On "Technomethodology": Foundational Relationships between Ethnomethodology and System Design

Paul Dourish

Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto
CA 94304 USA

dourish@parc.xerox.com

Graham Button

Xerox Research Centre Europe
61 Regent Street
Cambridge CB2 1AB
United Kingdom

button@xrce.xerox.com

Draft (submitted for review). Please do not cite or distribute without authors' permission.

Abstract

Over the past ten years, the use of sociological methods and sociological reasoning have become more prominent in the analysis and design of interactive systems. For a variety of reasons, one form of sociological enquiry, ethnomethodology, has become a something of a favoured approach. Our goal in this paper is to investigate the consequences of approaching system design from the ethnomethodological perspective. In particular, we are concerned with how ethnomethodology can take a foundational place in the very notion of system design, rather than simply being employed as a resource in aspects of the process such as requirements elicitation and specification.

We begin by outlining the basic elements of ethnomethodology, and discussing the place that it has come to occupy in CSCW and, increasingly, in HCI. We discuss current approaches to the use of ethnomethodology in systems design, and point to the contrast between the use of ethnomethodology for *critique* and for *design*. Currently, understandings of how to use ethnomethodology as a primary aspect of system design are lacking. We outline a new approach and present an extended example of its use. This approach takes as its starting point a relationship between ethnomethodology and system design which is a foundational, theoretical matter rather than simply one of design practice and process. From this foundation, we believe, emerges a new model of interaction with computer systems which is based on ethnomethodological perspectives on everyday human social action.

1 Introduction

One of the more significant trends in HCI research and practice over the past decade has been the increasing influence of sociological perspectives in the design and evaluation of interactive systems. Sociological understandings and methods have been used to study the settings in which work has conducted, to inspire and guide the design of interactive systems, and to evaluate those systems in real working conditions. The uptake of sociological research has been most pronounced within the domain of Computer-Supported Cooperative Work (CSCW), where, clearly, sociological approaches lend themselves well to a primary focus on interaction between individuals and groups, rather than simply between the eponymous human and computer of HCI. CSCW research has highlighted the social setting of computer use, and so set up for itself a framework within which sociology has direct applications. More recently, however, sociological perspectives have begun to permeate HCI more generally. Within HCI, sociological methods complement (and sometimes challenge) the technical and psychological perspectives around which the field was originally organised, and have become increasingly accepted and even expected as a component of HCI research.

It would be unwise to imagine, however, that "sociology" is all of a piece—far from it. There are any number of particular branches, each with their own perspectives, orientations, methods and concerns, coming together under the sociological umbrella. Shapiro (1994) provides something of a "travel brochure" for some of this vast terrain. In this paper, we will be concerned with one particular branch of sociological investigation, ethnomethodology. For a variety of reasons, some of which we will explore, ethnomethodology has become something of a favoured (or, at least, more prominent) perspective amongst the sociological positions exploited in CSCW and HCI.

We come at this as representatives and practitioners of each discipline. One of us (Dourish) is a computer scientist, and one (Button) an ethnomethodologist. Over the past few years, in work conducted separately and together, we have been concerned with how the design of collaborative and interactive systems can be grounded in sociological understandings of action and interaction. Our goal has not simply been to develop a model of design which is responsive or respectful to observations of particular social settings, nor have we been attempting to formulate a design method by which sociologists and computer scientists can work together on design problems. Rather, we have sought to develop a form of technological design which is fundamentally grounded in the understandings which sociological perspectives employ. Our fundamental position is that the relationship between social and computational sciences is *more than a practical problem*. Our goal has been to develop a stance in which ethnomethodology and computer science play equally significant roles (rather than grafting one onto the other), and so our approach is radically novel for both disciplines. We use the term "technomethodology" to emphasise that it is something new, drawing from each side, but different from either.

Our goal in this paper is to motivate, introduce and illustrate the approach we have arrived at. As a consequence, this paper is rather unusual amongst those in HCI or CSCW emerging from collaborative work by sociologists and computer scientists. We do not present a set of technical requirements derived from a field study, nor do we present a system design which incorporates the lessons of ethnographic investigation. Instead, this paper is concerned with the basis on which those other sorts of research can be conducted. Since we are more concerned with computational design per se than with specific system-design efforts, our primary illustration is not a particular system, but a reconceptualisation of a particular foundational element of interactive system design (viz., the notion of abstraction). Before we can proceed to this, though, we need to spend some time considering just what ethnomethodology *is*, and how it has come to play its current role in HCI.

2 What is Ethnomethodology?

Despite the interest in ethnomethodological ideas in CSCW and increasingly in HCI research, those ideas themselves remain remarkably poorly understood. We can speculate about the reasons. Perhaps it is because ethnomethodology has largely been conducted by ethnomethodologists, rather than becoming a more available approach to researchers at large; or, perhaps it is due to the relative opacity of much of its writings ("I know of no discipline," comments Eric Livingstone (1988), "which has suffered more at the hands of its expositors than ethnomethodology.") In this section, we will attempt to give a flavour of the ethnomethodological position[1], with two concerns: first, to introduce some conceptual foundations on which we will build later; and second, to give a flavour of the way in which ethnomethodology differs from other approaches.

2.1 The Origins of Ethnomethodology

Historically, ethnomethodology has its roots in the work of Harold Garfinkel, beginning in the late 1950's and subsequently developed through the 1960's and early 1970's by Garfinkel and colleagues, perhaps most notably Harvey Sacks, the founder of Conversation Analysis.

Garfinkel's objective was a *respecification* of the subject matter and methodological approaches of sociology. At the time, the prevailing school of (especially American) sociological thought was structural-functionalism, most fully developed at that point in the work of Talcott Parsons and especially his "The Structure of Social Action" (Parsons, 1937). Ethnomethodology arises from Garfinkel's confrontation with this perspective: "Inspired by *The Structure of Social Action*, ethnomethodology undertook the task of respecifying the production and accountability of immortal, ordinary society" (Garfinkel, 1991). So Garfinkel's project was not merely to critique Parsons, but to use Parsons has a springboard for the respecification of sociology—a respecification in which the very terms of reference of sociology were re-examined and the questions of what sociology could be and how it could be conducted were opened to scrutiny.

2.2 The Objective Reality of Social Facts

A critical focus of this respecification is ethnomethodology's rejection of the traditional approach to the relationship between practical social action and the sociological "rules" by which stable social order is established and maintained (known as the "problem of social order"). Parsons had considered this problem as a matter of concerted action. He described how, in performing activities in accordance with reciprocally shared rules and norms, social actors achieve the co-ordination of their activities. In this, he followed sociology's historical belief that "the objective reality of social facts is sociology's fundamental principle", and so went about investigating how those facts governed everyday social action.

In his respecification, however, Garfinkel drew attention to the *work* that goes into the production of social order, underscoring how social order is "made to work" in the actions and interactions of its members. Social order does not simply exist, and social action is not simply determined from it; and so, social order and social action cannot be approached independently. So, for ethnomethodology, the "objective reality of social facts" is not sociology' fundamental principle, but its fundamental *phenomenon*. As a result, precisely how it is this phenomenon is its achieved and manifested become the primary subject matter for investigation. Ethnomethodology turned its analytic attention to the ways in which

1. Readers who wish to pursue this in more detail are referred to Garfinkel (1967) or Heritage (1984).

everyday social action was achieved, looking directly within the circumstances of action for evidence of the methods by which individuals *achieved* precisely the stable social order which traditional sociology had defined by theoretical fiat.

2.3 The Accountability of Social Action

In looking at the emergence of social facts from the everyday details of what people do, a primary concern for the ethnomethodologists was not just how individuals engaged in rational social behaviour, but also how they could be *seen by others* to be engaged in it. After all, the ethnomethodologists reasoned, "rational social behaviour" was observable not only to learned professors of sociology who knew "the rules", but also to individuals engaged in everyday practical action. Everybody, everyday, knows what it is and what it looks like.

The emphasis on *practical* social action is both intentional and highly significant. It implies two things. First, it emphasises that social action unfolds in people going about everyday activities; that is, there are matters at hand to be attended to, and the object of activity is to attend to these matters, not to reproduce the stable structures of society. When I call a plumber and ask him to fix a leaking pipe, my concern is to avoid a pool of water on the floor, not to reproduce a pattern of social interaction based on contracts and wage-labour. Second, focusing on practical social action draws attention to the way in which those practical concerns, the matters at hand, take up a critical role in the understanding and production of action. Understandings of social action are formed "for practical purposes".

Garfinkel held that the procedures for the production of action, and the procedures for observing people as being engaged in rational social action, were one and the same. Human social action is "reflexively accountable"—that is, that in the way in which it is organised it provides the means to recognise it as what it is (accountable), and does so within the very fact of its production (reflexivity), rather than within some wider frame of "social meaning". So, the organisation of action, within specific (and innumerable) circumstances of its production, furnishes to others the means to recognise, observe and report upon it. For example, an "ordinary social fact", such as a conversational greeting, is produced in conversation *as* an observable greeting, not just by using a recognisable greeting term such as 'Hello', but also by placing that term in an interactionally-organised position within the course of a conversation. The "work" which the word does is a feature of how the word is used rather than simply what the word is. As a general concern, this property of human social action—*accountability*—became one of the critical analytic features of ethnomethodological studies[2].

The idea of the reflexive accountability of action also provided ethnomethodology with an analytical warrant for a particular form of investigation; one which looked in extreme detail at specific instances of action, looking *within those circumstances* for the exhibition of members' methods of acting and seeing. This is, perhaps, most vividly demonstrated in the development of Conversation Analysis (CA), a particular form of ethnomethodology, de-

2. In our experience, one confusion about ethnomethodology in HCI has arisen precisely from Garfinkel's use of the term "accountable". As we have outlined here, the accountability of action lies in the way in which it "gives an account of itself as itself", and is "observable and reportable" as such. It does not refer, then, to a political or moral accountability for ones actions, or provide for an opportunity to be taken to task over them. The fact that action is accountable has nothing to do with the fact that someone may be "held accountable" for their action.

veloped primarily by Harvey Sacks (1992). CA studies used fragments of naturally occurring conversation as data, looking within the conversational data itself for the mechanisms by which conversation was organised. Reflecting ethnomethodology's concern with practical action, CA views conversation as social action, rather than as the articulation of internal mental states; and, on that basis, analyses this action to see how aspects of conversation (such as topic nomination) are managed as a practical activity. So, for example, in a CA investigation of greetings, the focus is not on what sorts of terms (like the word "hello") might carry "greeting-ness" as intrinsic properties which people then deploy in different circumstances. Instead, it focuses on the interactional work they do, the implications they have for what comes next, and for how they are used in the solution of the problem of how to concert social action on this occasion.

Since accountability is a fundamental and "irremediable" property of social action, the ethnomethodologists contended that these sorts of analysis could be conducted not only in the domain of conversation, but in any domain of social action. Their concern with how social action was made to be and seen to be rational allowed them to treat *any* social action as an occasion of lay "sociological theorising". Conversely (and perhaps rather mischievously), it also allowed them to treat "professional sociological theorising"—that is, the professional practice of sociology, carried out in books, journals, conferences and lecture halls—as just as another domain of everyday practical action. While this might well have been an interesting example of the universality which the ethnomethodologists were seeking, it has not endeared them to other sociologists.

From their concern with conversation and with language as social action, Garfinkel and Sacks (1970) emphasised a focus on "member" not as an instance of a delineable social grouping, but as "mastery of natural language use". For them, the mastery lies not in the grasp of syntax or grammar, but rather in the competent use of language (that is, of course, social action) in appropriate (social) settings. Such competent language use means to be able to use language, or to act, appropriately for the setting, which in turn depends upon the appropriate exercise of understandings of (in Garfinkel's term) "what everyone knows that everyone knows". For example, Sacks' (1984) essay, "On doing 'being ordinary'" illustrates the way in which such understandings are applied in everyday conversation so as to explicitly render some elements of everyday life as commonplace and ordinary, while others are made particular and exceptional, as appropriate for the setting within which the conversation/action unfolds.

The emphasis on the practical, situated, account-able and ordinary character of social action form the relevant core elements of ethnomethodology that are relevant for our purposes here. At the same time, they are core elements which are in opposition to the prevailing currents, then and now, of "conventional sociology". Ethnomethodology, taking the stability of social facts as a phenomenon to be studied rather than as a principle to be assumed, is concerned with how, in the conduct of their everyday affairs, people manage to find and make rational social actions. In particular, it looks at how they do this by exercising understandings of "what everyone knows that everyone knows" and by finding, within the immediate circumstances of action, the means to understand and interpret it for practical purposes.

2.4 *Ethnomethodology and Ethnography*

Although it is not immediately relevant to the argument which we wish to develop in this paper, it is nonetheless important as part of this introduction to make clear the distinction between ethnomethodology and ethnography. The fact that ethnomethodologists often use ethnographically-generated materials in their analysis (Hughes et al., 1993) may lead those who are not sufficiently familiar with the disciplines to conclude that they are one and the same thing. They are decidedly not.

As outlined above, ethnomethodology is a particular analytic orientation to the practical issue of the problem of social order. It sets out a policy for the study of practical social action. Ethnography, on the other hand, is a form of investigative fieldwork and analysis. Ethnography considerably predates ethnomethodology. Modern ethnography emerges primarily from the work of Bronislaw Malinowski in the early part of the this century, particularly his work in the Trobriand Islands during the First World War.[3] Ethnography is best seen in contrast with other methods in anthropology; qualitative rather than quantitative, with an emphasis on the "member's point of view", and, critically, with a focus on the member's *experience* rather than simply his or her action. Ethnography has grown to be the predominant perspective of anthropological field workers, not simply for the collection of their materials, but also for their organisation, interpretation and presentation. Within ethnography, however, numerous analytic orientations may operate. For instance, ethnographic field techniques have also been used by many in the "Chicago school" of Human Ecologists and Symbolic Interaction in the study of social life and of work.

Part of the confusion between these terms and approaches in HCI rests on the fact that ethnomethodology often makes use of ethnographically-gathered materials. An ethnomethodologist going into the field to collect data is likely to use ethnographic techniques, and so to an observer of field-workers, might seem indistinguishable from, say, a symbolic interactionist doing the same. Indeed, they might both be adequately labelled "ethnographers"[4].The point of difference comes into play in the "analytic mentality" they display in the selection of phenomena and topics for investigation and in the issues they would want to draw attention to in the materials gathered. Some of the confusion, then, arises in the way in which these concerns have entered the domain of HCI research, which is the topic that shall now concern us.

3 The Rise of Ethnomethodology in HCI Research

Ethnomethodology has become a prominent form of sociological analysis in HCI and CSCW. This is particularly intriguing for us since, as should be clear from discussion above, ethnomethodology is only one amongst a wide range of sociological perspectives (and a fairly small one at that). Our goal here is to consider why, other than a partisan belief that ethnomethodology is a methodologically and analytically superior form of sociological reasoning, this has come to be the case.

3. The historical context of Malinowski's work is a different but fascinating story in itself. Malinowski spent the war years in the Trobriand Islands as an arrangement to avoid internment as a foreign national at the outbreak of hostilities. See Anderson (1996).

4. Care must be taken here, too. Anderson (1991) makes the point that what system design often sees as the value of "ethnography" is often simply the value of field-work, and discusses how ethnography itself also comes with its own analytically and politically predispositional baggage.

More or less ethnomethodologically-oriented investigations are now regularly presented at CSCW conferences[5] and increasingly at HCI conferences[6]. Ethnomethodology is being used to inform design through:

1. fieldwork investigations that develop an understanding of work and organisations from the "inside", providing innovative insights into the organisational situatedness of work and the methods and practices through which work activities and interactions are assembled and which may be used in the design of technology to support work; and
2. developing an understanding of the temporal organisation of activities and interactions, revealing them to be a moment-by-moment organisation, and in so doing furnishing new concepts around which to generally consider the design of technology.

These understandings, further, allow a new focus on the relationship between technology and the accomplishment of work. From this analysis comes the opportunity to use ethnomethodological analyses as the basis of design and redesign of work technologies.

There are a number of inter-related reasons why some within the design community are taking up these two sets of issues.

3.1 Plans and Situated Actions

One primary reason for the widespread influence of ethnomethodology in interactive systems design is the role of Lucy Suchman's book, "Plans and Situated Actions" (Suchman, 1987). Suchman's book formulated a telling and forceful critique of the user modeling and planning-based approaches common in both HCI and Artificial Intelligence. It is very widely read and cited (perhaps more widely cited than read) in the HCI literature, and firmly established the relevance of sociological and anthropological reasoning for the problem of human-computer interaction. As such, the book and the argument it puts forward have come to occupy an almost iconic position within the field: one which, due to a number of misunderstandings, Suchman has repeatedly been forced to clarify in the decade since its publication.

The disturbingly common caricature of her position is that there are no plans, but only "situated actions"—improvised behaviours arising in response to the immediate circumstances in which actors find themselves and in which action is situated. In fact, as Suchman has been at pains to point out, she did, in fact, accord an important status to plans as *resources* for the conduct of work; her argument was that plans are one of a range of resources which guide the moment-by-moment sequential organisation of activity, rather than laying out a sequence of work which is then blindly interpreted.

The argument which Suchman lays out in "Plans and Situated Actions" was partly founded in the work of ethnomethodology. Garfinkel used the term "judgemental dope" to characterise traditional sociology's view of members' practical decision making, as they blindly act in accordance with theoretically formalised systems of rules and norms. Garfinkel attempted to relocate practical decision making to a realm of relevantly-invoked situated ac-

5. Examples include Anderson et al., 1993; Bentley et al., 1992; Bowers, 1994; Bowers et al., 1995; Heath et al., 1993; and Rouncefield et al., 1994.
6. Examples include Bowers and Pycock, 1994; Button and Sharrock, 1995; and Heath and Luff, 1991.

tions in local circumstances. Similarly, Suchman emphasised a perspective on purposeful human action as situated in (and organised around) the context of particular circumstances. Suchman illustrated her argument with detailed examples drawn from a laboratory study of the use of a complex photocopier, and as she has pointed out, laboratory studies are hardly the stuff of ethnomethodology. However, her argument and analysis drew strongly on the ethnomethodological tradition, and introduced it to the HCI community. The HCI community has never recovered. In a recent book collecting essays on different social perspectives on HCI (Thomas, 1995), eleven out of the twelve essays cited “Plans and Situated Actions” or Suchman’s subsequent work.

Suchman’s book has had a significant influence in HCI design, and in related areas concerned with the design of computer systems supporting working activity. Many within HCI and CSCW have taken up Suchman’s concern with work settings and the detail of everyday working practices; and, as they have taken on board Suchman’s arguments, they have also taken on, perhaps unwittingly, an ethnomethodological influence.

3.2 Participatory Design

One particular group who have been particularly influenced by “Plans and Situated Actions” deserve especial mention. For a long time, there has been, within HCI, a strong and vocal group who have consistently argued that the requirements for technology should be developed directly around the work situation of the technology’s users. The Participatory Design movement, in particular, has made considerable strides in developing methods and perspectives on interactive systems design from this position, for both practical reasons (concerning the efficient and fluid accomplishment of work, and support the acceptance of technology) and political ones (emphasising the importance of the worker’s voice in issues of workplace management and development).

Since ethnomethodology is generally concerned with the “detailed and observable practices which make up the incarnate production of ordinary social facts” (Lynch et al., 1983), its investigations of particular work domains contain rich descriptions of work practice. They begin with what is involved in the everyday accomplishment of work, not in abstract models of the activity, whether those abstract models are laid down by “professional sociology” or by management. This may suggest to those already concerned with the relationship between work and design that they have an “analytic ally” in ethnomethodology; that it can provide a methodological warrant for a primary concern with the details of work practice in the design of new technologies. So ethnomethodological perspectives, and ethnographic field techniques, found a receptive audience in this community, who found in it a resource for methodological sustenance or even empirical descriptions of work.

3.3 Ethnography

The rich descriptions of work on which ethnomethodological studies are often based have similarly played a role in making ethnomethodology (or ethnomethodologists) appealing to those seeking to ground the design of interactive technologies on studies of the performance of work. Ethnomethodology is here adopted as part of a general concern with the use of field investigations in design (Plowman et al., 1995).

Of course, as we have already described, ethnomethodology is scarcely the first form of sociological investigation to make use of ethnographic or other field techniques in working settings. The "Chicago School" of sociology which emerged in the 1920s used ethnography in turning an anthropological eye not to the tribes of the south Pacific but to the life of the American cities, and so ethnography has become a technique which has been widely applied to the studies of work settings from the perspective of technological design and evaluation.

So, drawing on the sorts of confusions which we attempted to resolve earlier, it is possible that ethnomethodology comes, in part, to HCI and CSCW research in the guise of experienced fieldworkers, and, as a result, ethnomethodology may, to an extent, be basking in the sun of ethnography.

3.4 HCI in Transition

Finally, here, Grudin (1990) has argued that HCI has passed through a number of stages in its development, to reach its current position. In his characterisation, it is currently moving from the fourth stage, which is focussed on a dialogue with the user, to a fifth stage. This fifth stage which we are currently approaching is one focussed not around the individual, but around the work setting.

From this perspective, we can, again, see that ethnomethodology's concern with the organisational situatedness of work might be appealing to many concerned with HCI design. Ethnomethodology may be, for some, a port in the storm of transition. With its focus on the setting and situation of working activity, it may be seen as offering candidate solutions to the problems of incorporating into HCI design understandings of work setting as well as work practice.

4 Ethnomethodological Studies of Technological Work

As we have stated, a variety of studies have applied ethnomethodological understandings to the lived experience of work with technology, and have been used, in turn, to support the development of new technologies and new approaches to computational support for work. As might be expected, a focus on the variety of ways in which the sequential organisation of working activities is organised, and the detailed practices by which they manage their work, have been a common focus of these studies.

Our goal here is not to reproduce the detailed findings of these studies. Instead, we are concerned with the relationship between the disciplines of ethnomethodology and computer systems design which these investigations embody. In particular, our focus in this section will be on how ethnomethodological understandings make their way into novel system designs—on how system design "learns" from ethnomethodology.

4.1 Learning from Ethnomethodologists

To date, the most numerous examples of ethnomethodologically-informed system design have been conducted by bringing together ethnomethodologists and computer scientists in multi-disciplinary design teams. The investigation of Air Traffic Control by a group from Lancaster University (Bentley et al., 1992; Hughes et al., 1993) exemplifies this approach, and shows how valuable it can be. In this approach, a disciplinary division of labour typically emerges. Ethnomethodologists are sent into the field, and return brimming with observations and an analytic frame within which to interpret them. These observations become requirements for the system design process, more or less formally. The ethnomethodologists will typically also be involved in the ongoing evaluation of design alternatives, acting as proxy for the end-users, or, perhaps more accurately, as proxy for the work setting itself. Otherwise, they hand their requirements to their computer science colleagues, who then build the system on this basis. The fact that these requirements are derived from ethnomethodological studies and understandings is all but invisible.

This approach, and the division of labour it sets up, has become so paradigmatic at least within CSCW that it has even driven others to frame their work *as if* it were being conducted in this way, even when the setting is quite different. Plowman et al. (1995) have observed that, in order to be published in the CSCW literature, it is almost required of field study reports that they conclude with a section on "Implications for Design" and a set of bulleted points framing observations as requirements for design, whether or not the study is actually conducted as part of an explicit design effort.[7]

Since our concern in this paper is with the disciplinary relationship between ethnomethodology and computer science in this work, it is reasonable to ask, Where is the ethnomethodology in this process? In what way has ethnomethodology come to be integrated into the system design process? And what aspects of design have changed as a result?

Perhaps surprisingly, given the role of ethnomethodology in accounts of this form of system design (Sommerville et al., 1992), ethnomethodology does not seem to have entered the process at all, save in one way—that part of the process is conducted by ethnomethodologists. The locus of ethnomethodology in this approach, then, is the ethnomethodologists's head; it consists in how part of the process (requirements capture) is conducted. Otherwise, all remains as it was; the process has not changed, and nor have the artifacts it produces.

4.2 Learning from Ethnomethodological Accounts

Less numerous, but still significant, are studies in which there is a greater disconnection between the ethnomethodological work and the system design. In these cases, the implications and requirements for design are not drawn directly from the ethnomethodologists' interpretation of their field work, but from the ethnomethodological accounts of such studies.

Some of our own earlier work at the Rank Xerox Research Center illustrates this approach. Bowers et al. (1995) report on an investigation of the use of workflow technologies in the production printing industry. They describe the use of a particular technology at various sites of Establishment Printers, one of the UK's largest production printing organisations. In particular, they detail, first, the ways in which the model of the printing process embodied by the technology and its relationship to the management of the work systematically undermines the practices by which the print workers manage the flow of work through the print shop; and, second, the variety of ways in which the print workers undermine the technology in order to get the work done. The disparity between work process (represented explicitly within workflow technologies) and work practice (the detailed ways in which the process is actually performed) is a common focus of ethnomethodological studies of work, and is highlighted by their observations.

7. It seems ironic that in disciplines which have (quite rightly) rejected the right of system designers to set themselves in the place of psychologists and sociologist, the right of psychologists and sociologists to set themselves in the place of system designers seems assured.

Subsequently, the Freeflow project (Dourish et al., 1996b) focussed on the design of workflow technologies which would be more sensitive to the variety and fluidity of work practice. This was a systems design project, exploring new conceptual and architectural approaches to the design of workflow systems, but at its heart was an attempt to resolve precisely the sorts of problems which Bowers and colleagues had uncovered, both abstractly and in particular.

This project exemplifies this second use of ethnomethodological investigation as a basis for the design of new technologies. The locus of ethnomethodology is the account of work. In many ways, this can be seen as a more satisfactory way to proceed, at least from the point of view of disciplinary connection; after all, ethnomethodology (as opposed to other analytic perspectives on social interaction) becomes a part of what is communicated. On the other hand, the "disconnection" between work site and design which is implied by this approach can also be problematic, as it undermines, to an extent, precisely the sort of motivations which have led us, in HCI, to a deeper concern with the users of technology. If the user remains a "scenic feature of the design space" (Sharrock and Anderson, 1994) then it scarcely matters whether or not that feature is one painted with an ethnomethodological brush.

5 Ethnomethodology for Critique and Design

Taken together, ethnomethodologically affiliated studies have produced a strong critique of the design of technology at work. They have displayed that technology, at best, often fails to support the work it is designed for, and, at worst, does not allow people to actually engage in their work, because the technology is not aligned to the practices through which they organise their actions, interactions and work. Heath et al. (1994:147) summarise this conclusion for CSCW in the following: "Despite impressive technological developments in CSCW, it is widely recognised that there are relatively few examples of successful applications in real world settings. [...] it is suggested that the lack of success of CSCW systems derives not so much from their technological limitations, but more from their insensitivity to the organisation of work and communication in real work environments".[8]

Suchman's (1995) discussion of the technical and theoretical basis of Winograd and Flores' "THE COORDINATOR" provides a salutary case in point. Suchman's telling analysis of the use of speech act theory in this system, based in part upon Conversation Analysis and the work of Harvey Sacks, emphasises the ways in which the stipulative organisation imposed by the system undermines the interactionally contingent aspects of language use; but yet, at the same time (and as is plaintively acknowledged in some commentaries on her article appearing in the same issue), it would seem almost to leave technology with nowhere left to go.

5.1 Two Paradoxes of Ethnomethodologically-Informed Design

Ethnomethodological analyses have been used in a range of circumstances to critique technological design in particular working settings and situations. Ethnomethodology, in attending in particular to the details of everyday action and work practice, has been able to

8. More expansively, Cooper and Bowers' (1995) discussion of the "disciplinary rhetoric of HCI" points to the way in which construals of "user" in HCI embody an explicit move away from technology and from technological determinism, and so a focus on critique should not surprise us.

expose an unfortunate paradox in the design of technologies for collaborative activity (or socially-constructed action). This is the *paradox of system design*—that the introduction of technology designed to support "large-scale" activities while fundamentally transforming the "small-scale" detail of action can systematically undermine exactly the detailed features of working practice *through which* the "large-scale" activity is, in fact, accomplished. It points, fundamentally, to the interdependence of minute practice and grand accomplishment.

However, in so doing, ethnomethodology finds itself caught in a second paradox—the *paradox of technomethodology*.[9] Given the concern with the particular, with detail, and with the moment-by-moment organisation of action, how can ethnomethodology be applied to the design of new technologies? Certainly, ethnomethodologists have urged that designers take into account the methods and practices through which social action, interaction and categories of work are organised; but in the face of the unavoidably transformational nature of technology and system design in working settings, it would seem that ethnomethodology becomes relatively powerless. Its tradition is in analysing practice, rather than "inventing the future". Even Heath and Luff's (1992) observations of the relative impotence of gaze and gesture in video-mediated communication are, to an extent, undermined by accounts of the active transformation of visual conduct in video-mediated interaction by which the same communicative *accomplishments* are reconstructed; reconstructed, ironically enough, out of the *lived detail* of video interaction (Dourish et al., 1996a).

5.2 Critique and Design

Whatever the historical context and the factors which shaped the emergence of particular ideas at particular times, we take it as fundamental that the crucial (or prosaic) reason for ethnomethodology's new-found place in the design disciplines is that there are a number of ethnomethodologists who are interested in design. The elementary components of everyday life which proved so interesting to Garfinkel, Sacks and colleagues in the early days of ethnomethodological enquiry are, in the Nineties, increasingly dominated by (or at least suffused with) technology. For many people, everyday interactions increasingly include interactions with computer technology in one form or another. Our goal, then, is not simply to look at how ethnomethodology can be used to critique technologies, crucial though that is, or even to apply ethnomethodological understandings in order to better understand the conditions in which technology comes to be developed (as has been one focus, for instance, in the Participatory Design movement). Rather, alongside those investigations, we have been engaged in a different one; to understand how ethnomethodological understandings of human social action and interaction can be used, directly, in designing interactive technologies. Our focus is on *design*, not on *critique*; but it is also, critically, on the artifacts we design and the conceptual and technical apparatus by which that activity is performed, rather than on the design of specific systems.

6 Technomethodology: Drawing Foundational Relationships

In contrast with the approaches to ethnomethodologically-informed design discussed above, our approach can be stated quite simply. We take the relationship between ethno-

9. Taken together, these two paradoxes constitute what Grudin and Grinter (1995) refer to as "the ethnographer's dilemma".

methodology and system design in a design context to be more than a practical matter. For us, it is a matter of analytic orientation rather than project management. We take the ethnomethodological perspective on human social action to have detailed and deep consequences for what Suchman called the "problem of human-machine communication", and therefore see it as concerning the foundational concepts of system design, not simply the process by which system design proceeds. The interaction between our disciplines takes place in the interactions between the foundational elements of each discipline, not in their application to particular problems (although, of course, it is in the application that the interaction becomes valuable and visible).

We refer to this approach as "technomethodology". Although this term is rather whimsical, its force is in how it emphasises that this is not simply technology design, nor is it simply ethnomethodology. Rather, it is something new, equally radical in its consequences for its "parent" disciplines, but different from either.[10]

What do we mean by "drawing foundational relationships"? The answer is best given in terms of examples, and much of the rest of this paper will provide an extended example of the technomethodological approach, drawing on a relationship between the notion of "abstraction" in system design and the notion of "accountability" in ethnomethodology. These are examples of the foundational elements which we draw upon; elements which are conceptually central to the disciplines. Abstraction is the very stuff of system design; accountability is one of the primary elements of ethnomethodological reasoning. Similarly, concepts of number, identity, grouping, membership, formalisation and stability are grist to the technomethodological mill.

Fundamentally, then, our concern is not with the findings of particular ethnomethodological studies of working settings, but with the analytical frame within which those studies are conducted. Similarly, we are not immediately concerned with the design of this system or that system, but with the design of systems. System designers learn from ethnomethodology, not from ethnomethodologists or their observations; ethnomethodologists learn from computer science, not from computer scientists or their applications.

7 Abstraction, Accounts and Accountability

One example of the technomethodological approach currently under investigation is a user interaction model we call "accounts". This research is founded in a re-evaluation of the role of abstraction in designing interactive systems, and that re-evaluation is grounded in ethnomethodology's concern with the accountability of practical social action.

In this section, we will begin by outlining the role of abstraction in conventional system design, and subsequently introduce a novel architectural approach (called Open Implementation) that aims to address some practical, technical problems which arise in its use. These problems, traditionally tackled at the level of software infrastructure development, also arise in interactive systems. We employ the ethnomethodological perspective not only to consider the impact of these problems as interactional issues, but also how the technical so-

10. It might be observed, though, that, as a foundational investigation, it perhaps has *more* in common with the core concepts of either "computer science" or "ethnomethodology" than the simply applied perspectives might have.

lution provides an opportunity to fuse ethnomethodological understandings with an interaction design perspective.

7.1 Abstraction in System and Interface Design

Abstraction is the most fundamental tool of system design. Abstraction allows systems to be considered at different level of detail, to be broken down into individual components, and to be reassembled again. The act of systems design is the creation and manipulation of abstractions. User interface behaviours (file copying, printing, selection) are abstractions over the behaviour of the programs which they control; the programs are sets of abstractions (procedures, arrays and loops) which programmers manipulate to control the computer; and even our views of computers are couched in terms of abstractions (instruction sets, memory architectures, bus interfaces) over raw transistors and electrical pulses. Even at that level, we cannot escape the abstraction of binary signals, imposed over continuous voltages.

Abstractions help us manage complexity by allowing to selectively hide it. In systems design, abstractions typically function as “black boxes”. They are defined by the nature of their interactions with the outside world (human users or other pieces of code—the “clients” of the abstraction), which are typically defined in terms of the available functionality, procedure call conventions and return values. The system’s internal mechanisms, which describe and control how it goes about doing the work it does, are intentionally not available to inspection. By hiding mechanism in this way, the two main uses of abstraction in system design can be achieved. First, systems can be built in terms of complex components with simple interfaces, rather than in terms of basic, raw mechanism (allowing us to build systems like spreadsheets out of mathematical packages and interface packages). Second, systems with the same interface can be regarded as equivalent (as is the case with programming language compilers or microprocessors).

In interface design, the same models of abstraction show through. Users interact with abstract interfaces to the system's functionality (such as a print dialogue, or a direct manipulation view of a filesystem) which provide simple, consistent interaction by hiding the complex realities of the system mechanisms (creating a Postscript file and sending it to the printer, or copying files from a local disk to a server across a long-distance network connection). The goals of interface abstraction are similar to those of systems abstraction—reduction of complexity, modularity, consistency—and arise out of the use of similar techniques. Of course, since interactive systems are computational, the use of abstraction techniques in interface design derives from their use in computational systems generally.

One use of abstraction in interface design is the support for metaphoric interaction. Metaphors are brought into being by drawing equivalences between two abstractions, and function precisely through the hiding of mechanism. Breakdowns tend to occur where the mapping no longer holds, or where the details which the abstraction hides become suddenly relevant (such as when the network suddenly makes its presence felt by introducing delays into file copying, or when it suddenly becomes clear that the Mac trashcan is actually an abstraction for different sets of deleted files on different disks). However, it is significant that everyday interactions with the physical and social world notably do not display the same sort of “information hiding” characteristic (and hence tend not to exhibit these sorts of failure). Real-world machines produce noises and respond to physical interference, and their physical embodiment allows us to perceive their operation and even sometimes become involved in it. Human actors allow us to query their actions and motivations. In other words, we organise our actions not simply around abstractions of possible action, but around the detail of the *production* of action and behaviour in particular circumstances.

7.2 Abstraction and Glossing Practices in Everyday Interaction

This aspect of human action has been a central element of the ethnomethodological study programme; and ethnomethodology's analysis offers lessons for the reconsideration of abstraction in interactive system design.

Recall that, for ethnomethodologists, the critical property underpinning rational social action is accountability. Garfinkel argues that human social action is organised accountably, that is, it is organised so as to reveal itself as precisely the sort of action it is, and to make it rationally recognisable to others. Accountability is the property of action organised in this way; organised so as to be observable and reportable. For ethnomethodology, the key element of rational social action is that it is organised so that it can be rationalised. Garfinkel and Sacks (1970) use the notation "doing [being a researcher]", say, to describe not only simply the performance of the research, but the work of doing it so that it is organised to be recognisable to others as research.

Critically, this meta-work—the organisation of activity in this way—is not done externally to the activity itself, but is rather a phenomenon of the activity. Activity itself is *made* observable and reportable, rather than being pointed to, observed and reported *upon*.

For the purposes of human-computer interaction, then, the critical observation is this. What computational abstractions share with the abstractions of natural, everyday interaction is that they are organised to reveal certain things for certain purposes. What they hide is as important as what they reveal; and decisions about what to hide and reveal are the key to their effective use in any particular situation. What they do *not* share with the abstractions of everyday activities—and what becomes particularly relevant when we apply ethnomethodological understandings the investigation of human-computer interaction—is the observable-reportable nature of everyday action which is at the heart of ethnomethodological investigation. Garfinkel and Sacks (1970) refer to the mechanisms at work here as *glossing practices*—"methods for producing observable-reportable understandings... a multitude of ways for exhibiting-*in*-speaking and exhibiting-*for*-the-telling that and how speaking is understood" [pp. 333–334].

In other words, the key property of human action is the way in which it is made observable and reportable in the course of its own unfolding. It is organised accountably. As someone speaks, this is how "he provides the very materials for *making out* what he says". Computational abstractions, being static, atomic and unexaminable, provide no such means. Computational behaviour is not accountable; and, for this reason, the forms of contingent, improvised (situated) action become problematised.

7.3 Towards Observable-Reportable Abstractions: Open Implementation

What this leads us towards, then, is a way of opening up abstractions so that they can convey aspects of the mechanisms which lie behind them. By revealing more of what lies behind them, these more "translucent" interfaces would provide cues as to not only what the system was doing, but why it was being done, and what was likely to be done next. A view onto the mechanism provides a context to make system activity rationalisable, and to do so within the circumstances of its activity rather than outside of it (as, say, manuals might do). Critically, this is not a call for more complex, more mechanical interfaces; those would be ones which do away with the abstractions, and we need to retain the abstractions to retain clarity, consistency and ease of use. However, like social interactions in everyday life, we would like our interactions with computational systems to be organised in terms of abstractions which are supported by their own unfolding, rather than opaque (and brittle) black boxes.

A recent line of development in software architecture has, for quite different reasons, been moving in just this direction. Open Implementation (OI) (Kiczales, 1996) is an approach to system design which recognises that the implementation features which abstractions hide are often ones embodying design decisions critical to the effective use of the abstraction. Kiczales' concern is with software abstractions offered to programmers and to other system components, but precisely the same arguments apply to interactive systems; and the techniques which OI develops are also applicable in this domain.

One key principle which underlies much OI practice is *computational reflection* (Smith, 1982). Reflective systems are computational entities which contain a representation of their own structure and behaviour. Critically, this representation is *causally connected* to the behaviour which it describes. The result of this causal connection is that not only will changes in the system's behaviour be reflected by equivalent changes in the model, but also the model can be changed in order to change the system. This principle was originally applied in the design of programming languages, providing languages with models of their own execution which could be used to introduce new programming features in an implementation-independent way. The Open Implementation approach has applied it more widely to the problems of abstraction, using a reflective "meta-level" to offer a model of the internal mechanism lying inside an abstraction, as a means to observe and control how that abstraction will be realised when the system is used (figure 1).

The full detail of the OI approach, and in particular the practicalities of implementing reflective systems, are beyond the scope of this paper; the reader is referred to more technical descriptions such as that of Kiczales et al (1991) for a fuller account of the techniques by which Open Implementations are realised. Here, however, we will sketch relevant aspects of OI's reformulation of computational abstraction.

The goal of Open Implementation is to provide flexibility and the opportunity for reconfiguration of the mechanism which lies behind a traditional computational abstraction. Drawing on examples of current practice in a wide range of domains, OI argues that details of implementation *strategy*—details which are explicitly hidden by traditional abstraction techniques—will often be critical to the way in which clients will make use of the abstraction. Traditional abstractions, embodied in programming languages, library APIs and user interfaces, talk only in terms of input and output, request and result, but never in terms of *how* the input will be mapped to the output, or how the result will be generated when the request is made. If a programmer knows that creating a new window in a window system is a fast operation with low memory overhead, then she can program her application to use windows whenever a new screen object is required. On the other hand, if she knows that

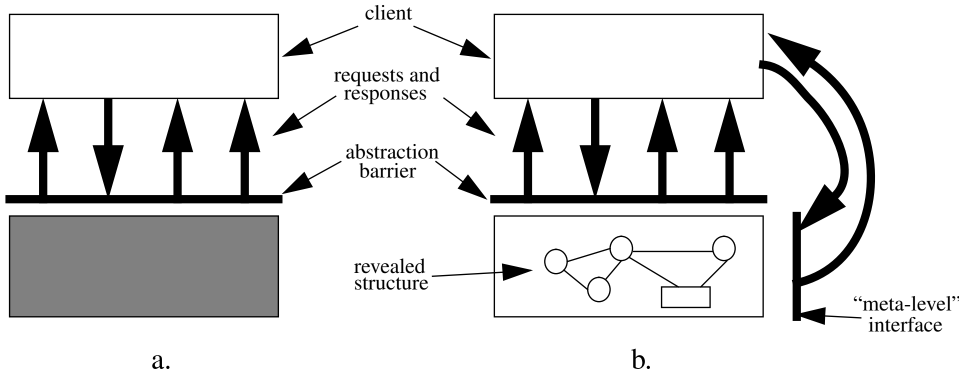

FIGURE 1: (a) Clients interaction with traditional black-box abstractions through standard abstraction barriers. (b) Open implementations also reveal inherent structure.

there are significant performance and memory overheads involved in creating windows, she might choose instead to allocate just one large window for her application, and handle the allocation of space inside it herself. The traditional abstraction model provides no opportunity for gaining this sort of information, because those aspects of implementation strategy which are relevant are locked within the "black box".

The Open Implementation strategy is to provide two interfaces to the implementation, at different levels. The first is the traditional interface, by which clients can make use of the abstraction (e.g. create, draw in and destroy windows). This is called the *base-level interface*. In addition, it provides a second interface, called the *metalevel interface*. The abstraction offered at this interface is a rationalised model of the inherent structure of the implementation; and the controls offered at the metalevel interface can be used to control aspects of the implementation of the base-level interface; that is, it provides the programmer with a view into the mechanism, by which aspects of that mechanism can be controlled.

This approach to computational architecture has been fruitfully applied in a number of areas of system design, most notably in the design of programming languages (e.g. the Common Lisp Object System (Bobrow et al., 1988)) and operating systems (e.g. Apertos (Yokote, 1992)). Briefly, its value is that, in opening up to scrutiny how traditional abstracts are to be realised, it allows programmers to make critical distinctions between *what* they want to do with system abstractions and *how* they want the system to go about providing its functionality. Further, the approach does this in a way which makes clear the relationship between what and how; they are not divorced from each other.

For our purposes, then, the critical feature of the Open Implementation approach, and the feature on which we rely in developing its use as in interactive systems, is the way in which it allows us to forge and articulate the relationship between what is *done* (the implementation behaviour), and what is *done by what is done* (the achievement of application ends).

7.4 Accounts and Accountability

We are currently developing an approach to the design of interactive systems in terms of Open Implementation. One aspect is to adopt the Open Implementation model in the same way in which it has been applied to other domains, that is, as an engineering technique which can be used to provide considerably greater flexibility than would be available otherwise. For instance, Prospero (Dourish, 1996) is a CSCW toolkit which employs Open Implementation in this way, offering programmers the opportunity to become involved in the implementation of infrastructure mechanisms which support their applications.

We will be concerned here with a second approach. This is a more fundamental one, and aims to address the disparity between traditional process-driven models of interface design and the more improvised model revealed by sociological and anthropological investigations such as Suchman's. This approach is based on a re-reading of Open Implementations' reflective self-representations as *accounts which systems offer of their own activity* (Dourish, 1995).

As should be clear from what has gone before, the term "account" is chosen particularly to emphasise a metaphorical frame drawn from the ethnomethodological perspective on the organisation of action. So what is important about this approach is not the account itself (the explanation of the system's behaviour) but rather *accountability* in the way this explanation arises. In particular, the account arises reflexively in the course of action, rather than as a commentary upon it, and concerns the way in which that action is organised so that it can be made rational in particular circumstances. These features, which arise directly from the Open Implementation approach, allow us to use these accounts as a means for users to rationalise the activity of the system and therefore to organise their behaviour around it, as interaction proceeds, for their own practical purposes.

8 Example: Accounting for File Copying

To make these ideas more concrete, this section will present an example of working with accounts. The role of the example is to illustrate not only the role of accounts in interactive systems, but also how accounts operate as an example of applying the technomethodological approach to the relationship between abstraction and accountability.

8.1 The File-Copying Scenario

The file-copying scenario is familiar to users of graphical desktop environments, in which a single "folder" or "directory" abstraction is used for all containers of files. Folders can be specified as targets for copying operations simply by dragging the icons for files to be copied and dropping them on the destination folder.

In many such environments, initiating a copy operation of this sort will cause a status indicator such as a "percentage-done bar" (or "thermometer") to appear, indicating how much of the operation has been completed. So graphical interaction (drag and drop) initiates the copy operation, and the percentage-done bar reports progress. As more of the files are copied, more of the bar is filled in, until eventually all the files are copied and the bar is completely filled.

However, consider an alternative scenario which is possible in this case. Imagine that the folder to which the files are copied is not actually a folder on your local hard disk, but rather is a link to a folder on a remote volume accessible over a network. Let us consider a case in which someone drags a large set of files onto the folder. A percentage-done bar appears, and starts to fill; but when it reaches 40% complete, the copy operation fails. There are many reasons why it might have failed; the remote volume may have become full, or the server may have become unavailable, or the network may have failed (or perhaps never been connected in the first place).

In this case, what resources are available to the user to understand what has happened, and to understand what options are now available? What does it mean that the percentage-done bar filled to 40%? Were 40% of the files written onto the remote volume? Were 40% of the files read from the local disk? Did all of the files get 40% of the way to their destination?

From a technomethodological perspective, consider why these questions are important, and how they arise. Ethnomethodology talks about the way in which people find, within the circumstances of action, the means to find that action rational. Further, in terms of Suchman's work, these resources provide the means to articulate abstract plans of action and to organise the specifics of action. However, the abstraction which has been offered by the system—the folder—hides the details upon which such understandings could be based. The differences between local and remote folders, the difference in the operation of local and remote copy operations, and the consequences of these differences, are hidden from view.

What's more, it is not sufficient simply to offer two different kinds of folders, providing a distinction between local and remote. The reason why this is not an adequate solution is that actions are situated within the specific circumstances of their production, not within abstract characterisations of them. What is important here is not the differences between two abstract types of copying (local copying and remote), but the specifics of *this or that copying operation*. There are far too many different features of the occasion (including distance, available network bandwidth, other people's activities, the types of files involved and even the type of network infrastructure) for designers or users to be able to distinguish between in the abstract model which the system offers.

8.2 An Account of File Copying

Instead of trying to provide different abstractions for all the different circumstances in which copying might take place, the accounts model provides a mechanism for dynamically relating manipulation of the abstraction to the detail of what is actually happening. The account provides a means for users to see what copying means in this case, by providing a view into the mechanism by which copying is carried out.

What is needed in the first instance, then, is a model of the copying mechanism—an account of file copying, in terms of which a specific account of a specific copy can be formulated. In seeking such an account, we look for the structural properties of the system's behaviour with reference to which those elements that concern us can be made manifest in the interface. The "account" here (the interface's representation of the system's behaviour) is not an explanation as such, but a backdrop against which the system's behaviour can be played out and made understandable.

As an example, we can characterise the structure of file copying as shown in figure 2. Between the file source and destination are arrayed a number of staging posts (data buckets). File data flows from the start-point to the end-point by moving from one bucket to another along a data path. As data flows from one bucket to the next, the buckets are related to each other by flow strategies, by which the movement of data from one to the next is regulated. The structure of a flow path and the strategies used will vary in different circumstances; the number of buckets and the variety of strategies by which data flow is controlled character-

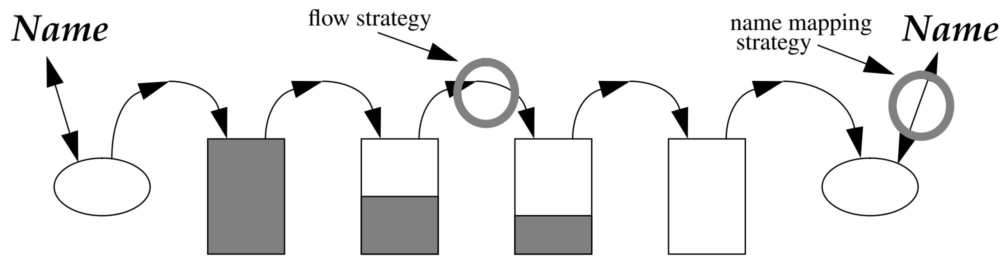

FIGURE 2: A structural model of the file copying example in terms of data buckets and the connections between them. Connections between elements of this model are the points at which strategies and policies can be identified.

ise the different ways of doing file copying. So, in different situations, data buckets may be used to describe file system caches, network interfaces and networks themselves. The flow of data through these, and the activation of the flow strategies, provides a framework for the relationship between the action in which the system engages and the reading and writing of data files.

In addition, this structure also provides the means to answer the sorts of questions which were raised earlier when the file copy failure was observed for network copying. There were two particular problems when those questions arose. First, the system had no means to draw distinctions between the different circumstances in which file copying might be done. The data bucket and flow strategy mechanism provides just this opportunity. Second, the system provided no terms in which the role of the percentage-done bar played, making it impossible to interpret its behaviour. However, with this structure in place we can describe "where" in the flow the percentage-done bar is connected, and thus allow the user to make some sense of what its is actually reporting.

8.3 Accounts and Ordinary Operation

Although our scenario earlier described a failure in network copying (and so a requirement to provide more information to allow the user to understand what had gone wrong), it is important to notice that, by being offered within the action rather than outside it, these sorts of interface accounts provide not just for recovery from failure, but also for more detailed ongoing monitoring of action. It is from this provision of information in the course of activity that the system supports the sorts of improvised activity which Suchman brought to the analysis of behaviour at the interface. By enriching the system's detailing of the circumstances in which it is acting, we similarly enrich the potential resources for the user's moment-by-moment decision making. In the case of file copying, an explanatory system organised around failure would be useless in order to make decisions like, "why is this taking so long?" or "will this finish before my ride home arrives?"—the sorts of questions which potentially lie at the heart of decisions to stop the copy, to do it in another way, to copy a subset of the files, and so on. In other words, an explicit failure model sets limits not only on the sorts of questions which might be asked, but also, in organising them around specific breakdowns, on the *reasons* that those questions might arise. Our goal is to avoid this problem by providing information without making a prior commitment to the reason that information might be useful.

The account, then, is not simply a new form of error reporting system, but rather becomes part and parcel of ordinary interaction with a computer system. Again, this draws on the ethnomethodological perspective on interaction, where accountability arises not out of specific requests for information, but, first, as part and parcel of everyday activity, and second, as a crucial feature allowing concerted action to arise.

8.4 Accounts and Mental Models

On brief inspection, there might appear to be considerable overlap between the accounts-based notion of interaction offered here and the body of work on mental models in HCI (see, for example, Gentner and Stevens (1983) for an introduction and overview of this work). It is instructive to detail why this is not so.

The fundamental distinction is quite straight-forward. A mental model[11] is a model of the operation of a system, formed *by a user* of that system, and which the user employs in the planning of their activity. An account, on the other hand, is a model of the system's activity offered *by the system* to account for and cast light on its own action. While this distinction is easy to state, and the fundamental difference in the "location" of the model is clear, it hides a range of other issues. Two will concern us here.

The first is that, as can be seen in the phrasing of the terms above, the traditional notion of mental model, as derived from a cognitivist perspective, originates from a very different perspective on human action than that which we are considering here. This is not the place for a debate on the relative positions and merits of phenomenological sociology and cognitive science. However, it is important to emphasise the way in which the accounts model arises from a model of situated, resource-driven activity rather than one constituted in terms of abstract models of behaviour. It is this same perspective which leads us to frame the account in terms of what the *system does* rather than what the *user might mean*.

The second issue is that, as part of the artifact, the account is itself designed along with the rest of the system. It is part of the interface (albeit at a metalevel, rather than concretely instantiated in buttons and menus). The questions which this observation raises—of how the artifact is designed, of the principles for its appearance in the user interface, and of the means by which its structure is determined—are ones subject to ongoing investigation (e.g. Dourish and Curbow, 1997). Our goal, however, is not to develop a design method by which the "right" account can be found and embodied in the system, but rather to investigate this "two-level" approach to interface design as a way of making ethnomethodological perspectives on accountable action "real" in the abstractions of interactive systems.

9 Abstraction and Particularity

> "... the reported phenomena are only inspectably the case. They are unavailable to the arts of designing and interpreting definitions, metaphors, models, constructions, types or ideals. They cannot be recovered by attempts, no matter how thoughtful, to specify an examinable practice by detailing a generality."
> —Harold Garfinkel (1991)

Our approach seems, on the surface, to be in contradiction with some widely-held views. In CSCW, particularly, where the disciplinary constitution of the field means that these de-

11. The term "mental model" has been used by different authors at different times to refer to different things. We take this meaning, a conceptual model possessed by the user of a system, to be the primary meaning of the term, and the one most relevant here.

liberations arise frequently, ethnomethodologists are characterised, somewhat unfairly, as unwilling or unable to traffic in the generalisations which system design needs. Computer scientists, in turn, are sometimes seen by ethnomethodologists as over-eager to generalise findings, stripping them in the process of precisely the finely-balanced features of the setting which are the very essence of the ethnomethodological accounts. To those who are familiar with these oft-repeated observations (whether in the pages of learned journals or grumbled around the bar), we would seem to be setting ourselves an impossible task—to exploit generalisations in ethnomethodology rather than working from particular field studies, and, what's more, to exploit these generalities not in terms of particular application settings, but in terms of the conceptual underpinnings of systems design.

The resolution of this problem lies a distinction between the ways in which our two disciplines use abstraction. For computer science, abstractions are generative; that is, they are used to generate behaviours. There is no way to put a character on a screen without manipulating the abstractions of windows, fonts, and display procedures. The results of much of what computer scientists do every day (building architectures, descriptions and programs) is based on the application and manipulation of abstractions.

Despite appearances to the contrary, ethnomethodology also traffics in abstractions, but they are abstractions of a different sort. For ethnomethodology, abstractions are analytic; used to characterise and explain behaviour (but not necessarily taking an active role in how that behaviour comes about). What ethnomethodology rejects is not the notion of widely application abstractions and concepts, but rather the "ideal types" of conventional sociology. Ideal types, a concept first put forward and exploited by Weber, strip away distracting detail to reveal the "essential patterns" of social truth underneath. In effect, they provide the sociological theorist with a means to step outside the vagaries of everyday concrete phenomena in providing a sociological account of the world, the very phenomena which ethnomethodology embraces and revels in. Ethnomethodology rejects "abstraction" only in as much as it rejects an approach to sociological theorising which deals with the problem of social order in terms of ideal types and social structure.

In other words, there is a considerable difference between the abstractions of conventional sociology and those of ethnomethodological investigations, in terms of their ontological status. Where conventional sociology sought the elements of a theoretical mechanics of social order, ethnomethodologists sought to uncover only those regularities which they could directly observe in naturally-occurring data.

The power of those regularities, however, is precisely how they occur on different occasions, and indeed many of the social "mechanisms" that ethnomethodology has described are found across many different social circumstances. For example, Garfinkel and Sacks (1970) describe "cohort independence" phenomena, by which they mean social phenomena that are not tied to the scenic features of their production. Thus, for example, the model for turn-taking in conversation (Sacks et al., 1974) is, in some crucial respects, cohort independent in that it operates across local circumstances such as gender, ethnicity, race, occupational identity, etc. It is these sorts of generally operative social processes, explicated by ethnomethodology, which we turn to in our technomethodological approach. These processes are, in Sacks et al.'s terms, "context free, yet context sensitive".

The crucial feature of these abstractions is that, abstract though they are, they are uncovered and operate in the particular. This concern with the particular is a fundamental methodological tenet of ethnomethodology, dealing with naturally-occurring everyday behaviour rather than with generalisations of social action. They are abstract in that they are not concrete; they do not set out how it is that any given conversation will unfold. But they are abstract, none the less.

The distinction between these two dimensions—between the abstract and the concrete, between the general and the particular—provides us with the opportunity to engage in technomethodology. In particular, the move represents an attempt to work with a set of *sensibilities* rather than with the details of specific activity, even though, of course, those sensibilities arise out of the ethnomethodology's very concern with the grounded and specific experience of everyday activity. In other words, technomethodology attempts to align system design not so much with the details of *specific working practices*, as with the details of the *means by which such working practices arise and are constituted*.

Consider an example to illustrate this distinction. In recent years, there has been an interest in utilising the insights of ethnomethodology for the development of dialogical interfaces. Attempts have been made to build in the *specifics* of Sacks et al.'s turn-taking model, such as the rules associated with speaker transfer, into computer interfaces. However, our argument is that the value of the turn-taking model described by Sacks et al is in the way it which it shows how the abstractions of conversational flow are *sustained*, rather than rote procedures by which they might be *enacted* (Button and Sharrock, 1995). It is this notion of the ongoing management of conversation, rather than the specifics of any human dialogue, which provides an abstraction for design. When we fail to make the distinction, we fall foul of the paradox of technomethodology.

10 Conclusions

Although the influence of research work conducted from the perspectives of sociology and anthropology has become increasingly prominent in HCI in recent years, what we typically observe finding its way into new research systems is sociological observation rather than sociological analysis. Ethnomethodology, which has become particularly prominent, especially in the field of CSCW, seems to disappear as an analytic position in favour of the observations which ethnomethodologically-inclined field workers report from their observational studies. The goal of our work, reported here, has been to seek a new position on the relationship between computer science and ethnomethodology in the design of interactive software. This position regards the relationship between our disciplines as a foundational, analytic concern rather than simply a practical one, and so emphasises how it is that the ethnomethodological position on the problem of social order can inform, respecify and reconceptualise foundational elements of system design.

Our investigation of these issues takes place on three levels. First, we are concerned with developing a basic analytic position on the foundational relationship between our two disciplines. Our goal is a position which moves away from a "service" relationship between our disciplines, in which ethnomethodologist uncover requirements for system design or computer scientists develop systems organised around specific working practices. Second, we are engaged in developing a collection of specific disciplinary relationships both as illustrations of our position and as tools of a new model of design. The work presented here on accountability and abstraction is one example of a relationship of this sort; ongoing work is directed towards other particular relationships. Third, we are looking at how these relationships can be exploited in specific design projects, in order to chart a course from analytic reasoning to working software systems. We hope to report on these investigations at a later date.

Acknowledgments

The seeds of the work reported here grew from the long-term interest in the relationship between social and design sciences which was a core research focus at the Rank Xerox Research Centre and the Xerox Palo Alto Research Center. We would like to acknowledge the valuable contributions, to both our thinking and writing, made by many colleagues there and elsewhere, particularly Annette Adler, Bob Anderson, Victoria Bellotti, Beki Grinter, Austin Henderson, Yvonne Rogers, Lucy Suchman and Randy Trigg.

References

Anderson, R. (1991). Representation and Requirements: The Value of Ethnography in System Design. *Human-Computer Interaction*, 9, 151–182.

Anderson, R., Button, G. & Sharrock, W. (1993). Supporting the Design Process Within an Organisational Contex. *Proceedings of the European Conference on Computer-Supported Cooperative Work ECSCW'93* (Milano, Italy). Dordrecht: Kluwer.

Anderson, R. (1996).: Work, Ethnography and Systems Design. *Encyclopedia of Microcomputers*, 20, 159–183. New York: Marcel Dekker.

Bentley, R., Rodden, T., Sawyer, P., Sommerville, I., Hughes, J., Randall, D. & Shapiro, D. (1992). Ethnographically-Informed Systems Design for Air Traffic Control. *Proceedings of the ACM Conference on Computer-Supported Cooperative Work CSCW'94* (Toronto, Canada). New York: ACM.

Bobrow, D., Demichiel, L., Gabriel, R., Keene, S., Kiczales, G. & Moon, D. (1988). *Common Lisp Object System Specification*. X3J13 Document 88-002R. New York: American National Standards Institute.

Bobrow, Linda Demichiel, Richard Gabriel, Sonya Keene, Gregor Kiczales and David Moon, "*Common Lisp Object System Specification*", X3J13 Document 88-002R, June 1988.

Bowers, J. (1994). The Work to Make a Network Work: Studying CSCW In Action. *Proceedings of the ACM Conference on Computer-Supported Cooperative Work CSCW'94* (Chapel Hill, North Carolina). New York: ACM.

Bowers, J. & Pycock, J. (1994). Talking Through Design: Requirements and Resistance in Cooperative Prototyping. *Proceedings of the ACM Conference on Human Factors in Computing Systems CHI'94* (Boston, Mass.). New York: ACM.

Bowers, J., Button, G., & Sharrock, W. (1995). Workflow from Within and Without. *Proceedings of the Fourth European Conference on Computer-Supported Cooperative Work ECSCW'95*. Dordrecht: Kluwer.

Button, G. & Sharrock, W. (1995). Simulacrums of Conversation. In P. Thomas (Ed.), *The Social and Interactional Dimensions of Human-Computer Interfaces*. Cambridge: Cambridge University Press.

Button, G. & Sharrock, W. (1995). Ethnomethodology and HCI. Presentation at the CHI'95 Research Symposium (Denver, Colorado.)

Cooper, G. & Bowers, J. (1995). Representing the user: Notes on the disciplinary rhetoric of human-computer interaction. In P. Thomas (Ed.), *The Social and Interactional Dimensions of Human-Computer Interfaces*. Cambridge: Cambridge University Press.

Dourish, P. & Curbow, D. (1997). Exploring the Interface to Accountable Systems: From Direct Manipulation to Direct Experience. Unpublished technical report, Apple Computer, Inc., California.

Dourish, P. (1995). Accounting for System Behaviour: Representation, Reflection and Resourceful Action. *Proceedings of the Third Decennial Conference on Computers in Context*.

Dourish, P. (1996). *Open Implementation and Flexibility in CSCW Toolkits*. PhD Thesis, University College, London.

Dourish, P., Adler, A., Bellotti, V. & Henderson, A. (1996a). Your Place or Mine? Learning from the Long-Term Use of Audio-Video Communication. *Computer-Supported Cooperative Work*, 5, 33–62.

Dourish, P., Holmes, J., MacLean, A., Marqvardsen, P., & Zbyslaw, A. (1996b). Freeflow: Mediating Between Representation and Action in Workflow Systems. *Proceedings of the ACM Conference on Computer-Supported Cooperative Work CSCW'96*. New York: ACM.

Garfinkel, H. (1967). *Studies in Ethnomethodology*. New Jersey: Prentice Hall.

Garfinkel, H. & Sacks. H. (1970). Formal Structures of Practical Action. in J. C. McKinney and E. A. Tiryakian (Eds), *Theoretical Sociology*, New York: Appleton Century Crofts.

Garfinkel (1991) Respecification: evidence for locally produced, naturally accountable phenomena of order, logic, reason, meaning, method, etc. in and as of the essential haecceity of immortal ordinary society (I) – an announcement of studies. In Button (Ed), *Ethnomethodology and the Human Sciences*. London: Routledge.

Gentner D. & Stevens A. (1982). *Mental Models*. New Jersey: Laurence Erlbaum Associates.

Grudin, J. (1990). The Computer Reaches Out: The Historical Continuity of Interface Design. *Proceedings of the ACM Conference on Human Factors in Computing Systems CHI'90*. New York: ACM.

Grudin, J. & Grinter, R. (1995). Ethnography and Design. *Computer-Supported Cooperative Work*, 3, 55–59.

Heath, C. & Luff, P. (1991). Disembodied Conduct: Communication through video in a multi-media environment. *Proceedings of the ACM Conf. Human Factors in Computing Systems CHI '91* (New Orleans, Louisiana). New York:ACM.

Heath, C. & Luff, P. (1992). Media Space and Communicative Asymmetries: Preliminary Observations of Video-Mediated Interaction. *Human-Computer Interaction*, 7, 315–346.

Heath, C., Jirotka, M., Luff, P. & Hindmarch, J. (1993). Unpacking Collaboration: The Interactional Organisation of Trading in a City Dealing Room. *Proceedings of the Third European Conference on Computer Supported Cooperative Work ECSCW'93* (Milano, Italy). Dordrecht: Kluwer.

Heath, C., Jirotka, M., Luff, P., & Hindmarsh, J. (1994). Unpacking Collaboration: the Interactional Organisation of Trading in a City Dealing Room. *Computer-Supported Cooperative Work*, 3, 147-165.

Heritage. J. (1984). *Garfinkel and Ethnomethodology*. Cambridge: Polity.

Hughes, J., Randall, D., & Shapiro, D. (1993). From Ethnographic Record to System Design: Some Experiences from the Field. *Computer Supported Cooperative Work*, 1, 123–141.

Kiczales, G., des Rivieres, J., & Bobrow, D. (1991). *The Art of the Metaobject Protocol*. Cambridge: MIT Press.

Kiczales, G. (1992). Towards a New Model of Abstraction in the Engineering of Software. *Proceedings of the IMSA Workshop on Metalevel Architectures and Reflection*.

Kiczales, G. (1996). Beyond the Black Box: Open Implementation. *IEEE Software*, January, 8–11.

Livingstone, E. (1988). *Making Sense of Ethnomethodology*. London: Routledge, Kegan and Paul.

Lynch, M., Livingstone, E., & Garfinkel, H. (1983). Temporal Order in Laboratory Work. In Knoll-Centina and Mulkay (Eds), *Science Observed*, London: Sage.

Parsons, T. (1937). *The Structure of Social Action*. New York: McGraw-Hill.

Plowman, L., Rogers, Y. & Ramage, M. (1995). What are Workplace Studies For? *Proceedings of the European Conference on Computer-Supported Cooperative Work ECSCW'95* (Stockholm, Sweden). Dordrecht: Kluwer

Rouncefield, M., Hughes, J., Rodden, T. & Viller, S. (1994). Working with Constant Interruption: CSCW and the Small Office. *Proceedings of the ACM Conference on Computer-Supported Cooperative Work CSCW'94* (Chapel Hill, North Carolina). New York: ACM.

Sacks, H. (1992). *Lectures on Conversation*. Cambridge: Blackwell.

Sacks, H. (1984). On doing 'being ordinary'. In Atkinson and Heritage (Eds), *Structures of Social Action: Studies in Conversation Analysis*, 413–429.

Sacks, H., Schegloff, E., & Jefferson, G. (1974). A Simplest Systematics for the Organisation of Turn-Taking in Conversation. *Language*, 50, 696–735.

Sharrock, W. and Anderson. R. (1994). The User as a Scenic Feature of the Design Space. *Design Studies*, 5–28.

Shapiro, D. (1994). The Limits of Ethnography: Combining Social Sciences for CSCW. *Proceedings of the ACM Conference on Computer-Supported Cooperative Work CSCW'94*. ACM: New York.

Smith, B. (1982). *Reflection and Semantics in a Procedural Language*. Report MIT-TR-272, Laboratory for Computer Science. Cambridge: MIT.

Sommerville, I., Rodden. T., Sawyer, P., & Bentley, R. (1992). Sociologists can be Surprisingly useful in Interactive Systems Design. *People and Computers VII*, 341–353. Cambridge: Cambridge University Press.

Suchman, L. (1987). Pla*ns and Situated Actions: The Problem of Human-Machine Communication*. Cambridge: Cambridge University Press.

Suchman, L. (1994). Do Categories have Politics? The language/action perspective reconsidered. *Computer-Supported Cooperative Work*, 2(3), pp. 177–190.

Thomas, P. (1995). *The Social and Interactional Dimensions of Human-Computer Interfaces*. Cambridge: Cambridge University Press.

Yokote, Y. (1992). The Apertos Reflective Operating System: The Concept and Its Implementation. *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems and Applications OOPSLA'92*. New York: ACM.