UNDERSTANDING EXPLANATION

Claire O'Malley

May 1987

Cognitive Studies Research Papers

Serial No. CSRP 088

The University of Sussex
School of Cognitive Sciences
Arts Building,
Falmer, Brighton BN1 9QN

Understanding Explanation

Paper presented to the Ergonomics Unit, University College London, 22nd May 1987.

Claire O'Malley

Cognitive Studies Programme
University of Sussex
School of Social Sciences
Arts Building
Brighton BN1 9QN
(0273) 606755

Introduction: Help and explanation systems

An important issue in human-computer interaction research is how to design effective help, explanation and advice-giving facilities for complex computing environments and expert systems. Take, for example, environments such as UNIX or Poplog. The characteristics of such systems, users and their tasks, makes traditional approaches to providing help inadequate for several reasons:

(a) the level of experience of users tends to vary considerably — ranging from fairly casual acquaintance with the system to quite substantial experience;

(b) there isn't a straightforward continuum from novice to expert — a study by Draper (1985) showed that expertise in a system like UNIX is characterised more by experience with a fairly small working set of commands, and that users' working sets overlap only marginally with each other;

(c) thus, the notion of "specialist" is more appropriate than "expert";

(d) expertise in such systems cannot be characterised simply by amount of knowledge of the system — the system is too large and complex, and there are many levels and kinds of knowledge, depending on the nature of the task the user commonly performs;

(e) finally, the system is highly modular — there are many ways to do things with the available commands, making the system powerful, but leading to problems in designing help facilities, since as a result there aren't always set procedures for achieving tasks.

This view of expertise with the system is also reflected in the use of the help and documentation. In a study of the use of the online UNIX manual (O'Malley, Smolensky, Bannon, Conway, Graham, Sokolov & Monty, 1984), it was found that the vast majority of queries for help concerned the need for a procedure for performing a specific task, for which there was no one-to-one correspondence with the information as presented in the documentation. So, there is often a huge gap between the way a user's problem is stated and how the information needed is represented and organised in the documentation or help

system.

Not surprisingly, users often prefer to go to another person for help than to try and find the information in the documentation. Interestingly, a study by Scharer (1983) suggests that when users consult the local expert, she doesn't always try to answer the query or solve the problem, but more often tells users how to find the information that will help them figure out their problem. (The "local expert" also tends to be the heaviest user of the manual — cf. Draper, 1985.) The general point here is that one of the ways in which other people provide help is by bridging the gap between the user's problem and the information needed — by helping users specify their problem so that it's closer to the way the information is organised.

Other studies of social interaction in the workplace (e.g., Fikes, 1982) show that procedures are often *negotiated* between people rather than being set down as standard rules. These informal procedures may emerge as set ways of doing things, but this information, being emergent, is not captured by an a priori analysis, and therefore not represented in the documentation.

Another way in which other people help is by "contextualising" information that exists in formal manuals and procedures. In fact, a recent study of fault diagnosis by photocopier technicians (Orr, 1986) suggests that personal anecdotes or "war stories" are essential in coping with cases not covered by the documentation. These war stories serve as "community memory" and help in problem solving by combining abstract information about the machines with the context of a specific situation.

Finally, as Owen (1986) has pointed out, learning "the system" is often quite a serendipitous process, achieved by looking over someone's shoulder in the terminal room and seeing them do a "neat hack", or chatting in the coffee room, and so on.

What kinds of questions do users ask?

There are several possible reasons for users needing to ask a question. They may have a goal to accomplish but don't know how to go about achieving it, or they may need a description of a term or a concept. However, they may also just need confirmation or verification of a solution they are considering, or help in testing a hypothesis. They may also be considering several alternative solutions and need help in choosing among them.

There are some types of question that are reasonably well supported by traditional kinds of help system. However, there are other kinds of questions that it is not possible to ask, let alone answer, with traditional help systems:

(a) Some questions are fairly straightforward, such as *what is grep?* The user needs a description of the command, which can be very brief, or may involve a longer exposition, especially if it is a concept they are asking about (e.g., a buffer) rather than a command. Although most systems are able to provide descriptive help of this kind more or less effectively, there is a variant of the *what is?* question that is often not taken into account: the question of the form *what is the difference between?*, where the user needs to know the difference or relationship between one or more things.

(b) Another common type of question that help systems often do address is the *how?* question — i.e., requests for information about how to accomplish some goal. However, most systems presuppose that the user has specified this question well enough for the system to produce an immediate solution in the form of a procedure to carry out. In many cases, answering *how?* questions involves help in planning how to accomplish goals.

(c) *What if?* questions are a special case of *how?* questions. These are questions about hypothetical situations, rather than being requests for recipes for performing some task. The answer to these kinds of question requires that the system be able to simulate — or allow the user to simulate — the hypothetical case in some way.

{d) *Why?* questions are usually interpreted as being of the form *what caused?* and are viewed as requiring explanations based on the system providing a trace of the steps it went through to produce some result. Other interpretations of *why?* questions involve wanting to know why an undesirable result occurred. In other words there is a difference between the question *where did I go wrong?* and *why is this wrong?*

A simple way of supporting users in trying to work out where something went wrong is to show them where to look, without necessarily giving any further explanation. In other words, if the problem is due to a simple slip, the user can figure out what to do once it is pointed cut — e.g., typographical errors. However, there are other cases where the system interprets the user's command alright, but the output isn't what was intended for some reason. In this case, the user needs to know what produced the effect. The only difference in these cases is whether the system or the user detected the existence of the problem. In both cases, a simple backtrace to the origin of the problem without any deeper explanation, would suffice.

There are other cases, however, where even when the source of the problem has been pointed out, the user still doesn't understand why something is wrong — perhaps because of some fundamental misconception. This distinction is similar to that made by Norman (1981) between slips and mistakes. Slips (or questions of the form *where did I go wrong?)* can be handled by drawing the user's attention the the source of the error. Mistakes (or questions of the type *why is this wrong?)* are cases where there is a "bug" in the user's plan or model. In these cases some deeper explanation is required.

What are explanations for?

The advent of expert systems has raised considerable interest recently in the design of systems that can provide more flexible and intelligent explanations. However, research in artificial intelligence and expert systems tends to focus on the natural language aspects of questions, in order to determine the user's intention from the surface expression. This has led to a good deal of work recently on classifying question types and so on. However, there are several reasons to question the feasibility of this approach:

(a) The surface expression of a query (i.e., the question asked) rarely specifies the type of answer sought (i.e., the intention behind the utterance). Firstly, there is a problem of disambiguation, which requires attention to pragmatic and contextual factors.

(b) Explanations are always relative to a particular query and context, so, secondly, it makes no sense to consider explanations as objective and independently existing accounts that just have to be "accessed[11] by asking the right question.

(c) This leads to the view that many explanations are not objective accounts to be "given" or communicated in some way, but are better viewed as solutions that require joint negotiation — where both the inquirer and the explainer contribute to the eventual solution.

Research in expert systems has also tended to place more emphasis on how to give an explanation, than on why an explanation might be needed. There has also been a considerable history of philosophical research into explanation, but little of it bears any relation to cognitive processes, or to what might prompt people to ask for explanations. With complex and powerful computing environments, and given the variety of users, tasks and contexts which need to be taken into account in designing help and explanation facilities, we need to ask not what counts as an explanation in any *absolute* sense, but what, psychologically, is an explanation *for?*

An account of human-human explanation should address (at least) the following questions:

(a) Why do things puzzle people and prompt them to find/construct explanations (i.e., what is it about a phenomenon or event that makes people ask "why did X happen [rather than Y]?")?

(b) What steps do people take initially to solve the problem for themselves (e.g., experimenting, consulting manuals and documentation, constructing hypotheses, etc.)?

(c) When people resort to getting help/explanations from others (or documentation and manuals) how do they formulate an appropriate question to ask? (E.g., what sort of knowledge does the inquirer assume of the explainer, and how is this determined?)

(d) How does the explainer determine the inquirer's intention from the question asked, the problem context, etc.?

(e) How can the explainer determine the right kind of explanation to give — given that there are many possible solutions, constraints, etc.?

(f) How should the explanation be presented?

(g) How does the explainer determine that the explanation is both understood and solves the inquirer's problem? In many cases beyond the simplest, it isn't a question of finding the solution and then communicating it, but more a case of negotiating the solution with the inquirer, so it's both an iterative and cyclic process.

(h) Finally, an issue related to the last — why does an explanation stop where it does? What makes this slightly different to the last question is that here we need to look not just at how the explainer determines that the explanation is acceptable, but also at how the inquirer knows that it's solved the problem — how the goal is actually satisfied by the solution.

Charles Sanders Peirce, though a philosopher writing in the late nineteenth century, offers perhaps one of the earliest *psychological* accounts of what prompts people to seek an explanation (cf. Burks, 1958). He talks about the "surprise" at coming across the unexpected as prompting a search for an explanation:

> "...nothing can appear as definitely new without bring confronted with a background of the old. At this, the infantile scientific impulse ... must strive to reconcile the new to the old ... Thus it is that all knowledge begins by the discovery that there has been an erroneous expectation of which we had before hardly been conscious." (Burks, 1958, p. 111-112.)

Most philosophers, in discussing explanation, have taken the motivation or the circumstances under which an explanation is sought, as given. Peirce asks explicitly what it is about a phenomenon that makes it surprising. He argues that what is important is the relationship between the phenomenon to be accounted for and prior knowledge. In other words, the search for an explanation involves some notion of conflict or inconsistency between the observed event and what was expected to happen.

This point may seem fairly obvious, but in the expert systems field *why?* questions (e.g., why did X happen?) are typically interpreted as being of the form *what caused?* In this view what is required for the system to provide an explanation is to give a trace of the steps it went through (the rules that were fired) to produce a result, with perhaps some additional justification for the explanation. However, if the user's problem is due to a particular event not matching what was expected, the question *why?* actually implies *why not?* In this case an appropriate explanation is one that is related to the expectation that the user had.

In her studies of consultation dialogues, Kidd (1985) has noted that even users who were relatively inexperienced in the domain, still had their own ideas, preferences and expectations about an appropriate kind of remedy for their problem. Miller's ATTENDING system (Miller, 1984) incorporates what might be one attempt to address this issue, by the use of what is called a "critiquing" approach in providing explanations. That is, instead of basing an explanation on some objective account and from the system's perspective, the explanation is given in terms of a critique of the user's plan.

Learning and problem solving

So, in general, in order to understand what makes an explanation acceptable as the solution to a puzzle, we need to examine the relationship between the explanation and the inquirer's prior knowledge, expectations or (in the case of understanding systems such as computers) the user's mental model:

Theories of mental models have generally focussed on structural or surrogate models, and view such models as powerful in that they are self-contained or closed, thus allowing users to mentally simulate the possible actions of a system and thereby predict its behaviour. In reality, however, users* models seem to be distributed and fragmentary, rather than structural and complete. They are often heavily context-dependent: Lewis (1986) has shown how users generate explanations of a command's function in a specific context but these explanations often do not predict the command's function in another context.

Coming up with a single coherent view of a system may involve structural models and rules that are quite difficult to learn. If a system is to be used in a variety of ways, multiple models are likely to be developed. It has also been suggested (Carroll & Mack, 1985) that it is the very *open-endedness* of fragmentary or distributed models which stimulates *active learning.*

In general, research on mental models has focused more on how people use models rather than on how they are developed. The few studies that do deal with learning tend to look at how an explicit and externally imposed model is assimilated. We need to look more at how actual models are formed and change. Research by Lewis and others (e.g., Lewis & Mack. 1984; Mack, Lewis & Carroll, 1983) has shown that learners often developed explanations of events they observed during training, even when they weren't asked to. Lewis (1986) suggests that such explanations may be valuable in allowing learners to develop generalisations from one or a few examples, and that determining how their actions relate to observed outcomes is crucial in allowing learners to build new procedures for accomplishing novel tasks.

Kahneman and Miller (1986) have recently proposed a theory to account for, amongst other things, the phenomenon of "surprise" or "violation of expectation". They have also noted that why-questions imply a violation of expectations or "norms":

> "A why-question ... presupposes that some state X is the case, and also implies an assertion that not-X was normal." (Kahneman & Miller, 1986, p. 148.)

They also argue that an appropriate answer to a why-question (at least, those of a deniable kind — i.e., for which the reply "why not?" might be a sensible answer) is *not* the explanation of the *event* per se, but the explanation of an *effect* — that is, the contrast between an observation and a more "normal" alternative.

However, although I have argued that we need to look at the relationship between the puzzling event and the learner's expectations, this is not to imply that learning by building explanations is strictly expectation-driven (i.e., truly planful). Kahneman and Miller (op. cit.) argue, in contrast to the more usual schema-driven approach, that norms are evaluated *after* the event, rather than in advance:

> "Reasoning flows not only forward, from anticipation and hypothesis to confirmation or revision, but also backward, from the experience to what it reminds us of or makes us think about." (Kahneman & Miller, 1986, p. 137.)

Ross and Moran (1984) have also shown that aspects of a task (even those which may be irrelevant to how the task is to be accomplished) can trigger remindings of earlier events in the learning situation, which then influence how learners perform. These remindings often tend to occur early in learning, and where the tasks faced seem difficult. Ross and Moran suggest that remindings serve as a backup process when the learner can't remember or reconstruct a correct method.

Explanation based learning

The explanations constructed by users in the studies by Lewis and others (Lewis & Mack, 1984; Mack, Lewis & Carroll, 1983), and in the Big Trak study by Shrager and Klahr (1986), seem to involve inferential processes of an *abductive* nature. Users tended to try to generate a hypothesis to account for some observation, usually based on very limited evidence, and then tried to verify these hypotheses — sometimes by explicit experimentation — within these same limitations of information. Even behaviour that was only nearly correct was taken as confirming evidence, sometimes leading to slight distortion of orginal predictions. When they came across evidence conflicting with their hypothesis, users tended to change their theory, again based on very little evidence, and often contradicted by other available information.

So, these "abductive strategies" often lead users into trouble. However, abduction can also be a valuable strategy. It can be a fairly cheap way of gaining new knowledge, since it doesn't require extensive pre-existing knowledge. It allows people to select information that's relevant from a whole host of possible interpretations. In fact, as Peirce noted, abduction, or coming up with an appropriate hypothesis, is the first step in any scientific reasoning:

> "Accepting the conclusion that an explanation is needed when facts contrary to what we should expect emerge, it follows that an explanation must be such a proposition as would lead to the prediction of the observed facts, either as necessary consequences or at least as very probable under the circumstances. A hypothesis then, has to be adopted, which is likely in itself, and renders the facts likely. This step of adopting a hypothesis as being suggested by the facts. is what I call abduction." (Burks, 1958, p. 121-122.)

Ideally, new consequences are deduced from the hypothesis and tested against additional data, then inductive reasoning is used to make a judgement about the likelihood of the hypothesis being true, given the accumulating evidence. However, when learning in unfamiliar domains, users are often led into the trap of what Lewis and Norman (1986) call "cognitive hysteresis" — where it is easier to stick to a (possibly erroneous) hypothesis, than to give it up.

This has also been labelled the "confirmation bias", and is usually interpreted to mean that people tend to test those cases that have the best chance of verifying the current hypothesis than of falsifying it. However, Klayman and Ha (1987) have recently argued that many phenomena under the label "confirmation bias" are better understood in terms of a more general "positive test strategy". That is. there is a tendency to test those cases that

are expected to have the property of relevance or interest rather than those expected to lack that property. Thus, as Lewis (1986) has argued, the confirmation bias may be better viewed as a *relevance* bias.

Klayman & Ha (op. cit.) suggest that the positive test strategy is a default heuristic used in cases where concrete, task-specific information is lacking, or when cognitive demands are high. They suggest that it is a sensible strategy to adopt in cases of little information or domain knowledge, and where obtaining falsification is somewhat like searching for a needle in a haystack.

When users construct explanations to account for system behaviour, the choice of hypothesis is not completely random. Shrager and Klahr (op. cit.) noted that subjects in their study tended to generate only a very limited set of alternative hypotheses. Peirce also pointed out that analogy plays a major role in abductive inference:

> "The mode of suggestion by which, in abduction, the facts suggest the hypothesis is by resemblance — the resemblance of the facts to the consequence of the hypothesis." (Burks, 1958, p. 137.)

Lewis (unpublished manuscript) has recently taken up this idea in his attempt to model explanation-based learning by users, by a process of analogical generalisation. It seems that work on machine learning can be usefully applied here. However, some of the problems with the machine learning technique of explanation-based learning are that it requires quite a rich domain theory, it assumes that all instances are correctly classified (i.e., there is no "noise"), and that the concept or model to be acquired is all or none. In contrast, users seem to perform partial mappings between new observations and previous events, recall the rule or explanation that accounts for the past episode, and then transform it, using the new observation to constrain the transformation, to produce the new hypothesis. What's needed is an account of explanation-based learning that doesn't depend on a lot of pre-existing knowledge, and that accounts for the acceptance of partial explanations. Lewis's work suggests that work on learning by transformational analogy is relevant here.

Learning and (socio)cognitive conflict

In summary, the value of abduction appears to be that it allows users to develop explanations of events in completely unfamiliar domains, leading to the development of fragments of knowledge and partial explanations. It is possible that users continue to have many models to account for different aspects of the system, and that no single model is robust and complete. This is the "distributed" perspective suggested by DiSessa (1986), which says that improvements in skill result from learning to apply the right model at the right time and perhaps refining the model with context-specific knowledge. However, naive and erroneous models may persist due to learners misinterpreting or distorting information to fit their model; learners may have several models for different instances of the same phenomena; or they may focus on only highly salient aspects of an event and ignore other aspects. Users may hang onto erroneous models perhaps because their real-world (as opposed to instructional) experiences tend to be such that inconsistencies don't occur. So one way in which understanding a system might be helped could be by revealing to learners the inconsistencies or conflicts amongst their different models.

The notion of conflict as a stimulus for learning is a Piagetian idea, and has generated a good deal of research in developmental psychology. However, it seems to be difficult to create conflict between observations in individual children. Research in developmental psychology also suggests that social (peer) interaction serves as an impetus for learning and development. Although the evidence is not entirely unequivocal, in general it seems that children working together on a problem benefit more individually than children working alone. Researchers within both the Vygotskian and Piagetian traditions argue that social interaction is only likely to be effective when there are some initial differences in perspectives between the members of the dyad, corresponding to what Wertsch (1984) calls different "situation definitions". In the course of communication, the conflict between these different perspectives can be resolved and a shared situation definition may be attained. When a child interacts with another in solving a problem the difference between them produces conflict which forces each child to restructure their representations or alter their strategies to resolve the conflict.

As Doise and Mugny (1984) point out, this is both a social and a cognitive disequilibrium:

> "It is cognitive disequilibrium in that the cognitive system is unable to integrate simultaneously its own responses and those of others within a single coherent whole... It is social disequilibrium since this is not simply cognitive disagreement. It involves relations between individuals for whom this conflict poses a social problem." (Doise & Mugny, 1984, p. 160.)

Constructive interaction

I have been interested in looking at joint-problem solving as a means of studying mental models and explanations developed by users. In successful joint problem-solving:

(a) because there are two people working on the problem, there is more of a possibility for alternative perspectives or interpretations;

(b) being confronted with a different point of view helps one to overcome "cognitive hysteresis";

(c) having different models or points of view stimulates people into articulating their point of view and using it to argue and criticise the other's point of view;

(d) if one member of the pair proposes a solution or an experiment to test a hypothesis, she is obliged to say why, or the other can be expected to object and ask for an explanation;

(e) thus, the pair naturally explain not only what they are thinking, but also why they think it — making a usually invisible process visible;

(f) the dialogue is intrinsic to the task of solving the problem, rather than (as is the case with typical single-subject think-aloud studies) being additional to the task.

Miyake (1986) has called this kind of joint problem-solving "constructive interaction". She asked pairs of subjects to try and figure out how a sewing machine makes its stitches, and from their protocols, she developed what she called a "function/mechanism hierarchy" to describe the explanation that her subjects developed about how the sewing machine worked. According to this framework, Miyake's subjects' understanding proceeded from a global functional understanding to local mechanistic understanding by descending "levels". When a function at one level of mechanism is identified and questioned as a problem (that is, subjects puzzle over how it is achieved — e.g., how does the upper thread intertwine with the lower thread?) it opens up a search for a lower level mechanism to explain it. This mechanism can then be decomposed into its subfunctions, and one of these functions can be posed as a problem, and so on.

What was "constructive" about the interaction in Miyake's study was seen in the way subjects divided up the task: subjects appeared to take different roles depending on their different focus or point of view. So, while one person led the interaction by engaging in a local task, the other observed and provided help by criticising and suggesting new ways of approaching the problem. The constructive part consists in that, when a subject working alone claims to have solved a problem, it can be difficult for her to come up with a counter example to test it. This testing process (criticism and validation checking) occurs naturally with two people — because each participant works from a different perspective or "starting schema", what is natural and obvious for one may not be so for the other. In fact Miyake found that self-criticising only accounted for 12% of incidents, implying that validation checking is hard to obtain with an individual system.

I have been interested in applying this "constructive interaction" paradigm to human-computer interaction. Some initial studies (cf. O'Malley, Draper & Riley, 1985) have given encouraging results. For example, in one of these studies the subjects' task was to solve a particular problem together concerning underlying processes to do with the Unix operating system. This study provides an interesting example in illustrating an aspect of the acceptability of explanations concerning convincing a user to abandon or change a pre-existing model or schema in the face of conflicting evidence. The evidence was provided in part by an experiment generated by the subjects in order to test out the correctness of the model that one of them had.

The topic being discussed by these subjects was the UNIX C-shell command interpreter, and the rules governing when variable values get passed to subordinate processes. The two participants knew the system fairly well but were not experts. The session revealed that they were both seeking different kinds of explanation, based on their different models of the system.

One of the subjects (A) had two problems that he wanted to figure out, and he felt that they were related. One concerned the fact that in using UNIX he had noticed that he often got processes listed when he typed *ps* (the command for listing processes running) that he hadn't explicitly set up himself, and that differed from the jobs he had running. He wanted to know what process had control over creating subprocesses and spawning new shells. He had also noticed a difference between the variables that were set in his environment (via ".login") and in the shell (via ".cshrc"). His model of operating systems viewed the command interpreter (the C-shell) as being the parent of all other processes, which never dies nor duplicates itself. So far as his model was concerned, the C-shell takes input from the terminal and runs the ".login" and ".cshrc" initialisation files upon login. B pointed out that this wasn't right because the login program is running prior to the C-shell. It is the login program which waits for input from the terminal.

B then started to construct an explanation of what the underlying job of the C-shell was, based on his knowledge as a programmer of the system primitives for process creation (called *fork* and *exec)*, in the hope that the puzzling surface behaviour of the C-shell might be understood. In this model, the shell is a program which can run several jobs. It generates several programs which listen to terminals (logins). Upon login, according to B, the shell does an *exec* in which it dies and is replaced by another C-shell.

Although he showed every sign of understanding B's explanation, A was still puzzled, and wondered what happens to the top-level process which spawned the login processes. (He was still convinced that the C-shell has overall control.) A suggested they try an experiment of explicitly executing an *exec* command from the C-shell. (This is possible, because the C-shell recognises *exec* as a command.)

As a control, the subjects typed *lf* (the command for listing files), which listed the files in their current directory, and, as they expected, returned the normal shell prompt (%). They then typed *exec lf*, which again listed their files, but instead of printing the normal shell prompt, it printed the login prompt. In other words, their session had been terminated and they were logged out.

This seemed to serve as an illuminating confirmation for B's theory. On this account, normally when a process is created the shell forks and creates a copy of itself, inheriting all the variables and their values, then "execs", or replaces itself with an instance of the program to be run. The other shell (the copy) waits for this process to finish or die. In the case of *exec lf*, the exec primitive overwrote the calling program with an instance of the new program. Thus, the C-shell had overwritten itself with the directory listing program, which had run normally and terminated. The system had detected that there were no more processes associated with that terminal and had prompted for a new login. (The information about the system primitives which could form the basis of an explanation is in fact contained somewhere in the documentation, but not in terms that related to observable events on the screen, e.g., to do with what happens when you login, and so on.)

When B offered this explanation, A didn't fail to understand it, but couldn't accept it because of a fundamental conflict with his model of operating systems (based on experience with a different system), in which the command interpreter is a part of the operating system which can never "die", nor allow itself to be replaced by some other program. It wasn't until B began adopting A's model himself, and trying to couch his explanation in those terms, that they started to make progress.

There were several parallels between what happened in this study and what Miyake noted from her studies. For example, in both studies it was noted that subjects implicitly divided the task so that one person tried to come up with solutions while the other played the role of criticising these solutions and suggesting alternatives. Interestingly though, it was the inquirer, or the one who knew the "least", who did the criticising, suggested new ways of looking at the problem, and provided validation checks.

This division of labour (the leader/follower or task-doer/observer pattern) in co-operative dialogues has also been found in recent studies by Garrod (1987; Garrod & Anderson, in press). Garrod argues that in dialogue, meaning can be viewed as a property that *emerges* from the interaction, rather than an inherent and "fixed" property of language. His studies (involving maze-following co-ordination tasks) concern how participants come to a mutual view through both semantic and conceptual co-ordination. This co-ordination is most successfully achieved through *implicit*, rather than explicit, means. This sort of

strategy has also been discussed in Power's work on joint planning (Power, 1987). Power suggests that a successful means of contributing to a joint plan is to adopt a strategy of "presumption" (i.e., behaving as if the goal is already present) rather than the more explicit strategy of agreement (i.e., proposing and accepting goals explicitly). In presumption, the agent performs an action which commits the other person to go along with it.

There are of course considerable differences between the kind of study done by Miyake and the study by O'Malley et al. (O'Malley, Draper & Riley, 1985) involving computing systems. Firstly, Miyake used a physical device. In the case of the computer system, the structure of the device is not visible, but has to be inferred through observing the behaviour of the system. So, secondly, whereas Miyake could observe "conceptual points of view" in the sense of focussing on a physical spot, the "conceptual point of view" in the case of the computer system is the perspective from one person's *model* of the system.

Recent work on children's joint problem solving with computers (Crook, 1987) also shows that the nature of the task is an important factor in making interaction constructive for both participants. One of the tasks used by Crook was a maze following game, which wasn't specifically designed to involve co-ordination of participants moves, as was the task Garrod used. This task required distributing attention between the maze itself and the sequence of steps being written to traverse it. Most children were reluctant to do this at times when they weren't keying in the instructions themselves, so collaboration rarely went beyond taking turns at the whole task. Another task involved solving anagrams — here there was a tendancy for one child to spot the solution immediately, leaving the other to just key it in. The successful tasks involved having to construct or discover rules, such as completing series, or an adventure game. In these tasks, the approach to the solution was more differentiated, prompting discussion of competing hypotheses.

Another important difference between Miyake's study and the computer study described above can be seen in terms of the kind of explanation subjects were producing, based largely on the nature of the task. The task for Miyake's subjects was well-structured: She used a physical device and the explanations generated by her subjects were largely reductive, and in general they stuck to the function/mechanism hierarchy she described, mostly proceeding "downwards" through the levels. However, the nature of the system used in the UNIX study meant that the task was much more ill-structured: subjects had to *infer* a structure within which to explore functions and mechanisms, and from which to generate testable predictions. Thus, there was much more opportunity to observe the kind of "abductive" inferences discussed earlier.

Explanations are not given but (jointly) constructed

My interest in these joint problem solving situations is not just in using them as a source of data for studying the development of users' models. At the beginning of this paper I claimed that we need to study human-human explanation in order to be able to design systems that can provide intelligent explanations, give advice and interact cooperatively. The discussion up to this point has emphasised conflict, rather than co-operation, as being a stimulus for joint problem solving. In fact, conflict could not be useful unless the participants had a common aim to resolve it — to be co-operative. This highlights a difference between explanations that are simply a case of so many steps towards a solution (where the explanation exists in some objective sense and is unique) and the *joint construction* of a solution. The latter involves compromise on the part of the inquirer or the explainer, or both. So, although it is claimed here that situations of

constructive interaction can make visible what is often hidden when people solve problems on their own, it is not claimed that all such processes are revealed. Moreover, it may be that what is observed in the interaction is different in several ways to what goes on in the head of the individual problem-solver. Antaki (1985) argues that the elaboration of causal structures to be found in explanations occuring in conversation, concerns the use of conventional devices or rhetoric, rather than representing individuals' cognitive structures.

According to Antaki, in ordinary explanations (i.e.. justifying or accounting for some behaviour or attitude), the desire to actually persuade may be less keen than the desire to show competence in using the conventions of persuasion. He bases this on studies which show that subjects use warrants (cf. Toulmin, 1958) more often than data to back their claims — Le., they concentrate on giving information concerning why the other particpant should believe them. However, this does not mean that subjects are less keen to persuade than to demonstrate their ability to argue. It could be that they are providing a justification for accepting their reasoning, since it is the means by which they came to make the link between the data and the conclusion. In other words, they are giving the other person a reason why there account should be believed.

Nevertheless, it is claimed that some of the knowledge which subjects bring to bear is revealed — the process of arguing against one person's point of view involves explaining how you view the situation. But, although I've argued that conflict between participants can be revealing about their models of the domain, the process of negotiation as *compromise* may obscure what goes on in individuals in an interaction. That is, participants may have to distort their explanation (based on their own knowledge or model) in order to be understood. That is, there may be some "shareability constraints" (cf. Freyd. 1983) on the interaction.

According to Freyd's "shareability hypothesis", when people get together certain dynamic processes, to do with the need to share knowledge, lead to the emergence of certain knowledge structures which are different to those individually held. It might also be that the very act of trying to observe an individual's knowledge structure demands that the person share the knowledge, thus causing an emergent structure. Freyd argues that *because* shared knowledge must be shared there are certain constraints that emerge on a second-order level — it is the interaction of human minds that forms knowledge systems. This is very close to the Vygotskian notion that all learning begins on an interpsychological plane, which, through social interaction, eventually becomes the intrapsychological plane. In fact, Miyake found that individuals* starting and ending models or schemas were very different, even though the jointly constructed explanation was mutually satisfactory.

Freyd (op. cit.) also argues that this account helps to explain the preponderance of analogies that people use in the explanations of new terms and concepts, since analogies work by isolating one or more dimensions and pointing out common values along these dimensions. Thus, the use of analogies in explanations may have a basis in the shareability of simple representational structures. Freyd argues that the shareability hypothesis accounts for the fact that even though concept terms may lie along a continuum, they attain discrete values along these dimensions:

> "... one would expect that of all the possible dimensions available for categorising real objects or abstract ideas, people would tend toward isolating a few dimensions that they can apply to a number of knowledge domains to ease the problems of agreeing on the meanings of new terms. In this way, ease of shareability would begin to shape the knowledge structure." (Freyd, 1983, p. 198.)

This is also the point that emerges from Bartlett's (1932) studies of memory. Edwards and Middleton (1986) note that Bartlett's studies were concerned not with the ways in which social factors affect individual cognition (where two heads are seen to be more effective than one) but rather with the inherently social basis of mentality itself. In discussing the serial reproduction studies, they conclude that "It is not so much that people are not very good at remembering, as that they are very good at making the past serve the present" — i.e., the present communicative and social purpose (Edwards & Middleton, 1986, p. 88). David Middleton (personal communication) makes a distinction between discourse as facilitative as opposed to inherent or constitutive. He notes that facilitative perspective is a slightly weaker position — two heads are better than one with respect to the number of ideas or alternatives that come up for consideration. The constitutive position is a stronger one — social interaction provides a qualitatively different set of constraints for problem solving.

Conclusions

Explanation is an inherently communicative activity. Our ordinary language use of explanation (e.g., "giving" an explanation) is often misleading, since it suggests that there is some objective or independently existing account that somehow has to be transmitted from one person's head to another's. It is also misleading to try and separate the "communication of an explanation" from the "process of explaining". Power (1987) has noted that, in joint planning, rather than thinking of participants as having private goals, we should think of them as executing together a single shared joint plan. In line with this, it may be a mistake to think about explanation as being either about individual cognitive change or social communicative processes, but that it should be viewed in the sense of "distributed cognition" — i.e., explanation is interactive not just in the sense that participants negotiate steps towards a solution, but in the sense that *explanation is an emergent property of the interactive situation,* contributed to by both participants. So in joint-problem solving (constructive interaction), the task is not so much to discover the mental models held by subjects individually, but to look at the development of a mental model, and at how it changes during the interaction.

This view of explanation as joint problem solving is similar to recent suggestions, made independently by Gaines (1986) and Hutchins (unpublished manuscript), that a new way of looking at performance in tasks is in terms of the "social organisation of distributed cognition". In this view, rather than looking primarily at the individual and then trying to see how a set of individuals could function together, one might take a group of people jointly performing a task as the fundamental unit. Communication among the actors is then seen as a process internal to the cognitive system. Computational media, such as diagrams, computer systems, etc., and the computations carried out on them are seen as representations and processes internal to the system. The suggestion is that because the cognitive activity is distributed across a social network, these internal processes and internal communications are directly observable.

References

Antaki, C. (1985). Ordinary explanation in conversation: Causal structures and their defence. *European Journal of Social Psychology*, **15**, 213-230.

Bartlett, F.C. (1932). *Remembering: A Study in Experimental and Social Psychology*. Cambridge: Cambridge University Press.

Burks, A.W. (1958). *Collected Papers of Charles Sanders Peirce, Vol. VII, Science and Philosophy*. Cambridge, MA: Harvard University Press.

Carroll, J.M. & Mack, R.L. (1985). Metaphor, computing systems, and active learning. *International Journal of Man-Machine Studies*, **22**, 39-57.

Crook, C. (1987). Computers in the classroom: Defining a social context. In J.C. Rutkowska & C. Crook (Eds.) *Computers, Cognition and Development: Issues for Psychology and Education*. Chichester: John Wiley.

DiSessa, A.A. (1986). Models of computation. In D.A. Norman & S.W. Draper (Eds.) *User Centered System Design: New Perspectives in Human-Computer Interaction*. Hillsdale, N.J.: Lawrence Erlbaum Associates.

Doise, W. & Mugny, G. (1984). *The Social Development of the Intellect*. Oxford: Pergamon Press.

Draper, S.W. (1985). The nature of expertise in UNIX. In B. Shackel (Ed.) *Human-Computer Interaction — Interact'84*. Amsterdam: North-Holland.

Edwards, D. & Middleton, D. (1986). Conversations with Bartlett. *Quarterly Newsletter of the Laboratory of Comparative Human Cognition*, **8**, 79-89.

Fikes, R. (1982). A commitment-based framework for describing informal cooperative work. *Cognitive Science*, **6**, 331-347.

Freyd, J.J. (1983). Shareability: The social psychology of epistemology. *Cognitive Science*, **7**, 191-210.

Gaines, B.R. (1986). Foundations of knowledge engineering. In M.A. Bramer (Ed.) *Research and Development in Expert Systems III*. Cambridge: Cambridge University Press.

Garrod, S. (1987). Some observations on semantic and conceptual co-ordination in task oriented dialogue. *Proceedings of the 2nd Alvey IKBS Workshop on Explanation*. (University of Surrey, January 8-9 1987.)

Garrod, S. & Anderson, A. (In press). Saying what you mean in conversational dialogue: A study in semantic co-ordination. *Cognition*, in press.

Kahneman, D. & Miller, D.T. (1986). Norm theory: Comparing reality to its alternatives. *Psychological Review*, **93**, 136-153.

Kidd, A.L. (1985). What do users ask? — Some thoughts on diagnostic advice. In M.

Merry (Ed.) *Expert Systems '85.* Cambridge: Cambridge University Press.

Klayman, J. & Ha, Y-W. (1987). Confirmation, disconfirmation, and information in hypothesis testing. *Psychological Review,* 94, 211-228.

Lewis, C. (1986). Understanding what's happening in system interactions. In D.A. Norman & S.W. Draper (Eds.) *User Centered System Design: New Perspectives in Human-Computer Interaction.* Hillsdale, N.J.: Lawrence Erlbaum Associates.

Lewis, C. & Mack, R. (1984). Learning to use a text processing system: Evidence from "thinking aloud" protocols. In A. Janda (Ed.) *Hitman Factors in Computing Systems — /.* Amsterdam: North-Holland.

Lewis, C.H. & Norman, D.A. (1986). Designing for error. In D.A. Norman & S.W. Draper (Eds.) *User Centered System Design: New Perspectives on Human-Computer Interaction.* Hillsdale, N.J.: Lawrence Erlbaum Associates.

Mack, R.L., Lewis, C.H. & Carroll, J.M. (1983). Learning to use word processors: Problems and prospects. *ACM Transactions on Office Information Systems, 1₉* 254-271.

Miller, P.L. (1984). *A Critiquing Approach to Expert Computer Advice: ATTENDING.* London: Pitman.

Miyake, N. (1986). Constructive interaction and the iterative process of understanding. *Cognitive Science,* 10, 151-177.

Norman, D.A. (1981). Categorization of action slips. *Psychological Review,* 88, 1-15.

O'Malley, C.E., Draper, S.W. & Riley, M.S. (1985). Constructive interaction: A method for studying human-computer-human interaction. In B. Shackel (Ed.) *Human-Computer Interaction — Interact*84.* Amsterdam: North-Holland.

O'Malley, C.E., Smolensky. P., Bannon, L., Conway, E., Graham, J., Sokolov, J. & Monty, M. (1984). A proposal for user centered system documentation. In A. Janda (Ed.) *Human Factors in Computing Systems — /.* Amsterdam: North-Holland.

Orr, J.E. (1986). Narratives at work: Story telling as cooperative diagnostic activity. *Proceedings of the CSCW'86 Conference on Computer-Supported Cooperative Work.* (Austin, Texas, December 3-5 1986.)

Owen, D. (1986). Answers first, then questions. In D.A. Norman & S.W. Draper (Eds.) *User Centered System Design: New Perspectives in Human-Computer Interaction.* Hillsdale, N.J.: Lawrence Erlbaum Associates.

Power, R. (1987). Efficiency in conversation. *Proceedings of the 2nd Alvey IKBS Workshop on Explanation.* (University of Surrey, January 8-9 1987.)

Ross, B.H. & Moran, T.P. (1984). Remindings and their effects in learning a text editor. In A. Janda (Ed.) *Human Factors in Computing Systems — /.* Amsterdam: North-Holland.

Scharer, L.L. (1983). User training: Less is more. *Datamation,* October 1983, 175-182.

Shrager, J. & Klahr, D. (1986). Instructionless learning about a complex device: The paradigm and some observations. *International Journal of Man-Machine Studies*, **25**, 153-189.

Toulmin, S. (1958). *The Uses of Argument*. Cambridge: Cambridge University Press.

Wertsch, J. (1984). The zone of proximal development: Some conceptual issues. In B. Rogoff & J. Wertsch (Eds.) *Children's Learning in the "Zone of Proximal Development"*. San Francisco: Jossey-Bass.