MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 719 April, 1983

SEMANTIC SUPPORT FOR WORK IN ORGANIZATIONS

Gerald Barber[1], Peter de Jong, Carl Hewitt

Present day computer systems cannot implement much of the work carried out in organizations such as: planning, decision making, analysis, and dealing with unanticipated situations. Such organizational activities have traditionally been considered too unstructured to be suitable for automation by computer. We are working on the development of computer technology to overcome these limitations. Our goal is the development of a computer system which is capable of the following: describing the semantics of applications as well as the structure of the organization carrying out the work, aiding workers in carrying out the applications using these descriptions, and acquiring these capabilities in the course of the daily work through a process which is analogous to apprenticeship.

This paper describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology and at Institute National de Recherche en Informatique et en Automatique, Le Chesnay, France. Major support for the research at MIT was provided by the System Development Foundation. Additional support was provided by IBM and by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N0014-80-C-0505.

---

[1]Currently at INRIA, Le Chesnay, France

1. Introduction

Organizational work is performed in offices, laboratories, warehouses, production lines, etc. The workers spend a great deal of their time in problem solving activities dealing with unanticipated situations. Many of the purely routine activities have been automated and therefore no longer are performed by the workers. Providing aid for organizational planning and problem solving activities requires computer systems quite different from those currently available. New kinds of computer systems are needed that *know* the goals of the organization and its applications. We are developing such a system. At the heart of the system is a description language in which the structure, goals, and applications of an organizations can be described. This is a semantic description. It can be used to generate the applications for automation of some organizational functions, and it can be used to assist in organizational problem solving activity.

2. Open Systems

An organization is a type of system which is open-ended and incremental--undergoing continual evolution [Hewitt, de Jong 83]. In open systems it becomes very difficult to determine what objects exist at any point in time. For example a query which looks for all the purchase orders might never finish, since new purchase orders are being continuously created in various parts of the organization. The notion of completeness is hard to achieve in general. Usually this problem is solved in data base systems by declaring that the items in the data base are all the items which exist, this is called the "closed world assumption" [Reiter 81]. In a very distributed environment like an organization, all that can be achieved are islands of relatively complete knowledge. In our description system, repositories serve this function. A query to a repository such as the personnel employed in a branch office will give an answer which is accurate as to some period in the past. Repositories are relatively expensive to maintain. In general a search for information in an organization will return an adequate answer, rather than the best answer. This is known as the principle of statisficing [March, Simon 63]. For example when attempting to purchase an item, it is usually sufficient to find the item at a reasonable price rather than at the lowest price. The organization might have guide lines which act as constraints on the purchaser in order to define what *reasonable* might be. Descriptions which are not detailed enough to specify in an algorithmic way how an application is to be performed, can act as constraints on that application when it is performed manually by a person. Management plans and rules are sometime formulated as constraints.

An organization is continually undergoing change: procedures change, the goals of the organization change,and the workers develop individual, distinctive styles as they become more skilled at performing their work. Therefore, it is not practical to completely separate the describing of an application from the performing of an application. At this time the technology does not exist for a semantic support system to

change its descriptions, by itself, to better match the requirements of the organization. But the semantic support system can operate as an apprentice to a worker. In this way a system could initially be delivered to a worker with a base of knowledge about the general nature of the work to be done. In the course of the work the system would acquire more knowledge about the worker's style of performing the application. The new knowledge is then related to the existing knowledge in an incremental manner [Barber 82].

In an open system there are no global objects. The only thing that the various subsystems have in common is the ability to communicate. If two different subsystems, independently developed need to cooperate, then they must know enough about each other in order to process each others messages. In the general case, the subsystems will have to *negotiate.* In a purchasing application, one organization's purchasing procedure will need to negotiate with another organization's order-entry procedure. Order-entry procedures across organizations, although similar in function, are different in details. For example, the purchasing organization negotiates with the vendor's order-entry subsystem to determine how to place the order and to agree on a price.

3. The Information-Action Models

Information-Action models are an active research area today for computerization of organizational work. They focus on the information used in office work and the actions performed on the information. Information-action models are concerned with issues such as: what action is done, when it is done, what information the action needs as input, and what information the action produces as outputs. In addition the routing or flow of information is important. Office work is thus characterized by actions along with their decision criteria.

Examples of forms flow models that focus on structuring information and routing it through an organization are: the Form Manipulation System [Tsichritzis 79], Officetalk [Ellis, Nutt 80], and the System for Business Automation [de Jong 80]. These systems include the routing of information to and from databases as well as between office workers.

Information Control Networks [Ellis 79] can be used to specify the flow of information between repositories and actions. The ICN work focuses on a precise definition of information flow and can be used as an analysis tool and a tool to guide restructuring of information flow within the office.

The Office Procedure Specification Language [Zisman 77] uses Petri nets augmented with production system rules to describe relevant actions when information arrives at a particular work station. The production rules specify what action is to be taken with incoming information, including sending information to other work stations.

The Office Specification Language [Kunin 82] concentrates on being able to express both the structure of procedures and the structure of data in a semi-formal language. OSL uses the idea of objects in the operation of office procedures. Each procedure is composed of attributes such as a focal object to be operated on, a main line description of the procedure, variations and exceptions, and timing constraints. The TAXIS system uses semantic nets composed of organizational objects to describes both the structure and procedures which comprise an organization [Borgida, Mylopoulos, Wong 82].

In all these systems information is treated as something on which office actions operate producing information that is passed on for further actions or is stored in repositories for later retrieval. These types of systems are suitable for describing office work that is structured around actions (e.g. sending a message, approving, filing); where the sequence of activities is the same except for minor variations and few exceptions. None of the above systems explicitly describe the goals of the office work and how each action is related to the accomplishment of the overall goal of the work. Thus it becomes difficult to describe work where the goal can be achieved via several different methods or where the actions necessary to accomplish the goal cannot be known ahead of time. These systems do not deal well with unanticipated conditions.

4. Knowledge, Goals, and Problem Solving

The *goals* of the organizational work provide a basis for dealing with work that does not fit within the information-action models. A goal presents a problem to be solved and can be decomposed into subgoals. This decomposition step simplifies the goals. In some cases some of the subgoals may be delegated as subtasks to other individuals of the organization. The problem decomposition process is repeated until subgoals are reached that may be easily attained by the workers. Often the process of decomposing a problem into subproblems is not easy and requires intimate knowledge of the particular instance at hand.

As an example consider a sales manager's work. The manager's goal is to sell products; in order to accomplish this goal his task is broken down into subgoals such as overseeing the activities of a group of sales persons and selling products to large customers and accounts. Thus part of the manager's work is delegated and part is handled by the manager. Because large sales orders are complicated and not made frequently, the manager breaks down such sales into subprocesses such as information gathering, negotiation with customers, issuing quotations, and finalizing sales. Each of these subprocesses may be achieved in different manners for different customers. Some customers may prefer to negotiate in a relaxed manner over meals, while others prefer the morning when they are most alert.

The Information-action model does not do a good job of describing how a sales manager performs his work. Each situation is unique and dependent on the customer. A Semantic Support System can remind the sales manager that a customer hasn't been informed of a new product, or that certain aspects of a contract

have not been finalized. A Semantic Support System should function in a manner analogous to an apprentice: reminding, suggesting, and in some cases performing the task.

There is a vast literature concerning organizational goal structures centered around the behavioral theory of the firm. A seminal book in this area which combines a view of the organization with the early work on Artificial Intelligence is [March, Simon 63]. More recently, Lucy Suchman has looked at organizations from an anthropological point of view [Suchman 79]. Further research into organizational goal structures and human problem solving in organizations is important to the development of Semantic Support Systems. Other work in this area includes: [Wynn 79], [Goldstein 80], and [Fikes, Henderson 80].

5. Describing an Application

To illustrate the use of a Semantic Support System, we will describe a purchasing application. In this application (see figure 5-1) there is a Purchasing department, an Accounts payable department, a shipping department, a Purchaser, and a Vendor. The Purchasing department contains buyers who fill out purchase orders when a Purchase request form is received. A copy of the purchase order is sent to a Vendor, and the Accounts Payable department. A receiving ticket is sent to the Shipping and receiving department. The Vendor sends an Invoice to the Accounts payable department, and the item purchased to the shipping department. The shipping department matches the item received with the receiving ticket, sends the item to the person who ordered it, and a notification of receipt to the Purchasing department and Accounts payable department.

The complete application will not be described, only enough to illustrate the use of descriptions. Also an end-user interface needs to be constructed to make it easier for the application to be specified by the worker [de Jong, Byrd 80]. In this paper we concentrate on describing the kinds of knowledge structures which need to exist in the computer system.

Below we show an instance description with concept `Buyer`, with attribute relation `responsibility`, and with attribute filler `Computers`.

```
(a Buyer (with responsibility Computers))
```

A particular buyer in purchasing may be described with the is statement shown below.

```
(Johnson is (a buyer (with responsibility Chemicals)))
```

The above description states that Johnson is a buyer whose responsibility is chemicals. We could further state that:

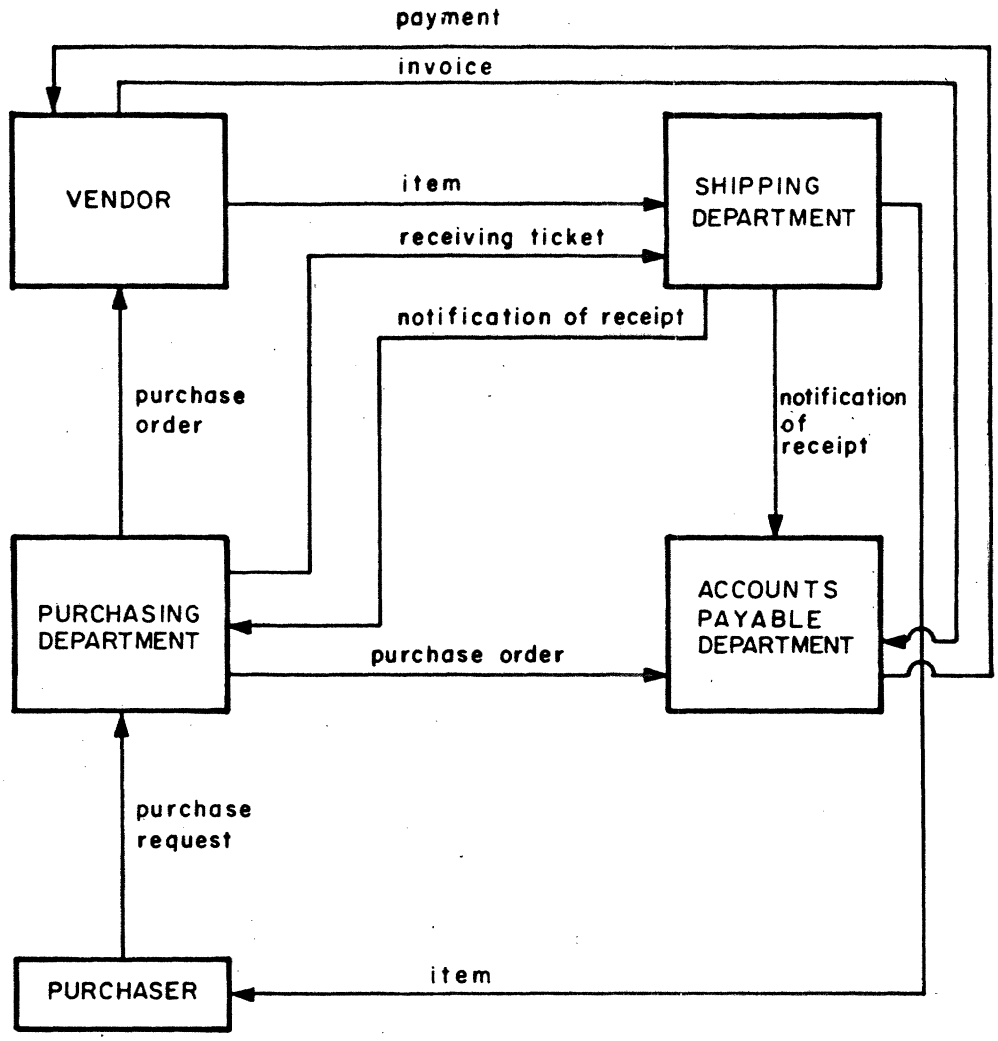

Figure 5-1: Purchase Procedure

```
(Smith is (a buyer (with responsibility Chemicals)))
```

Now both Smith and Johnson are buyers with the responsibility Chemicals. If we now state the following:

```
((a buyer) is (an employee))
```

then both Johnson and Smith would inherit from the description an employee. Descriptions have well defined inference rules which specify how descriptions acquired in an incremental manner relate to each other [Attardi, Simi 81].

For the purchasing application, we will first describe part of the organization of the purchasing

department:

```
(a Purchasing-department

  (with personnel-roles
    {(a buyer (with responsibility Computers))
     (a buyer (with responsibility Chemicals))})

  (with files
    {closed-purchase-orders
     outstanding-purchase-orders}))
```

The above description describes a purchasing department with two personnel roles and two files. The "{...}" notation is used to indicate sets. In this case the set of files contains `closed-purchase-orders` and `outstanding-purchase-orders`.

In a like manner the rest of the organization of the purchasing department and the other departments can be described. The behavior of the department is described in terms of the communications it receives, the communications it sends, the changes it makes to its files, and the objects it creates. For example a part of the behavior of the Purchasing Department when it receives a purchase request is described:

```
(a Purchasing-department

  (with handler
    (a request-handler

      (with message-received (a purchase-request))

      (with messages-sent
        {(a communication
           (with target (a vendor))
           (with message (a purchase-order))

         (a communication
           (with target (a shipping-dept))
           (with message (a receiving-form)))

         (a communication
           (with target (an accounts-pay-dept))
           (with message (a purchase-order)))})))))
```

More detail could be added to the above behavior. If enough detail is added the support system could perform that task; although, in some cases, even if the system knows enough to perform the task itself, it may be necessary to gain approval for reasons of accountability. Suppose that added to the above description is the following behavior -- when a purchase request is received, the buyer responsible for that category of purchase receives the request:

```
((a purchasing-department
   (with personnel-roles (a buyer (with responsibility =r)))

is

 (a purchasing-department
   (with handler
     (a request-handler
       (with message-received (a purchase-request (with type =r)))
       (with message-sent
         (a communication
           (with target (a buyer (with responsibility =r)
           (with message (a purchase-request)))))))))
```

The notation "=r" is used to indicate a variable. The above description states that if a buyer has a certain responsibility then he or she will receive purchase requests for items of the same type as their responsibility. At this point we can describe the behavior of a buyer when a purchase request is received. Since the buyer is part of the purchasing department, his or her behavior will be constrained to conform to the behavior already specified for this role in the purchasing department. If additional behavior is not specified, the buyer will get the purchase request by electronic mail, at which point he or she can manually create the purchase order. Once the purchase order is sent to a vendor by the buyer, the purchase order will be routed to the correct vendor by the behavior specified for the purchasing department.

Over a period of time the purchasing application will grow and change. As the purchasing application changes, so must the descriptions in the system. In this way organizations can be incrementally described and evolved.

6. Problem Solving

One of the main goals of the work we are doing is to create a system which will not only support normal activity, but also help in the problem solving necessary when unanticipated things happen. The descriptions with which an application is described can be used not only to support or perform the work, but also to provide the information necessary to help solve problems.

As an example, suppose that an item was ordered and estimated to arrive within two weeks. After two weeks, the person who ordered the item notifies the buyer that the item has not arrived. To initiate the finding of the item, the buyer, or computer, disseminates the following goal:

```
(disseminate
  (Goal (=1 is (a location (of item PO-64)))))
```

The system will then attempt to find the location of the ordered item. Disseminating a goal sends a message

throughout the system. Within the system are objects called *sprites*. A sprite will trigger when a disseminated message matches the message pattern of the sprite. The when clause states the message pattern on which the sprite will trigger. In this example, the sprite below will trigger on the message disseminated above. The sprite will then attempt to find the item by disseminating additional goals. In this case the sprite will try to establish whether the item was shipped by the vendor, at the same time it will try to establish the negative of the goal. A standard technique used with sprites is to have *skeptic* sprites try to show the impossibility of achieving goals that other sprites are trying to find. When a sprite establishes a goal or its negative, it disseminates an assertion. The assertion usually triggers the sprite that initially disseminated the goal. In this example, the sprite, after disseminating the goals `(shipped-by-vendor)` or `(not(shipped-by-vendor)`, is prepared to receive the assertions of the same name. If it receives the `(shipped-by-vendor)` assertion, it will disseminate the new goals `(delivered-to-dock)` and `(not(delivered-to-dock))`. If it receives the `(not(shipped-by-vendor))` assertion, it will disseminate the assertion `(=1 is (an at-vendor))`.

```
(when
  (goal (=1 is (a location (of item =PO))))

    (disseminate
      (goal (a shipped-by-vendor (with item =PO))))

    (disseminate
      (goal (not(a shipped-by-vendor (with item =PO)))))

    (when
      (assertion (a shipped-by-vendor (with item =PO)))

        (disseminate
          (goal (a delivered-to-dock (with item =PO))))

        (disseminate
          (goal (not(a delivered-to-dock (with item =PO)))))

        (when
          (assertion (not(a delivered-to-dock (with item =PO))))

            (disseminate
              (assertion (=1 is (a in-transit (with item =PO)))))))

    (when
      (assertion (not(a shipped-by-vendor (with item =PO))))

        (disseminate
          (assertion (=1 is (an at-vendor (with item =PO)))))))
```

The sprites can be programmed to act in a manner appropriate for the goal it is pursuing. If a sprite is

trying to determine if an item is on the shipping dock, it would ask the shipping department whether the item is there. In other cases it can examine the appropriate files in one of the many description data bases spread throughout the organization. The standard way of using sprites is to continue to break down the goals into subgoals until a description is found which matches a subgoal. Once a subgoal is achieved and the fact asserted, other sprites pick up this assertion, and assert that their subgoals have been achieved. Eventually the original goal is given the information it was seeking.

The more information a system has about an organization, the better job it can do in supporting problem solving procedures. If the computer system can't solve the problem, it might help the worker by generating hypotheses on what caused the problem. Once a hypothesis is shown to be the correct description of a problem, the system can either solve the problem, or guide the worker in solving the problem.

7. Conclusion

We have argued that much of the work which has heretofore been considered unstructured and thus difficult to computerize can be dealt with in terms of goals, problem solving, and knowledge of the relation between actions and goals. We propose that a system that has access to a description of this teleology can more effectively support workers in activities such as management, dealing with unanticipated situations, and planning.

The members of the Message Passing Semantics Group have participated in the implementation of many *separate* systems to deal with different aspects of the issues involved in Semantic Support Systems. Actor systems are systems with mathematical semantics, serializers, parallelism, independence of machine boundaries, unification of procedural and object-oriented languages [Lieberman 81], [Theriault 82], [Hewitt, de Jong 83]. Omega is a system with inheritance, lattice of descriptions, instance descriptions, attributions, sprite attachments, deduction, model theory OMEGA-80 [Attardi, Barber, Simi 80], OMEGA-82 [Barber 82]. Ether is a system with goals, assertions, inferencing, sprites, skeptics, based on scientific community metaphor [Kornfeld 82], [Kornfeld, Hewitt 81]. The System for Business Automation(SBA) is a system with description of organizational structure and objects, forms flow, end-user interface, triggering of objects, and the integration of a relational data base with the SBA language [de Jong 80], [Byrd, Smith, de Jong 82]. We are currently constructing a system which *unifies* these technologies.

Acknowledgments

The ideas presented in this paper have developed over a period of several years. We have particularly benefited from discussions which took place at the Office Semantics Workshop [Barber 81] and the September 1980 IEEE Workshop on Office Information Systems at Lake Arrow Head. In addition we have been strongly influenced by the work of Skip Ellis, Richard Fikes, Ira Goldstein, Austin Henderson, Lucy Suchman, and Eleanor Wynn at Xerox Parc.

This paper describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology and at Institute National de Recherche en Informatique et en Automatique, Le Chesnay, France. Major support for the research at MIT was provided by the System Development Foundation. Additional support was provided by IBM and by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N0014-80-C-0505. We would like to thank Najah Naffah, Charles Smith, and Patrick Winston for their support and encouragement.

References

[Attardi, Barber, Simi 80]
Attardi, G., Barber, G. and Simi, M.
Towards an Integrated Office Work Station.
Automazione e Strumentazione (3), March, 1980.

[Attardi, Simi 81]
Attardi, G. and Simi, M.
Semantics of Inheritance and Attributions in the Description System Omega.
In *Proceedings of IJCAI 81.* IJCAI, Vancouver, B. C., Canada, August, 1981.

[Barber 81]
Barber, G. R.
Record of the Workshop on Research in Office Semantics.
AI Memo 602, MIT, March, 1981.

[Barber 82]
Barber, G. R.
Office Semantics.
PhD thesis, Massachusetts Institute of Technology, 1982.

[Borgida, Mylopoulos, Wong 82]
Borgida, A., Mylopoulos, J. L., Wong, H. K. T.
Generalization as a Basis for Software Specification.
In Brodie, M. L., Mylopoulos, J. L., Schmidt, J. W., editor, *Perspectives on Conceptual Modeling.*
Springer-Verlag, 1982.

[Byrd, Smith, de Jong 82]
Byrd, R. J., Smith, S. E., de Jong, S. P.
An Actor-Based Programming System.
In *Conference on Office Information Systems.* ACM SIGOA, June, 1982.

[de Jong 80]
de Jong, S. P.
The System for Business Automation(SBA): A Unified Application Development System.
In *Proceedings of the 1980 IFIP Congress.* IFIP, Tokyo, 1980.

[de Jong, Byrd 80]
de Jong, S. P. and Byrd R. J.
Intelligent Forms Creation in the System for Business Automation.
IBM Research Report RC 8529, IBM, 1980.

[Ellis 79]
Ellis, C. A.
Information Control Nets: A Mathematical Model of Office Information Flow.
In *Proceedings of the Conference on Simulation, Modeling and Measurement of Computer Systems,*
pages 225-240. ACM, August, 1979.

[Ellis, Nutt 80]
Ellis, C. A. and Nutt, G. J.
Computer Science and Office information Systems.
Computing Surveys 12(1), March, 1980.

[Fikes, Henderson 80]
Fikes, R. E. and Henderson, D. A.
On Supporting the Use of Procedures in Office Work.
In *Proceedings of the First Annual AAAI Conference.* American Association for Artificial Intelligence, August, 1980.

[Goldstein 80]
Goldstein, Ira.
PIE: A Network-Based Personal Information Environment.
In *Proceedings of the First Annual AAAI Conference.* American Association for Artificial Intelligence, August, 1980.

[Hewitt, de Jong 83]
Hewitt, C., de Jong, P.
Open Systems.
In Brodie, M. L., Mylopoulos, J. L., Schmidt, J. W., editor, *Perspectives on Conceptual Modeling.* Springer-Verlag, 1983.

[Kornfeld 82]
Kornfeld, W.
Concepts in Parallel Problem Solving.
PhD thesis, Massachusetts Institute of Technology, 1982.

[Kornfeld, Hewitt 81]
Kornfeld, W. A. and Hewitt, C.
The Scientific Community Metaphor.
IEEE Transactions on Systems, Man, and Cybernetics SMC-11(1), January, 1981.

[Kunin 82]
Kunin, J. S.
Analysis and Specification of Office Procedures.
Technical Report 275, MIT Laboratory for Computer Science, February, 1982.

[Lieberman 81]
Lieberman, H.
A Preview of Act-1.
A.I. Memo 625, MIT Artificial Intelligence Laboratory, 1981.

[March, Simon 63]
March J. G. and Simon H. A.
Organizations.
John Wiley & Sons, Inc., 1963.

[Reiter 81]
Reiter, R.
On closed world data bases.
In Gallaire H., and Minker, J., editor, *Logic and data bases.* Plenum Press, 1981.

[Suchman 79]
Suchman, L.
Office Procedures as Practical Action: A Case Study.
Technical Report, XEROX PARC, September, 1979.

[Theriault 82]
Theriault, D.
A Primer for the Act-1 Language.
A.I. Memo 672, MIT Artificial Intelligence Laboratory, April, 1982.

[Tsichritzis 79]
Tsichritzis, D.
A Form Manipulation System.
In *Proceedings of the NYU Symposium on Automated Office Systems.* NYU, 1979.

[Wynn 79]
Wynn, E.
Office Conversation as an Information Medium.
PhD thesis, Department of Anthropology, University of California, Berkeley, 1979.

[Zisman 77]
Zisman, M. D.
Representation, Specification and Automation of Office Procedures.
PhD thesis, Wharton School, University of Pennsylvania, 1977.