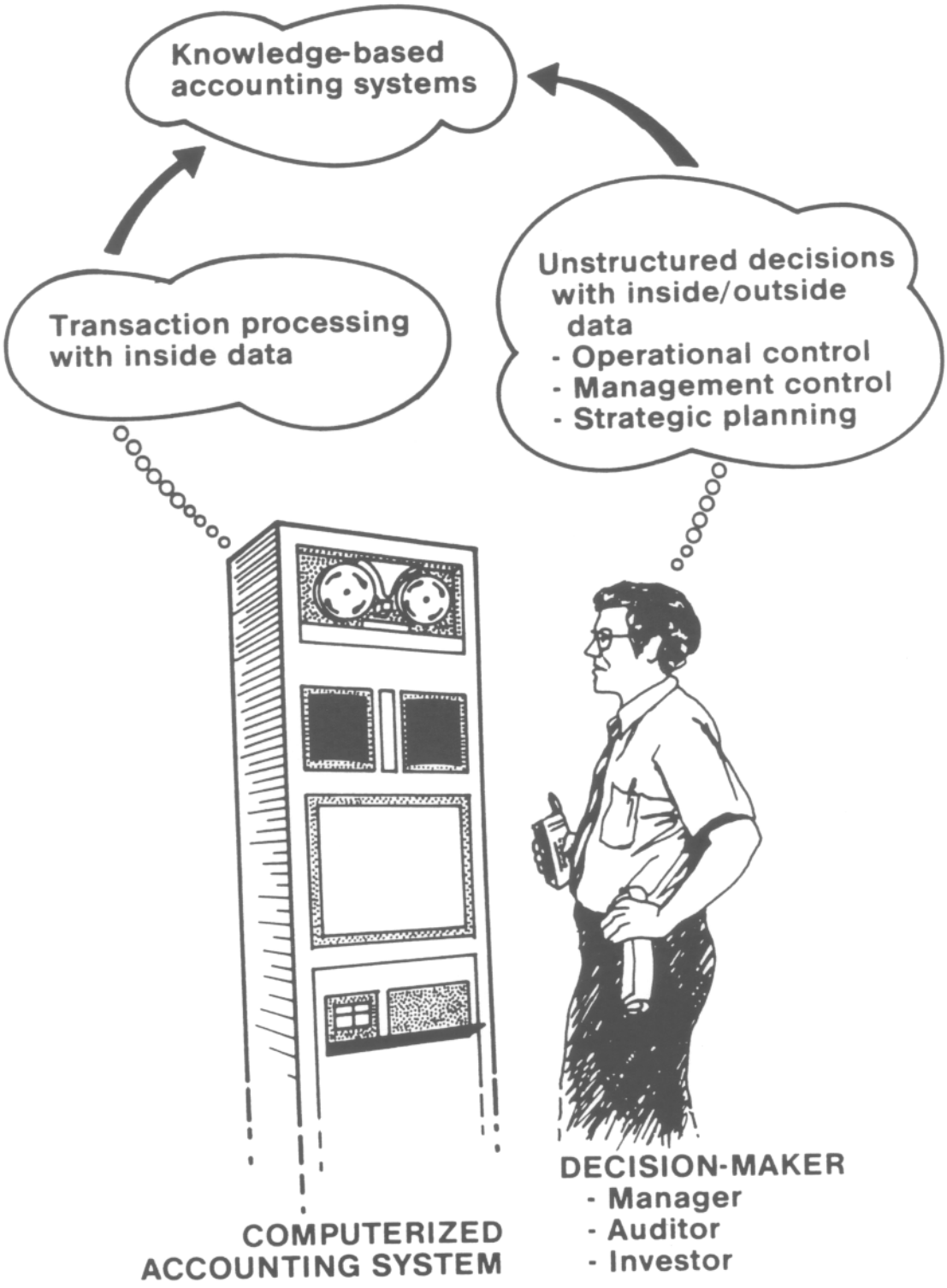
Even with only this toy example as a reference, readers can see that the ratio of types to tokens in the DB world is heavily skewed toward large amounts of homogeneous occurrence data. This high degree of homogeneity allows for extremely efficient processing, hence the DB emphasis on computational realism in its choice of real world problems to attack. Payroll, general ledger, and nearly all other kinds of normal accounting processing arising from transaction cycles also fall into this DB domain. This is why these accounting functions have been so heavily computerized for 20 years or more.

The world of AI applications, on the other hand, deals less with large sets of instance data and more with types. AI type/token ratios are much closer to unity. For example, audit judgement research will usually isolate many concepts to be considered in a final decision, but there will rarely be multiple instances of each concept to be reviewed in a single case. Emphasis in the AI arena of computer modeling swings decidedly toward an ability to represent definitional features of the environment faithfully (epistemological adequacy).

Recent work in expert database systems illustrates the point that many applications will require a mix of these two orientations. In business organizations today, there is certainly a heavy bias toward the DB world, but that bias is tempered by increasing needs for decision-making support in relatively unstructured areas of operational control, management control, and strategic planning. Such decisions were formerly the exclusive property of human managers, auditors, and investors (see Figure 2) who made their choices or plotted their strategies with information gleaned from internal reports and databases along their own understanding of the outside corporate environment. Expert system and decision support research has indicated that at least some of this decision-making structure is amenable to computerized processing or support. However, such computerized support must also be able to access and "interpret" the pool of corporate data arising from cyclic transaction processing, hence the need for movement toward hybrid systems as shown in Figure 2.

For a system to combine the efficiency and capacity of database management with the flexibility and functionality of an AI system, it must be operating from a common definitional base. The basis for that commonality lies in the area of knowledge representation methods (the AI term) and semantic data modeling methods (the DB term). Both of these fields have borrowed heavily from each other (and from other disciplines such as psychology and linguistics), and their concepts will be reviewed without undue emphasis on either in the next section.

Fig. 2




SOURCE: Adapted from Patrick, p. 139.

### *Knowledge Representation Methods*

As illustrated in Figure 3, knowledge-based systems have three basic types of architecture. Readers should realize that the distinctions which I will be making among these three are not absolute in the sense that the types share many common features and in the sense that knowledge in one form is readily translatable to another. There also exists a number of other representation and modeling methods, and my categorizations are not ones with which all computer scientists would agree. Nonetheless, presentation of these three representational formalisms will pinpoint some basic philosophical differences which in turn will give readers an overall appreciation for the definitional nuances of both DB and AI. The three architectures are these:

1. ***Logic***—This is the most theoretically appealing knowledge representation (KR) technique, and it forms the basis for the use of logic programming systems such as PROLOG. Based on the predicate calculus [Dahl, 1983], it has been used extensively in research work combining DB and ES realms (especially in Europe and Japan).

   In logical representation, each atomic fact (such as "Art is the father of Bob") is represented in a *proposition* such as "F(Art,Bob)") wherein the "F" is a *predicate* which relates the *arguments* "Art" and "Bob" via a father relationship. These propositions can be tied together by implication as seen in the second sentence of Figure 3 which states that a father relationship between two people generalizes to a parent relationship and in the third sentence of Figure 3 which states that a parent of a parent relationship implies a grandparent relationship.

2. ***Rules***—Rule-based systems are the dominant form of ES architecture today. This is especially true for microcomputers whose software market is replete with expert system "shells" able to manage the creation and maintenance of rule knowledge bases quite efficiently.

   As seen in Figure 3, knowledge representation in rules takes the form of one or more *antecedent* clauses (such as "IF sex is male") and a *consequent* clause (such as "THEN eligible to run in Boston") which logically follows.

3. ***Semantic Networks***—This category can be subdivided into different types of networks, and these types collectively form the closest ties with traditional database research in the area of data modeling.

   Again as illustrated in Figure 3, semantic networks represent knowledge about some domain in the form of graphs which relate *nodes* (such as "concept A" and "concept B") together with labeled *edges*. Interpretation of the edges determines the type of network, although it is certainly quite common to have different types of networks used together (see for example [Smith and Smith, 1977] and [Winston, 1984]).

In the figures and text which follow, rules and semantic networks will be covered more extensively. My decision to exclude logical systems from more detailed enumeration is not meant to imply that such systems have little promise for accounting. My coverage of rules and networks is keyed more toward easier intuitive understanding and toward easier integration with past modeling work in accounting domains.

Fig. 3

## Knowledge-based systems architecture

### 1. LOGIC

- Art is the father of Bob

F (Art, Bob)

- Art is Bob's parent if Art is the father of Bob

P (Art, Bob): - F (Art, Bob)

- Art is Cap's grandparent if Art is Bob's parent and Bob is Cap's parent

G (Art, Cap): - P (Art, Bob), P (Bob, Cap)

SOURCE: Genesereth and Ginsberg, pp. 934-35.

### 2. RULES

IF Sex is male, and age is less than 40, and Marathon time is less than 3:00:00

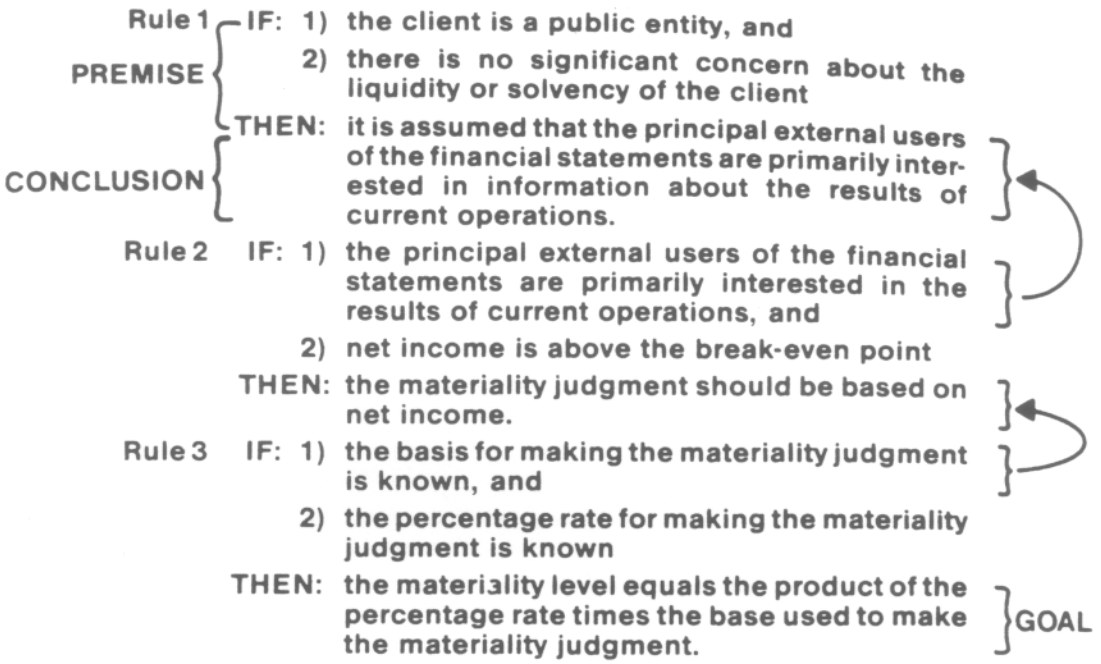
THEN Eligible to run in Boston

### 3. SEMANTIC NETWORK


## KNOWLEDGE STRUCTURES

### *Rule-Based Knowledge Representation*

Figure 4 illustrates a rule-based representation (also called a production system)




## RULE-BASED KNOWLEDGE AND INFERENCE

Fig. 4 **SOURCE: Steinbart, Chap. IV.**

of audit judgement. This subset of rules is taken from an expert system developed by Paul Steinbart [1984, 1987] to capture the expertise of an audit firm partner in the area of materiality judgements (that is, how auditors determine the threshold dollar amounts to investigate empirically during the conduct of an audit). Readers may see how "chunks" of knowledge are represented as a series of IF.....THEN rules where the first part or the antecedent of the rule is called the *premise*, and the second part or the consequence is termed the *conclusion*. When a premise is judged to be true, the rule is "triggered," and its conclusion becomes a fact known to the rest of the production system. The particular production system shown in this figure uses *backward chaining* to mimic expert inference, because it first establishes a "goal" conclusion to work toward and then attempts to prove that goal by having all of its premises established as facts. Premises are established:

(1) by finding other rules which have those premises as conclusions and by then working to trigger those other rules,

(2) by appeal to the user of the expert system in the form of an interactive question (such as an inquiry concerning the control consciousness of a company's CEO which would be a very difficult fact to establish empirically [Gal, 1985]), or

(3) by reference to a database of facts.

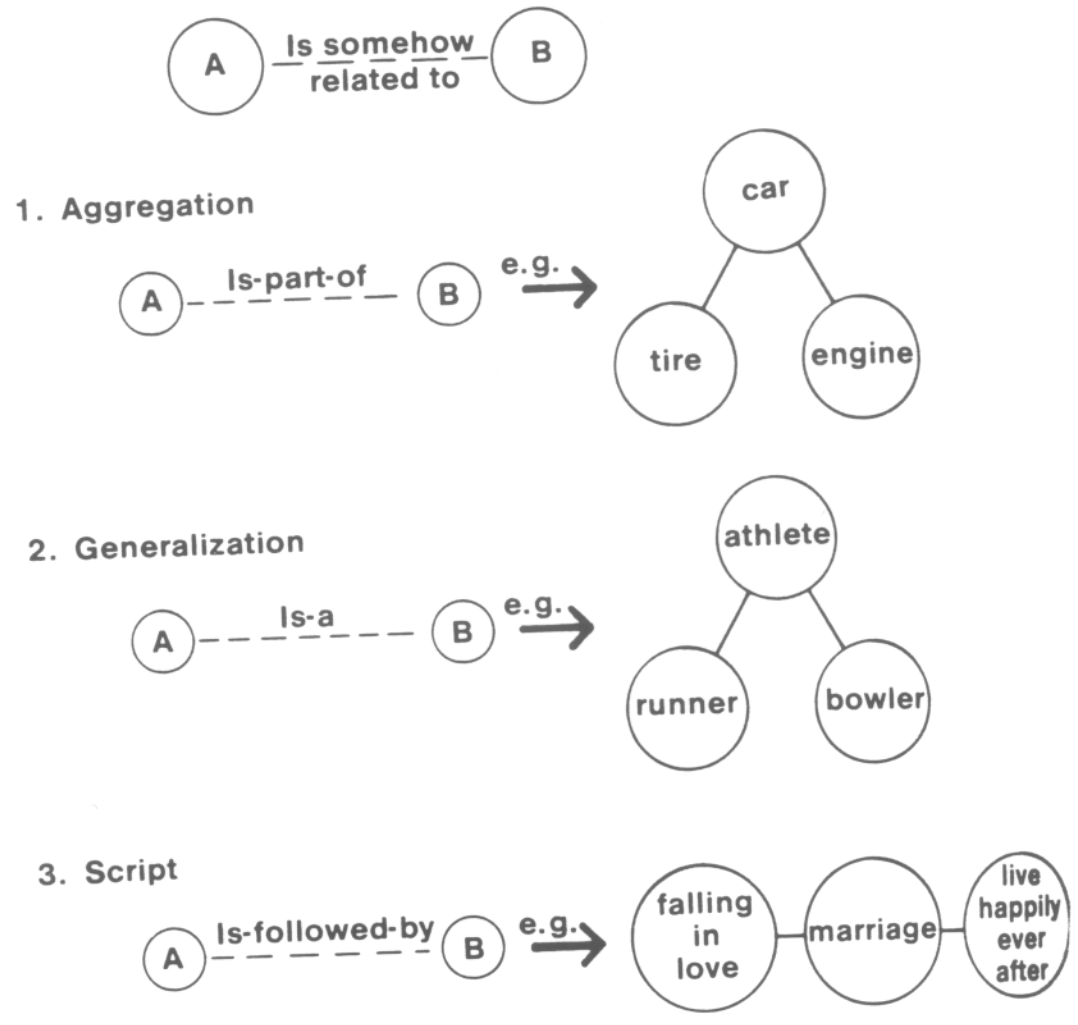
The chaining in the Steinbart example is illustrated with the backwards arrows, and it proceeds simply by finding other rules which can help it to resolve its purpose of triggering the goal rule.

As mentioned previously, production systems are the dominant ES architecture today, especially in accounting research work on audit judgement (see also [Merservy, 1985] and [Dungan and Chandler, 1985]). In all of these implemented cases, the process of knowledge engineering consisted of having experienced audit decision-makers work through a number of possible cases while the researcher attempted to elicit their expertise, capture it in the form of rules, and use it to build a system which could then perform in much the same manner as the expert auditors themselves would.

### *Semantic Network Knowledge Representation*

Figure 5 illustrates the basic idea of semantic nets: there are two concepts (which I have labeled as A and B) and there is some kind of relationship between them. Concepts are represented as nodes in the network, while relationships are portrayed as

**Semantic Networks** Fig. 5



labeled edges or links drawn between the nodes. The three examples of Figure 5 illustrate some common interpretations of semantic nets:

(1) Aggregation ..... A *is part of* B
*example*: An engine is part of a car.

(2) Generalization ..... A *is a* B
*example*: A runner is an athlete.

(3) Script ..... A *is followed by* B
*example*: Falling in love is followed by marriage.

In each of these three cases, part of the impetus involved in the development of these interpreted networks was the desire of some computer scientists to use knowledge representation formalisms which bear strong resemblances to human cognition and belief structures in a particular domain of discourse. For instance, if I were to state "I have my car parked outside this building," readers would logically assume with a high degree of certainty that "my engine" was there as well, because an engine is almost always a component part of a car. In the same vein, less certain statements such as "a bowler is an athlete" or "love and marriage go together" could be modeled with similar formalisms. However, the KR system in this case should allow for some degree of uncertainty or vagueness in the interpretation of the link. Some network representation methods do allow for such an interpretation, but such vagueness is rare in database management systems which adhere predominantly to the concept of "strict typing" [Tsichritzis and Lochovsky, 1982]. Similar uncertainties, incidentally, can be accommodated in both rule-based and logic-based KR systems with methods such as uncertainty factors and fuzzy logic.

Each of the three types of semantic nets listed above is explained in sections and figures which follow. For each network type, I will first use a common sense example and follow it with some accounting interpretations.

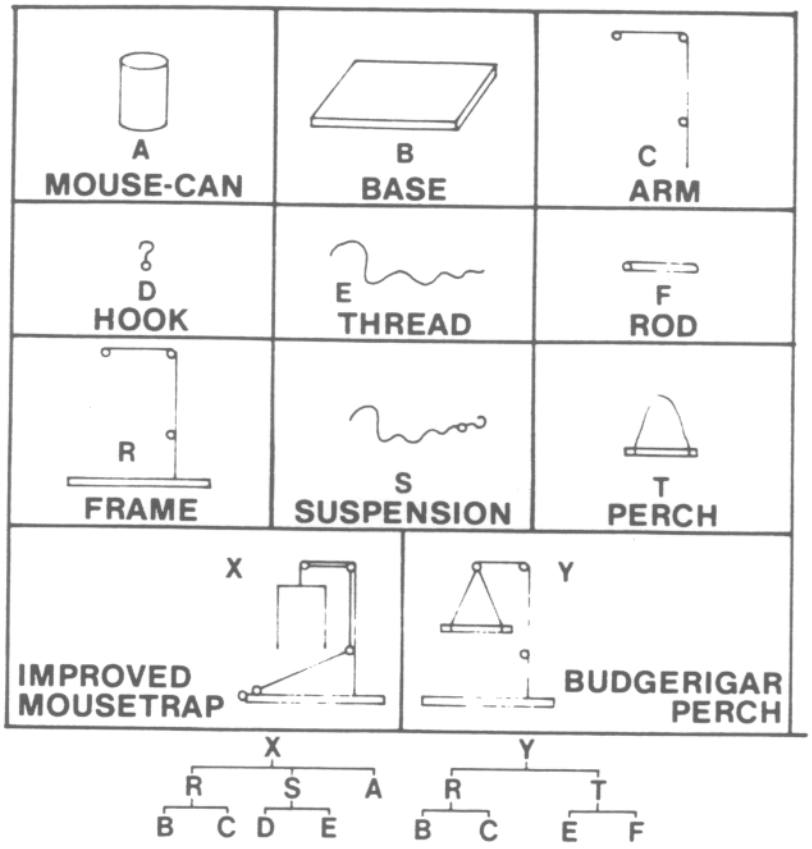
### *Aggregation*

Figure 6(b) portrays a semantic network in which aggregation is designated between the nodes as one goes up the page. In the first part of this figure, the component structures of two of the Figure 1 pet store merchandise items (the "improved mousetrap" and the "budgerigar perch") are shown pictorially, while the bottom half shows the detailed hierarchical representation of just the latter.

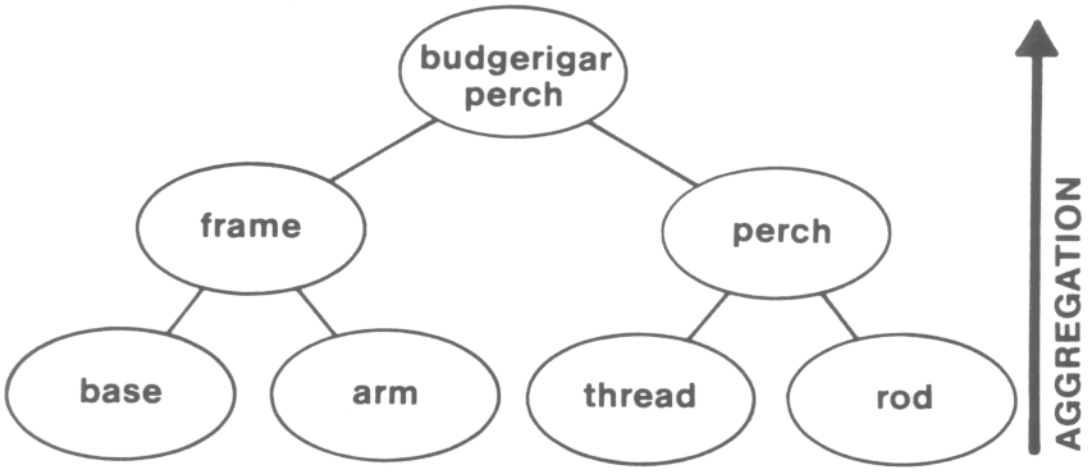
With the pictures, readers can see that a "frame," which itself is a component of two other items, has two parts: (1) a "base" and (2) an "arm." Much like the car and engine example, this semantic network structure illustrates how complex assemblages of ideas can be conceptualized at a higher level of abstraction as just one thing while simultaneous access to lower and more detailed levels is maintained for other purposes.

In Figure 6(b), interpretation of each link would be "is-part of" as one moves up the hierarchy. More commonly in business domains, this aggregation interpretation would refer less to the physical component structure of the merchandise (although such information would still be maintained in a bill of materials) and more to its data

Fig. 6




(a) Two component structures



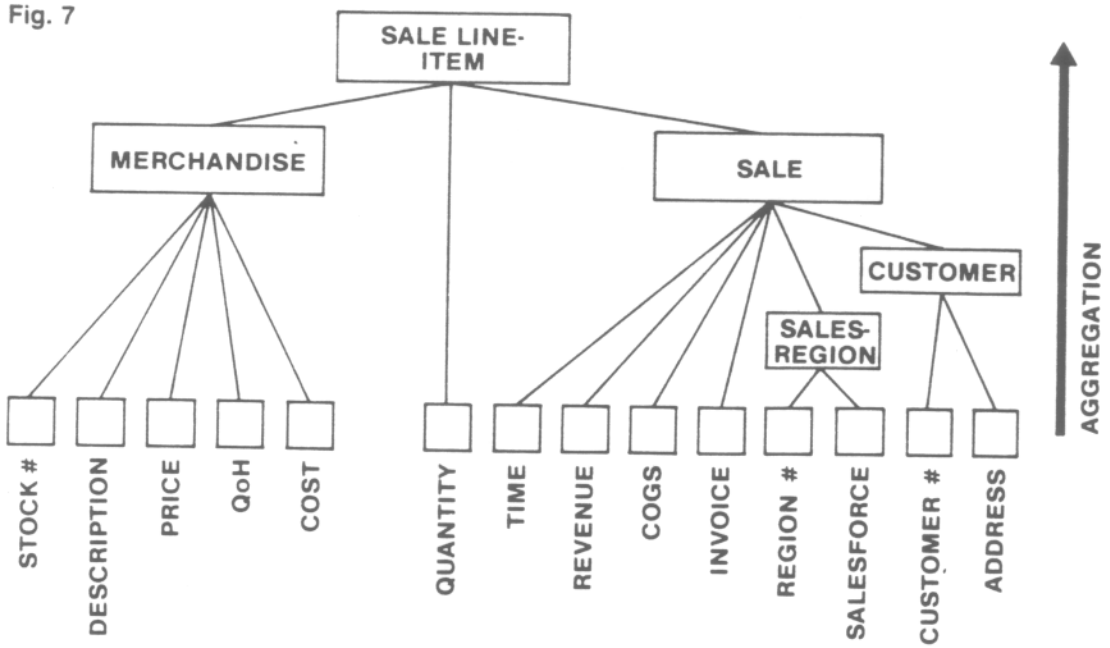

(b) Aggregation hierarchy for budgerigar perch

**AGGREGATION HIERARCHY**

**SOURCE: Adapted from Howe, p. 138**

properties and to its relationships with other concepts. Thus we would find that the primary processing of merchandise information in a company would concern itself with characteristics like the price and cost of various inventory items. Figure 7 illustrates an aggregation hierarchy [Smith and Smith, 1977] for merchandise data items. Link interpretation would still be "is-part of," although "has property of" and "is-related-to" might be more intuitively descriptive. Higher data objects in the hierarchy, like "sale" and "customer," are abstractions which allow easier handling of the more detailed lower items. In most traditional data processing applications, which actually are driven by little or no "modeling" orientation, data aggregation like that illustrated in Figure 7 is accomplished with COBOL-type *records* and *fields*.

Fig. 7

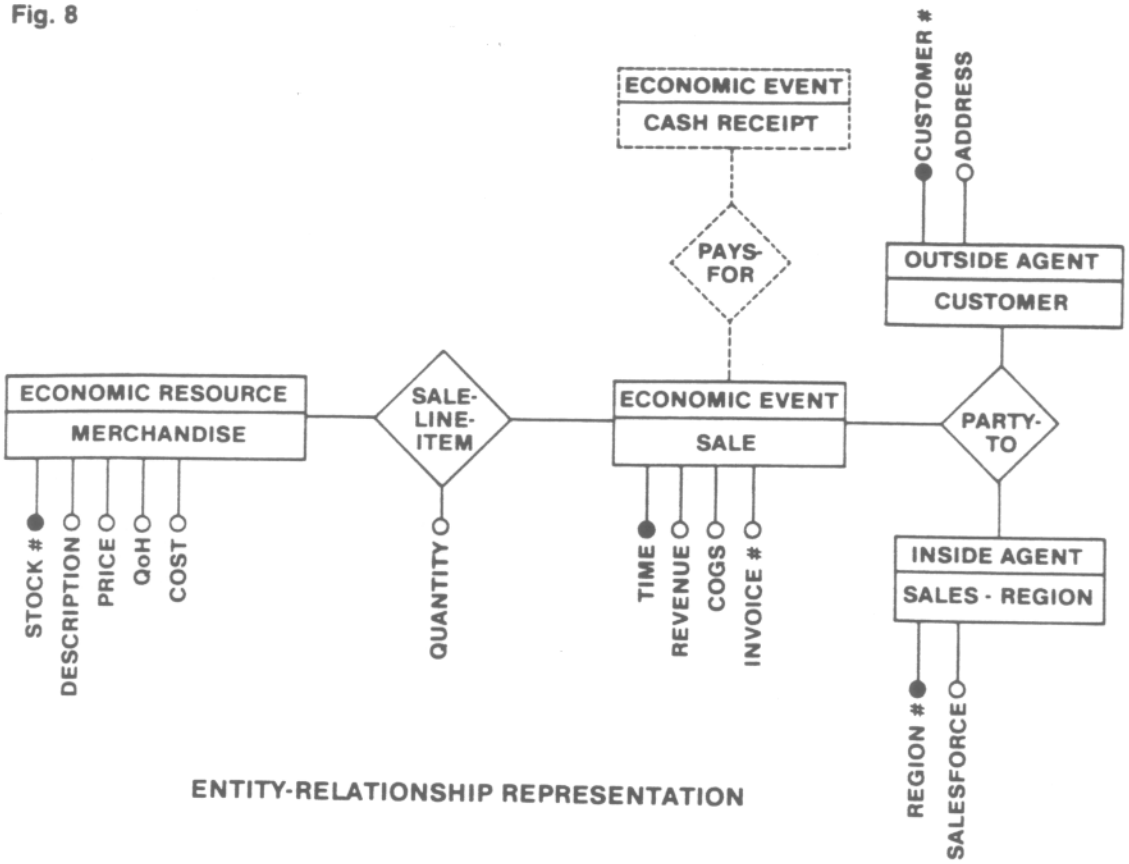



**AGGREGATION HIERARCHY**

Figure 8 takes the same example as Figure 7 (along with additional data shown as dashed figures) and recasts it into another formalism for data aggregation: the widely-used and easily understood entity-relationship data model [Chen, 1976] [McCarthy, 1979]. This type of semantic network does not represent data aggregation as a hierarchy of objects being related as components of other objects, but as a collection of entities (boxes), relationships (diamonds), and attributes (circles). Each entity box in this example has also been labeled in accord with the REA prototype of a typical accounting transaction cycle [McCarthy, 1982]. This REA approach to accounting data modeling has been successfully used in a number of implementations specifically tailored for the type of hybrid transaction-processing/decision-support-system environments shown in Figure 2 [Gal and McCarthy, 1986a], [Denna and McCarthy, 1986].

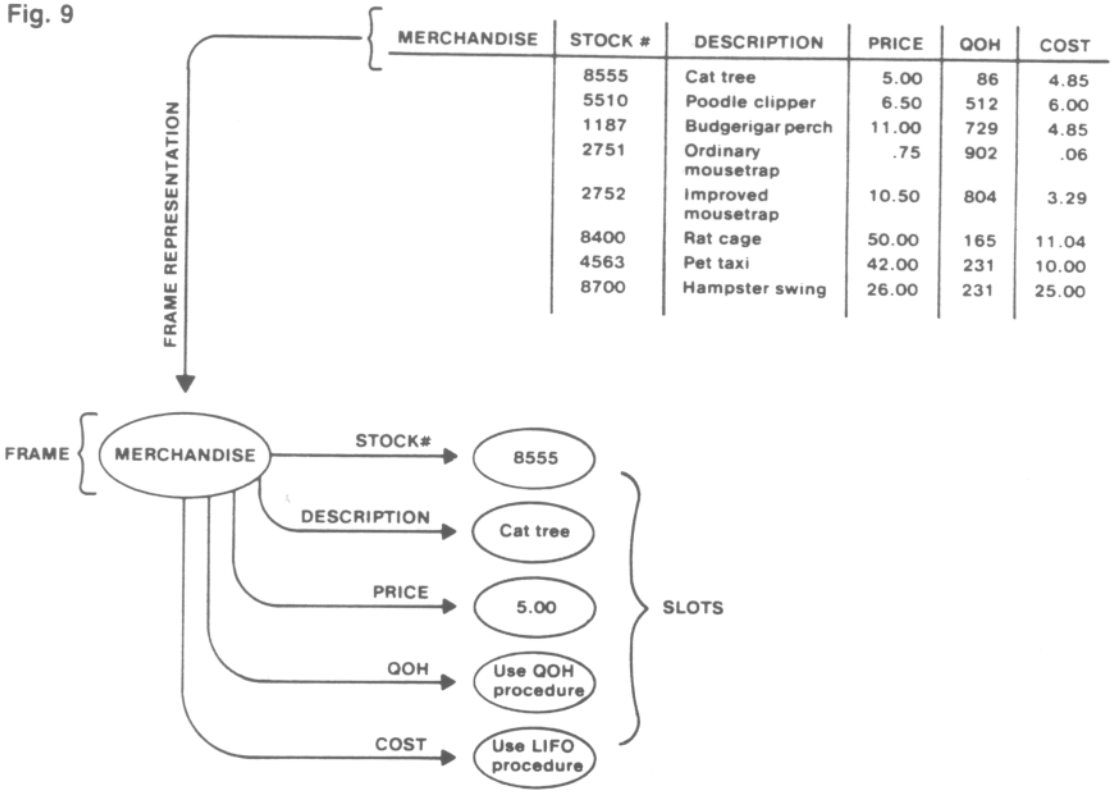
Figure 9 depicts how aggregation is commonly handled in the AI world: as *frames* [Minsky, 1975]. Frames are knowledge structures which represent prototypical constellations of elements or attributes. Figure 9 shows what a frame reconceptualization of the merchandise part of Figures 7 & 8 would look like. The central concept itself is the frame name, and its typical parts or properties are called "slots." The

Fig. 8




ENTITY-RELATIONSHIP REPRESENTATION

Fig. 9




| MERCHANDISE | STOCK # | DESCRIPTION | PRICE | QOH | COST |
|---|---|---|---|---|---|
| | 8555 | Cat tree | 5.00 | 86 | 4.85 |
| | 5510 | Poodle clipper | 6.50 | 512 | 6.00 |
| | 1187 | Budgerigar perch | 11.00 | 729 | 4.85 |
| | 2751 | Ordinary mousetrap | .75 | 902 | .06 |
| | 2752 | Improved mousetrap | 10.50 | 804 | 3.29 |
| | 8400 | Rat cage | 50.00 | 165 | 11.04 |
| | 4563 | Pet taxi | 42.00 | 231 | 10.00 |
| | 8700 | Hampster swing | 26.00 | 231 | 25.00 |

FRAME REPRESENTATION

individual property tokens (like "8555" and "cat tree") are said to *fill* the slots, and as readers can see, there is generally no requirement that slot-fillers be declarative (for example, last-in-first-out periodic inventory costs could be virtual and materialized only when needed). As a matter of fact, it seems that nearly any kind of computable object, including procedures and assertions, can be placed there. Procedures, such as those shown for the last two slots of Figure 9, which are invoked when a certain behavior pattern (such as an insertion of new data or a data query) is recognized in a system, are called *triggers* or *demons*.

As readers may have noticed in this brief treatment of aggregation, different symbolic interpretations of the same real world phenomena are almost inevitable in any knowledge-based system. Such variability has been the concern of researchers (such as Smith and Smith [1978], and Sowa [1984] among others) who have studied the problem of *conceptual relativity*. Using terms already described in the paper, the problem of conceptual relativity can be restated as follows:

- One person's entity is another person's relationship is another person's attribute,
- One person's procedure is another person's declaration is another person's constraint,
- One person's type is another person's token,
- etc., etc.

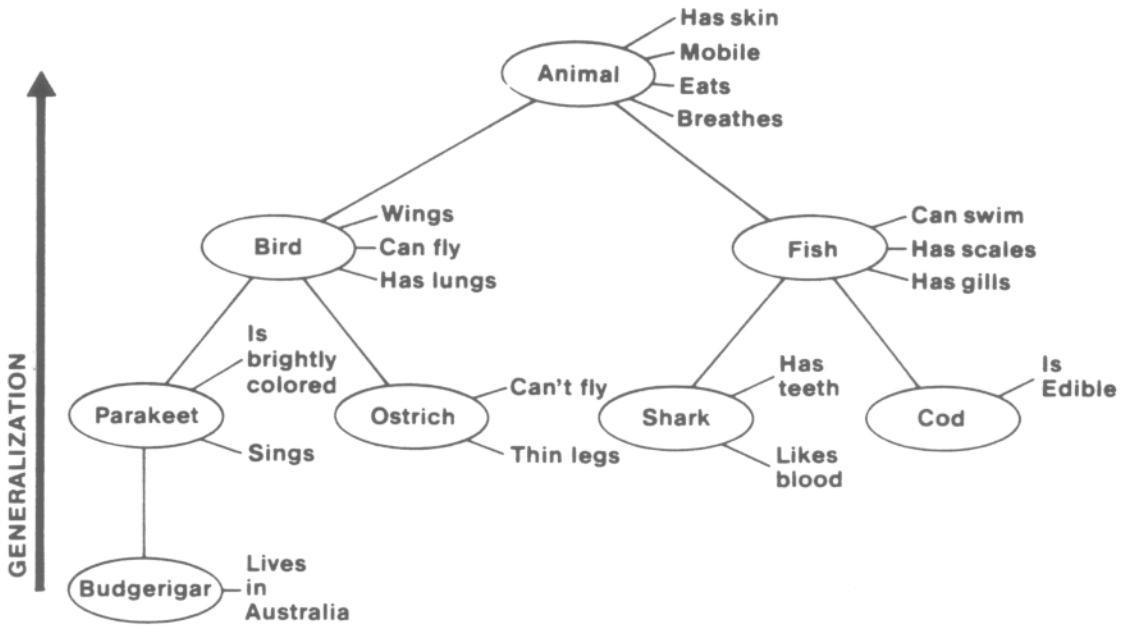
The paradox of fixed modeling primitives is that they facilitate individual thinking while simultaneously making the integration of different individuals' conceptualizations difficult. However, the facilitated thinking is worth the risk, and I believe that anybody who approaches this problem dogmatically can be rightly accused of pedanticism and overly-scholastic thinking (besides which, they will probably fail in their system building efforts). It is perfectly justifiable to use entity-relationship modeling, for instance, even if one cannot give an absolute rule for deciding what *is* or *is not* an entity. Likewise, it makes little sense to agonize indefinitely over a representation choice such as modeling inventory quantity-on-hand as a fixed data value or as a procedure which materializes that value when needed. Such choices involve relative criteria, and they just cannot ever be characterized as absolutely right or wrong. My advice to accounting workers in this area (advice, incidentally, which applies to all types of representation choices) is to use the concepts rigorously first, but with a strong degree of flexibility when integration with the work of others is contemplated. In an excellent discussion on the issue of conceptual relativity, John Sowa summarizes this modeling viewpoint when he says that "Concepts are useful fictions that are not absolute" [1984, p. 350].

My discussion of the methods of Smith and Smith [1977], Chen [1976], and Minsky [1975] completes this paper's review of *aggregation* which is clearly the dominant type of semantic network used in database work. The next two types of networks I will address involve concepts whose intellectual roots deal more with the AI problem domains of property inheritance and natural language processing: *generalizations* and *scripts*.

## *Generalization*

Figure 10 portrays a network in which nodes going down the page represent increasing specialization of the concept at the top, while concepts at the bottom generalize upward. I have also cast each node in this hierarchy as a frame with an aggregation of properties, and there is a strong notion of property inheritance for the lower level nodes. Thus, readers can see that a shark "has teeth" and "likes blood," but they should also note that a shark "is-a" fish which in turn "is-a" animal, so it also possesses characteristics like gills and skin.

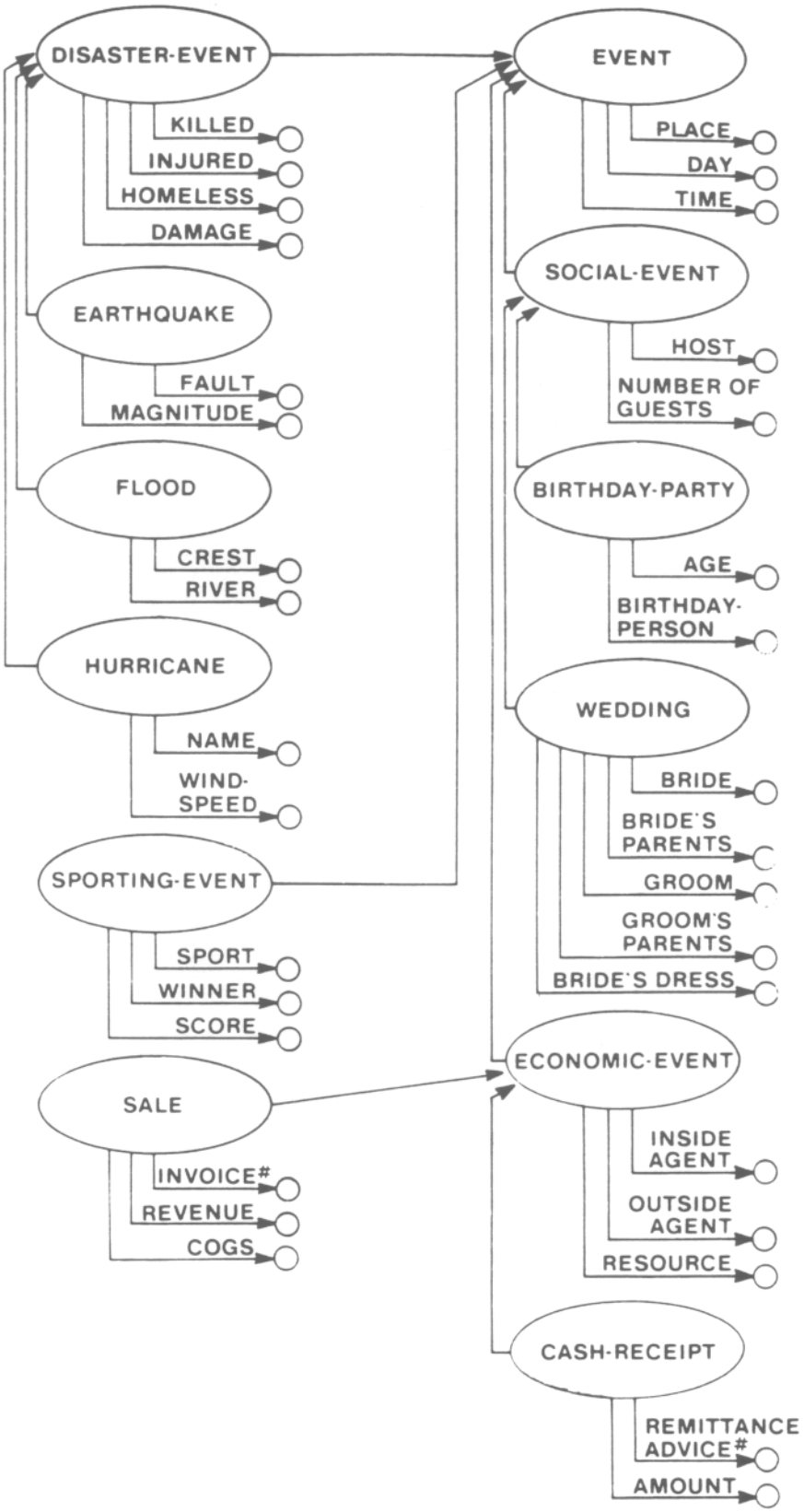
The generalization and the "is-a" link interpretation of Figure 10 denote type-type relationships, but token-type generalizations could be specified as well. For example, I could have added a node with "Ollie" under the "Ostrich" branch of this tree and a "Sammy" under the "Shark" branch to denote two individual animals of those classes. Readers should be aware, however, that some AI and DB researchers would use different names and different link interpretations for those two different kinds of generalization abstraction. Token-type generalization is sometimes called "classification," and the links in type-type networks are sometimes labeled "AKO" (a-kind-of).




**GENERALIZATION (ISA) HIERARCHY**

**Fig. 10** **SOURCE: Adapted from Brachman, p.31.**

Figure 11 illustrates another frame-oriented generalization example [Winston, 1984], this one taken from natural language processing research work designed to interpret newspaper stories. If readers have followed the aggregation and generalization examples given earlier, they could see that an instantiation of a concept would involve filling slots for that individual frame and for all of its ancestor frames up the hierarchy. Thus, an "earthquake" story would need in addition to other attributes




**FRAME-ORIENTED AGGREGATION AND GENERALIZATION**

Fig. 11 **SOURCE: Adapted from Winston, p. 267.**

information on the quake's "magnitude" on the Richter scale, its "damage" estimates, and its approximate "time."

In the lower part of Figure 11, I have included some accounting frames which illustrate in a different manner the REA [McCarthy, 1982] template structure for the "sale" event first shown in Figures 7 & 8. Slots to be filled for this example would include "invoice#," "revenue," "cogs," and "time," the last of these coming from two hierarchical levels away. The "inside agent," the "outside agent," and the "resource" involved in the exchange event would also be needed, and in the case of the "sale" transaction, these would be filled by appropriate sales-region, customer, and merchandise tokens.

Figure 11 is adapted from a section of Winston's artificial intelligence book [1984, chapter 8] entitled "Digesting News Seems to Involve Frame Retrieving and Slot Filling" wherein he discusses the problems involved in representing symbolically the commonsense knowledge people use in performing the intelligent albeit simple task of understanding daily news events. In terms of the split between the machine domain of processing and the human domain of processing that we first saw in Figure 2, this act of "digesting news" would be very similar to the human information processing which would take place when a manager, auditor, or investor makes some kind of unstructured decision using accounting transaction data. Such human-computer linking illustrates the motivation for development of knowledge-based accounting systems. Systems which will mimic the expertise of accounting experts (like managers) will have to have access to the commonsense knowledge structures of transaction data (like its aggregation components or its generalization categories).

### *Scripts*

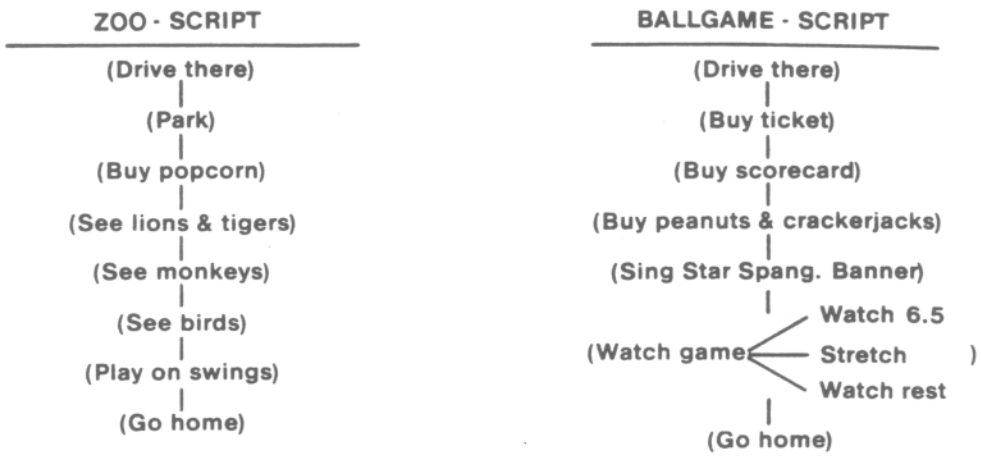
If people of various ages and backgrounds were to separately read sentence (1) at the top of Figure 12, there might certainly be some different interpretations of Graham's activities (did he go to a zoo or did he go to a ballgame?). To at least the sports-oriented reader of sentences (2) and (3) however, it would be fairly obvious what Paul and Howard did last night, because "Cubs" would seem to refer to the baseball team rather than to small animals and there are no "expos" at a zoo. Additionally, readers might be able to infer from experience other non-enunciated information about these activities such as the fact that Paul's night game didn't take place in Chicago (because the ballpark has no lights) or the fact that Howard's game might have included the singing of "O Canada" (because the Expos are a Montreal team).

Whatever ambiguity was present with regard to Graham's activities would probably dissipate for most readers when sentence (1) was followed with either sentence (1a) or sentence (1b), because this later information would certainly evoke the correct "script" from the reader's mind. This would be either the "zoo-script" following under sentence (1a) or the "ballgame-script" following under sentence (1b). (This particular ballgame-script incidentally is mine and thus it models my experience and understanding of what happens when one goes to a baseball game; the zoo-script is my daughter Molly's.)

(1) Graham went to see the tigers last night.
(2) Paul went to see the Cubs last night.
(3) Howard went to see the expos last night.

(1a) He saw Eric in the monkey house.

(1b) He saw Eric during the National Anthem.




ACCOUNTING SCRIPT

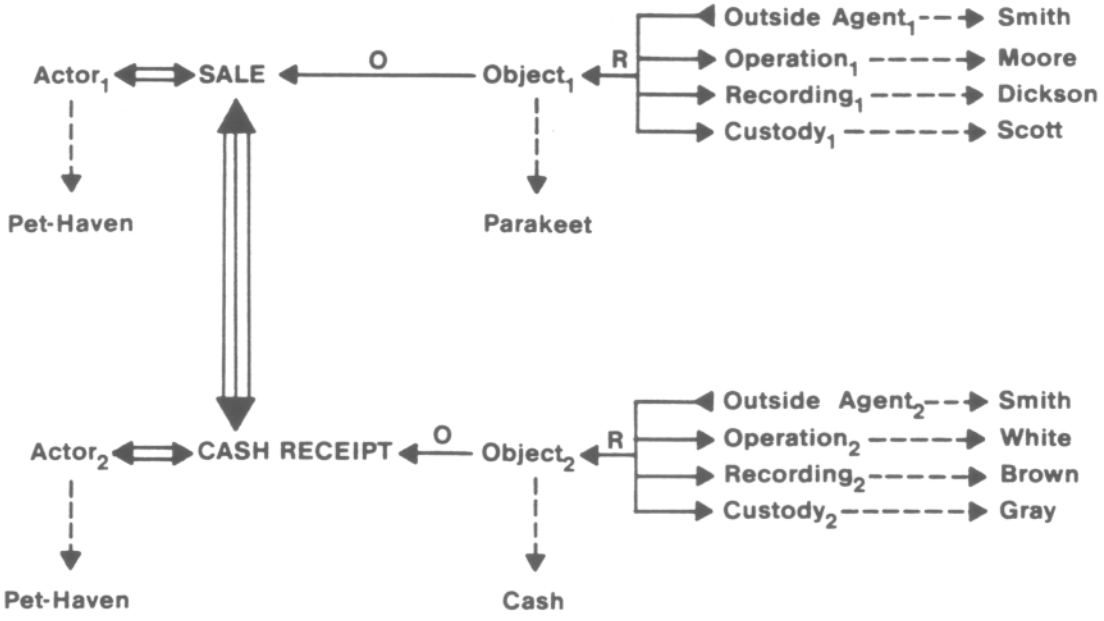
(Subscribe to stock) → (Pay money) → (Get certificate) → (Receive dividends)

Fig. 12 **VARIOUS TYPES OF SCRIPTS**

Scripts are knowledge structures which represent sterotypical sequences of actions, and their primary AI use has been in the research areas of natural language processing and human memory understanding [Schank, 1984]. Scripts can be conceptualized as semantic networks where the nodes represent the actions or *scenes* and the links interpret as ''is-followed-by.'' Each scene can be a script in its own right (for example ''watch game'' which has three subscripts in Figure 12), and these sub-parts can be broken down all the way to their most elementary actions (such as the physical transfer of money involved in ''buying popcorn'') [Schank and Ableson, 1978]. Additionally, each scene could itself be conceptualized as another type or node in a semantic net such as *a frame* (with slots for ''home team'' and ''starting pitcher'' for example in ballgames) or *a generalization hierarchy* (with an ancestor of ''sporting-event'' for ballgames). Conceptually, scripts can be specified at both higher and lower levels of abstraction to explore issues such as learning and analogous thinking [Lehnert, 1981]. For example, a researcher might use this mechanism to explore why going to zoos might remind people of going to ballparks or to explore what happens to general cognitive structures when particular script instances don't comply (such as when a different national anthem is played or when the players use aluminum bats).

At the bottom of Figure 12, I have included an example of a script from the domain of accounting to show readers the types of actions which might need to be impounded in a machine representation of background knowledge for an accounting decision-maker. Again, each of these four actions would probably be broken down into a number of more detailed subscripts. From the perspective of a corporation for instance, the "receive dividend" script for a stockholder might consist of "declare dividend" and "pay dividend," and each of these actions could be broken down further in turn.

In Figure 13, a much more elaborate script is presented which details not just sterotypical sequences of events, but also the allowable sets of actors and roles for those events. These particular representations and mechanisms for enforcing them in large complicated databases are currently being explored with regard to automating the images that auditors have of "good internal controls" for a company [Gal and McCarthy, 1985, 1986b]. Audit knowledge to be represented in this domain would include, for instance, the concepts of "an approved transaction" and "sufficient separation of duties."




**REVENUE - CYCLE SCRIPT**

**Fig. 13** **SOURCE: Adapted from Gal and McCarthy, 1986b.**

### *Knowledge Representation Methods Summary*

We have now completed our review of three different types of knowledge representation formalisms: (1) logic, (2) rules, and (3) semantic networks. The semantic networks were covered in particular depth because of their relationship to this paper's primary thesis which is the future integrated use of both DB and AI systems. Readers are reminded again, however, about the arbitrary nature of my review. Formalisms that I have deemphasized, such as logic, can be used extensively

in combined DB-AI work, and representation examples that I have presented in one way are easily translated to other methods.

## *Future Directions*

At the outset of this presentation, I speculated that AI-type or knowledge-based systems represent an arena into which accounting information system designers should naturally move as they become more interested in providing accounting data support for unstructured decision-making in areas of management control and strategic planning. In this last section of the paper, I intend to discuss what the architecture of these newer systems might look like and what changes in systems analysis methods might be required.

### *Architecture of ES-DBMS Environment*

Figure 14 illustrates the tight coupling which will have to occur between the database software environment and the expert system software environment for the movement toward knowledge-based accounting systems to be successful.

At the top of Figure 14 is portrayed the database environment whose design iterates through the traditional systems analysis life cycle with the exception that much more attention is paid to semantically-oriented data model specification [Armitage, 1985]. The database which results from this process is controlled and used via DBMS software, and it consists of two kinds of information: (1) the database *extension* which is the accumulated store of various accounting transaction data and (2) the database

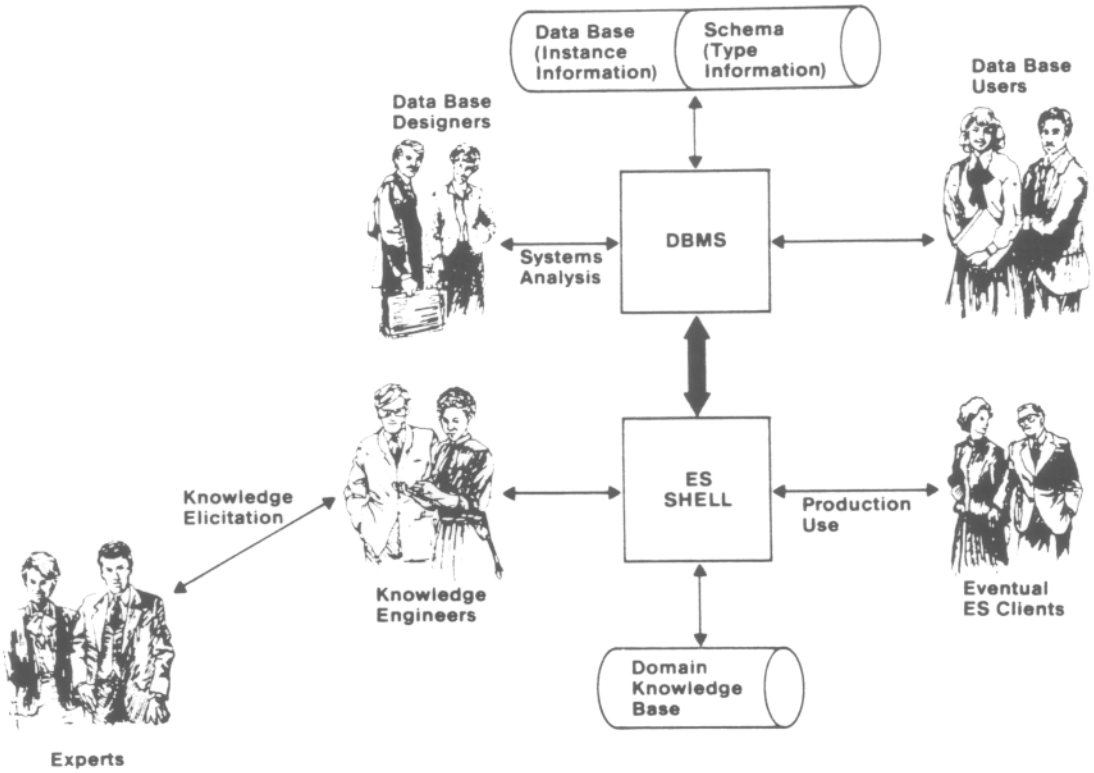



Fig. 14 ES AND DBMS COUPLING

*intension* which is the type information designated in the language of one of the knowledge representation formalisms covered earlier.

The bottom of Figure 14 illustrates the expert system environment for some type of accounting-oriented expertise whose structure has been elicited and captured by knowledge engineers with an ES "shell." The shell is a commercially available piece of software which facilitates the construction of a system by automatically managing procedures such as chained inferencing and property inheritance. The shell takes care of the ES overhead, while the actual specifics of a certain expert process are stored in a separate "domain knowledge base."

For the coupled systems to work well together, the knowledge representation formalisms used in each will have to be compatible. Up to the present time, most accounting ES that have been built have not had to face this problem, because they have not depended on access to large accounting transaction databases. In other words, their "facts" came from outside the company or from the experts themselves. Quite obviously, there is a great opportunity for newer classes of computerized accounting systems whose processes on the one hand require heuristic or ES-type specification, but whose data needs on the other hand require access to elements of the large corporate database.

Technologically, the tight links shown in Figure 14 are not yet supported on a widespread basis, although there has been some encouraging work completed in research settings. In the meantime, there are some nontechnological changes which accounting information systems researchers and practitioners can explore, so that the structures of their accounting systems and the methods by which they build those systems can be ready when the new software technology is.

## *Changes in Analysis and Design Methods for Accounting Information Systems*

*More Semantic Infrastructures*. First and foremost, there needs to be continued movement toward the goal of making accounting transaction databases more "epistemologically adequate." After our review of knowledge representation structures, I hope that readers would agree that our present methods of transaction processing and data representation in accounting are less ambitious than they need to be. Most computerized accounting systems remain philosophically closer to the insular mindset of the bookkeeper than to the more modern philosophy of shared usage and knowledge representation. As illustrated in Figure 15, this accounting restructuring requires a movement towards an *events* [Sorter, 1969] [McCarthy, 1981] orientation which means that accountants move away from their "observe-interpret-record" methods of value-accounting and toward simpler "observe-record" systems. Events models are driven by the same knowledge representation philosophies that database systems are driven by in the sense that both primarily endeavor to build information systems whose users can act as if they had actually observed the actual phenomena of interest (that is, the company's transaction activity). Too often, modern accounting systems contain account classifications and valuations which are interpretations indigenous only to bookkeeping. Such interpretations are not epistemologically correct, and they should be relegated to user views. As I have argued elsewhere, [McCarthy, 1984] I believe that REA-type structures form a solid basis on which to build conceptual structures of accounting activities.

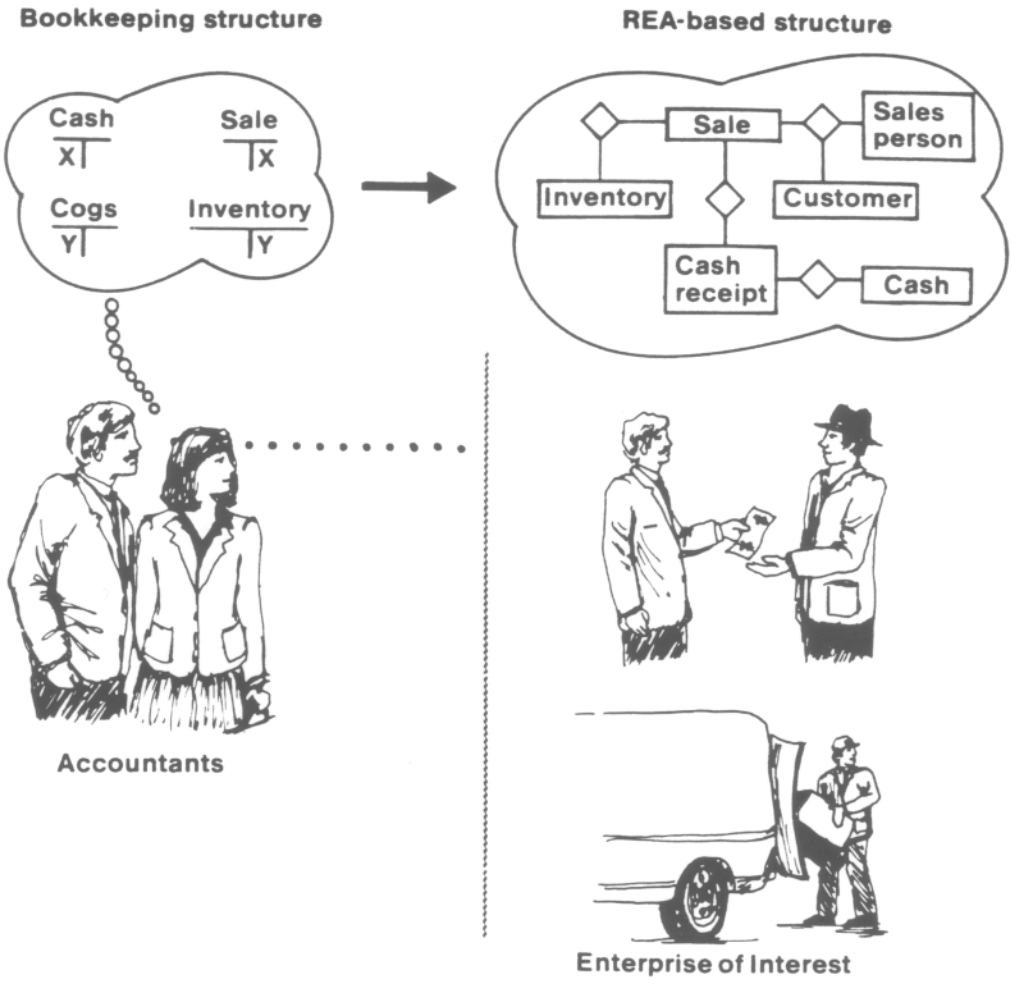
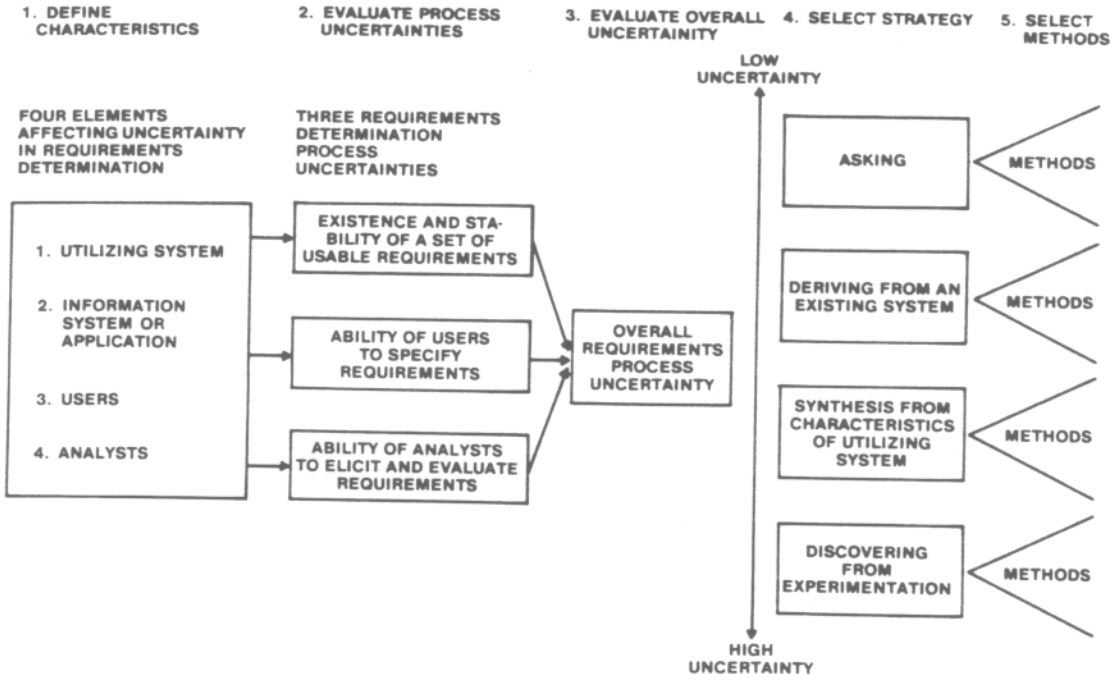



Fig. 15

*More Use of Enterprise Modeling and Prototyping*. Second, many of the more fundamental activities used in analysis and design of accounting systems will have to be supplemented with newer methods specifically designed for the less-structured environment. For example, Davis's contingency theory for the specification of information system requirements (see Figure 16) predicts that environments characterized by high uncertainty because of the absence of a stable set of requirements and because of user and analyst inexperience will not be served well by traditional accounting system life cycle methods (like asking users or using existing systems as models). This is exactly the type of environment in which knowledge-based systems will operate. In other words, the "way we have always done it" will *not* be the model of the future. Designers will find it necessary to become familiar with more heuristic methods associated with the high uncertainty end of the spectrum shown in Figure 16. These will include methods that emphsize firm-wide analysis (such as enterprise data modeling) and methods that emphasize the use of experiemental AI prototypes as a method for explicating the genuine needs of users.




DAVIS CONTINGENCY APPROACH TO INFORMATION REQUIREMENTS DETERMINATION

SOURCE: Davis and Olson, p. 489.

**Fig. 16**

## *Summary*

I have titled this paper, "On the Future of Knowledge-Based Accounting Systems." We have looked together at some initial definitions of knowledge-based systems and taken a fairly extensive review of the conceptual structures used within them. I finished with an enumeration of my own ideas of the changes that these techniques would bring upon accounting information system users and designers. As I mentioned in the paper, technology in the areas of both database systems and artificial intelligence is still changing rapidly, and the study of computerized systems which combine these two technologies is still in its infancy. Commercial interest and activity seems to indicate, however, that the effect of all of these areas upon corporations will be pervasive. Since most of this new technology will ultimately depend upon data sourced from accounting transaction systems, I believe that the changes upon accounting will be quite dramatic as well.

## *References*

Armitage, H. M. (1985), *Linking Management Accounting with Computer Technology* (Society of Management Accountants of Canada Research Monograph Series, 1985).

Biggs, S. F. and M. Selfridge (1986), "GC-X: A Prototype Expert System for the Auditor's Going Concer Judgement," Working Paper, University of Connecticut, January 1986.

Brachman, R. J. (1983), "What IS-A Is and Isn't: An Analysis of Taxonomic Links in Semantic Networks," *Computer* (October 1983), pp. 30-36.

Brodie, M. L., J. Mylopoulos, and J. W. Schmidt, eds. (1984), *On Conceptual Modeling*, (Springer-Verlag, 1984).

Chen, P. P. (1976), "The Entity-Relationship Model—Towards a Unified View of Data," *ACM Transactions on Database Systems* (March 1976), pp. 9-36.

Dahl, V. (1983), "Logic Programming as a Representation of Knowledge," *Computer* (October 1983), pp. 106-11.

Davis, G. B. and M. H. Olson (1985), *Management Information Systems: Conceptual Foudnations, Structure, and Development* (McGraw-Hill, 1985).

Denna, E. and W. E. McCarthy (1986), "An Events-Accounting Foundation for DSS Use," *Proceedings of the NATO Advanced Study Institute*, Maratea, Italy, 1986.

Dungan, C. W. and J. S. Chandler (1985), "Auditor: A Microcomputer Based Expert System to Support Auditors in the Field," *Expert Systems* (October 1985), pp. 210-221.

Gal, G. (1985), "Using Auditor Knowledge to Formulate Data Model Constraints: Expert Systems for Internal Control Evaluation," Ph.D. Dissertation, Michigan State University, 1985.

Gal, G. and W. E. McCarthy (1985), "Specification of Internal Accounting Controls in a Database Environment," *Computers and Security* (March 1985), pp. 23-32.

_____ (1986a), "Operation of a Relational Accounting System," *Advances in Accounting* (1986).

_____ (1986b), "Semantic Specification and Automated Enforcement of Internal Control Procedures Within Accounting Systems," Working Paper, Michigan State University, 1986.

Genesereth, M. R. and M. L. Ginsberg (1985), "Logic Programming," *Communications of the ACM* (September 1985), pp. 933-41.

Howe, D. R. (1983) *Data Analysis for Data Base Design* (Edward Arnold, 1983).

Lehnert, W. (1981), "Plot Units and Narrative Summation," *Cognitive Science* (Vol. 5, 1981).

McCarthy, W. E. (1979), "An Entity-Relationship View of Accounting Models," *The Accounting Review*, (October 1979), pp. 667-86.

_____ (1981), "Multidimensional and Disaggregate Accounting Systems: A Review of the 'Events' Accounting Literature," *MAS Communication* (July 1981), pp. 7-13.

_____ (1982), "The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment," *The Accounting Review* (July 1982), pp. 554-78.

_____ (1984), "Materialization of Account Balances in the REA Accounting Model," Invited Presentation to the British Accounting Association, Norwich, England, April, 1984.

Meservy, R. D. (1985), "Auditing Internal Controls: A Computational Model of the Review Process," Ph.D. Dissertation, The University of Minnesota, 1985.

Minsky, N. (1975), "A Framework for Representing Knowledge," in P. H. Winston (ed), *Psychology of Computer Vision* (McGraw-Hill, 1975), pp. 211-277.

Patrick, R. L. (1978), "Auditing and DP:" Redressing the Relationship," *Datamation* (15 November 1978), pp. 139-44.

Schank, R. and R. Abelson (1977), *Scripts, Plans, Goals and Understanding* (Erlbaum, 1977).

Schank, R. (1984), *The Cognitive Computer* (Addison-Wesley, 1984).

Smith, J. M. and D. C. Smith (1977), "Database Abstractions: Aggregation and Generalization," *ACM Transactions on Database Systems* (June 1977b), pp. 105-133.

_____ (1978), Principles of Database Conceptual Design," *NYU Symposium on Database Design*, (New York University, 1978), pp. 35-49.

Sorter, G. H. (1969), "An 'Events' Approach to Basic Accounting Theory," *The Accounting Review* (January 169), pp. 12-19.

Sowa, J. F. (1984), *Conceptual Structures: Information processing in Mind and Machine* (Addison-Wesley, 1984).

Steinbart, P. J. (1984), "The Construction of an Expert System to Make Materiality Judgements," Ph.D. Dissertation, Michigan State University, 1984.

_____ (1987), "Materiality: A Case Study Using Expert Systems," *The Accounting Review* (January 1987).

Tsichritzis, D. C. and F. H. Lochovsky (1982), *Data Models* (Prentice-Hall, 1982)

Winston, P. H. (1984), *Artifical Intelligence* (Addison-Wesley, 1984).