COGNITIVE SCIENCE 6, 331–347 (1982)

A Commitment-Based Framework for Describing Informal Cooperative Work*

RICHARD E. FIKES

Cognitive and Instructional Sciences Group
Xerox Palo Alto Research Center

In this paper we present a framework for describing cooperative work in informal domains such as an office. We argue that standard models of such work are inadequate for describing the adaptibility and variability observed in offices, and are fundamentally misleading as metaphors for understanding the skills and knowledge needed by computers or people to do the work. The basic claim in our alternative framework is that an agent's work is defined in terms of making and fulfilling commitments to other agents, and that the tasks described in those commitments are merely agreed upon methods for fulfilling the commitments. Determining the intended meaning of those task descriptions in specific situations is an important component of the work, and the agents making the commitments decide in any given situation how and whether a given commitment has been fulfilled. We analyze subcontracting relationships, noting that the tasks's client and contractor both play a role in defining the subcontractor's work. Also, we analyze the use and role of procedures in informal domains, noting the interdependencies among the agents performing them, the problem solving required to perform each step, and ways in which they can be made more adaptive. Our analysis indicates that "intelligent" capabilities such as planning, plan monitoring, and negotiation are required to do even simple cooperative work.

INTRODUCTION

In this paper we present a framework for describing cooperative work in domains where there is no agreed upon formal model of the work to be done, nor of the methods for doing it (i.e., in informal task domains). Most of the work people do is in such informal domains where it is not feasible to create precise statements of the tasks to be done, the situations in which they are to

*Lucy Suchman is a major participant in the research effort from which this paper emerges and has made significant contributions to the ideas presented herein. Also, I would like to thank Danny Bobrow, Austin Henderson, Tom Malone, Al Newell, and Kurt VanLehn for providing valuable feedback and ideas in support of this paper.

be done, the resources available, nor the capabilities of the processors doing them. For example, people are regularly confronted with task descriptions such as "write a progress report of the project," "describe the items to be purchased," "yield right of way," "keep your eye on the ball," or "slice chicken breasts very thinly into julienne strips".

The project that produced this framework has been focused on the problems of automating office work involving the use of prespecified procedures, and our experiences with those problems motivate and provide examples for the discussions in this paper. However, the results reported here are intended to be generally applicable in any task domain where tasks and procedures are informally specified, and agents enlist each other's help to achieve their individual goals.

Inadequacies of a Program Execution Model

We began our office automation efforts by attempting to develop a model of procedural office work that would enable computer-based systems to track the progress of tasks, perform portions of them, and provide advice about how to get them done. Our initial thesis was that office procedures are like computer programs, and they are "executed" by a collection of office workers in a manner analogous to a collection of computers executing a program. It seemed a simple matter to automate office procedures by storing them in a computerized data base as if they were programs. Then, at each step in the execution of the procedures in an office, the computer could do the step itself, or tell the person doing the step what operation to do and then monitor the results.

As we proceeded, fundamental problems arose that led us to question our thesis (Fikes & Henderson, 1980). One such problem was that our model did not account for the variability in the way tasks are accomplished. For example, an office worker has more options in following a procedure than our model described. He can choose to ignore some of the requirements of a task (e.g., leave some fields of a form blank), renegotiate the requirements of a task (e.g., request extension of a deadline), or use some method other than the standard procedure for performing a task.

A second problem was that our initial model did not account for the difficulties related to working with informally specified tasks, functions, and procedures. For example, the informality of office work makes infeasible the specification of precise algorithms for performing tasks. Situations occur in which the available procedures do not indicate what to do (e.g., a vendor claims that goods were delivered, but no record of their arrival can be found), what is indicated cannot be done (e.g., a deadline has already past), or what is indicated is not the preferred method for performing the task (e.g., because an unexpected resource is available). Hence, the work

involved in using office procedures is qualitatively different from the work involved in executing a formal algorithm.

We concluded, then, that modeling procedural office work as simple program execution is an inadequate basis for automating it and is misleading as a metaphor for understanding the skills and knowledge needed by computers or people to do the work. That conclusion led us to search for alternative ways of modeling cooperative work that would account for the way office tasks are *actually* performed, and would inform us regarding the required skills and knowledge. This paper presents the initial results of that search, an alternative based on the agreements made by agents performing the work and agents for whom the work is done (see Flores & Ludlow, 1981 for an analysis of communication and management in offices based on such agreements).

THE SOCIAL NATURE OF TASKS AND FUNCTIONS

Tasks

We begin by analyzing a simple work situation in which an agent has a *task* that he wants done. For the purposes of this discussion we will consider a task to be defined as a set of goals to be achieved while maintaining a set of constraints. The basic tenet of our model is that tasks are essentially social in nature in that they are done by one agent (a *contractor*) for some other agent (a *client*). The situation where an agent does a task for himself is the special case in which the client and contractor are the same agent.

A client (i.e., the agent who wants the task done) can choose to enlist some other agent to be the contractor for a task. The client accomplishes that enlistment by establishing a *task contract* with the contractor (see the work on contract nets, [Smith, 1978] and [Smith & Davis, 1981], for a detailed model of how such contracts are established). We model a *contract* as consisting of a collection of *commitments,* each made by one of the *contracting agents* to another of the contracting agents. A task contract is an agreement between two agents containing a commitment by one of the agents (the contractor) to do a task for a second agent (the client). The contract may also contain other commitments, such as a commitment by the client to compensate (i.e., achieve some goal for) the contractor in return for accomplishing the task.

In order for a task contract to be established, the client and contractor must agree on the task that is to be performed. Their negotiations will produce an agreed upon *description* of the task, and the commitment will be a statement of intent to do the described task. Therefore, we consider cooperative tasks as being defined by a social process, and as representing a negotiated agreement between the client and contractor.

Note that a task contract establishes a goal for the contractor of fulfilling his commitment to the client. Performing the described task is only a *method* for achieving that goal. Those two observations form the basis for our explanations of the behavior variations observed in human cooperative work situations. That is, the agreed upon task description provides the contractor with a set of sufficient conditions for fulfilling his commitment, and therefore represents one method of achieving his goal. However, *any* actions by the contractor which result in the client agreeing that the commitment has been fulfilled will achieve the contractor's goal. For example, the contractor may choose to achieve some variation of the task's goal, ignore some of the constraints, convince the client that the task shouldn't be done, etc. He is free to use whatever method he thinks will succeed and is most desirable for him in the context of his other goals and constraints.

How is a model based on these observations useful? First of all, it indicates the basic criteria for successful task completion, namely satisfaction of the client. Thus, for example, if a task tracking system is not able to follow the details of how a task is done in a particular situation and has available a description of the task contract, then it can determine when and if the task has been completed by simply querying the client as to whether he was satisfied.

Second, the model provides a categorization of the information in task and procedure descriptions that highlights the decisions available to the contractor. For example, a standard office procedure is only a typical or suggested method for performing a task, and the task itself is only a standard method for fulfilling a commitment, and the commitment is only a method for achieving the goals for which the other contracting agents have agreed to be the contractors, etc. The contractor has options to select alternative methods in any of these categories.

Third, by highlighting the contractor's decisions, it indicates the problem solving requirements of doing even the most routine cooperative work and begins to suggest the problem solving skills and knowledge needed by a contractor.

Functions

Often in human work situations, a person will agree to perform a given type of task whenever a given set of conditions occur; that is, he will agree to perform a *function*. For example, a buyer in a corporate procurement department may agree to issue a purchase order whenever a properly executed purchase request is received. Also, procedures are typically designed as methods for performing functions rather than individual tasks and are used whenever the function's task is to be done (e.g., the procedure for issuing purchase orders). Hence, in order to describe those situations, we will generalize our discussion to include functions as well as tasks.

As with tasks, one method available to a client who wants a function performed is to establish an agreement (a *function contract*) with a contractor in which the contractor commits to do the function for the client. We will say that such a contract *installs* the function. The contract will contain an agreed upon description of the function to be performed by the contractor. For our discussion, we will consider a function description to consist of a parameterized task description and a parameterized set of *preconditions* such that any given instance of the preconditions defines an instance of the task. Whenever an instance of the preconditions becomes true, the contractor agrees to perform the corresponding instantiated task.

As with tasks, the contractors goal is to fulfill his commitment and the agreed upon function description provides him with a set of sufficient conditions for achieving that goal. Each time the function's preconditions become true, the contractor can choose to do whatever actions he thinks will satisfy the client.

Note that in the transition from task to function, a new subtask has been introduced for the contractor; namely, recognition of the occurrence of the preconditions. Hence, performing a function requires ongoing monitoring to recognize situations in which an instance of the function's task is to be performed.

Tasks and Functions in Informal Domains

A function contractor depends on the function description to specify the situations in which he is to do something and in each of those situations the task that he is to perform. In informal domains, use of those descriptions becomes problematic because of their inprecision and incompleteness (Suchman, 1980) (e.g., what is a "properly executed purchase request"?). Hence, the contractor is confronted in each situation with the new subtasks of interpreting the function description to determine whether a task is to be done, what the task would be, and then after doing something whether the task has been accomplished.

Another basic claim of our model is that the sole criteria for an acceptable interpretation of these descriptions is agreement by the contractor and client. That is, the function and task descriptions are part of the contract between the contractor and client, and those agents are the final authority as to what those descriptions mean and whether they have been satisfied. For example, the meaning of "describe the items to be purchased" on a purchase requisition form is worked out in each case by the requisitioner and the procurement department buyer, for whom the description is being created.

If a commitment description is not sufficiently precise or complete for the contractor to determine in a given situation the client's expectations, then additional negotiations with the client are necessary. Hence, in in-

formal domains, the negotiation processes that produce commitment descriptions continue during the fulfillment of those commitments and become an integral part of the work required to fulfill them.

These observations imply that function contractors in informal domains must be capable of determining appropriate interpretations of imprecise descriptions and of recognizing when a description is sufficiently inadequate to warrant renegotiation with the client. When agents are skilled in those capabilities, the difficult and time consuming process of creating comprehensive function and procedure descriptions can be avoided. Descriptions can be allowed to build up incrementally by generalizing the experiences gained in particular situations. Current computer systems that automate office functions rarely have those capabilities and therefore their use imposes a severe overhead and a rigidity that limits their effectiveness. To the extent that this model aids in understanding these requirements for performing functions in informal domains, it indicates a set of design goals for making such systems more effective.

Installed Functions as Operators for Planning

Installed functions (i.e., functions that contractors have committed to perform) play the same role as operators in standard Artificial Intelligence planning and problem solving frameworks (for example, Fikes & Nilsson, 1971) in that they can be used by agents as methods for achieving goals.

We said earlier that an agent who wants a task done can enlist a second agent to do the task by establishing a task contract with the second agent. Installed functions provide an alternative method of enlisting a second agent to do a task. That is, if the second agent is a contractor who has made a commitment (it doesn't matter to whom) to provide a function, and the task that the first agent wants done is an instance of that function's task, then the first agent can cause the contractor to do the task by persuading him that the appropriate instance of the function's preconditions are true. We will say that an agent who uses a function in this manner is a *consumer* of the function.

For example, if an employee of a small company wants to obtain some equipment for use in his work, then he can achieve that goal by obtaining the appropriate authorizations and submitting the appropriate forms to the company's procurement department. The procurement department has made a commitment to the company president to be the contractor for the function of purchasing equipment, and receipt of the appropriate forms and authorizations is the precondition for that function. The employee becomes a consumer of that function by convincing the contractor that an instance of its preconditions have become true.

If a function contractor refuses to do a task, then the consumer can appeal to the function's client, attempting to convince him that the precon-

ditions are satisfied and that the contractor did not fulfill his commitment to accomplish the task. In the procurement example, for instance, if the procurement department refuses to purchase the equipment, the employee can complain to the company president that they are not performing their function.

In deciding to use a function, the consumer has replaced his original task with the new task of enlisting the contractor to do the original task. Notice that the method for accomplishing the new task is to convince the contractor that an instance of the preconditions have been satisfied, rather than simply to make an instance of the preconditions true. The consumer is free to negotiate with the contractor as to what he will accept as satisfactory evidence that the preconditions are true. For example, the employee requesting an equipment purchase might convince the procurement department that a phone call from the employee's manager is sufficient in that case to authorize the purchase.

If the preconditions of a function are informally described, then there is the additional issue to be resolved in those negotiations of determining an agreed upon interpretation of the descriptions for the specific situation. For example, the employee might ask the procurement department to accept a memo requesting the purchase rather than the standard form. This is another case where negotiations during the performance of a task are vital to its completion and where variability is introduced by the one-time agreements that result from those negotiations.

Subcontracting to Perform Tasks and Functions

Consider again the basic situation in which a client wants a task done and has obtained a commitment from a contractor to do the task. We could then describe the contractor's situation as one in which he wants a task done, and that he has the option of persuading yet a third agent (a *subcontractor*) to do some or all of the task for him. The subcontractor then is in the same situation and has an option to enlist a fourth agent (a *subsubcontractor*), etc. The same description holds for functions as well as tasks.

We are interested here in examining the role that the original client plays in the work of a subcontractor. For that purpose it is sufficient to consider the three agent case where a contractor and client have an agreement in which the contractor commits to perform a function, and the contractor instead of performing the function himself establishes an agreement with a subcontractor in which the subcontractor commits to perform the function. In that case, the original client then becomes the consumer for the subcontractor's function.

To illustrate, we can augment our purchasing example by considering the function contract between the company president and the employee (see Figure 1). In that contract, the president commits to purchase equipment

Function Contract:

Client: Employee

Contractor: Company president

Function Description: Purchase equipment for the employee whenever he submits an authorized request.

Function SubContract:

Client: Company president

Contractor: Procurement department manager

Consumer: Employee

Function Description: Purchase equipment for the employee whenever he submits an authorized request.

Figure 1. Example Subcontracting Situation in an Office

for the employee whenever he submits an authorized request. Hence, the president is the contractor and the employee is the client. Instead of doing the purchasing himself, the president contracts with the procurement department manager to do it. In that contract, the president is the client and the procurement department manager is the contractor. The overall result is a subcontracting relationship in which the employee is the consumer, the company president is the contractor, and the procurement department manager is the subcontractor.

The contractor wants the function done in order to fulfill his commitment to the consumer. The commitment of the subcontractor to perform the function can therefore be considered as being to fulfill the contractor's commitment to the consumer. Since the consumer has authority to determine when the contractor's commitment to him has been fulfilled, obtaining the consumer's satisfaction is one of the sufficient conditions and the primary method for the subcontractor to fulfill his commitment. The subcontractor is therefore free to do whatever he thinks will convince the consumer that the contractor's commitment to him has been fulfilled.

This analysis implies that the consumer is an additional agent with whom the subcontractor can negotiate to determine what is required of him in a given situation. If the subcontractor and the consumer agree on what is to be done, then the contractor need not enter into the negotiations or even know what was agreed on because his commitment to the consumer is being fulfilled and the commitment to him by the subcontractor is being fulfilled.

In our purchasing example, when the employee requests the equipment purchase, the procurement department buyer may attempt to satisfy the employee by convincing him that he should use previously purchased equipment or that he should rent equipment. He may persuade the employee to help find an appropriate vendor, and in return agree to obtain the authorizations for the purchase that are normally part of the employee's responsibility, etc. Such localized one-time agreements between agents occur

regularly in office settings, and are an important aspect of the variability and adaptibility that characterize office work. Standard computer program description techniques (e.g., flow charts) are hopelessly inadequate for describing such activity.

If, in a given situation, the subcontractor and consumer cannot agree on the task to be done, then they both can appeal to the contractor for help. The subcontractor can argue that his commitment to the contractor does not include what the consumer is asking for, and the consumer can argue that the contractor's commitment to him is not being fulfilled. Hence, the contractor needs to enter into the negotiations only when the subcontractor and the consumer cannot agree.

We see, then, that the consumer is a source of information for the subcontractor about what is to be done and an authority on when the task has been completed. Also, the consumer acts as a monitor for the contractor as to whether the subcontractor has done his job, since it is the consumer who cares whether or not the task is accomplished. The interdependencies among the consumer, contractor, and subcontractor discussed in this section are summarized in Figure 2.

For the consumer:

The subcontractor:
Performs the desired task.

The contractor:
Settles disputes with the subcontractor.

For the contractor:

The subcontractor:
Fulfills the commitment to the consumer.

The consumer:
Provides compensation for doing the task, and
Monitors the subcontractor's work.

For the subcontractor:

The consumer.
Helps interpret the task description, and
Indicates when the task is completed.

The contractor:
Provides compensation for doing the task, and
Helps settle disputes with the consumer.

Figure 2. Summary of the Consumer, Contractor, Subcontractor Relationships

The Social Nature of Procedures

Now consider a situation in which an agent has a function he wants done and a procedure describing how to do it. We will call the agent who has the

function "the *procedure's manager*" and the function "the *procedure's function*". A procedure describes a method for doing a function in terms of a collection of steps to be done in a specified order, and thereby provides a means for the procedure's manager to organize a collection of agents to perform the procedure's function. That is, the procedure's manager has the option for each of the procedure's steps of obtaining a commitment from some other agent to do the step (i.e., of "installing the step"). If he obtains such a commitment for each step of the procedure (i.e., if he "installs the procedure"), then the agents who agreed to do the steps (i.e., the "*step contractors*") will do the function for him. For example, if a procurement department manager is assigned the function of purchasing equipment for employees, then he can either find or create a procedure for performing that function and install the procedure by obtaining commitments from the people in his department to be step contractors for each of the procedure's steps.

In formal domains, action descriptions can be provided for each step in a procedure that are guaranteed to satisfy the designer's intention for the step (e.g., add x to y). Then the commitment of a step contractor is to perform the step's action in a manner that satisfies the formal description. The contractor need not have any model of the results expected from his step or of the role they play in performing the procedure's task. His total sphere of concern is to rotely perform the action as specified. That is the style of procedure execution done, for example, by a typical programming language interpreter.

In informal domains, there are no guarantees that a procedure will successfully accomplish its task. Those guarantees are lost because the procedure, its task, and the situations in which it will be used are imprecisely described. Hence, procedures in informal domains are only prototypes of methods for performing tasks. They suggest a way of decomposing a task into steps, and perhaps indicate how the task is typically performed, but they do not alleviate the need for problem solving in each specific situation to determine how to perform a task. The user of an informal procedure is confronted with the subproblems of determining the meaning of the procedure in the specific situation and whether it will be applicable or effective.

A basic problem in informal domains with installing procedures to perform functions is that the installer must commit at the time of installation to the decomposition specified by the procedure. If indeed as we argued above, that decomposition is only suggestive and needs to be reexamined each time the procedure is used, then the strategy of installing a procedure is an ineffective means of transferring the work from the procedure's manager to the step contractors. The challenge then for the procedure's manager is to describe and install procedures in a manner that maximizes their adaptibility and flexibility.

Procedure Steps As Functions

An important way of meeting the challenge of compensating for the inadequacies of informally specified procedures is to describe each procedure step as a function to be performed (i.e., as a type of task to be done whenever a given set of conditions occur) in addition to being an action to be performed. For example, add to a step described as "Submit to procurement an authorized purchase request" the function description "Whenever an employee wants equipment purchased, achieve the goal: Procurement knows the employee wants equipment purchased and has the information and authorizations necessary to make the purchase". Viewing procedure steps as functions implies that installing a step involves establishing a function contract between the procedure's manager and the step contractor.

A function description specifies the requirements of a step without reference to *how* those requirements are to be met. The contractor can choose whatever method is appropriate in a particular situation to accomplish the function's task. Such a description therefore clearly distinguishes for the contractor the procedure manager's requirements from suggestions about methods (actions) for meeting the requirements. The agent performing a step can use the function description to evaluate whether the action described for the step is an appropriate method in a given situation, to plan alternative methods for performing the step, and to evaluate whether the actions he takes accomplish the step.

Adding function descriptions to steps results in procedures applicable to a wider range of situations because it allows the agents performing the steps to take into consideration properties of the situation such as resource limitations and interactions with other tasks that may not have been known at the time the procedure was designed. The work involved in using those function descriptions is significantly different from the work of performing steps described as actions. In particular, it involves subtasks of *planning* to determine a method to use, and *monitoring* to determine whether the method accomplished the function. However, an agent capable of effectively performing those subtasks can better determine the appropriateness of his results and can successfully perform his step in unexpected situations (Fikes, 1981).

Subcontracting Within Procedures

Our description of procedure installation thus far would predict that each time a procedure step is performed and the step contractor does something other than the task described in his agreement with the procedure's manager, that the contractor must obtain an acknowledgement from the manager that what he did satisfies his commitment. In actual practice in offices, there is a

broad variability of behavior in the performance of procedure steps, and only rarely is that behavior accompanied by interaction with the procedure's manager (typically the step contractor's supervisor). Instead, there are frequent negotiations among the agents doing the steps of the procedure. Those agents are not generally working for each other and have made no apparent commitments to each other. How do we explain their negotiations and the role those interactions play in their work? The explanation becomes apparent, as we shall see, by examining the subcontracting relationships established among the step contractors during procedure installation.

We apply our analysis of subcontracting to the performance of procedure steps by identifying the commitments made during a procedure installation and considering the *procedural role* played by each step. A step's procedural role is the rationale used by the procedure designer for including the step in the procedure (e.g., achieve a goal of the procedure's function, satisfy a precondition of some other step in the procedure). That rationale is therefore the defining basis for the function to be performed at that step (VanLehn & Brown, 1980). For example, the procedural role of the "submit to procurement an authorized purchase request" step is to satisfy a precondition of procurement's purchasing step.

A procedure step's function has a set of preconditions as part of its description. The designer of a procedure must assure that when a given step is to be performed, its preconditions are satisfied. That design goal is satisfied by including other steps earlier in the procedure that will cause those preconditions to be satisfied. The procedural role of those earlier steps, therefore, is to initiate performance of the later step.

We can characterize a function's preconditions as consisting of *activation conditions,* the satisfaction of which signals the contractor that an instance of the function's task is to be done, and *enabling conditions,* the satisfaction of which provides the resources needed by the contractor to perform the task. For example, the function performed by a buyer in a procurement department is activated when he receives a purchase request and is enabled by "delegation of authority" forms which he uses to authenticate the authorization signatures on the request. We distinguish, therefore, between steps whose procedural role is to *activate* other steps and those whose role is to *enable* other steps.

We make use of that distinction in describing the contract that installs a procedure step. That contract contains a commitment by the step contractor to perform the step's function whenever the step's activation conditions are satisfied and typically also a commitment by the procedure's manager to satisfy the step's enabling conditions whenever the step is activated. For example, an accounting department clerk (the step contractor) may make a commitment to his supervisor (the procedure's manager) to respond to vendors' invoices whenever one is received. The supervisor, in

turn, agrees to provide the clerk with the purchase order, packing slips, and other supporting documents needed to respond appropriately to the vendor.

The procedure's manager assures that a given step is initiated at the appropriate time by installing the steps whose procedural role is either to activate or enable the step. The contractor for an activating step has the goal of informing the contractor for the step being activated that an activation condition has been satisfied (in addition to possibly causing the condition to become true). The contractor for an enabling step has the goal of providing the contractor for the step being enabled with an enabling resource. Hence, the contractor for an enabling or activating step is in effect a subcontractor whose consumer is the agent performing the step being initiated. For instance, in the accounting department example above, the agents who supply the clerk with the supporting documents are subcontractors whose consumer is the clerk.

Our earlier comments about the role a consumer plays in the work of a subcontractor therefore apply to enabling and activating steps, and provide the explanation we are seeking. That is, an agent doing a step being initiated and an agent satisfying an enabling or activating condition negotiate with each other to determine what the initiator's task is in problematic situations, and the procedure's manager is brought into the negotiations only when they cannot agree. Also, the agent being initiated acts as a monitor on the enablers and activators for the procedure's manager.

The analysis of subcontracting applies to any procedure step whose procedural role involves providing a result to some agent other than directly to the procedure's manager. In those cases the agent providing the result is fulfilling a commitment made by the procedure's manager to the consumer of that result (or to a client of that consumer). Hence, the consumer and producer can work out together what is to be provided.

SUMMARY AND CONCLUSIONS

In this analysis we have described a framework that identifies sources of variability and adaptibility observed in human cooperative work situations. Our basic claim is that an agent's work is defined in terms of making and fulfilling commitments to other agents. The tasks described in those commitments are merely agreed upon means for fulfilling the commitments. The agents involved in the agreement are free in any given situation to decide how and whether a given commitment has been fulfilled. Hence, nonstandard methods and outcomes may be considered acceptable even though they do not correspond to the described tasks, functions, and procedures.

We claimed that descriptions of tasks and functions result from negotiations between clients and contractors, and that in informal domains,

those descriptions are necessarily incomplete and imprecise. Determining their intended meaning in specific situations is an important component of the work. That determination involves continuing negotiations, and the sole criteria for an acceptable interpretation of the descriptions is agreement by the contractor and client.

We noted that when a contractor enlists subcontractors to fulfill his commitments that the original client plays the role of a consumer in the subcontractor's work. Since one of the subcontractor's commitments in such a circumstance is to satisfy the consumer, there is an additional source of variability in that the consumer is another agent with whom the subcontractor can negotiate to determine what is required of him in a given situation.

Procedures provide a means for organizing a collection of agents to perform a function. In informal domains, procedures represent only prototypes of methods whose meaning and applicability in specific situations is unclear. Their use requires problem solving and negotiation to determine an effective sequence of actions in a given situation.

We argued that describing procedure steps as functions provides the step contractors with the information and freedom they need to determine and perform the appropriate actions. We introduced the idea of a step's procedural role to represent the rationale for the step's inclusion in the procedure, and argued that any actions by the step's contractor that fulfill the step's procedural role are sufficient for successful execution of the step. Finally, we argued that installing a procedure involves establishing subcontractor-consumer relationships among the step contractors so that they negotiate among themselves about what to do in particular situations without needing to involve the procedure's manager.

Information Needed To Do Cooperative Procedural Work

The framework we have presented characterizes the basic information needed by agents doing cooperative work and the role that information plays in their work. In general, it indicates that an agent needs descriptions of the task, function, and procedure contracts to which he has agreed, and the functions available to him.

For each commitment to perform a task or function, the contractor needs to know the agreed upon task or function description (because it provides a set of sufficient conditions for fulfilling the commitment), the client (so that the contractor knows whose satisfaction he is trying to obtain), and the consumer of the results in the case where the commitment is a subcontract (because satisfying the consumer is one of the sufficient conditions for fulfilling the commitment). If the commitment is to perform a procedure step, then the contractor needs to know the procedural role of the step

(since satisfying that role is a sufficient condition for completion of the step).

An agent needs to know the functions available to him so that he can use them as steps in plans he forms to accomplish his tasks. In order to use a function, he needs a description of its task (so that he can determine whether the function can be used to accomplish a given task), its preconditions (because they describe a means for initiating performance of the task), the identity of the contractor (so he will know who he must persuade to perform the task), and the identity of the client (so he will know to whom to appeal if the contractor is not performing the task to his satisfaction).

The framework also indicates information to include in the description of an informal procedure in order for the procedure to be used adaptively and flexibly. The description should identify the procedure's manager (so that each step contractor knows whose satisfaction he is trying to obtain), and each step of the procedure should be described as a function (so that the step contractor can choose his own method of performing the step). If satisfaction of an enabling or activating condition of a step is subcontracted to another step, then the description of the step being initiated should identify the initiating step and who is performing it (so that the contractor for the initiated step can negotiate with the initiator and also monitor his performance). Finally, as noted above about all functions, if a step is a subcontract, then its description should identify the consumer of the subcontract (because satisfying him is a sufficient condition for fulfilling the commitment).

Implications For Designing Systems That Support Informal Cooperative Work

A primary goal of our attempt to explicate the nature of informal cooperative work is to clarify the challenges involved in constructing computer-based systems for supporting such work.

Our discussion indicates that in informal domains, "intelligent" capabilities such as planning, plan monitoring, and negotiation are required to do even simple cooperative work. Systems that claim to automate such work and do not have those capabilities require much more precise descriptions of their function and how it is to be performed than do people, and can "commit" to doing only a formalizable approximation of the function desired by the client. They are incapable of performing the function in situations that do not match the assumptions of the formalization, and can not adapt their methods to account for unexpected features of a particular situation such as resource limitation changes or interactions with other tasks. In addition, they require more effort by the client to establish their task or

function contract since they have no capability of interpreting vague descriptions and only very limited capabilities for recognizing situations where a description is inapplicable.

All too often, designers and installers of automation systems do not realize the unformalizable subtleties of the work being automated, and therefore do not anticipate the differences between what the systems are going to do and what the people did whom they are replacing. Those differences often cause major organizational upheavals because they change the work requirements of all the agents who interact with the systems. A major goal of the analysis presented here has been to provide a model of the unformalizable aspects of cooperative work being overlooked by current automation efforts so that the differences in functionality introduced by the automation can be predicted and compensated for.

Systems designed to assume the role of *expert assistants* to human agents doing informal cooperative work seem far more promising in the foreseeable future than do systems designed to replace people. The framework presented here provides a basis for describing in such expert systems the work to be supported. Commitment-based descriptions facilitate task tracking, for example, in that they indicate who is to do each task (the contractor) and provide basic criteria for task completion (i.e., client and consumer satisfaction, and initiation of a step being enabled or activated). In addition, they form a useful data base for providing agents with information they need, when they need it, and in a way that indicates clearly what is required of them and where choices are available to them in doing the work. The available information is useful both to agents that do the work (e.g., answering "how to" questions) and to managers (e.g., determining information and paper flow, and determining the effects of proposed procedural changes).

REFERENCES

Fikes, R. Automating the Problem Solving in Procedural Office Work. *Proceedings of the AFIPS Office Automation Conference,* Houston, TX, March 1981.

Fikes, R., & Henderson, A. On Supporting the Use of Procedures in Office Work. *Proceedings of The First Annual National Conference on Artificial Intelligence,* Stanford, CA, August 1980.

Fikes, R., & Nilsson, N. STRIPS: A New Approach to the Application Theorem Proving to Problem Solving. *Artificial Intelligence,* 1971, *2,* 189–208.

Flores, F., & Ludlow, J. Doing and Speaking in the Office. In G. Fick & R. Sprague (Eds.), *DSS: Issues and Challenges.* London: Pergamon Press, 1981.

Smith, R. G. A Framework for Problem Solving in a Distributed Processing Environment. Department of Computer Science, Stanford University, Stanford, CA, STAN-CS-78-700 (HPP-78-28) December 1978.

Smith, R. G., & Davis, R. Frameworks for Cooperation in Distributed Problem Solving. *IEEE Transactions on Systems, Man, and Cybernetics,* January 1981, *SMC-11 (1).*

Suchman, L. *Office Procedures as Practical Action.* Presented at the workshop on Research in Office Semantics, Chatham, MA, June 1980.

VanLehn, K., & Brown, J. S. Planning Nets: A Representation for Formalizing Analogies and Semantic Models of Procedural Skills. In R. Snow, P. Frederico, & W. Montague (Eds.), *Aptitude Learning and Instruction: Cognitive Process Analyses.* Hillsdale, NJ: Lawrence Erlbaum Associates, 1980.