

This excerpt from

Where the Action Is.
Paul Dourish.
© 2001 The MIT Press.

is provided in screen-viewable form for personal use only by members of MIT CogNet.

Unauthorized use or dissemination of this information is expressly forbidden.

If you have any questions about this material, please contact cognetadmin@cognet.mit.edu.

Social Computing

Embodied Interaction draws together ideas from two areas of recent research in Human-Computer Interaction. The previous chapter dealt with the first, the tangible computing perspective that blends computation and physical design to extend interaction “beyond the desktop.” This chapter will deal with the second, social computing.

Broadly speaking, social computing refers to the application of sociological understanding to the design of interactive systems. *Sociology*, though, is a very broad term. As a discipline, it encompasses a diverse set of interests, topics, objectives, concerns, and methodological approaches. Similarly, many different methods and topics from sociology have been applied to the design and deployment of interactive computer systems, from questions of the economic and social consequences of the information economy to studies of how the conversational paradigm for software compares with our own conversational behavior. Not all of these will be of concern here. In particular, nothing will be said here about the broad range of work on the social and economic consequences of computerization and automation, or other investigations that we might characterize as “social consequences” or “social policy” concerns. Although those are critically important and often neglected as aspects of the design process, they are not immediately relevant to the issue of embodied interaction.¹ Instead, this chapter will explore the way in which sociological methods and reasoning have increasingly been adopted as a part of the design, development, and evaluation of interactive systems. In other words, the relationship in “social computing” between sociological and technical issues is not just that of a sociologist talking about technology, but of sociologists and technologists working together in the design process.

At first blush, it might seem strange to look at interactive system design from a sociological perspective. Sociology is concerned with the structure and function of society, while interactive systems are tools that people use; we might as well study the sociology of screwdrivers.² However, although that position seems immediately appealing, the significance of a sociological approach becomes clear when we look at the *context* in which computation is put to work. Context can mean many things; it might be the tasks that the system is being used to perform, the reasons for which the tasks are being carried out, the settings within which the work is conducted, or other factors that surround the user and the system. The context, though, is as much social as technical. When we think in these terms, we can see that the work that computation does, and the uses to which we put it, are very much the sort of thing that a sociological perspective might help us to understand. Computation is part of a richer fabric of relationships between people, institutions, and practices that sociology can help us explore.

Another critical aspect of context that is often overlooked is the interaction between the designer and the user *through* the system. Human-computer interaction can be thought of as a form of mediated communication between the end user and the system designer, who must structure the system so that it can be understood by the user, and so that the user can be led through a sequence of actions to achieve some end result. This implies that even the most isolated and individual interaction with a computer system is still fundamentally a social activity. The communication between designer and user takes place against a backdrop of commonly held social understandings. Even the metaphors around which user interfaces are constructed (“private” files versus “public” ones, “dialog” boxes, electronic “mail,” documents, wizards, and “publishing” a web page) rely on a set of social expectations for their interpretation and use.

However, sociology is not a unified discipline, but something of an umbrella term for a variety of theoretical and methodological approaches. Unsurprisingly, widely different sociological perspectives have been applied to different aspects of human-computer interaction and system design. In the discussion that follows, I am not going to attempt to describe them all, but instead will focus on a few particular approaches. They share three

common characteristics. First, they are concerned with the details of the organization of social conduct rather than broad social trends. Second, they are primarily oriented toward real activities and experiences rather than abstractions or models. Third, they all adopt an anthropological perspective on collecting, interpreting, and using field materials. Although this is only a fraction of the possible range of concerns that could be addressed, two features recommend these approaches in particular. The first is their relevance to issues of embodiment, and so the sort of argument that I will develop from them; the second is their prominence, perhaps even dominance, in current HCI research.

As an introduction, I will begin by setting out some of the historical background and characteristics of the perspectives that we will encounter. That background will set the stage for a discussion of the adoption of sociological approaches in HCI.

Sociology, Ethnography, and Anthropology

One common feature of the set of sociological approaches that will be discussed here is their reliance on field materials, or observational studies of behavior. A fieldwork orientation contrasts with other possible approaches in sociology, including the use of laboratory studies, surveys, statistical techniques, or primarily theoretical analysis. In basing their investigations on fieldwork, the different approaches reveal a common tradition that originates not in sociology but in anthropology.

Sociology and anthropology are closely related, and in some places almost overlap; it can be hard to see where one ends and the other begins. Clifford Geertz (1973) suggests that whereas sociology examines the emergence and maintenance of social structures and patterns of social interaction, anthropology studies the cultural webs of signification that give those structures and interactions meaning. This distinction is perhaps unavoidably approximate, but it nonetheless sets out a rough and ready basis for distinguishing between the disciplines.

Anthropology, as a distinct body of enquiry, emerged in the mid-nineteenth century. In Europe, this was a period of colonial occupation, and it was the colonial experience that served as the basis for European anthropological research, drawing, in the first instance, on the reports

and experiences of explorers, travelers, traders, and colonial officers.³ At around the same time, American interest in anthropology grew out of the encounter with the Native American cultures, as represented particularly by the work of Franz Boas in the Pacific Northwest (Boas 1921).

One distinction between European and American approaches to anthropology at that stage was in their use of fieldwork and observational methods. By fieldwork, here, I mean extensive, detailed observations of the daily life and practices of other peoples. With the immediate and local availability of the native society, American anthropology was based in fieldwork from its inception. In Europe, however, the peoples who constituted the objects of anthropological interest were generally at some considerable distance from the anthropologists themselves. The result of this was that a good deal of European anthropology of the time was conducted from the safety and comfort of libraries, and compiled second-hand from field reports. The separation between experience and theorizing was largely taken as read, as illustrated by this snippet from a travel handbook:

It is the duty of every civilised traveller in countries newly opened up to research to collect facts, pure unvarnished facts, for the information of those leading minds of the age, who by dint of great experience, can ably generalize from the details contributed from diverse sources. (Johnson 1889:398)

So, perhaps the most radical change in European anthropology was to arise in the role of field work investigations, and in particular in the development of ethnographic methods.

The Emergence of Ethnography

Arguably, the dominant figure in developing the role of ethnography in anthropology is that of Bronislaw Malinowski. Malinowski was a Pole working in the United Kingdom in the early part of the twentieth century. In 1914, he joined a field trip to Australia and New Guinea under the leadership of C. G. Seligman. Unfortunately, war between Britain and Germany was declared while Seligman's party was en route from the United Kingdom to Australia; and Malinowski, as a subject of the Austro-Hungarian empire, was subject to internment on reaching the destination. However, he persuaded the authorities that, if their concern was merely to have him in a safe place where he could do no harm, there was no need to

lock him up in an internment camp; they could, instead, let him spend the time ensconced in some remote place, safely out of the way.

The result was that Malinowski spent most of the following few years on the Trobriand Islands, an archipelago to the east of New Guinea. He spent this time in the detailed study of the culture and practices of the native population. His explication of the “Kula Ring,” a hitherto mysterious practice involving arduous and dangerous sea journeys undertaken to engage in the ritual exchange of seemingly worthless gifts, is perhaps the best known of a landmark series of writings (Malinowski 1922, 1930, 1935) that emerged from this period of intensive observation of daily life among the Trobrianders, and in which he set out the principles that helped establish him as one of the preeminent anthropologists of his day.

Malinowski’s work in the Trobriand Islands, however, is also a landmark in a different sense, not simply for what it revealed, but for how it was conducted. It is not least from Malinowski’s long-term, in-depth engagement with the Trobriand Islanders, his analytic approach and form of reportage, that modern ethnographic fieldwork has been developed. In many ways, Malinowski established ethnographic fieldwork as the dominant paradigm for anthropological research.⁴

Ethnography places an emphasis on the detailed understanding of culture, through intensive, long-term involvement and what anthropologist Clifford Geertz (following Gilbert Ryle) calls “thick description.” It is often based upon participant-observation, in which the ethnographer immerses himself or herself in the culture in question. The central element is to explore the member’s own view of his or her life and culture. That implies the need to be able to describe not just what the members of that culture *do* but *what they experience* in doing it; why it is done and how it fits into the fabric of their daily lives. Of course, the idea that the member’s perspective is primary does not mean that the ethnographer’s job is simply to ask people what they’re doing, write it down, and bring it home. The member’s own report is clearly a major element in the story, but it cannot be accepted blindly. Rather, in attempting to represent the culture from the member’s point of view, the ethnographer attempts to avoid preconceptions or analytic orientations from outside the specific setting of the investigation. In this way, the development of ethnographic fieldwork was a radical departure

from approaches, such as survey-based work, that preceded it, at least in the European tradition.

Although ethnography developed as an approach to the anthropological investigation of “exotic” cultures, it has come to occupy an important place as a basis for fieldwork not just in anthropology but in sociology too. The cross-fertilization between two disciplines as closely related as anthropology and sociology is, of course, a continuous process; but for our purposes here, trying to trace the particular strands of development that have led to the role of ethnography in interactive system design, the work of the Chicago School of sociology is particularly relevant.

Ethnography and Sociology: The Chicago School The Chicago School emerged from research conducted at the Department of Sociology at the University of Chicago. Particularly in the period from the 1930s until the 1960s, under the direction Robert Park and, later, Everett Hughes, Chicago sociologists engaged in a wide-ranging program of investigations focused particularly on urban and working life in contemporary America. In addition to a distinctive analytic perspective that they brought to these investigations, the Chicago School sociologists also adopted ethnographic, participant-observer approaches to the collection of field materials. The anthropological aspect to their writings is heightened by the fact that their topics are frequently subcultures on the fringes of ordinary society, such as those of tramps and hoboes, alcoholics, recreational drug users, and jazz musicians.

One particularly relevant feature of the work of the Chicago School was its detailed exploration of particular modes of work. Examples include Becker, Geer, Hughes, and Strauss’s (1961) classic study of the “career” of medical students, as well as investigations of the working lives of nurses (Davis 1968), funeral directors (Hebenstein 1954), airline pilots (Wager 1959), janitors (Gold 1964) and schoolteachers (Becker 1952). The critical role that these played was to introduce a concern with the detail of how work gets done, and the use of ethnographic methods in studying working practice. This perspective, and the understandings that emerged about how work was conducted, subsequently came to play an important role in the adoption of sociological

approaches in Computer-Supported Cooperative Work, such as in Gerson and Star's (1986) seminal paper on office procedures.⁵

By that time, though, sociology was already being adopted as an approach to understanding Human-Computer Interaction in other ways.

Sociology in HCI

HCI's "origin myth" traces its emergence to the development of a relationship between psychology and computer science. In particular, it originates (for many) in, first, the application of techniques and models from cognitive psychology to the problem of understanding what goes on when people work with computers and, at the same time, how those understandings can be reflected back into the design of those systems. Given the psychological background, then, it is perhaps not surprising that the first appearance of "the social" in HCI was not sociology but social psychology.

Social psychology is concerned with how an individual's thoughts and emotions are affected by interactions with others. With the advent of digital communication systems and computer networks, social psychologists became interested in how these interpersonal relations could be manifested in communication mediated by computer systems (e.g., Kiesler, Siegel, and McGuire 1984). For example, phenomena such as "flaming"—indulging in abusive and heated electronic mail exchanges—are extreme examples of social interactions whose existence seems to suggest some particular characteristic of electronic communication. More generally, computer-mediated communication provided a novel environment in which to study issues such as self-presentation, attribution of personality traits, and other features of everyday communication, and in turn to refine our understandings of how these take place in other communicative media.

So, in the same way that psychologists could help inform the design of interactive systems by understanding the cognitive implications of particular forms of design, sociologists could lend an understanding of the settings in which these computer systems would be deployed, and the ways in which they would both affect and be affected by those settings. At the same time, sociologists also introduced a set of techniques by

which the workings of those settings could be examined and uncovered; methods, like ethnography, that could be used to gain detailed understandings of how work was conducted.⁶ The effect of the introduction of these techniques was to broaden the scope of inquiry and show how the use of computer systems involved more than simply the user and the computer, but also the context of the activity that the user was engaged in—a turn, as one writer characterized it, from “Human Factors” to “Human Actors” (Bannon 1991).⁷

Currently, social science research features most prominently in HCI as part of the processes of requirements gathering and system evaluation. The use of ethnographic materials is most common here. Advocates for socially based studies of work have found that ethnographic approaches can be used to uncover requirements for a system design through the detailed observation of the working setting. In contrast, more traditional approaches—based perhaps on functional specifications or on laboratory-based usability studies—tend to be disconnected from the lived detail of the work. Usability evaluations are generally concerned with the detail of interactional features of software systems, are carried out in laboratories in controlled conditions, and measure performance on artificial tasks across a range of subjects; they are designed to answer questions such as, “Does our new visualization improve the speed with which people can find information?” From an ethnographic perspective, these sorts of questions are meaningless when decontextualized and examined in the sterile confines of a laboratory. Ethnographers look for a more direct engagement. The ritual gift exchange of the Kula Ring can only be understood within the context of the life of the Trobriand Islanders; by the same token, the only way to come to a good understanding of the effectiveness of a software system is to understand how it features as part and parcel of a set of working practices, as embodied by a group of people actually using the system to do real work in real working settings. Ethnographic studies, then, tend to take a broader view of the relationship between technology and work.

On a more analytic level, the use of ethnographic methods for this sort of work is also rooted in a distinction between *work processes* and *work practice*. Work processes are the formalized or regularized procedures by which work is conducted; procedures for authorizing payments, for

ordering supplies, for repairing machines, or whatever. Work processes are captured and codified in rulebooks, manuals, recipes, and similar artifacts. In contrast, ethnographers in HCI have frequently drawn attention to work practice—the informal but nonetheless routine mechanisms by which these processes are put into practice and managed in the face of everyday contingencies. Work practice is frequently informal and seemingly innocuous, but often provides the lubrication that prevents formalized processes from seizing up. Filling out a form in pencil so that work can proceed before the numbers are finalized, proceeding without a rubber-stamp authorization when relevant managers are not available, routinely “calling ahead” to inform other people of work that might be coming their way, or responding to such a call by beginning to schedule time for jobs that have not yet arrived—these are all examples of the practices that people develop to make processes work. What the “working practice” researchers observe is that the actual, practical business of making processes work involves a considerable amount of approximation, invention, improvisation, and ad hoc-ery. “Getting things done” means being able to step outside the rules, being able to interpret and anticipate them, and so achieve a smooth organization of work despite everyday problems like lost receipts, broken fax machines, unreadable notations, missing ingredients, and absent colleagues.

It is important to recognize that the duality of practice and process is inevitable. No matter how clearly or carefully framed, a process description can never eliminate the need to interpret it for specific occasions. Similarly, the ways in which people may deviate from formalized procedures tend to reflect a better or more fruitful adaptation of the process to the specific circumstances in which the activity is carried out. So, in understanding and uncovering the everyday practices through which people manage and accomplish their work, the goal is not to eliminate them, nor is it to turn those practices into processes by rigidly appropriating them. Attempting to eliminate or stabilize practice would result in effects similar to the labor practice of the “work-to-rule”: a rigorous adherence to process and procedure in which effective work grinds to a halt. Practice is always dynamic, arising as a way to mediate between processes and the circumstances in which they are enacted. The reason to study practice is to understand how this dynamic mediation takes place.

This is particularly relevant for the development of information systems where, all too often, designers presume that the formalized work processes set down in the organizational handbook constitute a perfectly adequate description of what actually goes on. They are encoded into software systems without accounting for the flexibility with which they will be put into practice. Work practice studies emphasize that the handbook's description is, inevitably, only a part of the story. Their view of the relationship between people and technology emphasizes the critical creative involvement of the people doing the work even in cases of what seems like rote procedure, ripe for automation. The emergence of Computer-Supported Cooperative Work from the "office automation" community was partly a result of different perspectives on this process/practice dichotomy.

The most common form of this sort of research, then, is the ethnographic investigation of a particular domain of work, with an eye toward the technological opportunities it offers or design constraints that it imposes. I will give two examples here, from the domains of air traffic control and factory production printing.

Ethnography of an Air Traffic Control Center

One of the best-known ethnographic field investigations of work carried out in the domain of CSCW is that into air traffic control conducted by a multi-disciplinary team of sociologists and computer scientists based at Lancaster University in England (Hughes et al. 1995).⁸ In fact, commercial Air Traffic Control (ATC)—the real-time management of the air space occupied by commercial flights—has become something of a canonical workplace example in CSCW; in addition to the pioneering Lancaster work, ATC has been explored by a number of other researchers (e.g., Rognin, Salembier, and Zouinar 1998; Berndtsson and Normark 1999; Mackay and Fayard 1999). Although certain common features arise across these different studies, the Lancaster work will be the main focus here.

As an ethnographic study, the topic for this investigation was not the rules and procedures of air traffic control as they might be laid down in an organizational manual. Instead, it looked at the actual practice of air traffic control as it occurs and unfolds, moment by moment, day to day, and as it is executed and experienced by the air traffic controllers themselves.

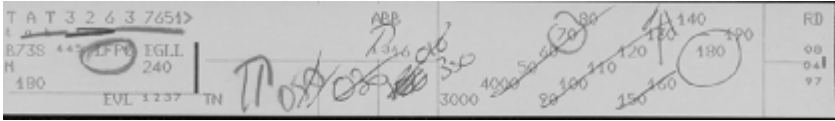
One seemingly obvious thing to say about air traffic control, as it is done by air traffic controllers, is that it happens in air traffic control centers. Location is important. The fact that these sorts of studies are often referred to as “workplace studies” is significant; the work happens in a place, and the actual practical activity that is the object of investigation cannot be divorced from the environment or setting in which it unfolds. So the first place to look to understand air traffic control is the center itself.

At the air traffic control center, the focus of the controller’s activity is one of a number of stations or “suites.” Each suite is generally responsible for one “sector” of airspace and is manned by two air traffic controllers, two assistants, and one sector chief. Each suite provides staff with various resources to manage the air traffic, including radar displays and communications equipment.

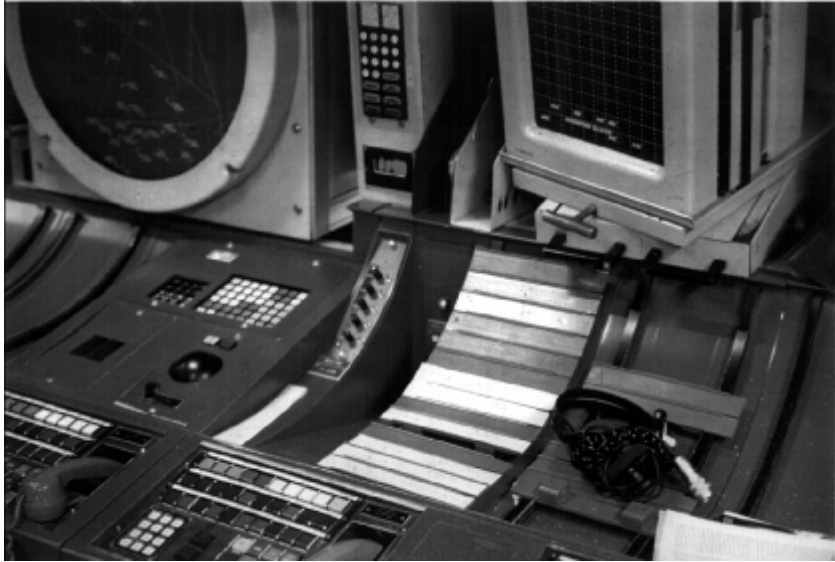
In exploring the conduct of the work in situ, the Lancaster group began to uncover the particular importance of one of the resources by which the work is managed—the “flight strip.” A flight strip is a strip of card approximately eight inches long and one inch tall that is used to represent a particular airplane en route through the sector. The various items on the strip show relevant information such as the call signal of the airplane, its heading and altitude, airspeed, and so forth. Flight strips are arranged in racks, arrayed in the working space of the air traffic controllers (see figure 3.1).

Although relevant flight information is maintained in a database and shown on the radar displays, it is not the electronic representations but rather the physical flight strips that are the primary focus of the air traffic controllers’ activity and their primary resource in managing the air space. Indeed, the ethnographers observe, “It is possible, and on occasion necessary, to perform ATC with the strips but without radar. To do so with radar but not flight strips, however, would pose extreme difficulties” (Hughes et al. 1992:115).

Essentially, the controllers *manage the airspace by managing the strips*. For example, when a flight is instructed to move to a different level, the controller will write the new level on the strip underneath the current level indication; and when the pilot acknowledges the instruction, then the old flight level will be crossed out. During the transition, the flight strip represents not



(a)



(b)

Figure 3.1

Air traffic controllers use flight strips (a), arrayed in a bay by their console (b), to represent the work of managing air space. Photographs courtesy of Wendy Mackay and the Centre d'Etudes de la Navigation Aerieenne.

just the plane, but also the plane's movement and the controller's work in managing the airspace. Strips can convey information not just through their contents, but also through their physical configuration. The arrangement of strips in the different racks available to the controller indicates the status of the various flights, and the controllers are observed to "cock out" strips (pulled out of alignment) to draw attention to them and indicate that there is some pending work to be done.

So, by the ways in which they employ flight strips, the air traffic controllers transform some of the work of managing airspace into a physical

process of monitoring and managing the set of strips corresponding to current traffic. This is more than simply using the strips to represent the state of the airspace. The work of managing the space becomes the embodied performance of physical activity, arising around the specific details of the work site itself.

Although strips are manipulated by individual controllers working at their suites, their role goes beyond individual controllers. Recall that the controllers work in teams, with two controllers working together with assistants and a coordinator. So, in addition to the work of managing the airspace, controllers also have to deal with the work of coordination between members of the team. Strips are used to tackle these problems too; the work of the team and the coordination of distributed activities are also intricately bound up with the physicality that the flight strips lend to the task. Flight strips arrayed around the workspace are not only a way of coordinating the work, but also a way of making it publicly accessible to others who understand how to read and interpret the physical configurations. Critically, the flight strips do not just represent the state of the airspace, but also represent the *activity of the controller* managing that space. The strips provide an easy, at-a-glance summary of the state of the work, manifested as part of the immediate environment, and this provides a resource to others who have to coordinate their work in that environment. So, for example, controllers coming on shift were seen to stand for a while behind the working controller, watching their activity to become familiar with the current state of the airspace. Observing the activities within the physical setting allows them to learn about the state of the work.

The Lancaster study looked at the management of air traffic not as an abstract, clinical affair, but as the practical and practiced everyday work of air traffic controllers. What it uncovered was the way in which this work is in every way organized around the features of the setting in which it takes place. The flight strips, for example, do not just record information, but are part of the very way in which the work is done, both for an individual controller and for the others whose activity must be coordinated with that of the controllers. For the system designers, partners in the investigation, this turned the design away from the management of decontextualized information toward the ways in which

information resources could not only represent the work but also themselves be the medium through which the work was conducted.

Ethnography of a Print Shop

The second example I want to present here explores a different workplace. In this case, the setting is a production print facility. This is an industrial print shop handling small-scale print runs (that is, brochures, forms, manuals, and similar materials, rather than commercial books) for a variety of clients. The print shop is capable of handling all aspects of the printing process, including design, prepress, reproduction of various sorts, binding, and finishing. They handle large and small jobs, both regular and one-off. Bowers, Button, and Sharrock (1995) present an ethnographic investigation of this setting. In particular, they are interested in the relationship between, on one hand, the set of practices through which the print workers organize and arrange their daily activity, and, on the other, their use of a computer system that embodies a formalized model of the production printing process—in other words, the two sides of the process/practice coin.

One of the issues they investigate is how the print shop operators achieve what they call a “smooth flow of work” through the print shop. By a “smooth flow,” they mean a balanced work load that keeps the print machines busy, avoids bottlenecks and backlogs, and ensures that all the print jobs are completed in a timely manner. These are all important concerns—expensive print machines are only producing income when they are running, bottlenecks put an undue load on people and resources, and timely completion is important if the print shop is to satisfy its customers. On the other hand, these goals may conflict, especially because the print shop has, at best, limited control over the jobs it will be asked to handle and the order in which they might arrive.

So, as Bowers and his colleagues observe, ensuring a smooth flow of work is not a straightforward process. It involves making skilled judgments about the requirements of each print job, anticipating likely future load, interleaving or concurrently carrying out different jobs, and so forth. It is a continual juggling of resources, demands, and expectations. Knowing that a regular job is about to arrive, for instance, the operators can anticipate the demands that will have to be met and orga-

nize their current workload more effectively in anticipation (perhaps clearing a machine that will soon be needed for another job). Similarly, the mix of jobs that are waiting to be performed at any given moment can affect the decisions about which jobs, or which *parts* of which jobs, should be performed next. Bowers, Button, and Sharrock detail a range of these practices and show how they feature as part of a system of practical reasoning about the work of the shop. Through these practices, the print-shop workers manage the practical problems of achieving a smooth flow of work.

However, the particular occasion for this investigation was the introduction of digital reprographic technologies and an associated system that offers the opportunity to manage and regulate the work of the shop. Digital reprographic machines offer a variety of features that traditional machines cannot, such as separating scanning from printing, storing digital files for printing later, offloading jobs from one machine to another, and so forth. In addition, because these new print machines are essentially computers, they can also communicate across a network with a management system designed to handle scheduling and job management. It could achieve this through an internal representation of the process of handling a print job—a “workflow” representation—that it could use to coordinate and account for print-shop activity. Although this offers a number of advantages and opportunities to both the management and the clients of the print shop, the fieldwork shows the negative impact that the system’s introduction had on the ways in which the print workers organized their work.

Of course, the workflow system also set out to achieve a smooth flow of work. However, the formalized procedures around which it was built did not account for the ways in which the practical work of the print shop would often involve stepping *outside* the regularized and formal procedures. For instance, where the print-shop operators could, themselves, preallocate machine time to regular jobs whose arrival could be anticipated, the computer-based system could not begin to process a job that had not yet arrived in the shop. Similarly, while the print-shop workers could divide their time between two jobs, or complete a short job on one machine while a long job ran unattended on another, the computer system maintained a rigorous one-to-one relationship between

jobs and operators that could not account for the standard practices by which the operators would actually manage their activities.

The outcome of the ethnographic work is to see the management of activity on the print-shop floor as a “setting-ed,” “occasioned,” or “situated” activity. In other words, the actual moment-to-moment organization of the work is contingent on the setting in which it emerges; the physical environment, the time of day, the stack of jobs awaiting completion, the materials available at hand, the understanding of the context of the work, and so forth. This is in contrast to the decontextualized view that the workflow system took, a view that focused on *this* job or *that* job but not on *the whole work of managing the print shop*. It failed to account for the real work of the print shop by divorcing that work from the setting in which it was carried out.

Both of these studies, then—in the print shop and at the air traffic control center—use detailed observation of the conduct of working activities to draw attention to the ways that people accomplish their work. The work doesn’t just “happen”; it has to be *made* to happen by the people who do it. In turn, this “making” is not an abstract process, but one that is firmly tied to the setting in which the work takes place and the specific circumstances in which it emerges.

One of the reasons that both of these studies focus on the occasioned nature of the practical accomplishment of work features is that they are both drawn from a particular analytic position. Both of these investigations are ethnographies, of course, but in addition they both draw on a sociological perspective called ethnomethodology. Ethnomethodology is a sufficiently common perspective in the HCI literature that we should explore it a little more deeply.

An Analytic Perspective

Although, as I have shown, there had long been a general sociological interest in interactive technology, a decisive turning point for the role of sociology in HCI was the publication of Lucy Suchman’s book *Plans and Situated Actions* (Suchman 1987). What Suchman set out in this book was a detailed analytic critique of the then-dominant paradigm for modeling human behavior in Artificial Intelligence; but, in focusing on “the problem

of human-machine communication,” she also provided the basis for a radical reorientation of the role that sociology could play in the development and analysis of interactive systems.

The primary focus of Suchman’s critique is the notion of a “plan” as it then featured, both technically and conceptually, in the domain of Artificial Intelligence. The “planning” paradigm in AI is as old as the discipline itself (which, of course, is not so old). It models human activity in terms of the formulation and execution of plans. According to this paradigm, plans are scripts for sequences of actions. Plans are formulated through a set of procedures beginning with a goal, which is then decomposed into a sequence of subgoals. The subgoals may themselves be similarly and repeatedly decomposed, until the resultant subgoals are primitive enough that they can be achieved through simple actions. These simple actions can be pieced together in sequence to form a plan for achieving the main goal. As the plan is executed, the results of each action are monitored to make sure that the system is still on track; in case of failure, a new plan may be formulated. So, if I want to drink some coffee, I can reduce this to a sequence of subgoals (fetching the beans, grinding them, mixing them with water, and so on) that must be achieved in order to attain my primary goal. These subgoals have an ordering; to grind the beans, I will first have to fetch them from the jar where I keep my coffee; the water has to be boiling before I can use it, which means I’ll have to fill the kettle, and so forth. If there’s no coffee in the jar, I may have to make a new plan to go to the store. By a continual process of this sort, a plan is formulated, and then the process of actually making myself some coffee involves simply progressing through the plan and carrying out the actions it describes, monitoring and replanning as necessary (see figure 3.2).

The “planning” model was, at the time Suchman was writing, the dominant paradigm for the development of intelligent applications. It could be used to guide robot motion (goal: reach the door) or to create systems that derive mathematical proofs (goal: show that $x^2 > 2x$ for $x > 2$). It could be applied across a range of domains because of its seemingly natural occurrence as a feature of human problem-solving. What Suchman did, however, was to show how this model failed to take into consideration a range of ways in which social sciences had radically revised our

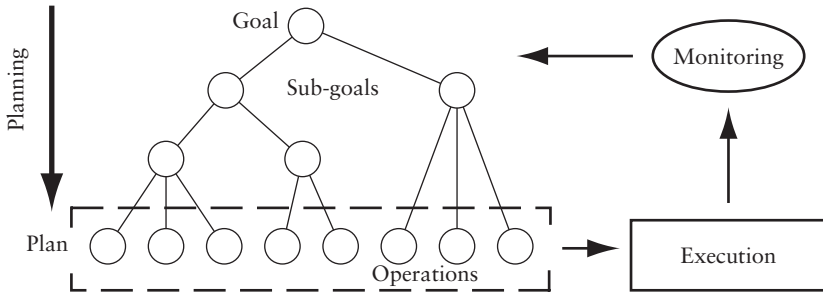


Figure 3.2
The planning model, in outline.

notion of how people act in the world. Drawing on a range of analytic perspectives, especially ethnomethodology, Suchman showed how the basic elements of the planning model had been under a sustained attack that questioned the cognitivist position. The planning model sees features of the world (and of our interaction with it) as stable, objective phenomena; this enables the relatively unproblematic execution of a plan formed around these objective phenomena. In contrast, Suchman presented a model of interaction with the world in which the apparently objective phenomena of the cognitivist model were, instead, active interpretations of the world formed in response to specific settings and circumstances. In this model, a “plan” might provide a resource that guides an individual’s action, but the plan is not “executed.” Instead, it is one of the features that shapes active individuals’ moment-by-moment responses to the situations in which they find themselves. The sequential organization of behavior, in Suchman’s model, is an ongoing, improvised activity. Our actions are organized in response to the features of the setting in which they arise; action is “situated.”

Suchman used this perspective to explore the problem of human-machine communication. Her observation is that the planning model that so dominated cognitive science was also the basis of the design of interactive devices. Interaction models assumed that users engaged in actions according to preformulated plans, and used the same techniques in order to build up representations of users’ skills and intentions. In terms of the planning model, intentions, requirements, steps, problems,

and outcomes are all stable features of interaction to be used to build models of what the user needs and how to achieve it. Suchman provided detailed analyses of the interactional problems arising from the mismatch between, on one hand, the clean-cut, abstract, and stable models that a system might have of interaction and, on the other hand, the much more messy, immediate, and fluid circumstances in which the system's users find themselves.

One of the particularly important features of Suchman's work for the account I am developing here is that her investigation was an analytic one. Her goal was not to investigate the use of computer technology in some particular working setting, but rather to tackle the ways in which the account of cognitive action that the planning perspective represents is incommensurate with the analytic account of situated social action she presents. In doing so, Suchman turned sociological discourse in HCI away from the purely empirical, and introduced an analytic perspective to the discussion. In fact, as it turned out, she brought into particular focus the specific analytic position from which her argument was developed, that of ethnomethodology. Through the influence of Suchman's work, ethnomethodology came to be a prominent sociological position within HCI.

Ethnomethodology

Although ethnomethodology is a far from dominant or even widely understood position within sociology, it has nonetheless come to occupy a position of some significance within HCI, and particularly within CSCW. Suchman's early work is certainly one reason for this, but since that time a number of other ethnomethodologists have also turned their attention to technological issues. In fact, the ethnographies described earlier in this chapter—of air traffic control and of print-shop operation—both form part of a program of ethnomethodological investigations in CSCW.

Ethnomethodology originated in the work of Harold Garfinkel in the 1960s, first brought together in his book *Studies in Ethnomethodology* (Garfinkel 1967). Ethnomethodology is not simply another theory of social action, sitting comfortably alongside the others that have been put forward. Ethnomethodology rejects the very notion of abstract theorizing on which most analytic accounts are based, and practiced by “the

worldwide social science movement and its armies of social analysts.”⁹ Ethnomethodology’s program is a *respecification* of the issues, methods, and very terms of reference of sociology.

It arises from Garfinkel’s reconsideration of one of the fundamental problems of sociology, the “problem of social order.” This problem, essentially, deals with the question of how stable and orderly social facts and relations can arise out of the independent action of individuals. How is it that each of us acting on our own nonetheless reproduces a stable social world? This is arguably *the* central sociological problem, and it had been taken as such by Talcott Parsons, whose landmark book *The Structure of Social Action* (1937) sets out the theoretical model that dominated American sociology for decades.

Garfinkel was not simply attempting to work out some small but knotty theoretical problem that Parsons had left unaddressed. Instead, he wanted to question the very foundations on which Parsons’s work, and all of conventional sociology, was built.

Emile Durkheim, one of the founding fathers of sociology, had observed that “the objective reality of social facts is sociology’s fundamental principle” (Durkheim 1938). This principle motivated the scientific investigation of that objective social world, which, in turn, spawned the development of a variety of theories cast in terms of grand themes such as class, capital, and gender. For Garfinkel, however, the problem of social order undermined the whole edifice of sociological theorizing. For him, Durkheim’s “objective reality of social facts” was not a principle at all, but a phenomenon. It was not to be assumed, formulated, and refined, but to be studied; it was the *topic* of sociology rather than the starting point. In Garfinkel’s view, the objective of sociology was not to develop abstract theories of social reality, but rather, to understand how social reality was achieved; how people *made it work*. This study of the commonsense methods by which people manage and organize their everyday behavior, he called “ethnomethodology” (for the study, *logos*, of native, or *ethno-*, methods).

One way to understand Garfinkel’s position on traditional sociological theorizing is to think about what sorts of things the theories are, and, even more importantly, who knows and applies them. These are, after all, theories of social action, governing the everyday action of members

of society. However, in conventional sociology, these facts and rules are not available to us as members of society, but rather, remain to be uncovered by sociologists. Garfinkel uses the term “cultural dope” to describe the actor as theorized by conventional sociology, blindly acting in accordance with a set of social rules of which he remains unaware. Critically, though, ethnomethodology observes that people *do* have reasons for acting the way they do. They continually operate according to explicable mechanisms by which they regulate and organize their action and understand the action of others. These commonsense understandings (“common” in the sense that they are shared—“what everyone knows that everyone knows”) are the object of ethnomethodology’s investigation. Garfinkel denies the right of the professional sociologist to any privileged insight into the rules that govern everyday, orderly social action; to ethnomethodology, “lay and professional sociological theorising are epistemologically equivalent.” In other words, in the course of everyday life, everyone, always, is engaged in “practical sociological reasoning,” when, as part and parcel of what they do, they have to figure out what other people mean and in turn figure out how to act themselves in order to get things done. The knowledge that people bring to bear in carrying out this practical sociological reasoning is no less valid than the theoretical models that professional sociologists might offer when *they* try to figure out what “society does.”

Ethnomethodology, then, turned its attention to the detailed analysis of actual practice, often drawing on ethnographic materials, and attempted to find, within them, evidence for the ways in which people achieved orderly social conduct. One particularly good example of this is the subfield of Conversation Analysis, pioneered by Harvey Sacks (1992). Sacks used recordings of real conversations as the basis for highly detailed analyses that attempted to uncover the mechanisms by which conversational interaction was structured. In these studies, again, we can see reiterated the theme of ethnomethodology, to look for the emergence of social order out of the details of what people do rather than from abstract theory. So, for example, Conversation Analysis claims that the word “hello” is not defined as a greeting in some vast social dictionary, but rather that specific *uses* of the word “hello” can *constitute* a greeting when interactionally organized in such-and-such a way or at such-and-such a

time. It acts as a greeting when people use it as a greeting (at the opening of a conversation, or when someone arrives) and, as a corollary, it can also be used as, and be heard to be, something other than a greeting—an exclamation, an inquiry, a solicitation, an exclamation of surprise, and so forth when used in other ways or at other times.

The two ethnographic studies discussed earlier both reflect an ethnomethodological orientation. They used ethnographic methods to collect and arrange the field materials, but relied on ethnomethodology to inform the analysis. It provides a particular stance toward the organization of action that is reflected in the analysis of what the materials demonstrate and the conclusions that can be drawn from them. This pairing of ethnographic fieldwork and ethnomethodological analysis has often been a source of confusion in HCI. It has sometimes led people to believe that ethnography and ethnomethodology are the same thing, or that one necessarily implies the other. Certainly neither is the case. However, these two studies follow in a tradition of ethnomethodological studies of work, in which the ethnomethodologists' attention is directed toward the practical logic not of conversation or generic action, but of specific domains of activity. Examples include the work of the police (Bittner 1967), mathematicians (Livingston 1982), 911 call operators (Whalen 1995), and a range of studies of scientists (Garfinkel, Lynch, and Livingston 1981; Lynch 1982; Lynch, Livingston, and Garfinkel 1983).

Across these specific domains of inquiry, ethnomethodology takes the same approach. On the basis of specific observations of activity, it attempts to uncover the commonsense methods by which people achieve the orderliness of action. People invoke these methods as practical solutions to practical problems. While the concern with practical rationality and commonsense methods is a hallmark of the ethnomethodological approach, it does not entirely originate with ethnomethodology. In focusing sociology's attention on the experiences of everyday life rather than on abstract theorizing, Garfinkel was quite consciously following in a philosophical tradition called phenomenology. Phenomenology had originated with Husserl, but Garfinkel drew primarily from the work of Alfred Schutz, who had explored the consequences of Husserl's thought for theories of social action. Like ethnomethodology, phenomenology

places primary emphasis on the experience of the everyday world rather than disconnected or abstract reasoning.¹⁰

The phenomenological character of ethnomethodology is an important element in the story of embodied interaction, and it will be explored at length in later chapters. First, though, I want to return to the issue of the relationship between social science and interactive system design.

“Technomethodology”

The two case studies presented earlier are examples of the most common way to use sociology in interactive system design. Sociological approaches can be harnessed to help us understand how work is conducted in real settings, and so, how interactive technologies can be designed to assist it (or at least, to hinder it less than they often do already). Although this approach can often result in better designs that are a much better fit for the ways in which people work, they do not go very far in addressing the kinds of critique that Suchman was making. Her target was not a specific design, but rather, the way in which a conceptual model was used to support a whole range of technologies. This is a much deeper issue. So a number of people have called for a deeper connection between sociological understandings and the design of interactive technologies. This would be an approach that deals not so much with *this* technology or *that* form of work, but rather more generally with interactive technology per se and the generally operative social processes that underpin any sociological account of behavior.

Graham Button and I coined the term *technomethodology* to describe a deeper relationship between technological design and ethnomethodology (Dourish and Button 1998). The word itself is a little flippant, but the model it proposes is not. By a “deeper” relationship, we mean one that satisfies two criteria. The first is that it attempts to draw not simply on a set of observations of a specific working setting, but rather on ethnomethodology’s fundamental insights about the organization of action as being a moment-to-moment, naturally occurring, improvisational response to practical problems. The second is that it attempts to relate these understandings not simply to the design of a *specific* interactive system aimed at a *specific* setting, but rather, at the basic, fundamental

principles around which software systems are developed—ideas such as abstraction, function, substitution, identity, and representation.

There are a number of reasons to look for connections at this more foundational level. One is that we believe that there are a number of *systematic* ways in which conventional system design undermines or removes the resources upon which human interactional behavior is based. (This is the sort of critique that Suchman made of the planning model.) If that is so, then we need to address the problems in an equally systematic way, considering not just this design or that design but the basic models around which those designs are built. A second reason is that we want to do this in a way that preserves the distinctive character of ethnomethodological reasoning, rather than simply the ethnographic observations of particular working settings.

Developing such a foundational relationship is a long-term objective. We have, so far, been approaching it by exploring particular areas of both technical and ethnomethodological interest, trying to find overlaps and mutual orientations to common issues. One that we have explored is the relationship between ethnomethodology's conception of "accountability" and the role that "abstraction" plays in the analysis and development of software systems. These two ideas are conceptually complementary, but in the differences between them lie some interesting problems for interaction.

Accountability The notion of "accountability" is a fundamental feature of the ethnomethodological perspective. As we have seen, ethnomethodology's concern is with the commonsense understandings by which people find the world rational, and so available for their own practical actions and activities. Again, "commonsense" here means "commonly held," even if the domain is high-energy physics rather than more mundane matters. This is important because it highlights ethnomethodology's contention that what it means to be a member of a language community (or, perhaps, an "action-community") is to share a set of understandings of how to act, and how to understand action, within that community. In other words, "acting rationally" and "perceiving action to be rational" are reciprocal aspects of the same set of understandings. This is the basis of ethnomethodology's notion of accountability:

[The] central recommendation [of ethnomethodological studies] is that the activities whereby members produce and manage settings of organized everyday affairs are identical with members' procedures for making those settings "account-able." [. . .] When I speak of accountable, my interests are directed to such matters as the following. I mean observable-and-reportable, i.e. available to members as situated practices of looking-and-telling. I mean, too, that such practices consist of an endless, ongoing, contingent accomplishment: that they are carried on under the auspices of, and are made to happen as events in, the same ordinary affairs that in organizing they describe; that the practices are done by parties to those settings whose skill with, knowledge of, and entitlement to the detailed work of that accomplishment—whose competence—they obstinately depend upon, recognize, use and take for granted; and *that* they take their competence for granted furnishes parties with a setting's distinguishing and particular features, and of course it furnishes them as well as resources, troubles, projects and the rest. (Garfinkel 1967: 1–2).

This characteristically dense passage deserves close reading. There are two aspects of it that are especially relevant here: first, the explanation of what accountability is from an ethnomethodological perspective, and second, an exploration of how accountability arises as a feature of conduct.

Let's start with the first of these, the definition. As a feature of action, accountability, in this sense, does not mean moral or political accountability, as it might in everyday speech, but rather means "observable and reportable." Other members can observe and report, that is, make sense of, the action in the context in which it arises. These two concepts, membership and context, are important. Members are those who share the "common sense understandings," and the context in which action arises provides part of the means by which that action can be interpreted as understood as normal, rational action by them. The observable-and-reportable nature of conduct is available *to members* and as *situated* practices.

There is more than this, though. Accountability lies in the *reciprocity* of action and understanding. Garfinkel's argument is not simply that action can be found to be rational by those who understand it, but rather that the methods of understanding and making sense of action and the methods for engaging in it *are the same methods* ("the activities whereby members produce [action] . . . are identical with [their] procedures for making those actions 'account-able'"). In other words, being a competent member of some setting *is* being able to engage in action in ways that are recognizable to other members. So, the accountability of

action is not simply the property of being recognizably rational as it emerges in context, but also that it is organized so as to allow this. The organization of action serves to demonstrate what that action is. Recall the example of conversation earlier; I suggested that the word “hello” does not carry “greeting-ness” as an intrinsic feature or property, but rather that it is *used* in such a way that people can understand that it is being used as a greeting, through the way it is said and interactionally organized within a conversation and a sequence of social action (e.g., when directed to a new person arriving in the conversation). This is a feature of conversational practice, that the organization of talk demonstrates features of that talk to members sharing an understanding of language practice. It is an example of the role of accountability.

The second aspect of accountability dealt with in the passage is *how* accountability arises as a feature of social action. Garfinkel details a number of features of the accountable nature of action, and in particular, the relationship between accountability and action. In particular, he emphasizes that action and accountability cannot be separated from each other (“made to happen . . . in . . . the same ordinary affairs that . . . they describe”). The accountable aspect of activity is never a “commentary” on the activity, standing separately from it; rather, it is an intrinsic and inseparable feature of how the activity is woven into the fabric of action and interaction. At the same time, he also emphasizes that the accountability of action is not an absolute matter. It is an “endless, ongoing, contingent accomplishment;” the account that matters is one that is good enough for the needs and purposes at hand, in the circumstances in which it arises and for those who are involved in the activity.

The analytic concept of accountability emphasizes that the organization of action, as it arises in situ, provides others with the means to understand what it is and how to respond in a mutually constructed sequence of action. It turns our attention away from simply the perceived result or outcome of an action, to include how that result is achieved. We pay attention not just to the destination, but also to the route taken to get there. Ethnomethodological investigations, such as those into the organization of conversation, show how this is critically important in providing a basis for rational mutual action.

However, this idea does not mesh very well with the way in which we currently design interactive software systems. In fact, in some ways, it seems that the design of software is carefully arranged to undermine this very principle. The problem lies in the way in which software relies on a notion of abstraction.

Abstraction Software systems are built from abstractions. The extent to which abstraction is fundamental to software systems can be hard to explain to someone who hasn't built them; while to someone who has, it can be so ubiquitous as to have become invisible. But the very essence of software system design is the manipulation, combination, and creation of abstractions.

User interfaces offer us abstractions in the form of generic user interface components or “widgets” such as menus, buttons, labels, and scroll bars. Widgets are arranged to form an interface by a programmer, who manipulates them in the form of objects with certain definable and controllable characteristics (such as color, size, or font). These objects are abstractions in the traditional software sense—entities whose essence lies in the range of manipulations they allow, and whose inner workings can be ignored as long as the external constraints are maintained. The very notion of “object” in a software system is itself an abstraction, exposing the “essence” of the object (that is, exposing the ways in which it can be examined and manipulated) while hiding a range of details of implementation (such as where the object resides in system memory, or how the details of its internal components will be represented). Object systems are themselves implemented on top of programming languages, which are abstractly manipulated by other programs to yield machine-language representations, sequences of operations in the instruction set of whatever processor will run the program. Here, again, we encounter abstractions. The instruction set is an abstraction that hides a variety of possible implementations (as we see when “clone” processors appear using completely different technology to implement the same instructions, or when a manufacturer produces a new generation of processors that are “backward compatible” with last year's instruction set). Similarly, the notion of a uniform processor abstracts away the specific details of each individual device. Software systems, in other words, constitute a tower of

mutually constituted abstractions, right down to the abstractions of binary logic that we use to isolate ourselves from the messy world of continuous voltages and variable current.¹¹

Before it sounds as if I somehow disapprove of computational abstractions, I should make it clear that I do not. There are extremely good reasons that abstraction is such a fundamental principle. First, at each point in the chain from binary digits to interface widgets, abstraction makes it possible for us to treat a complex set of computational behaviors as a simple, higher-level object out of which we can build something new and even more useful. Second, it also allows us to use a single abstract object (such as a scroll bar) to capture a range of potential needs and uses. Third, it helps isolate one component from another so that they can be managed and maintained separately. Without these properties, it would be impossible to build a modern software system. Abstraction also offers some very practical, everyday opportunities to the users of computer systems. The flexibility implied by reusable abstractions is one of the reasons for the success of general purpose computing systems today—it allows them to be a word processor one minute, a financial planner the next, and a game once you're done working. At the same time, we rely on the isolation offered by abstractions every time we install a new piece of software on an old PC (or upgrade the machine without having to buy completely new hardware).

The essence of abstraction in software is that it *hides implementation*. The implementation is in some ways the opposite of the abstraction; where the abstraction is the gloss that details how something can be used and what it will do, the implementation is the part under the covers that describes how it will work. If the gas pedal and the steering wheel are the abstraction, then the engine, power train, and steering assembly are the implementation. Hiding the implementation and dealing with something in terms of its abstraction allows us to *isolate* one piece of a system from all the rest, and so to adopt a modular approach to design that sees the system as an assembly of interoperating components, with all the advantages alluded to earlier.

By enabling a modular, component-based approach to design, the idea of “information hiding” has become critical throughout the design of software systems. However, at the user interface, the situation is more

problematic. Within a system, we know that the different components will interact in fixed and predictable ways. Users are less predictable, though, and their actions less fixed. Users may have different goals in mind, different reasons for using the system and different ways in which they want to use it. In just the same way as they approach all other activities, they need to be able to decide what to do in order to get things done. In everyday interaction, as we have seen, ethnomethodology argues that accountability is the key feature that enables them to do this. The way that activities are organized makes their nature available to others; they can be seen and inspected, observed and reported. But this feature—the way that actions are organized—is exactly what is hidden by software abstractions. Not by accident, either, but by design. In the “information hiding” approach, the information that is hidden is information about how the system is doing what it does, how the perceived action is organized.

Here is an example. Most computers store files in a hierarchical arrangement of folders and subfolders. Many, these days, also offer networked file servers, which store files centrally for users connected across a network. Generally, these file servers are arranged so that they appear as part of the local file system. They look like another disk on your PC; or perhaps you have a folder on your desktop that is actually located on the remote system, but which you use just as if it were local. It even looks the same. Dragging a file onto that folder doesn’t write it directly to disk, but transfers it across the network to be stored on the file server.

The transfer is “transparent”; it happens automatically as part of the action of moving the file. This transparency is something of an illusion, of course, because there are at least two ways in which transparency breaks down and the network comes into view. The first is performance; copying a file across a network is slower than doing the copy on your own disk, and so the operation will take a good deal longer. The second is “failure modes”; because there are more things that can go wrong (network errors, remote server crashes, and so forth), there are more and different sorts of errors that can result from the networked copy than the local one (which will generally only fail because your disk is full).

The abstraction argument claims that the differences between the local folder and the remote folder do not matter. Both of them implement the

“folder” abstraction, which captures all the relevant features of folders; any other differences are simply “a matter of implementation.” However, the ways in which the copy is not transparent *do* matter. The differences between the folders *are* consequential. Two folders on my desktop, one for my local disk and one for a file server, may look the same, but I will use them differently. I arrange my work around the ways in which they work; for instance, although it might be acceptable to keep document files on the file server, I need much faster access to the program source files that make up my current project when I try to compile them, and so I will partition my work across these two disks as is appropriate. However, the means to make my decision—that is, the different characteristics of these two folders—are not revealed by the interface. The whole point of the interface, in fact, is to ensure that a single abstraction, the folder containing files and subfolders, is supported uniformly by both the local and remote implementations. The features that matter to me as a user are ones that have been hidden by the interface and by the abstraction that it supports.

Accountability in Interface Abstractions The question is: Would it be possible to address this problem by introducing a form of “accountability” for the interface, or more generally for the abstractions in terms of which computer system design is organized? Accountability, in this sense, means that the interface is designed so as to present, as a part of its action, an “account” of what is happening. The goal of the account is to make the action of the system concrete *as a part of* an ongoing interaction between the system and the user. So, the account should not simply be an abstract description of the system’s behavior, but rather an explication of how the system’s current configuration is a response to the sequence of actions that has led up to this moment, and a step on the path toward completing the larger action in which it is engaged.

The design of a user interface that presents such an account is certainly challenging, and elsewhere I explore some specific examples that are suggestive of a general pattern (Dourish and Button 1998). The user interface determines the form of the account, but that is not what I want to focus on here. The important issue here is not the *account*, but *accountability*—that is, how the account is related to the behavior it describes.

The relationship between an account and the behaviour it accounts for is the key feature of accountability in Garfinkel's analysis of social action, and so it is for us here too. It requires a technical approach that provides three primary features. First, we need to find a way to ensure that the account that is offered of the system's behavior—a representation of that behavior—is strongly connected to the behavior that it describes. The goal is not to ensure that they can never disagree, but rather that the programmer can be in control of the ways in which they agree or differ. Second, we need to find a way to allow this representation to be tied to the action in such a way that the account emerges along with the action rather than separately from it. The account is not a commentary on the activity it describes, but is part and parcel of the activity as it is carried out. Third, we need to ensure that the account that is offered is an account of the current specific behavior of the system, in its current configuration and tackling exactly this piece of work, rather than a generic account that says little more than, "Oh, this is how the system generally behaves."

One promising way that we have been exploring to achieve these goals is through the use of a software design technique called "computational reflection" (Smith 1984; des Rivières and Smith 1984). The reflection technique emerged originally in the domain of AI programming, but has found perhaps its most widespread application in the design of programming languages (e.g., Kiczales, des Rivières, and Bobrow 1991). The observation upon which reflection rests is that there are two domains of concern in the execution of any program; the domain about which the program is dealing (e.g. cells and formulas for a spreadsheet, words and paragraph formats for a word processor, or tanks and spaceships for a game) and the domain of the program itself (comprising its internal structures, program encoding, execution state, and so on). Normally, these two worlds of representation are kept separate. However, reflection provides a link between them. The reflective link allows a programming system to change the domain of its operation from one to another; to perform computation using not only the representations that refer to the outside world but also those internal representations that refer to its own operation. This gives a program the ability to describe its own internal state and even to operate upon itself by revising those internal

structures. The link between the two domains is called the “causal connection” between the representation of a program and its own behavior. The connection is “causal” because it allows the program not only to view but also to change the way that the internal structures operate. The difference between a computer’s representation of a book in a library database and its representation of its own program is that it can effect changes on the program representation, whereas no amount of computation in the database system can move the book from a borrower’s desk back to the library shelf. The system’s representation of its own behavior is within its effective reach. What is more, the representation of the program not only *describes* the program, but also *gives rise* to it; the program is, in effect, no more than the “performance” of the representation.

So, the “causal connection” is a two-way relation between a program and its reflective representation. As the program executes, the representation is changed; as the representation changes, so the behavior of the program is transformed. Reflection’s causal connection is also a route toward accountability in interactive systems. It provides the features required of the relationship between an account (representation) and the actual behavior of a system (program). It ties the two together in a way that can be controlled (because the system itself is managing the representation), that ensures that the account emerges inseparably from the action it describes (because it is not only a description of it but also the source of that action, and therefore cannot be separated from it), and that it describes the specific, ongoing activity of the system rather than an abstract and generic specification of it (because it reflects the here-and-now).

The details of using the reflection approach in interactive system design are not relevant here, and are still a topic of ongoing research. What is important, though, is that it turns on the ability to change the subject matter of a computation, from the representations of external entities to the internal representations by which those other representations are sustained. This ability to redirect and refocus attention and reference will be a significant element of the account of embodied interaction to be developed shortly.

The proposal that reflection can provide a basis for interface accountability is a radical one, but not necessarily in the ways that people sometimes

imagine. For example, it is sometimes interpreted as a proposal that interface representations can be “meaningful” in the same sense as human ones, or that by this representational sleight-of-hand we can make computer systems interact in just the same ways as people do. Of course, this is not the case. Reflective interface design does not mean that computers and humans are “conversing” on the same level, or even that the structure of computer-based dialog will now mirror that of human conversation. The accountability of the user interface is not the accountability of human social action. Despite the fact that its implementation turns on a rather esoteric technique, the proposal can actually be thought of as being very straightforward. Put simply, it says that because we know that people don’t just take things at face value but attempt to interrogate them for their meaning, we should provide some facilities so that they can do the same thing with interactive systems. Even more straightforwardly, it’s a good idea to build systems that tell you what they’re doing.

What is radical about the proposal is something quite different. What is radical is the relationship it proposes between technical design and social understandings. It argues that the most fruitful place to forge these relationships is at a foundational level, one that attempts to take sociological insights into the heart of the process and fabric of design.

Technomethodology is, perhaps, the most extreme proposal to bring these two elements together. However, it is by no means the only attempt to take the relationship between sociological reasoning and system design to be a deep one. By contrast with the technomethodological approach, let us consider one of a rather different nature.

Space, Place, and Locales

Our second example is a rather more obvious place for sociological understandings to provide insight, inasmuch as it focuses on collaborative settings in which social action takes place. However, what it has in common with the technomethodology approach is that it attempts to forge deeper connections between the disciplines than simply requirements for the operation of a particular system in a particular working setting, and to provide a more general set of understandings that are applicable across a range of applications and domains. This example

concentrates on the development of spatial models and metaphors in interactive and, especially, collaborative systems.

The idea of “space” is a fundamental aspect of how many interactive systems operate. System designers create spaces of all sorts; virtual ones such as “name spaces,” and real ones such as the two-dimensional computer “desktop” on which files and icons are arrayed. Across these different sorts of spaces, there are certain common elements. For instance, things generally appear *within* the space. There can generally be only one object at any given point in space. Things tend to stay where they’ve been put. Spaces define distances; things can be nearby or far apart once they’re in the space. And so on.

There is no mystery to the pervasive use of space as an organizing principle in user interface or software design. Space is so fundamental to our everyday experience that it permeates the way we think. In their book *Metaphors We Live By*, George Lakoff and Mark Johnson (1980) explore the role that metaphor plays in human cognition, by looking for key metaphorical ideas around which a host of specific metaphors arise. For instance, the core metaphor “argument is war” leads to such expression as “He went on the offensive,” “She destroyed my argument,” “His position is indefensible,” “They won the point,” and so forth. Many of their examples emphasize the importance that spatial concepts play in our thinking and our language; notions of distance (such as when two positions are “far apart”), up and down (up, they observe, is generally good, and down bad), and so forth.

The use of spatial metaphors as a basic organising principle has been adopted particularly within certain areas of Computer-Supported Cooperative Work (CSCW). One feature of the spatiality of the everyday environment, after all, is that we share it in common, and so a number of researchers have argued that spatial models provide a natural metaphor for collaborative systems design. So, “shared workspaces” of one sort or another have become a common feature of many collaborative tools. In particular, the design of Collaborative Virtual Environments (CVEs) such as DIVE (Frecon and Stenius 1998) or MASSIVE (Greenhalgh and Benford, 1995) uses space as a way for people to manage their accessibility, orient toward shared artifacts, and provide a “setting” for particular forms of interaction. The spatial model in a CVE can be used to manage interactions

between individuals, by, for example, requiring that they stand close to each other and face each other in order to have a conversation. In turn, this use of the spatial setting means that others will be able to see that two individuals are oriented in that way, and so understand that they are talking to each other. So, the design of CVEs is arranged to exploit the ways in which we understand how to interact with each other in the real world; CVE designers argue that by reproducing the consequences of spatial arrangements in virtual environments, those same interactional patterns can be carried over to the online setting.

However, over the last few years, an alternative view has emerged concerning the role of space in collaborative settings. It argues that “space,” although important, is not the constitutive element of the ways that interactions are organized in these settings. Instead, the new view draws a distinction between those interactive phenomena that are derived from the nature of the *space* in which they unfold, and those that are instead predicated on an understanding of the *place* that is occupied (Harrison and Dourish 1996).

The distinction between space and place is, approximately, a distinction between the physical and the social. “Space” is largely concerned with physical properties (or metaphorical physical properties). It concerns how people and artifacts are configured in a setting; how far apart they are, how they interfere with lines of sight, how actions fall off at a distance, and so on. By configuring the space in different ways, different kinds of behaviors can be supported. A raised stage, facing banked rows of seats, supports presentations to large crowds, while clusters of low, comfortable seats are more conducive to intimate or informal conversations. However, space is not enough to account for the different kinds of behaviors that emerge in different settings. Two settings with the same physical configurations and arrangements of artifacts may engender quite different sorts of interactions due to the social meaning with which they are invested. For example, although the stage of an academic conference is physically configured in ways very similar to a concert hall, it is generally not appropriate to get up and sing there. Similarly, meeting rooms and dining rooms can have similar arrangements, but we behave differently in them. Our behavior in these environments is governed by social norms, not by physical constraints. So while “space” refers to the

physical organization of the environment, “place” refers to the way that social understandings convey an appropriate behavioral framing for an environment. It’s not for nothing that we use the term “out of place,” but not “out of space”; the idea of “place” often plays a much more central role in determining behavior.

Our reason for setting up this contrast, of course, is to consider design implications. What are the design implications of taking a view centered on “place” rather than “space”? I’ll outline three here.

The first is that it turns our attention away from the structure of the space and toward the activities that take place there. Activities take center stage, and the structure of the space in which they happen falls away except in as much as it features as a part of those activities. So, for example, in collaborative environments, an appeal to “real-world” metaphors—to three-dimensionality, to reciprocity of movement and access, and so forth—is valuable only in as much as it contributes *directly* to the activities that take place there. Of course, the structure of the environment is often a key issue in controlling how interactions develop. Tom Erickson (1999) provides an enlightening example centered on the collaborative production of limericks in a chat room, and he makes a very convincing argument for the way in which the specific design details of the chat software involved play a significant role in engendering the particular sort of word play at work in his example; Lynn Cherny (1999) has some similar examples drawing on interaction in textual multiuser environments. However, in both of these cases, the spatial features themselves are not brought into the foreground as design elements; they emphasize not how to *design the space*, but how to *design for the interaction*. This is an important difference.

The second consequence is that “place” reflects the *emergence of practice*. That is, it is knowledge that is shared by a particular set of people based on their common experiences over time. Practice emerges over time in the space; but at the same time, the space is also turned toward the particular needs of the moment. Take an everyday example. In a meeting room, chairs are moved around to fit the occasion and the group of people. Discussions may work best in a circle where everyone can see everyone else; presentations, on the other hand, require a different configuration of the environment. A meeting room in which the

chairs and tables were fixed down and did not support these kinds of fluid rearrangements would not be a popular or a pleasant place. We need to be able to customize the space to our changing needs; we need to be able to *appropriate* it to the purposes at hand. Similarly, in virtual environments, we also need the ability to turn and twist the setting to suit our immediate purposes, which in turn requires that the environment be malleable enough to support this sort of appropriation. The important point to recognize here is that these practices emerge not from the designers of the system, but from the actions of its users. This means two things; first, that true places emerge only when really occupied day-to-day, not in demonstrations or experiments that last a few hours; and second, that place can't be *designed*, only designed *for*.

The third consequence of the place-centric view is that an idea of place is relative to a particular *community* of practice. The practice that constitutes the place is shared by a particular set of people. The community of practice might be defined by a particular set of skills or training; it might be defined according to a particular point in space and/or moment in time. But places will be different for different communities in the same setting. So, for example, a shopping street will be a different sort of place early in the morning, when the delivery trucks arrive; in the afternoon, as children swarm out of school; in the evening, as people come out to meet and dine; and on a lazy weekend afternoon, as people sit in cafes, to chat, see, and be seen. Similarly, we can see these sorts of differences in technological settings, such as the wide range of responses that people might have and practices they might develop around communication technologies. The same sort of regular patterns of people associated with particular times of day can be seen, for example, in MUDs (Curtis 1992); and one of our experiences with media space technologies was the range of different expectations and practices that arose around privacy and connection models even with very similar technologies and overlapping groups (Dourish 1993).

This understanding of the ways in which behavior depends on the social connotations of physical settings has also been a topic for sociological investigation. Perhaps the best-known social scientist to tackle these questions was Erving Goffman. Goffman's book *The Presentation of Self in Everyday Life* (Goffman 1959) drew attention to the ways in which people

managed their conduct as a way of managing the impressions that others would form of them, and how, consequently, the ways in which they conduct themselves will vary with the particular “audiences” for their conduct at any given moment. Drawing on a theatrical metaphor, he contrasted “front-stage” and “back-stage” behaviors. So, for example, shop assistants will treat customers deferentially when “on-stage” (that is, on the shop floor) but act differently when among themselves in the back of the store. Front-stage and back-stage are not specific locations, of course, but rather are situations from which particular modes of conduct are drawn; and similarly, there may be multiple degrees of front-stage-ness or back-stage-ness.

The Locales Framework

In addition to Goffman’s work, the idea of place as a setting for action has also featured in other sociological writings. In Structuration Theory, Anthony Giddens (1984) uses the concept of “locale” to capture the same idea:

Locales refer to the use of space to provide the settings of interaction. [. . .] It is usually possible to designate locales in terms of their physical properties, either as features of the material world or, more commonly, as combinations of those features and human artifacts. But it is an error to suppose that locales can be described in those terms alone. (Giddens 1984:188)

So, if these examples demonstrate the way in which place has featured in sociological theorizing, then we can look for attempts to relate these understandings to the practice of design, in ways perhaps similar to the technomethodological approach we have already encountered. One is the framework developed by Geraldine Fitzpatrick (1998) as a foundation for the design of collaborative systems.

Fitzpatrick’s work on the “Locales Framework” uses locales as the central element of a framework designed to help system designers understand the social organization of activity and to support design activities that take these understandings into account. Although the term *locale* is that used by Giddens, the Locales Framework is in fact built around the theory of action developed by Anselm Strauss (1993). Various elements of Strauss’s model of activity are woven into the framework in such a way as to reveal their interconnectedness with respect to particular settings for study and design.

The Locales Framework has five primary components or *aspects*, called Foundations, Civic Structure, Individual Views, Interaction Trajectory, and Mutuality.

Foundations encompasses the social world being addressed, and the sites and means that make up the locale. The idea of a social world is drawn directly from Strauss. A social world is a group of individuals brought together by a common commitment to collective action; both the common action or goal, and the communicative means by which a collective orientation can be established, are central elements of the idea. Social worlds, too, are settings for action, and action is carried out against a backdrop of previous activities (and with an expectation of future ones), so the social worlds perspective attempts to address not merely social groupings, but the *dynamics* of social action. Elements of the social world to which the Locales Framework draws attention include membership, duration, structure, roles, culture, focus, and tasks. The activities of a social world are supported by *sites* (the spaces or domains of activity) and *means* (the “furnishings” of a site). Whether sites are real or virtual, the introduction of sites and means relates the actions of the social world to specific manifestations and objects of action. In Fitzpatrick’s framework, a Locale arises from the relationship between a social world and the sites and means by which its activity is carried out.

On top of this foundation, the other aspects of the framework incorporate other elements of social settings. Civic Structure examines how the locale relates to others. In the same way as the meanings of actions are constituted by the other actions that come before and after them, so, too, can locales only be understood in relation to others. Going in the other direction, locales are also made up of individuals who have their own perspectives, concerns, roles, and forms of participation; the Individual Views aspect of the framework addresses this.

Strauss uses the term *Trajectories* to refer to the emergence of a particular course of action as it may be evolved through time and involving multiple actors. In relation to the rest of the framework, this ties together the way in which actions are situated within particular histories, and the notion of the collective action of social worlds. Similarly, social worlds have trajectories, as do individuals. The Interaction Trajectory aspect of

the framework introduces Trajectory as a way of understanding how social worlds develop, how people enter and leave, and how the activities in which they engage contribute to the various courses of action in which the social world is engaged. The temporal aspect of Trajectory also provides the means to consider phases, rhythms, and schedules—dynamic aspects of action that are more than simply sequence.

Finally, the Mutuality aspect explores the way in which these elements are made manifest or present in a space as a consequence of the manifestation of entities in a shared environment, made mutually accessible to the other participants:

For interaction to happen in a locale, there are basic requirements for presence and awareness. First, the potential interactants need some form of representation or way of making themselves (or being made) *present* in the locale. Secondly, the potential interactants need some way of being *aware* of the other's presence. Both presence and awareness possibilities for the interactants are facilitated by various *mechanisms* that are part of the locale domain(s). [. . .] For the purpose at hand, the interactants will make selective choices, consciously or unconsciously, about the use of such mechanisms or features according to their *capability* to do so. Hence, mutuality is the interplay of presence and awareness for interaction purposes, mediated by capability and choice. (Fitzpatrick 1998:134; emphasis in original)

So, where the locale foundations explored the sites and means of action—the furnishings of the locale—Mutuality considers how the ways in which these sites and means are made manifest to members of the social world, and the ways in which those members and their activities are themselves made manifest to others *through* the sites and means of action, are key elements in the ongoing maintenance of the social world.

In terms of the directions explored in this chapter, the Locales Framework, first, takes the distinction between place and space and uses it to talk about the issues of the setting in which action unfolds; second, relates it to a specific set of theoretical insights into the organization of social action; and third, arranges this in such a way as to support the design of novel systems.

The goal of the Locales Framework is not to present a theory of social action, but rather to package a set of sociological understandings so that they can be used to inform the analysis of working situations and the design of technologies to support cooperative work. It reflects a distinct

orientation toward design, and so its effectiveness has to be judged in terms of the ways in which it can be applied in design settings.

The most fully developed application of the Locales Framework to design is in a system called Orbit (Mansfield et al. 1997), an environment for distributed collaborative work. Orbit was the successor to an earlier system called wOrlds (Fitzpatrick, Tolone, and Kaplan 1995), and it is perhaps in the differences between wOrlds and Orbit that we can see most clearly the influence that the Locales Framework had on the system's design. Orbit and wOrlds are both persistent collaborative workspaces, but they embody different approaches to the organization of activity in the workspace. If wOrlds was a system built around "space," then Orbit was one that took "place" as its primary focus. For wOrlds, the simulation of aspects of a shared physical environment was the primary mechanism by which activities could be migrated from the physical to the virtual environment. In Orbit, by contrast, the design is organized around the social action that takes place in the space and the ways in which the space provides a setting for the activities of people engaged in common tasks. Where wOrlds provided rooms that gave participants shared access to artifacts, Orbit provides a more nuanced sense of participation, in response to the ways in which people may occupy many different social worlds at a time, with greater or lesser degrees of intensity, and in different roles. Similarly, in Orbit, mechanisms are provided for maintaining a richer view of activity in the space, both synchronously and asynchronously, as uncovered by the mutuality aspect of the Framework.

The Locales Framework provides a basic set of orientations toward the problems of social action—ways of looking at social settings and systematically uncovering the issues at work—for the purpose of design. When the task is finding a way to make sociological understandings available to computer scientists for their unashamedly practical ends, there is clearly a set of compromises to be made. The same is true for the idea of technomethodology explored earlier. Relating sociological understandings to technical design principles is a different enterprise from either sociology or system design, with different goals and methods. What determines success at the end of the day is the ability to develop systems that resonate with, rather than restrict (or, worse, refute), the social organization of action.

Although the topics addressed by the Locales Framework and the technomethodology approach are radically different, they have some of the same spirit. In different ways, they attempt to draw out relationships between aspects of technological design and aspects of sociological analysis. They provide frameworks that help system designers invest interactive systems with sociological understandings. In their attempt to develop a model of “social computing,” they look for deeper connections than the ethnographic requirements capture approach. Although this is a riskier approach, it is one that offers considerable payoff.

Summary

This chapter has introduced a wide range of approaches to understanding social action and its relationship to the use of technology. These different approaches are not all compatible, and they reflect different sets of assumptions and commitments. Rather than focus on these differences, though, I want to try to draw out some common features.

First, a common theme through this chapter has been that social action is *embedded*. By this I mean that it is firmly rooted in the setting in which it arises, where that setting is not just material circumstances, but social, cultural, and historical ones as well. When talking about social action, we need to talk about the specifics of the action, when and how it arose, where and for whom it was conducted, and so forth. This concern with specifics often means, at the same time, a concern with the *mundane* aspects of social life, the background of taken-for-granted everyday action. This is a distinctive element of the sociological positions we have looked at here, even though they may themselves take different perspectives.

However, the observation that social action is inseparable from the uncountable minutiae of everyday life does not mean that it is completely chaotic. Social action is clearly organized. The focus of our attention, though, changes when we see social order as emerging from practice. The focus of attention becomes *how* orderly social conduct emerges from the detail of each setting in which it is undertaken, and how orderliness is *achieved* in the face of the endless contingencies to which it is subject.

When interactive technology enters the picture, it does so in a variety of ways. First, it is itself part of the setting; the specific features of each

technology and how they are deployed and used introduce transformations to the conduct of everyday action. So, for example, the ways in which electronic communication systems have tended to increase our connectedness to each other and access to information have, in turn, changed our expectations about the availability of other people or information sources; to be “out of touch” for a week seems strangely quaint (and ever harder to achieve). Second, technology is increasingly the medium within which activity takes place. We are used to the ways in which the physical world mediates our actions, and how it forms a shared environment whose characteristics are thoroughly predictable; when we converse face to face, I understand how my gestures will appear to you (and in fact, if I didn’t, there would be little point in making them). Technological systems as a medium for social conduct are very different inasmuch as the inherently disconnected, representational nature of computer systems means that actions can be transformed in unpredictable ways. Finally, technological systems are themselves embedded in a set of social and cultural practices that give them meaning at the same time as being constrained and transformed by them.

Putting these together yields the conclusion that, just as the embedded approach to social action turned attention to how the orderliness of social conduct was achieved (rather than simply assuming it), so too, given the role that technology places in social settings, the key question is to understand *how* the relationship between technology and social action comes to be worked out in different situations, and from these to understand how the features of technological design and the features of everyday social settings are related.

This excerpt from

Where the Action Is.
Paul Dourish.
© 2001 The MIT Press.

is provided in screen-viewable form for personal use only by members of MIT CogNet.

Unauthorized use or dissemination of this information is expressly forbidden.

If you have any questions about this material, please contact cognetadmin@cognet.mit.edu.