1981

Toward A Formal Definition of Task Representation

Richard D. Hackathorn
University of Colorado

Robert A. Fetter
University of Colorado

Toward A Formal Definition of Task Representation*

Richard D. Hackathorn
University of Colorado

Robert A. Fetter
University of Colorado

ABSTRACT

This paper addresses the issue of how tasks within an organizational context should be represented from the perspective of a single decision maker. Based on a previous paper (Hackathorn, 1981), this paper presents a formalism for task representation based on recent work in the Knowledge Representation area. The formalism is called Simple Associative Network (SAN). The implications of this formalism result in the discussion of several issues, such as: (a) the nature of task occurrence, (b) handling multiple task types of a task occurrence, (c) means and goals as a specialization of task types, and (d) control structures among task types.

INTRODUCTION

In a previous paper (Hackathorn, 1981), the argument was made that the concept of task needed to be formalized as a basis for understanding how a decision process could be supported using computer technology. A task was defined in this paper in a general manner as "a unit of purposeful human action" after Ackoff and Emery (1972) and Newell and Simon (1972). A distinction was made between "task representation" (i.e., a language for describing tasks) and "task description" (i.e., an actual description of a specific task). It is argued that decision support is the management of task descriptions.

The focus of this paper is to develop a formalization of task representation through the application of recent knowledge representation research. This formalization will deal with the ambiguity of decision processes within the organizational context and will show distinct levels of task abstractions. Finally, issues for contructing decision support tools will be discussed.

SUPPORT OF DECISION MAKING

An important theme of the Decision Support Systems (DSS) literature is the emphasis on "support" rather than "replacement" or "automation" (Scott

*The authors wish to express appreciation to the members of the Information Systems Doctoral Seminar at the University of Colorado, Boulder. Its members include: Paul Zeiger, Jean Bell, Tracy Hansen, and Dick Sowar. The advice and encouragement of Peter Keen, Ralph Sprague, Benn Konsynski, Joyce Elam, John Henderson, and Roger Weissinger-Baylon is also greatly appreciated. Finally, thanks once again to Marcia Bravard for making an infinite series of revisions into a polished and readable manuscript.

Morton, 1971; Gorry & Scott Morton, 1971; Keen & Scott Morton, 1976; Alter, 1975, 1976, 1979; Keen, 1980; and Sprague, 1981a, 1981b). The point is that technology should be used to assist managerial decision making, rather than replace the human involvement. The tasks that engage managers within organizational settings are extremely complex and fundamentally ambiguous. The specification of formal algorithms to accomplish these tasks is very difficult and probably inappropriate. Further, attempts to apply technology to automate managerial tasks have often led to failure.

The literature of such fields as systems analysis, information requirements analysis (Teichroew, 1974a, 1974b), and more recently office specification languages (Cook, 1980; Ellis, 1979; Ellis & Nutt, 1980; Hammer & Kunin, 1980; Tsichritzis, 1980; Zisma, 1977; Nawojski & Konsynski, 1981) deals with formalisms for describing activities of both humans and computers. The deficiency that this literature has is that it deals with the concept of "human purpose" in a very shallow fashion.

For the most part, human purpose is acknowledged but then eliminated from these formalisms. For most computer applications, the purpose or functionality of that application is specified at the beginnning of the development process and may be used at the end to evaluate the resulting application. Note that the assumptions about purpose are that: (1) the purpose is static, specifiable, and valid; and (2) the means of accomplishing that purpose can be formulated. These assumptions constituted the essence of the terms "routinization," "mechanization," and "automation."

In the DSS literature, there is a strong reaction against the automation of information processing related to managerial activities that are "unstructured." The term "unstructured" comes from the research of Simon and others (1969, 1976) to indicate the degree to which an activity can be specified and, hence, the degree of determinism. Other authors have noted that structuredness is a relative concept of the observer. Different observers note different degrees of structuredness for an activity depending on their understanding of the activity and assumptions about importance, etc. Stabell (1974, 1977) even states that the structuredness of an activity is often an organizational policy decision, one based on payoffs to the organization balanced with cost of information processing.

The motivation for "support" rather than "automation" is based on the notion of <u>ambiguity</u> with which a manager must cope. This ambiguity causes a dynamic tension between the pressure to act counterbalanced with the indecision of acting. More precisely, the dilemma for a manager is the choice of feasible actions that will result in the occurrence of desired events, leading to a goal. An important function for decision support systems is to assist in resolving the dilemma of planning actions. This paper addresses an aspect of that dilemma, that of <u>task formulation.</u>

The next section will elaborate on the notion of ambiguity in organizational activity.

AMBIGUITY OF ORGANIZATIONAL ACTIVITY

Consider the following situation:

> Imagine that you're either the referee, coach, player, or spectator at an unconventional soccer match: the field for the game is round; there are several goals scattered haphazardly around the circular field; people enter and leave the game whenever they want to; they can throw balls in whenever they want; they can say "that's my goal" whenever they want to, as many

times as they want to, and for as many goals as they want to; the entire game takes place on a sloped field; and the game is played as if it makes sense (Weick, 1976, from a private communication with James March).

The above quote of Karl Weick's captures the flavor of the decision situation that most managers find themselves in. This situation is in sharp contrast to traditional organization theory that focuses upon:

- Explicit selection of goals,
- Rational formulation of plans to achieve those goals,
- Efficient execution of the plans through division of labor, and
- Layers of authority relations and incentives to control the above.

As stated by Weick (1976), the problem with traditional organization theory is the following: "People in organizations find themselves hard pressed either to find actual instances of those rational practices or to find rationalized practices whose outcomes have been as beneficient as predicted, or to feel that those rational occasions explain much of what goes on within the organization." Further, Weick notes that parts of some organizations are heavily rationalized, yet the large, remaining section has consistently avoided rational analysis. One wonders how such inconsistent and loose systems retain a sense of identity across time, so that they can be recognized and dealt with in some formal manner.

A concept that attempts, at least in part, to answer the intriguing question of understanding consistency amid chaos is that of <u>loose coupling</u>. Loose coupling suggests that most organizations or systems can be decomposed into stable subassemblies that comprise the crucial elements of the organization or system. This decomposition allows the manager to handle the complexity issues through modularity. An event that is coupled to another or other events is responsive to those events, yet "still preserves its own identity and some evidence of its physical or logical separateness" (Weick, 1969). The notion of loose coupling is similar to self-contained tasks as described in Galbraith (1973, 1977).

The degree of coupling between two events has been characterized on the basis of the activity of the shared variables of the two events. The specification of events and couplings is not a one-shot activity, as both events and couplings may appear and disappear over time. Any theory of coupling must also take into account the nature and intensity of the couplings, as these can cause the creation and dissolution of events and couplings.

Cohen, March, and Olsen (1972) describe three general properties of organizations or decision situations:

1. <u>Problematic Preferences</u>. An organization is usually better described as a loose collection of ideas rather than as a coherent structure. An organization discovers preferences more through actions than acting on the basis of preferences. It is therefore difficult to impute a set of preferences to the decision situation.

2. <u>Unclear Technology</u>. The organization manages to survive, yet its processes are not understood by its members. It operates more on trial and error procedures and learning from past mistakes.

3. <u>Fluid Participation</u>. Participants vary in the amount of time and effort they devote to different

domains; their involvement varies from one time to another. The boundaries of the organization, therefore, are poorly defined and constantly changing.

These properties lead to several forms of ambiguity concerning the decision context within the organization (March & Olsen, 1976):

1. Ambiguity of Intention. Organizations and even individuals have inconsistent and ill-defined objectives.

2. Ambiguity of Understanding. The causal relationship among both internal and external events is uncertain.

3. Ambiguity of History. Past events are important, but they are not easily interpreted. What appeared to happen may not really have happened; what really happened may not have been intended. The flow of individual actions produces a flow of decisions that is intended by no one and is not related in a direct way to anyone's desired outcomes (March & Olsen, 1976).

4. Ambiguity of Attention. At any point, individuals vary in the attention that they give to various decisions. Participation in these decisions is fluid and continously changing.

These forms of ambiguity indicate that a revised theory of management is required in which goals are unclear, and standard procedures are not applicable (March & Olsen, 1976). Mintzberg (1973; Mintzberg, Raisinhani & Theoret, 1976) has begun to address the problem of identification of the manager's role in an organization in an attempt to develop a normative theory for management action in unstructured decision settings. It is recognized that current normative theories have a significant influence on the procession of the lower and middle levels (or subassemblies) of an organization, but have had virtually no influence on the higher levels. In fact, by applying normative techniques to a fundamentally ambiguous setting, the organization might actually be pursuing "inappropriate courses of action more efficiently" (Mintzberg, Raisinhani, & Theoret, 1976).

When one studies the individual decision maker, it appears that the decision maker's actions in complex settings could hold the key to understanding the decision process. Individual decision makers deal with complex, unstructured problems by breaking them down into more familiar, unstructured settings (Mintzberg, Raisinhani, & Theoret, 1976; Payne, 1976). Use of problem solving shortcuts such as satisficing is also in evidence. Mintzberg and his associates (1976), through a four year study of twenty-five decision processes, have been able to subdivide the twenty-five processes into seven distinct groups. Of these seven groups, four have been found to involve similar outcomes.

The implication is that, with an appropriate representation, it might be possible to develop theories to model the ambiguities of complex organizations. The next section outlines the initial development of such a representation scheme.

TASK REPRESENTATION USING ASSOCIATIVE NETWORKS

The previous section described the ambiguity of organizational activity with which the manager must cope. Needed is a representation technique that captures the essential aspects of this environment. This section suggests a simple representation for organizational activity based on associative networks.

The use of graph based structures to associate factual information was first introduced by Ross Quillian (1968, 1969) in his Ph.D. dissertation. Since then, many researchers have adopted similar techniques for knowledge representation (KR). See Woods (1975), Brachman (1979), and Mylopoulos (1981), for a review of previous research with associative networks (AN). Note that the term "semantic networks" is not used since it refers to the "representation for linguistic utterances to capture the underlying relations to words and to produce information from text" (Brachman, 1979). Hence, semantic networks have a narrower scope than associative networks.

For the purposes of exploring alternative representations for tasks, a simple representation using associative networks was formulated and is called SAN for "Simple Associative Network." SAN is a composite of the Procedural Semantic Network (PSN) of Levesque and Mylopoulos (1979) and KLONE of Brachman (1977). Further, the aggregation relation is handled in SAN in a way similar to the notion of frames (Minsky, 1975; Bobrow & Winograd, 1976, 1977, 1979). A frame is a structured collection of properties (i.e., slots) that surround an object, giving it the ability to represent complex situations in a compact manner. Many of the advanced, and more interesting, features of these AN representations are, however, not incorporated into SAN so that the complexity of SAN could be minimized for the purposes of task representation.

Representation Levels

The following sections will discuss the SAN representation in terms of "representation levels" that are built successively upon one another (Brachman, 1979). The intention is that the concepts and terminology of one level are used as the primitive units of the next level. This partitioning of the SAN aspects increases the modularity of the representation.

The first representation level (i.e., I-level) relates to the implementation aspects, such as the definition of the nodes and edges of the directed graph formalism.

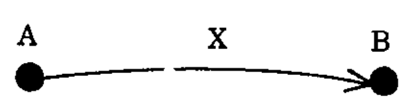
The second level (i.e., L-level) relates to the logical aspects, such as assertions of relations between two objects. The definition of assertions uses the terminology of nodes and edges of the previous level.

The third level (i.e., E-level) relates to the epistemological aspects. This level is usually missing in many AN representations (Brachman, 1979) and is the subject of current KR research. For example, the epistemological level deals with the distinction between object types and instances and generalization hierarchies of object types.

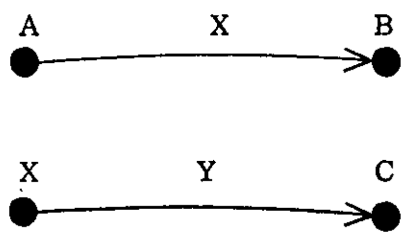
The fourth and final level (i.e., C-level) is called the conceptual level and deals with the knowledge specific to the problem context. In the case of task representations, the conceptual level will define an object called task along with its structural properties. Also, a control structure will be defined over a set of task types using relations with the object type condition.

Implementation Level of SAN

The primitive concepts of SAN representation are based on a directed graph consisting of nodes interconnected with edges. The node having an edge "going out of it" is called the out-node for that edge. Similarly, the node having an edge "going into it" is called the in-node for that edge. The triple of (out-node, edge, in-node) is called a link. Consider the following link:



The nodes are labeled "A" and "B" respectively and the edge is labeled "X." Labels on nodes and edges refer uniquely to tokens. The purpose of tokens is to allow nodes and/or edges with the same labels to be considered equivalent. Moreover, the same token can be used as both a node and an edge. For example, consider the token "X" in the following two links:



The implementation level of SAN can be summarized using the Relational Data Model of Codd (1970). A SAN "database" consists of tuples for the relation LINK having the form:

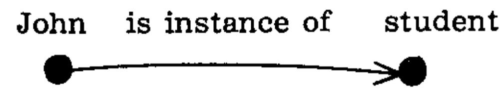
LINK (OUT-NODE, EDGE, IN-NODE)

where all three attributes act as the primary key and have the same domain of TOKEN. Note that in the relation LINK all tuples must be unique, implying that there do not exist identical edges between any two nodes.

Logical Level of SAN

The next level up from the Implementation Level is called the Logical Level, or simply the L-level. The L-level gives to tokens and links an interpretation equivalent to predicate calculus. Hence, the link acts as a binary relation that can be used to build an arbitrary complex network of assertions.

More precisely, a link at the I-level represents an assertion at the L-level consisting of a relation (corresponding to the edge) and two objects (corresponding to the nodes). The out-node represents the domain object, while the in-node represents the range object. For example, consider the assertion "John is a student." This assertion at the L-level can be represented at the I-level by the link:



We will adopt the following notation to indicate that a link represents an assertion:

(John; is instance of; student)

Note that the semicolon is a reserve symbol that separates the relation token from the two object tokens.

For a good discussion of the correspondence of predicate calculus to associative networks, see Nilsson (1980) where he discusses "structured object representations." Nilsson also suggests the extension of the L-level to first order predicate calculus by the addition of connectives (e.g., conjunction, disjunction, implication, and negation) and of universal and existential quantification. The extension will not be pursued in this paper since it is not yet needed for the purposes of task representation.

Epistemological Level of SAN

The next level in the SAN representation is called the epistemological level or E-level. Woods (1975) and Brachman (1979) criticize most KR research as having vague notions of their epistemological aspects. He argues for the "formal definition of knowledge-structuring primitives" as opposed to the individual pieces of knowledge values (e.g., "John is a student").

Distinction Between Object Instances and Object Types

The first aspect of the SAN epistemological level is the distinction between object types and object instances. An object type is a set or category of objects, such as "person." The object "person" does not actually exist in reality; however, what does exist are instances of "person," such as John, Mary, and Joe. Hence, an object instance is an actual object in reality, while an object type is a classification of a collection of equivalent object instances. This distinction is usually clear for tangible objects. There is, however, a significant degree of ambiguity when dealing with events and concepts.

The convention is followed using upper case letters for labels of generic object types and using lower case letters for object instances and problem-specific object types.

Abstraction Assertions

The E-level of SAN consists of three abstraction assertions:

1. Classification. An assertion is said to be a classification if a specified object token (e.g., John) is a member of an object type (e.g., "student"). A classification assertion is indicated in SAN by using the special token "is instance of" as an edge:

   (John; is instance of; student)

2. Generalization. An assertion is said to be a generalization if it relates one object type (e.g., "student") as a subset of another object type (e.g., "person"). A generalization assertion is indicated, using the special token "is type of" as an edge, as follows:

   (student; is type of; person)

3. Aggregation. An assertion is said to be an aggregation if it relates an object instance to its components (e.g., John has brown hair). An aggregation assertion is indicated in SAN as follows: The link above asserts that "person" has a property attribute of "has hair color of" through the special token "is part of." A second link below asserts that an object instance of "person" (i.e., John) has a particular property value of that attribute, namely brown hair.

   (has hair color of; is part of; person)

The link above asserts that "person" has a property attribute of "has hair color of" through the special token "is part of." A second link below asserts that an object instance of "person" (i.e., John) has a particular property value of that attribute, namely brown hair.

(John; has hair color of; brown)

A mild consensus is developing among the KR researchers that the above three abstraction assertions are "primitive" (Mylopoulos, 1980). Also note that Smith & Smith (1977) define aggregation and generalization along similar lines.

We will adopt the notion that the verb "is" is reserved for these three abstraction assertions, while the verb "has" is used to form property attributes of the aggregation assertion. Hence, the verb "is" is more primitive and focused on the E-level, while the verb "has" is an important building tool for the C-level.

Between the classification and generalization assertions, a hierarchy of object types can be formed. The object types higher in the hierarchy are generalizations of those lower. Conversely, the object types lower in the hierarchy are specializations of those higher. Hence, one can say that an object type is specialized into lower level object types.

Inheritance of Properties

In the example above, note that John (which is an instance of student) inherited the property attribute of "has hair color of" from the object type "person." This inheritance propagates through two levels of the abstraction hierarchy. Property inheritance is an important feature in many AN representations.

At this stage of the SAN representation, the mechanisms for property inheritance are not fully defined. We can, however, supply a few general properties of inheritance.

When an object is an instance of more than one class, it will inherit attributes from all the classes. In particular, when a class is a subclass of another class (i.e., student is a subclass of person) an object which is an instance of the subclass (i.e., John), will inherit attributes of both classes.

Inheritance of attributes is a convenient way to extend and refine existing classes, because of the cumulative nature of inheritance. It is in precisely this manner that inheritance serves as an abstraction mechanism, since objects are defined by showing how they differ from existing, more general objects. Details concerning the similarities between objects are suppressed (as shown by Levesque & Mylopoulos, 1979).

We will also separate inheritance of property attributes from inheritance of specific values for those attributes. For example:

(average IQ; is part of; person)

(student; is type of; person)

The class student will inherit the property attribute of average IQ, yet the value assigned to the attribute for the class person will not be inherited by the class student, as they may be radically different. In some cases, however, values will be inherited (particularly for default values). When this occurs, the inheritance of a value always presupposes the inheritance of the associated property attribute.

Definition of CLASS, RELATION, and VALUE

All objects (i.e., types and instances) are related together through a generalization hierarchy to the object type OBJECT (as suggested by Levesque & Mylopoulos, 1979). OBJECT is then specialized into CLASS, RELATION, and VALUE through the assertions:

(CLASS; is type of; OBJECT)

(RELATION; is type of; OBJECT)

(VALUE; is type of; OBJECT)

OBJECT is also given a special property attribute "has default of" that will be used later to given object instances a default property value if none is specified for that instance:

(has default of; is part of; OBJECT)

CLASS can be further specialized into object types appropriate to the problem context. As explained in the section entitled "Conceptual Level of SAN," CLASS will be specialized into person, task, and condition.

RELATION is specializd into ASSERTION and PROPERTY as follows:

(ASSERTION; is type of; RELATION)

(PROPERTY; is type of; RELATION)

PROPERTY differs from ASSERTION in that PROPERTY gives "structure" to an object type by attaching PROPERTY attribute to that object type. The implication is that instances of that object type should have (in the normal situation) associated property values for each property attribute. An example of the property attribute "has hair color of" was given in a previous section. On the other hand, ASSERTION specifies a relation between object types in general, rather than some structural component of a particular object type.

The only instance of the PROPERTY object type is "is part of" corresponding to the aggregation relation:

(is part of; is instance of; PROPERTY)

On the other hand, ASSERTION has two instances, corresponding to the classification and generalization relations respectively:

(is instance of; is instance of; ASSERTION)

(is type of; is instance of; ASSERTION)

To explicitly differentiate ASSERTION and PROPERTY object types, each are given differing property attributes. ASSERTION object type has the property attributes of domain and range:

(has domain of; is part of; ASSERTION)

(has range of; is part of; ASSERTION)

along with the general default of CLASS for both domain and range:

(has domain of; has default of; CLASS)

(has range of; has default of; CLASS)

The PROPERTY object type has only a "value" property attribute with a default of VALUE:

(has value type of; is part of; PROPERTY)

(has value of; has default of; VALUE)

The domain of a property attribute is understood to be the specified object type in the "is part of" assertion.

The VALUE object type can be specialized, as needed, into categories appropriate for the property values of the problem context. For example, the specification of the "state" of a condition has the range of TRUTH VALUE, that is defined as follows:

(TRUTH VALUE; is type of; VALUE)

(true; is instance of; TRUTH VALUE)

(false; is instance of; TRUTH VALUE)

To summarize the discussion of the E-level, Figure I shows the generalization hierarchy of OBJECT, etc. The boxes indicate object types, while the circles indicate object instances. The double arrows indicate the "is type of" assertion (i.e., generalization) and the single arrows indicate the "is instance of" assertion (i.e., classification). The E-level is composed of the objects above the dashed line, while the C-level (to be discussed in the next section) is below the line. The intent is that the objects within the E-level are relatively independent of particular problem contexts or situations, in contrast to the C-level whose function is to model the problem context.

Conceptual Level of SAN

The final representation level for SAN focuses on the knowledge aspects that are specific to tsk representations. As mentioned before, CLASS is specialized into person, task, and condition:

(person; is type of; CLASS)

(task; is type of; CLASS)

(condition; is type of; CLASS)

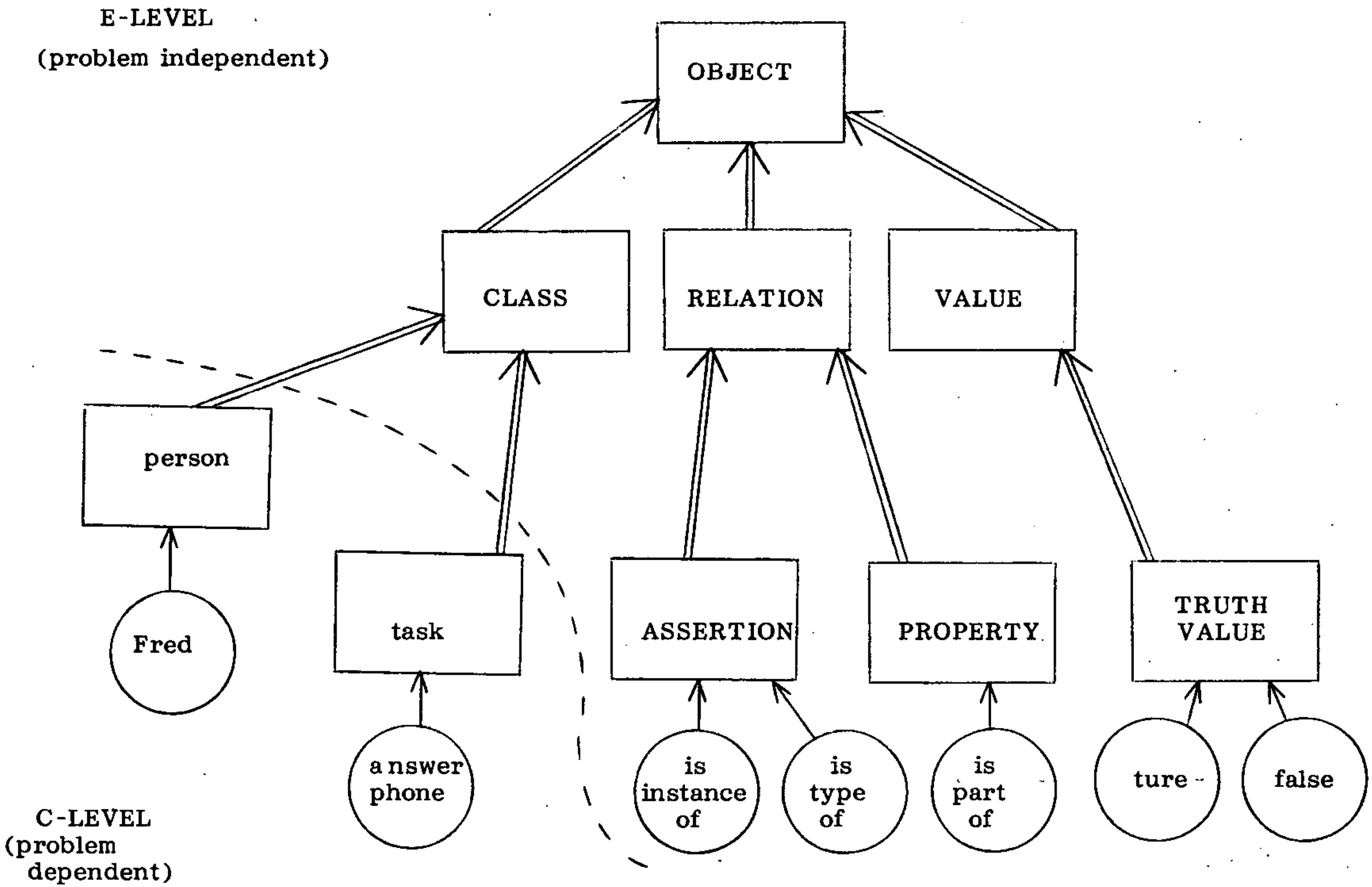


Figure 1. Generalization Hierarchy

The object type "person" is self-evident; the next sections will explain the definitions of the "task" and "condition" object types.

Definition of Task

Three aspects are explained relative to task definition: (a) property attributes of task, (b) task specialization into means/goals hierarchy, and (c) the nature of task instances.

First, the property attributes of task were chosen to be: time, actor, and pre-conditions:

(has time of; is part of; task)

(has actor of; is part of; task)

(has condition of; is part of; task)

The value types for these attributes are:

(has time of; has value type of; TIME VALUE)

(has actor of; has value type of; person)

(has condition of; has value type of; condition)

Secondly, task can be specialized into lower level task types, such as "answer phone" or "send literature." Through this specialization, a means/goal hierarchy can be constructed. The specialization of a task type can be said to be the means of that task type. Conversely, the generalization of a task type can be said to be the goal of that task type. After Ossorio (1978), higher level task types can be viewed as "how do you do that?" The means versus goal distinction introduces the notion of purposefulness into the concept of task (Ackoff & Emery, 1972; Simon, 1969). For example, consider the two tasks "market product" and "send literature." The second task could be considered a specialization of the first:

(market product; is type of; task)

(send literature; is type of; market product)

Finally, the nature of task instances needs to be specified. The approach used in this paper is to define a new assertion "is occurrence of" to distinguish task types from task instances:

(is occurrence of; is instance of; ASSERTION)

For example, consider the task "answer phone" and an occurrence of it:

(answer phone; is type of; market product)

(*01; is occurrence of; answer phone)

(*01; has time of; 12:34 3/13/81)

(*01; has actor of; Fred)

Note that a task occurrence (and most object instances for that matter) is referred to by an indefinite label, implying that the occurrence is denoted in conversation by its context (e.g., "the phone call that Fred had thirty minutes ago").

Definition of Condition

For the purpose of interrelating tasks in terms of a "control structure," the concept of object type "condition" is introduced into the task representation. The property attributes of a condition involve the condition state and the activation of that condition:

(has value of; is part of; condition)

(has state of; has value type of; TRUTH VALUE)

(has state of; has default of; false)

(has activation of; is part of; condition)

(has activation of; has value type of; task)

Note that the condition state is initially set to "false" and that the activation of a condition depends upon the occurrence of a task.

Task Control Structure

Through the use of the property attributes "has condition of" and "has activation of" for tasks and conditions, respectively, a control structure can be constructed that is equivalent to Petri Nets (Holt, 1971; Horning & Randell, 1973; Peterson, 1977). Petri Nets is a digraph formalism that has extensive mathematical treatment that incorporates two complementary ways of viewing a process: as a sequence of operations; or as a sequence of states.

There have been recent extensions of the Petri Net formalism. First, Hackathorn (1976, 1977a) and Meldman & Holt (1971) have used this formalism as a descriptive modeling technique. Secondly, Zisman (1977) used Petri Nets augmented with recursive nets and production rules (Newell & Simon, 1972) to define a specification language for office procedures. Finally, work on Information Control Nets (ICN) of Ellis (1979), Ellis and Nutt (1980), and Cook (1980) has further extended the Petri Nets formalism to include explicitly both control and information flows. A graphical technique has been refined to illustrate ICN's as descriptions of office processes.

In our notation, a task type is equivalent to the Petri Nets notation of transition, and a condition for a task is equivalent to the notion of place. The property attribute "has condition of" is the directed arc connecting a condition (source) to a task (sink), and the property attribute "has activation of" is the directed arc connecting a task (source) to a condition (sink). A task occurs (i.e., is enabled and fired) when all the necessary conditions for the task have the state of "true."

As an example, consider the process of making a sale. First, there must be product awareness on the part of a consumer:

(send literature; has condition of; person is interested)

When the following assertion is true:

(person is interested; has state of; true)

Then the task "send literature" occurs, and product awareness results:

(person is aware of product; has activation of; send literature)

Product awareness is the condition for making a sales pitch:

(make sales pitch; has condition of; person is aware of product)

and the assertion:

(person is aware of product; has state of; true)

will cause the activation of the task "make sales pitch." The occurrence of this task will result in the sales decision:

(sales decision; has activation of; make sales pitch)

This condition enables either one of the following two tasks to occur:

(take order; has condition of; sales decision)

(terminate contact; has condition of; sales decision) depending on the state of the decision. This type of control structure allows for the richness of conflict and resolution inherent in the Petri Nets formalism. The complete diagram is shown in Figure 2.

CONCEPTUAL ISSUES OF TASK REPRESENTATION

The current stage of using the SAN formalism for task representation has only dealt with simple notions of tasks, conditions, and persons. Even at this preliminary stage, several fundamental issues of conceptualizing task representations have surfaced:

- Instances versus occurrences of tasks
- Specialization of task types into means/goal
- Multiple task types for a single task occurrence
- Relation of control structure to task occurrence

Instances Versus Occurrences of Tasks

As mentioned in the definition of the "task" object type, a new assertion "is occurrence of" was defined to distinguish task types from task instances. The problem with this approach is that it creates a new assertion type with little epistemological basis for doing so. The argument is that a task instance denotes the occurrence of an event that occupies a singular point along some time dimension. This distinction is somehow fundamentally different than (John; is instance of; student).

There are two other alternatives for the "occurrence/instance" problem. First, just consider a task instance as a task instance. That is, there is no difference between "(*01; is instance of; answer phone)" and "(John; is instance of; student)." Whatever distinguishes these assertions should be handled as property attributes, such as "has time of" as an attribute of task. Secondly, create a new object type called "event" and remove any connotation of time from the task object type. The object type "event" could then have the property attribute:

(has occurrence of; is part of; event)

(has occurrence of; has value type of; task)

Specialization of Task Types into Means/Goal Hierarchy

From the organizational behavior literature discussed in the third section, it is apparent that specializing tasks into means and goals is a gross simplification. The question is whether the means/goal hierarchy is a useful simplification that can be extended in later work. In this paper, the means/goal hierarchy is used to focus attention on the following aspects:

a) Purposeful direction of action,

b) Mechanism for creating task hierarchies (Sacerdoti, 1977; Pearl, Leal, Saleh, 1981), and

c) Goal definition as justification of past actions, rather than direction of future action.

Multiple Task Types for a Task Type of Occurrence

Expanding on the means/goal hierarchy, consider the following example of a task occurence:

(*02; is occurrence of; make phone call)

(*02; is occurrence of; make sales pitch)

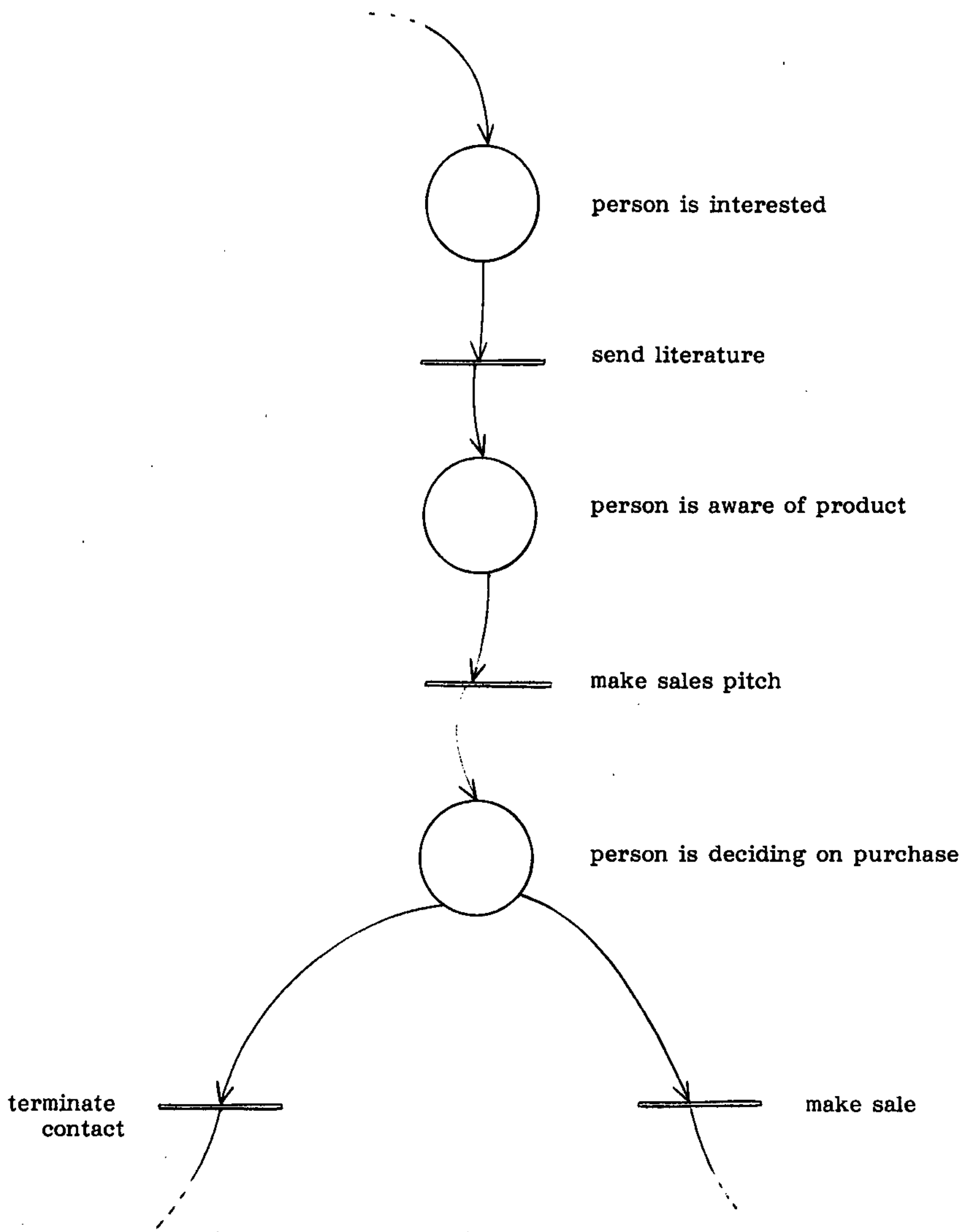


Figure 2. Petri Net Of Sales Contact

(*02; has time of; 15:20 3/20/81)

(*02; has actor of; Fred)

(*02; has object of; John)

In this example, Fred makes a phone call to John for the purpose of selling some product to John. Note that "*02" is the occurrence of two task types, one is a "means" type (i.e., make phone call) and the other is a "goal" type (i.e., make sales pitch). The situation can easily become more complex with multiple means for the same goal or multiple goals for the same means. If the ambiguity that was mentioned by Weick (1969), Cohen, March, and Olsen (1972), and March and Olsen (1976) is credible, then the connection of means and goals within task occurrence will be even more nebulous.

This problem with multiple object types for an object instance is well known to KR researchers. If an object instance (or type) has multiple object types as its generalization, then the rules for property inheritance are ambiguous. This situation is no longer a generalization hierarchy, but a lattice formed from the partial ordering of "is type of" and "is instance of" assertions.

Relation of Control Structure to Task Occurrences

As described in the section on task control structure, the property attributes "has condition of" and "has activation of" form a control structure that is equivalent to that of Petri Nets. The issue then arises as to what is the relation between the task control structure and task occurrences.

Some attributes of Petri Nets are particularly useful in modeling complex systems such as organizations. In particular, the feature of concurrency and non-determinism provide extreme flexibility (Peterson, 1977). Concurrency handles the notion of two or more kinds of independent entities in a system, each of which has events that relate solely to the particular entity. These events can occur independently without the need for synchronization. If, however, synchronization is necessary, this situation is also easily modeled. Non-determinism deals with the fact that in a Petri Net composed of a sequence of discrete events, the order of occurrence of events is one of many allowable sequences. If more than one task can occur at any time, then any of these tasks may occur. The choice of which task will occur is made in a non-deterministic manner (i.e., randomly or by factors that are not modeled). This feature reflects realistic situations where, when several things are happening concurrently, the order of occurrence is not unique.

Several approaches to exploring this relation can be adopted:

a) Simulation of control structure to produce occurrence behavior,

b) Analysis of occurrence behavior to infer control structure, and

c) Consistency of control structure to occurrence behavior.

CONCLUSIONS

This paper has formalized an initial representation for task based on recent work with associative networks. The first part discussed the nature of decision support and, in particular, the inherent dilemma faced by a manager. The next section elaborated on the ambiguity of organizational activity. The fourth section explained the SAN representation and its use for task representation. The fifth section explored several of the conceptual issues that surfaced from the SAN representation when used for task representations.

The contributions of this paper are:

a) An initial task representation built on primitive concepts,

b) Equivalence to Petri Nets for process coordination aspects,

c) Discussion of task instances versus occurrences,

d) Specialization of task types into means/goal hierarchy,

e) Discussion of multiple task types per occurrence, and

f) Relation of task control structure to occurrence behavior.

The issues raised by this paper are fundamental for evolving a practical representation for tasks within a decision support framework. These issues are not artifacts of the SAN representation, but are fundamental to the nature of organizational activity.

A future step is to integrate the concepts and issues of the SAN representation with related (but diverse) research, such as:

- Model Management Systems (Elam, Henderson, & Miller, 1980; Elam & Henderson, 1981; Konsynski, 1981);

- Process Descriptions (Ossorio, 1978; Jeffrey & Putnam, 1980;

- Information Control Networks (ICN) (Ellis, 1979; Ellis & Nutt, 1980; Cook, 1980);

- Planning systems (Sacerdoti, 1977; Hayes-Roth & Hayes-Roth, 1979; Schank & Abelson, 1977; Stefik, 1980);

- Office Specification Languages (Hammer & Kunin, 1980; Nawojski & Konsynski, 1981; Tsichritzis, 1980; Zisman, 1977);

- Networks of commitments (Flores & Ludlow, 1980; Searle, 1969); promises (Lee, 1980), or mental images (Barbarie, 1981) among persons;

- Belief versus Knowledge Systems (Abelson, 1973, 1979).

Finally, a previous paper (Hackathorn, 1981) listed the uses for a task representation:

a) <u>Description</u>. A descriptive statement of how tasks have been performed, such as in a case study.

b) <u>Specification</u>. A normative statement of how the tasks should be performed, such as in a system design.

c) <u>Analysis</u>. Inspection of the task description according to specified criteria for consistency, non-redundancy, etc.

d) <u>Optimization</u>. The permutation of a task description so that certain criteria are maximized.

e) <u>Maintenance</u>. The revision of a task description based on changes in task activity.

f) <u>Prediction</u>. Simulation of a task description into future time periods.

The most important use is the first. At the heart of understanding ambiguous activities within organizations is the ability to describe accurately individual events, along with general patterns. The imposing of organizationally sanctioned activities should be analyzed in the same conceptual context as those activities that emerged from recurring patterns. Therefore, the greatest research problem in decision support is the weak and costly methodologies for describing tasks. Hopefully, this paper has been a contribution toward this goal.

REFERENCES

Abelson, R.P. "The Structure of Belief Systems," R.C. Schank and K.M. Colby, eds., Computer Models of Thought and Language, W.H. Freeman, 1973.

Abelson, R.P. "Differences Between Belief and Knowledge Systems," Cognitive Science, Volume 3, Number 4, October-December 1979, pp. 355-366.

Ackoff, R.L. and Emery, F.E. On Purposeful Systems, Aldine-Atherton, Chicago, Illinois, 1972.

Alter, S.A. "A Study of Computer-aided Decision Making in Organizations," Ph.D. Dissertation, Sloan School, M.I.T., Cambridge, Massachusetts, 1975.

Alter, S.A. "How Effective Managers Use Information Systems," Harvard Business Review, 1976.

Alter, S.A. Decision Support Systems: Current Practices and Future Challenges, Addison-Wesley, New York, New York, 1979.

Barbarie, A.J. "A Decision Support System Based on Mental Representations," Working Paper WP-81-25, International Institute of Applied Systems Analysis, Laxenburg, Austria, February 1981.

Bobrow, D.G. and Winograd, T. "An Overview of KRL, a Knowledge Representation Language," Cognitive Science, Volume 1, Number 1, January 1977.

Bobrow, D.G. and Winograd, T. "KRL: Another Perspective," Cognitive Science, Volume 3, Number 1, January-March 1979, pp. 1-28.

Brachman, R. "A Structural Paradigm for Representing Knowledge," Ph.D. Thesis, Harvard University, Cambridge, Massachusetts, 1977.

Brachman, R. "On the Epistemological Status of Semantic Networks," N.V. Findler, ed., Associative Networks: Representation and Use of Knowledge by Computer, Academic Press, New York, New York, 1979.

Codd, E.F. "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, Volume 13, June 1970, pp. 377-387.

Cohen, M.D., March, J.G., and Olsen, J.P. "A Garbage Can Model of Organizational Choice," Administrative Science Quarterly, Volume 17, Number 1, March 1972, pp. 1-25.

Cook, L. "Streamlining Office Proceedings - An Analysis Using the Information Control Model," Proceedings of National Computer Conference, May 1980, pp. 555-565.

Elam, J.J., Henderson, J.C., and Miller, L.W. "Model Management Systems: An Approach to Decision Support in Complex Organizations," Proceedings of the First International Conference on Information Systems, 1980, pp. 98-110.

Elam, J.J. and Henderson, J.C. "Knowledge Engineering Concepts for Decision Support Systems Design and Implementation," Proceedings of the Hawaii International Conference on System Science, 1981, pp. 639-643.

Ellis, C.A. "Information Control Nets," Proceedings of the ACM Conference on Simulation, Measurement, and Modeling, Boulder, Colorado, August 1979.

Ellis, C.A. and Nutt, G.J. "Office Information Systems and Computer Science," ACM Computing Surveys, Volume 12, Number 1, March 1980, pp. 27-60.

Flores, F. and Ludlow, J.J. "Doing and Speaking in the Office," Decision Support Systems: Issues and Challenges, Proceedings of an International Task Force Meeting, Fick, G. and Sprague, R.H., eds., June 1980, p. 95.

Galbraith, J.R. Designing Complex Organizations, Addison-Wesley, Reading, Massachusetts, 1973.

Galbraith, J.R. Organizational Design, Addison-Wesley, Reading, Massachusetts, 1977.

Gorry, G.A. and Scott Morton, M.S. "A

Framework for Management Information Systems," Sloan Management Review, Volume 13, Number 1, pp. 55-77, Fall 1971.

Hackathorn, R.D. "Activity Analysis: A Methodology for the Discrete Process Modeling of Information Systems in Organizations," Ph.D. Dissertation, University of California, Irvine, California, 1976.

Hackathorn, R.D. "Modeling Unstructured Decision Making," Proceedings of the Conference on Decision Support Systems, Data Base, Volume 8, Number 3, pp. 41-42, Winter 1977 (a).

Hackathorn, R.D. "Task Representations and Management of Task Descriptions. An Approach to Decision Support, Proceedings of the 1981 Hawaii International System Science Conference, January 1981.

Hammer, M. and Kunin, J.S. "Design Principles of an Office Specification Language," Proceedings of National Computer Conference, May 1980, pp. 541-547.

Hayes-Roth, B. and Hayes-Roth, F. "A Cognitive Model of Planning," Cognitive Science, Volume 3, Number 4, October-December 1979, pp. 275-310.

Holt, A.W. "Introduction to Occurrence Systems," E.L. Jacks, ed., Associative Information Techniques, American Elsevier, New York, New York, 1971.

Horning, J.J. and Randell, B. "Process Structuring," ACM Computing Surveys, Volume 5, Number 1, July 1973, pp. 5-30.

Jeffrey, H.J. and Putman, A.O. "The MENTOR Project--Replicating and the Functioning of an Organization," Presented at the Annual Meeting of the Society for Descriptive Psychology, August 1980.

Keen, P.G.W. and Scott Morton, M.S. Decision Support Systems, Addison-Wesley, Reading, Massachusetts, 1976.

Keen, P.G.W. "Decision Support Systems: A Research Perspective," G. Fick and Sprague, R.H., eds., Decision Support Systems: Issues and Challenges, Proceedings of an International Task Force Meeting, June 23-25, 1980, p. 23.

Konsynski, B. "On the Structure of a Generalized Model Management System," Proceedings of the Hawaii International Conference on System Sciences, 1981, pp. 630-638.

Lee, R.M. "CANDID: A Logical Calculus for Describing Financial Contracts," Ph.D. Dissertation, University of Pennsylvania, Philadelphia, Pennsylvania, 1980.

Levesque, H. and Mylopoulos, J. "A Procedural Semantics for Semantic Networks," Findler, N.V., ed., Associative Networks: Representation and Use of Knowledge by Computer, Academic Press, 1979.

March, J.G. and Olsen, J.P. Ambiguity and Choice in Organizations, Universitetsforlaget, Bergen, Norway, 1976.

Meldman, J.A. and Holt, A.W. "Petri Nets and Legal Systems," Jurimetric Journal, Volume 12, Number 2, December 1971.

Micro Decisionware. "Micro-SEED Reference Manual and CP/M Operating Guide," Micro Decisionware, Boulder, Colorado, 1981.

Minsky, M. "A Framework for Representing Knowledge," The Psychology of Computer Vision, P. Winston, ed., McGraw-Hill, New York, New York, pp. 211-277, 1975.

Mintzberg, H. The Nature of Managerial Work, Harper and Row, New York, New York, 1973.

Mintzberg, H., Raisinghani, D., and Theoret, A. "The Structure of 'Unstructured' Decision Making," Administrative Science Quarterly, Volume 21, June 1976, pp. 246-275.

Mylopoulos, J. "An Overview of Knowledge Representation," Proceedings of the Workshop on Data Abstraction, Databases and Conceptual Modelling. SIGART Newsletter, Number 74, January 1981, pp. 5-12.

Nawojski, C. and Konsynski, B. "A Computer Aid for Office Automation Design," Proceedings of the Hawaii International Conference on System Sciences, 1981, pp. 690-698.

Newell, A. and Simon, J.A. Human Problem Solving, Prentice-Hall, Englewood Cliffs, New Jersey, 1972, p. 808.

Nilsson, N.S. Principles of Artificial Intelligence, Tioga Publishing Company, Palo Alto, California, 1980.

Ossorio, P.G. What Actually Happens: The Presentation of Real World Phenomena in Behavioral Science, University of South Carolina Press, Columbia, South Carolina, 1978.

Payne, J.W. "Task Complexity and Contingent Processing in Decision Making: An Information Search and Protocol Analysis," Organizational Behavior and Human Performance, Volume 16, 1976.

Pearl, J., Leal, A., and Saleh, J. "GODDESS: A Goal-Directed Decision Structuring System," Proceedings of the Hawaii International Conference on System Sciences, January 1981, pp. 546-558.

Peterson, J.L. "Petri Nets," ACM Computing Surveys, Volume 9, Number 3, September 1977, pp. 223-252.

Quillian, M.R. "Semantic Memory," Semantic Information Processing, M. Minsky, ed., MIT Press, Cambridge, Massachusetts, pp. 227-270, 1968.

Quillian, M.R. "The Teachable Language Comprehender: A Simulation Program and Theory of Language," Communications of the ACM, Volume 12, Number 8, pp. 459-476, 1969.

Sacerdoti, E. A Structure for Plans and Behavior, Elsevier, New York, New York, 1977.

Schank, R. and Abelson, R. Scripts, Plans, Goals, and Understanding, Laurence Erlbaum Associates, Hillsdale, New Jersey, 1977.

Scott Morton, M.S. "Management Decision Systems: Computer Based Support for Decision Making," Ph.D. Dissertation, Sloan School of Management, Cambridge, Massachusetts, 1971.

Searle, J.R. Speech Acts: An Essay in the Philosophy of Language, Cambridge University Press, London, England, 1969.

Simon, H.A. The Sciences of the Artificial, MIT Press, Cambridge, Massachusetts, 1969.

Simon, J.A. "Administrative Behavior," Third Edition, Free Press, New York, New York, 1976.

Smith, J.M. and Smith, D.C.P. "Database Abstractions: Aggregation and Generalization," ACM Transactions on Database Systems, Volume 2, Number 2, June 1977, pp. 105-133.

Sprague, R.H. "A Framework for Research on Decision Support Systems," G. Fick and R.H. Sprague, eds., Decision Support Systems: Issues and Challenges, Proceedings of an International Task Force Meeting, June 23- 25, 1980a, p. 5.

Sprague, R.H. "A Framework for the Development of Decision Support Systems," MIS Quarterly, Volume 4, Number 4, December 1980b, p. 1-26.

Stabell, C.B. "On the Development of the Decision Support Systems as a Marketing Problem," Paper Presented at the International Federation for Information Processing Congress, Stockholm, Sweden, August 1974.

Stabell, C.B. "Decision Research: Description and Diagnosis of Decision Making in Organizations," D. Heradstreit and O. Narvesen, eds., Decision Making Research: Some Developments, Norsk Utenriks Politisck Institute, Oslo, Norway, 1977.

Stefik, M.J. "Planning with Constraints," Ph.D. Dissertation, Department of Computer Science, Stanford University, Stanford, California, 1980.

Teichroew, D. "Problem Statement Languages in MIS," Couger, J.D. and Knapp, R.W., eds., System Analysis Techniques, Wiley, New York, New York, 1974a, pp. 310-327.

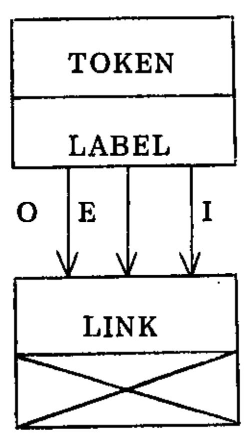
Teichroew, D. "Problem Statement Analysis: Requirements for the Problem Statement Analyzer," Couger, J.D. and Knapp, R.W., eds, System Analysis

Techniques, Wiley, New York, New York, 1974b, pp. 336-358.
Tsichritzis, D. "OFS: An Integrated Form Management System," Proceedings of Very Large Data Base Conference, October 1980, pp. 161-166.
Weick, K.E. Social Psychology of Organizing, Addison-Wesley, Reading, Massachusetts, 1969.
Woods, W.A. "What's in a Link?" Foundations for Semantic Networks," Representation and Understanding, Bobrow, D.G. and Collins, A.M., eds., Academic Press, New York, New York, pp. 35-82, 1975.
Zisman, M. "Representation, Specification, and Automation of Office Procedures," Ph.D. Dissertation, Wharton School, University of Pennsylvania, Philadelphia, Pennsylvania, 1977.

APPENDIX A. SIMPLE ASSOCIATIVE NETWORK EDITOR (SANE)

In the early stages of this work, it became clear that manual manipulation of any non-trivial AN was infeasible. Since the intent was to explore the implications of AN representations of tasks, a tool was constructed whose primary function was editing these networks. The editor for SAN, called SANE, was developed using the Micro-SEED CODASYL database management system (Micro Decisionware, 1981). The SANE program operates on 8080/Z80 microcomputers under the CP/M operating system. Given the limitations of 500K bytes for floppy disk storage, AN databases composed of 20,000 tokens and links can be reasonably manipulated. The interaction with SANE is in the tradition of LISP editors, with cursor control to the CRT terminal.

The SANE database uses a recursive schema definition similar to the typical bill-of-materials structure:



The two record types TOKEN and LINK are related through three set types O, E, and I that are the roles of out-node, edge, and in-node, respectively. The only data item is LABEL in the TOKEN record type.

APPENDIX B. LISTING OF ASSERTIONS

CLASS	is type of	OBJECT
RELATION	is type of	OBJECT
VALUE	is type of	OBJECT
has default of	is part of	OBJECT
ASSERTION	is type of	RELATION
PROPERTY	is type of	RELATION
is part of	is instance of	PROPERTY
is instance of	is instance of	ASSERTION
is type of	is instance of	ASSERTION
has domain of	is part of	ASSERTION
has range of	is part of	ASSERTION
has domain of	has default of	CLASS
has range of	has default of	CLASS
has value type of	is part of	PROPERTY
has value type of	has default of	VALUE
TRUTH VALUE	is type of	VALUE
true	is instance of	TRUTH VALUE
false	is instance of	TRUTH VALUE
person	is type of	CLASS
task	is type of	CLASS
condition	is type of	CLASS
has time of	is part of	task
has actor of	is part of	task
has condition of	is part of	task
has time of	has value type of	TIME VALUE
has actor of	has value type of	person
has condition of	has value type of	condition
is occurrence of	is instance of	ASSERTION
has state of	is part of	condition
has state of	has range of	TRUTH VALUE
has state of	has default of	false
has activation of	is part of	condition
has activation of	has value type of	task