DECISION SUPPORT USING OFFICE PROCEDURES

W. Bruce Croft
Lawrence S. Lefkowitz

Computer and Information Science Department
University of Massachusetts
Amherst, MA. 01003

Technical Report 83-33

ABSTRACT

Office systems that allow the specification and support of office procedures can be regarded as decision support systems. Using procedures, a system can be configured to make information and tools available according to individual or organizational preferences. The system will also be aware of when decisions should be made and the context within which they are made. In this paper, we describe the POISE system and how it uses office procedures to provide intelligent assistance to the users. POISE is able to both recognize a user's actions in the context of the office procedures and to plan a sequence of actions given a user-specified goal. Two examples of the way POISE can be used are described.

This research was supported by a Digital Equipment Corporation External Research Grant.

INTRODUCTION

Decision Support Systems (DSSs) have been characterized in many different ways. Factors that have been considered in producing definitions and taxonomies include the nature of the problems tackled by the system, the functionality of the system and the intended users of the system (see, for example, Keen [KEEN78]). For the purposes of this paper, the following DSS features are particularly relevant (taken from [KEEN78],[SPRA82]):

a. A DSS is designed to aid in decision making and implementation.

b. It should increase the effectiveness of individuals or organizations.

c. It should be able to deal with the less well-structured, underspecified problems that managers typically face. This is to be contrasted with the extremely structured, routine, mass-volume tasks such as preparing payrolls or customer invoices.

d. It should also have a user-friendly interface and be able to adapt to changes in the environment.

In this paper we shall describe a system, based on descriptions of office procedures, that can exhibit these characteristics. Office procedure formalisms have been used as a means of analysing offices ([ELLI80],[LADD80]) and for automating office tasks ([ZISM77]). These formalisms typically describe the major functions or tasks in an office in terms of the information used, transformations on that information and its flow through the office. The roles of the people involved in the tasks are sometimes included in the description. A common feature of the formalisms is that they make the decision points in an office task explicit. In fact, they indicate not only when a decision is needed, but also what the environment will be for making the decision. That is, factors such as the information available, time constraints, the people involved and the effects of a decision can all be specified, to some extent, in an office procedure. What is not specified is the method of actually making the decision. A system which uses office procedures could, therefore, provide a large degree of support to people making decisions as well as automating some simple tasks. To what extent, however, could office procedures support the decision making of upper-level managers?

People at all levels of a business make decisions. The people carrying out the clerical level tasks are dealing with more structured tasks than those dealt with by the management. This means that a large part of these tasks can be tackled algorithmically. A task which is extremely structured (such as the payroll preparation) can be solved using algorithms specified in application programs. As tasks become less structured, they involve more problem-solving and decision making. Office procedures can be used to specify typical or recommended ways of handling these semi-structured tasks. In fact, the tasks are structured by the specification of the procedures. However, even at this level, a system that uses procedures must incorporate a large degree of flexibility. Alternative ways of accomplishing the goals of the tasks must be permitted, recognized and perhaps even suggested.

Many of the "unstructured" management tasks could be handled by the same mechanisms. A means would be provided for users (the managers) to specify the typical method they use for tackling a problem or making a decision. This may involve steps such as obtaining certain information, displaying it in different ways and using various software tools to manipulate it. This "personal" office procedure would then be used to provide the appropriate environment when the same situation is encountered or when a user unfamiliar with the task needs suggestions. A system that uses office procedures specified by experienced managers is similar in concept to a simple "expert" system. A crucial feature of such a system is that the method of defining procedures must be very easy to use.

Fikes [FIKE80] and Barber [BARB83] have proposed the use of artificial intelligence techniques to handle the problem-solving aspects of office work. The work described here attempts to combine these techniques with a strong emphasis on the procedural nature of many of the tasks.

In the remainder of this paper we shall describe the POISE system developed at the University of Massachusetts and indicate how it could be used for decision support. POISE is designed to act as an intelligent assistant to the users of an office system. The office system is viewed as a collection of tools, some very general (such as an electronic mail facility) and some more specific (for example, a tool that provides simulations using accounting models or an expert system that gives advice on sales strategies). The office tasks are specified as a hierarchy of procedure descriptions. The procedure descriptions specify the typical steps involved in completing the task and the tool invocations which correspond to those steps. The goals of the procedures are also specified. The ability to combine recognition of user actions using procedure descriptions, and planning using the descriptions and goals gives POISE great flexibility in the type of assistance it can provide. The following is a list of POISE's main capabilities:

a. *Planning used to propose actions.* By using the goals and sequences of actions specified in the procedure descriptions, POISE can describe alternative courses of action to a user who is uncertain of how to carry out a task. It can also provide default values for incompletely specified actions.

b. *Planning used for task automation.* Whereas some procedures require human interaction, other tool instantiations in procedures can be invoked by the system.

c. *Propagating constraints to correct local and global errors.* Specific user actions apply constraints to the general procedure descriptions. By following the implications of a user's actions through a procedure, the system can recognize actions that, though syntactically correct, appear inappropriate in the context of what the user is trying to do. A user may, for example, destroy some entity that will be needed to complete the task. POISE could warn the user of such potential problems before an actual error occurs.

d. *Abstracting user actions.* Since the procedures represented in POISE are specified hierarchically (i.e. procedures located further up in a hierarchy represent more abstract tasks), the system is not only able to recognize a user's action, but is also able to interpret it as being a part of some higher level procedure and thus understand the action at a more abstract level. This capability is used in summarizing and predicting activities.

e. *Agenda maintenance.* The system can keep track, over a number of terminal sessions, of a user's activities. At any time, the users can ask for agendas of their activities which the system will present as partially completed procedures.

f. *Providing a higher-level interface.* Because POISE contains a hierarchy of procedures at different levels of abstraction, the user can interact with the system at various levels. For example, the user can invoke abstract tasks which do not correspond to actual tools.

The features mentioned above will use a natural language interface that is currently under development. This interface will be for user requests to POISE and for generating natural language descriptions of the current state [MCDO83]. In the context of decision support, all of these features are important but particularly the procedure invocation, planning and agenda maintenance capabilities. Using these facilities, the user is able to choose a task, invoke a procedure which will help carry out the task, and ask for suggestions on alternative courses of action.

The next section describes the POISE system and office procedure formalism in more detail. In the third section, we give an example of how POISE may be used for decision support.

THE POISE SYSTEM

POISE acts as an intelligent interface between the user and the tools available in an office system. A simplified diagram of the main POISE system components is shown in figure 1. Three types of information are used by the interface. The procedure library contains the descriptions of user tasks (or office procedures). The semantic database contains descriptions of the objects used in the procedures and descriptions of the available tools. The model of a particular user's state includes partial instantiations of procedure descriptions with parameters derived from specific user actions as well as instantiations of semantic database objects.

For example, in the procedure library there could be a procedure for filling out a purchase order form. In the semantic database, there would be a description of this form, the fields in it and its relationship to other forms and fields used in the system. After a user had started to fill in a particular purchase order form, the user model would contain a partial instantiation of the "fill_out_purchase_order_form" procedure with values derived from the actual values filled in by the user. There would also be an instantiation of the semantic database object that represents the purchase order form.

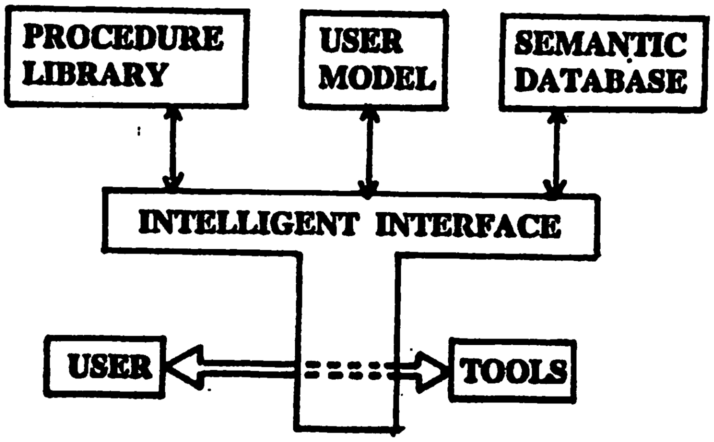

Figure 1: The POISE system.

The formalism used to describe the office procedures must be able to represent sequences of concurrent activities. Figure 2 gives an example of the formalism used in the POISE system. In this "Purchase_items" procedure, the IS clause specifies how the task is accomplished in terms of other procedures. In this case, it specifies that after a purchase request has been received, either a purchase requisition or a purchase order is processed. The task is completed by the steps involved in the complete_purchase procedure. To get the details of the steps involved in the complete_purchase procedure, for example, we would have to examine the corresponding descriptions. The more detailed procedures contain links to the tools available in the system. Figure 3 shows an example of a hierarchy of procedures where the lowest level correspond (approximately) to tool invocations.

The COND clause specifies constraints placed on the values and relationships of attributes of procedures. For example, the simplified COND clause in figure 2 specifies that either the purchase order amount or the purchase requisition amount must be the same as the amount received in the purchase request. The attributes of a procedure are defined in the WITH clause in terms of the attributes of its component procedures. For example, the purchaser in the purchase_items procedure is the same as that in the receive_purchase_request procedure.

```
PROC      Purchase_items

DESC      (Procedure for purchasing items with non-state funds.)

IS        (Receive_purchase_request
          * (Process_purchase_order | Process_purchase_requisition)
          * Complete_purchase)

COND      (and (or (eq Process_purchase_order.Amount  Receive_purchase_request.Amount)
                   (eq Process_purchase_requisition.Amount  Receive_purchase_request.Amount))
               .......

WITH      ((Purchaser    =Receive_purchase_request.Purchaser)
           (Amount       =Receive_purchase_request.Amount)
           (Items        =Receive_purchase_request.Items)
           (Vendor       =Receive_purchase_request.Vendor))
```

Figure 2: An example procedure specification.

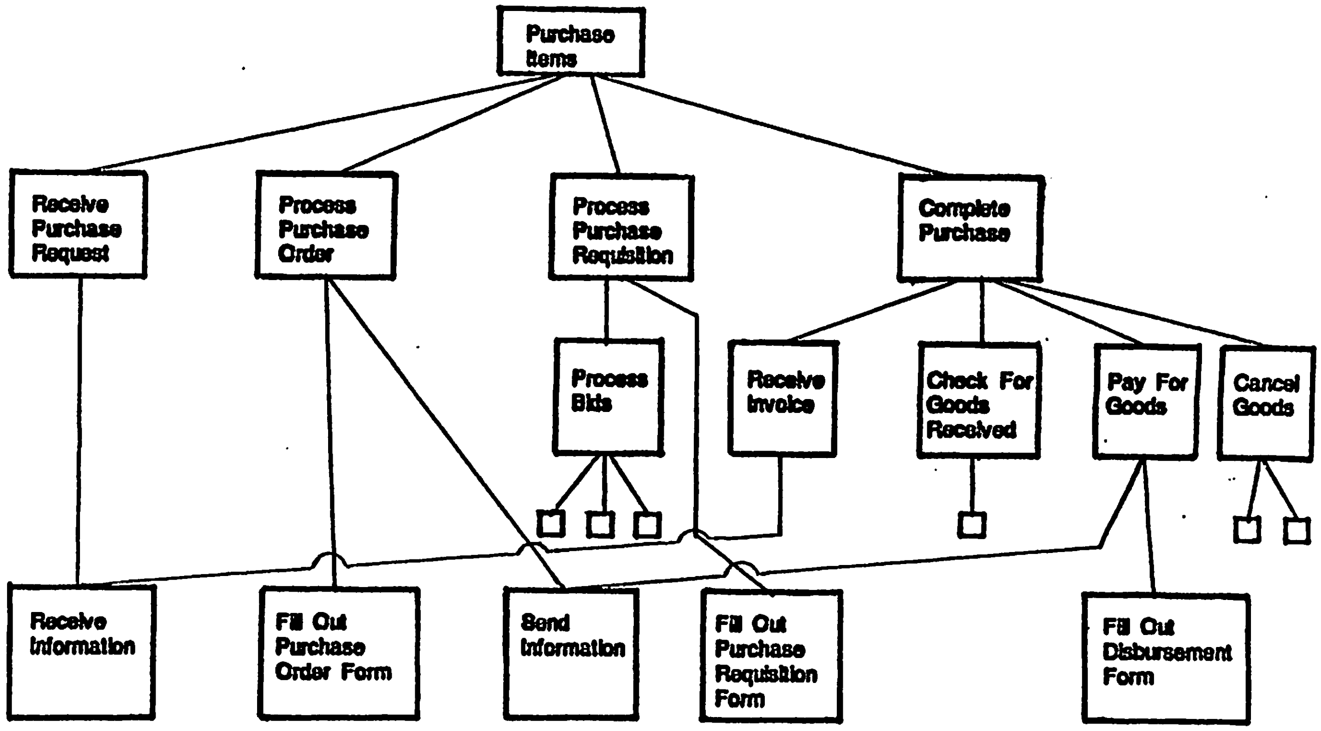

Figure 3: An example procedure hierarchy.

Although it is not shown in Figure 2, the POISE formalism also contains a description of the state of the environment that must exist in order for the procedure to begin and the conditions that must be satisfied when the procedure finishes. An example of this would be to specify that all goods ordered must be received and paid for before the purchase_items procedure can finish. This information serves both as an aid to the planning component of POISE and as an alternate means of recognizing the completion of a procedure. For more details on the POISE formalism, see Bates [BATE82] and Croft [CROF82].

It should be emphasised that the formalism described is an *intermediate* language. The procedure descriptions written in this language are mapped into an internal representation by the POISE procedure reader. The language is adequate for a systems analyst but obviously unacceptable for presentation to the users of an office system. Research is currently underway to investigate the use of natural language, examples and AI planners in the procedure definition process.

The POISE system can work in two different modes - interpretation or planning. In the interpretation mode, the user invokes tools directly and POISE attempts to recognize the user's goals in the context of the procedure library. In the planning mode, the user invokes a procedure and the system must then take responsibility for carrying out as much of that procedure as possible. In an actual environment, POISE will make use of whichever mode is appropriate. This will be illustrated in the next section.

Implementation issues such as the focussing and prediction mechanisms used to recognize user actions, the constraint propagation mechanism, the planning mechanism used for direct procedure invocation, and the semantic database representation will not be covered in this paper. Some of these mechanisms are explained in Croft *et al*[CROF83].

DECISION SUPPORT USING POISE

As a simple example of the type of assistance POISE can provide, we shall use the procedure hierarchy shown in Figure 3. The type of decision a clerical worker who purchases items may make is whether to cancel goods that have been ordered but not yet delivered. The standard procedure (derived from company policy) may be that if the goods are not received within one month after the invoice has arrived, then the order should be cancelled. POISE could provide the following assistance in this process.

a. If one month had elapsed since an invoice had been received, POISE could remind the clerk of the state of this order and suggest that a check be made on the goods received.

b. If the clerk entered the information that goods had not been received, POISE would then suggest that a decision be made on whether to cancel the order. Note that POISE is not expected to cancel orders on its own initiative (there is no tool invocation corresponding to this action).

c. If the goods had been received, POISE would fill in as many details of the disbursement form as possible and present it to the user for completion.

d. If the clerk decides not to cancel the order, POISE will maintain the same state for this procedure while the clerk goes on to other business. If the goods are received, POISE will no longer recommend cancellation but will go on to the pay_for_goods procedure.

We see in this example how POISE both prompts the user and responds to user input. In contrast to the type of task described by the purchase_items procedure, a manager will, in general, have less structured tasks that do not have prespecified organizational constraints. Instead, each person should be able to specify how they prefer to approach the problem. The example we shall use of this type of task is hiring new graduate students for a research project. The problem is to decide how many people can be hired and who they should be. Although it would be very difficult to write an algorithm to handle this task, most people could write a list of things that should be done. For example,

a. Check salary budget for the project.

b. Check current salary for graduate students.

c. Check space available.

d. Get a list of unfunded students and their details.

e. Advertise the positions available.

These steps can be directly specified in a POISE procedure along with other types of information. In step a, the project manager could specify the tool invocation that could retrieve the information (a database query) and how it should be displayed. Step b may involve a phone call to the department head and therefore would not have a corresponding tool invocation. POISE would, however, be able to remind the manager to get this information. For step d, the procedure could specify simply to make the student database available for browsing. Some additional control complexity could be added, such as specifying that if a definite decision is made on some likely students, then mail messages advertising the position will be sent to just these students, otherwise a mail message will be broadcast to all students.

The next time the project manager wanted to hire graduate students, this procedure could be called directly. POISE would then lead the manager through the steps, making the appropriate information available and automating some simple steps. The first step (assuming, for this example, that the procedure consisted of steps a through e, in order) requires checking the salary budget for the appropriate project. The degree to which POISE can accomplish this depends on the amount of information available to it. If it knows which project is to be used (perhaps because this is the manager's only project or because it was recently referred to), POISE may issue a query to the database to obtain the salary budget. Through its semantic database constraints, it can conclude that the salary budget is part of a grant contract. If the grant is available to the system, POISE can obtain this information for the manager, thereby fully automating the first step. Otherwise, it will tell the manager what information is needed and, where possible, suggest how to obtain it.

In a similar fashion, POISE would assist the manager in obtaining the current graduate student salary, available space and current student funding status. Where the information is accessible online, POISE will retrieve it. This may require interacting with the user if the query is not sufficiently specific (e.g. graduate student salary as specified by the department or by the university?). When a decision point is reached, such as deciding which students should receive the advertisement, POISE will present the manager with the information relevant to that decision. Furthermore, upon receiving the results of the manager's decision, POISE can validate some aspects of the results against known constraints. In this example, one or more of the students may be inappropriate candidates (e.g. they are about to graduate, are otherwise employed, etc.).

By helping users collect relevant information, informing them of available choices, and checking the validity of their actions, the system provides the decision-making environment for the user. Though this approach to decision support is not applicable to problems or tasks which only occur once, it is a powerful tool for environments where problems tend to reappear.

CONCLUSION

In this paper, we have indicated how office procedures can be used to support decision-making at all levels of a business. A system which uses descriptions of office procedures to provide intelligent assistance to users was described. The POISE system combines a procedural and goal-based representation of office tasks. It is capable of both recognizing a user's actions in the context of its procedure library and planning a sequence of actions when a procedure is invoked directly. The major component of POISE that remains to be developed is an interface that will allow users to easily specify their own procedures. This interface is a crucial factor in making the system accessible and useful to upper-level managers for decision-making.

Acknowledgements

The design of the POISE system was done in cooperation with Victor Lesser and Karen Huff. Dan McCue and Norman Carver made major contributions to the implementation.

References

BATE82 Bates, P.C.; Wileden, J.C. "EDL: A basis for distributed system debugging tools". *International Conference on Systems Science*; Hawaii, 1982.

BARB83 Barber, G. "Supporting organizational problem solving with a work station". *ACM Transactions on Office Information Systems*, 1: 45-67; 1983.

CROF82 Croft, W.B.; Lefkowitz, L.S. "An office procedure formalism used for an intelligent interface". COINS Technical report 82-4, University of Massachusetts, 1982.

CROF83 Croft, W.B.; Lefkowitz, L.S.; Lesser, V.R.; Huff, K.E. "POISE: An intelligent interface for profession-based systems". *Conference on Artificial Intelligence*, Oakland, Michigan, 1983.

ELLI80 Ellis, C.A.; Nutt, G.J. "Office information systems and computer science". *ACM Computing Surveys*, 12: 27-60; 1980.

FIKE80 Fikes, R.E.; Henderson, D.A. "On supporting the use of procedures in office work". *First National Conference on Artificial Intelligence*, Stanford, California, 1980.

KEEN78 Keen, P.G.W.; Scott Morton, M.S. *Decision Support Systems: An Organizational Perspective*, Addison-Wesley, Reading, Mass. 1978.

LADD80 Ladd, I.; Tsichritzis, D.C. "An office form flow model". *National Computer Conference*, 1980.

MCDO83 McDonald, D.D. "Natural Language Generation as a Computational Problem: an introduction". in *Computational Models of Discourse*, The MIT Press, Cambridge, Massachusetts, 1983.

SPRA82 Sprague, R.H.; Carlson, E.D. *Building Effective DSS*, Prentice-Hall, New Jersey, 1982.

ZISM77 Zisman, M.D. *Representation, Specification and Automation of Office Procedures*, Ph.D. Dissertation, Wharton School, University of Pennsylvania, 1977.