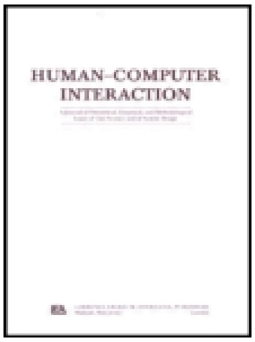


Human–Computer Interaction

ISSN: 0737-0024 (Print) 1532-7051 (Online) Journal homepage: http://www.tandfonline.com/loi/hhci20

The Intellectual Challenge of CSCW: The Gap Between Social Requirements and Technical Feasibility

Mark S. Ackerman

To cite this article: Mark S. Ackerman (2000) The Intellectual Challenge of CSCW: The Gap Between Social Requirements and Technical Feasibility, Human–Computer Interaction, 15:2-3, 179-203, DOI: 10.1207/S15327051HCI1523_5

To link to this article: https://doi.org/10.1207/S15327051HCI1523_5

Published online: 09 Dec 2009.

Submit your article to this journal

Article views: 1194

View related articles

Citing articles: 172 View citing articles

Full Terms & Conditions of access and use can be found at
http://www.tandfonline.com/action/journalInformation?journalCode=hhci20

HUMAN-COMPUTER INTERACTION, 2000, Volume 15, pp. 179–203
Copyright © 2000, Lawrence Erlbaum Associates, Inc.

The Intellectual Challenge of CSCW: The Gap Between Social Requirements and Technical Feasibility

Mark S. Ackerman
University of California, Irvine

ABSTRACT

Over the last 10 years, Computer-Supported Cooperative Work (CSCW) has identified a base set of findings. These findings are taken almost as assumptions within the field. In summary, they argue that human activity is highly flexible, nuanced, and contextualized and that computational entities such as information sharing, roles, and social norms need to be similarly flexible, nuanced, and contextualized. However, current systems cannot fully support the social world uncovered by these findings. In this article I argue that there is an inherent gap between the social requirements of CSCW and its technical mechanisms. The *social–technical gap* is the divide between what we know we must support socially and what we can support technically. Exploring, understanding, and hopefully ameliorating this social–technical gap is the central challenge for CSCW as a field and one of the central problems for human–computer interaction. Indeed, merely attesting the continued centrality of this gap could be one of the important intellectual contributions of CSCW. I also argue that the challenge of the social–technical gap creates an opportunity to refocus CSCW.

Mark Ackerman is an associate professor in the Computing, Organizations, Policy, and Society (CORPS) group in Information and Computer Science at the University of California, Irvine; he is currently on leave at the MIT Laboratory for Computer Science.

CONTENTS

1. INTRODUCTION

Over the last 10 years, Computer-Supported Cooperative Work (CSCW) has identified a base set of findings. These findings are taken almost as assumptions within the field. Indeed, many of these findings have been known and have been debated within computer science, information science, and information technology for over 20 years. I discuss the findings at length later, but in summary, they argue that human activity is highly flexible, nuanced, and contextualized and that computational entities such as information transfer, roles, and policies need to be similarly flexible, nuanced, and contextualized.

Simply put, we do not know how to build systems that fully support the social world uncovered by these findings. I argue here that it is neither from lack of trying nor from lack of understanding by technical people. Numerous attempts have been made, not only within CSCW, but within many other subfields of computer science to bridge what I call here the *social–technical gap,* the great divide between what we know we must support socially and what we can support technically. Technical systems are rigid and brittle—not only in any intelligent understanding, but also in their support of the social world.

Researchers and computer professionals have edged toward a better understanding of this social–technical gap in the last 10 years, and CSCW systems have certainly become more sophisticated. We have learned to construct systems with computer-mediated communication (CMC) elements to allow people enough communicative suppleness; yet, these systems still lack much computational support for sharing information, roles, and other social policies. Important CSCW technical mechanisms (e.g., floor or session control) lack the flexibility required by social life. The social–technical gap still exists and is wide. Exploring, understanding, and hopefully ameliorating this social–technical gap is the central challenge for CSCW as a field and one of the central problems for human–computer interaction (HCI). Other areas of computer science dealing with users also face the social–technical gap, but CSCW, with its emphasis on augmenting social activity, cannot avoid it. I also argue later that the challenge of the social–technical gap creates an opportunity to refocus CSCW as a Simonian science of the artificial (where a science of the artificial is suitably revised from Simon's strictly empiricist grounds).

This article proceeds in three parts. First, I provide an overview of CSCW, briefly reviewing the major social and technical findings of the field, particularly with regard to the construction of computational systems. Next, I argue that there is an inherent gap between the social requirements of CSCW and its technical mechanisms; I demonstrate this through a discussion of a particular CSCW research problem, privacy in information systems. Finally, I discuss potential resolutions for the social–technical gap. In the section, the requirements for a science of the artificial are evaluated, along with the need for such a viewpoint for CSCW.

2. A BIASED SUMMARY OF CSCW FINDINGS

Most of this section will be obvious to CSCW researchers but might be a useful overview for non-CSCW researchers. This section does not attempt to be a complete summary of CSCW assumptions and findings; rather, the emphasis is on those social aspects most germane to the social–technical gap.

Although March and Simon's (1958; Simon, 1957) limited rational actor model underlies CSCW, as it does for most of computer science, CSCW researchers also tend to assume the following:

- Social activity is fluid and nuanced, and this makes systems technically difficult to construct properly and often awkward to use. A considerable range of social inquiry has established that the details of interaction matter (Garfinkel, 1967; Strauss, 1993) and that people handle this detail with considerable agility (Garfinkel, 1967; Heritage, 1984; Suchman, 1987). (In this article, following Strauss, 1991, and others, I use *nuanced* narrowly to denote the depth of detail as well as its fine-grained quality. Connotations to the term include agility and smoothness in the use of the detail.) People's emphases on which details to consider or to act on differ according to the situation (Suchman, 1987). Yet, systems often have considerable difficulty handling this detail and flexibility.

  For example, Goffman (1961, 1971) noted that people have very nuanced behavior concerning how and with whom they wish to share information. People are concerned about whether to release this piece of information to that person at this time, and they have very complex understandings of people's views of themselves, the current situation, and the effects of disclosure. Yet, access control systems often have very simple models. As another example, because people often lack shared histories and meanings (especially when they are in differing groups or organizations), information must be recontextualized to reuse experience or knowledge. Systems often assume a shared understanding of information.

One finding of CSCW is that it is sometimes easier and better to augment technical mechanisms with social mechanisms to control, regulate, or encourage behavior (Sproull & Kiesler, 1991). An example is the use of chat facilities to allow norm creation and negotiation in commercial CSCW systems.

- Members of organizations sometimes have differing (and multiple) goals, and conflict may be as important as cooperation in obtaining issue resolutions (Kling, 1991). Groups and organizations may not have shared goals, knowledge, meanings, and histories (Heath & Luff, 1996; Star & Ruhleder, 1994).

  If there are hidden or conflicting goals, people will resist concretely articulating their goals. On the other hand, people are good at resolving communicative and activity breakdowns (Suchman, 1987).

  Without shared meanings or histories, meanings will have to be negotiated (Boland, Tenkasi, & Te'eni, 1994). As well, information will lose context as it crosses boundaries (Ackerman & Halverson, 2000). Sometimes this loss is beneficial in that it hides the unnecessary details of others' work. Boundary objects (Star, 1989) are information artifacts that span two or more groups; each group will attach different understandings and meanings to the information. Boundary objects let groups coordinate because the details of the information used in one group need not be understood completely by any other group.

  An active area of CSCW research is in finding ways to manage the problems and trade-offs resulting from conflict and coordination (Malone & Crowston, 1994; Schmidt & Simone, 1996).

- Exceptions are normal in work processes. It has been found that much of office work is handling exceptional situations (Suchman & Wynn, 1984). In addition, roles are often informal and fluid (Strauss, 1993). CSCW approaches to workflow and process engineering primarily try to deal with exceptions and fluidity (e.g., Katzenberg, Pickard, & McDermott, 1996).

- People prefer to know who else is present in a shared space, and they use this awareness to guide their work (Erickson et al., 1999). For example, air traffic controllers monitor others in their workspace to anticipate their future workflow (Bentley et al., 1992; Hughes, King, Rodden, & Andersen, 1994). This effect has also been found in other control room settings and trading floors (Heath, Jirotka, Luff, & Hindmarsh, 1994). An active area of research is adding awareness (i.e., knowing who is present) and peripheral awareness (i.e., low-level monitoring of others' activity) to shared communication systems. Recent research is addressing the trade-offs inherent in awareness versus privacy, and in awareness versus disturbing others (Hudson & Smith, 1996).

- Visibility of communication exchanges and of information enables learning and greater efficiencies (Hutchins, 1995b). For example, copilots learn from observing pilots work (i.e., situated learning, learning in a community of practice). However, it has been found that people are aware that making their work visible may also open them to criticism or management; thus, visibility may also make work more formal and reduce sharing. A very active area of CSCW is trying to determine ways to manage the trade-offs in sharing. This is tied to the issue of incentives, discussed later.

- The norms for using a CSCW system are often actively negotiated among users. These norms of use are also subject to renegotiation (Strauss, 1991). CSCW systems should have some secondary mechanism or communication back channel to allow users to negotiate the norms of use, exceptions, and breakdowns among themselves, making the system more flexible.

- There appears to be a critical mass problem for CSCW systems (Markus, 1990). With an insufficient number of users, people will not use a CSCW system. This has been found in e-mail, synchronous communication, and calendar systems. There also appears to be a similar problem with communication systems if the number of active users falls beneath a threshold (called the "melt-down" problem in Ackerman & Palen, 1996). Adoption of CSCW systems is often more difficult than for single-user systems because CSCW systems often require initial buy-in from groups of people, rather than individuals, as well as continued buy-in.

- People not only adapt to their systems, they adapt their systems to their needs (coevolution; Orlikowski, 1992a; O'Day, 1996). These adaptations can be quite sophisticated. People may use systems in ways completely unanticipated by the designers. One CSCW finding is that people will need to change their categories over time (Suchman, 1994). System designers should assume that people will try to tailor their use of a system.

- Incentives are critical. A classic finding in CSCW, for example, is that managers and workers may not share incentive or reward structures; systems will be less used than desired if this is true (Grudin, 1989). Another classic finding is that people will not share information in the absence of a suitable organizational reward structure (Orlikowski, 1992b). Even small incremental costs in collaborating must be compensated (either by reducing the cost of collaboration or offering derived benefits). Thus, many CSCW researchers try to use available data to reduce the cost of sharing and collaborative work.

Not every CSCW researcher would agree with all of the aforementioned assumptions and findings, and commercial systems (e.g., workflow systems) sacrifice one or more of them. The previous list provides an ideal type of what needs to be provided. Because some of the idealization must be ignored to provide a working solution, this trade-off provides much of the tension in any given implementation between "technically working" and "organizationally workable" systems. CSCW as a field is notable for its attention and concern to managing this tension.

3. THE SOCIAL–TECHNICAL GAP IN ACTION

Attempts to deal with online privacy nicely demonstrate the gap between what we need to do socially and what we can do technically. I use the example of the Platform for Privacy Preferences Project (P3P) of the World Wide Web Consortium. P3P is an attempt to create a privacy standard for the Web. It is inherently a CSCW system, and an HCI problem, because it deals with how people manage their private information with regard to other people, companies, and institutions:

> The goal of P3P is to enable users to exercise preferences about Web sites' privacy practices. P3P applications will allow users to be informed about Web site practices, delegate decisions to their computer agent when they wish, and tailor relationships with specific sites. (Cranor & Reagle, 1999, p. 48)

It is important to detail at some length how P3P works and what its initial design goals were. Regardless of whether one believes in the efficacy of such protocols for ameliorating privacy issues per se, P3P aims at a common collaborative problem, sharing information. As such, it must tackle the social–technical gap discussed previously. With regard to P3P, the gap is large. In the following description, it is not important to grasp the details as much as understand the information space under consideration:

> P3P is designed to help users reach informed agreements with services (Web sites and applications that declare privacy practices and make data requests). As the first step towards reaching an agreement, a service sends a machine-readable P3P proposal …, in which the organization responsible for the service declares its identity and privacy practices.
>
> Proposals can be automatically parsed by user agents such as Web browsers and compared with privacy preferences set by the user. If there is a match between service practices and user preferences, a P3P agreement is reached. Users should be able to configure their agents to reach agreement with, and proceed seamlessly to, services that have certain types of practices; users should also be able to receive prompts or leave when encountering services that engage in potentially objectionable practices. Thus, users need not read the privacy policies at every Web site they visit to be assured that information exchanged (if any) is going to be appropriately used. (Cranor & Reagle, 1999, p. 49)

Note that the desire is to deal with this information space automatically, with the exact mechanism determined by those writing P3P clients. The necessity to handle this appropriately was raised with the additional goal of automatically transferring data if the agreement is made between the service and the user's agent. This part of P3P has been shelved for Version 1, partially because no one was confident that it could be done well. However, the original intention is worth noting:

> Some P3P implementations will likely support a data repository where users can store information they are willing to release to certain services. If they reach an agreement that allows the collection of specific data elements, such information can be transferred automatically from the repository. (Cranor & Reagle, 1999, p. 50)

Even a cursory examination shows a wicked problem (in the computer science sense of *wicked,* meaning an ill-formed, intractable problem). If we follow Goffman (1961), a user would wish to control the release of his or her private information on an ongoing basis to the various individuals and institutions within the environment. Roughly, this translates to allowing the user to customize information transfer in two dimensions–by the recipient (i.e., all potential recipients, perhaps including any interaction effects among recipients) and by the data itself (i.e., all possible pieces of private information, as defined by the user).

There are insoluble user interface problems here; users must be able to handle essentially an infinite information space. However, this is not merely a user interface problem; the problem is conditioned by the underlying social requirements. By the findings explicated previously (going back to Goffman, 1961), people do this everyday. Except in unusual circumstances, we do not have to deliberate about these information spaces in detail, and we do not need to laboriously click and switch modes within everyday information dissemination. To require users to do anything else than the apparently seamless change between "faces" (Goffman, 1961) is to place users of P3P within a social–technical gap.

One technical solution might be to allow users to group potential information recipients together into roles or other collections. For example, I may wish to hide my telephone number from all students, but not from professional colleagues. Unfortunately, again, by the findings explicated previously, in everyday life I move people seamlessly among those groupings (especially roles). Furthermore, exceptions are common and must be accounted for–I may wish to give a prized honors undergraduate my home phone number for a consultation. Again, people do this every day in a nuanced and apparently seamless manner. Although considerable work goes into accomplishing this everyday activity, people still manage to do it in a manner quite unlike the awkwardness afforded and required by current systems.

The online privacy problem is even more complex than stated earlier. The protocol currently allows for the expression of eight dimensions. Still others, such as consequence (what might happen) and duration (how long the data might be kept) were discarded as being nearly intractable technically, socially, or politically.

With some important exceptions, these eight dimensions within P3P incorporate most of the details of everyday life. Yet, one can easily assert that no one knows how to construct a suitable user interface for such a protocol. Without a completely accurate grouping mechanism (or some manner of collapsing categories in a meaningful way), few users will be able to correctly categorize a situation without errors. Fewer yet may take the time to categorize, because normal social activity does not require this explicit categorization. Moreover, one of the CSCW findings was that such categorization (and especially how categories are collapsed into meta-categories) is inherently political. The preferred categories and categorization will differ from individual to individual.

To summarize, there are no current HCI mechanisms to straightforwardly mechanize the naturally occurring, everyday social activity of handling personal information in its entirety. We must necessarily restrict the problem from what we know is appropriate to the social circumstances. This is the social–technical gap.

Within the privacy problem, one can see that the social–technical gap inherent in P3P results from three aligned issues:

1. Systems do not allow sufficient nuance. People make very fine grained distinctions, often based on contextual or historical knowledge. Systems do not do this, and when they attempt to do so, they either lack the requisite background knowledge or simplify the number of states.
2. Systems are not socially flexible. People switch among states gracefully. For example, people fluidly move among their faces as social settings require. People do not make these switches explicitly, whereas systems require people to explicitly switch states (e.g., roles).
3. Systems do not allow sufficient ambiguity. In most settings, people are inherently ambiguous about, for example, which role they are currently playing or the import of the information they are disclosing. People do not inhabit the discrete states of a decision or action graph; they inhabit states that are only partially determined and seldom made explicit.

Although P3P agents are only one possible collaborative application (and the problems are partially interface and individual-use problems), they demonstrate the social–technical gap. The next section surveys some technical work squarely within the CSCW field and attempts to show that this gap is fundamental.

3.1. Technical Research in CSCW

Until the last 2 or 3 years, it was not uncommon to read CSCW articles analyzing some aspect of system use or workplace activity that essentially argued that system designers just do not sufficiently understand the social world. The problem, then, was centered by social scientists in the process of design. Certainly, many studies in CSCW, HCI, information technology, and information science at least indirectly have emphasized a dichotomy between designers, programmers, and implementers on one hand and the social analyst on the other.

Indeed, early collaborative systems were awkward. In the 1980s, many researchers made roles, rules, and even floor control necessarily explicit when using a system. The coordinator (Flores, Graves, Hartfield, & Winograd, 1988) has been much maligned over its explicit roles and rules; one necessarily had to respond to requests for action. However, one can see explicitness as a design criterion in other CSCW systems, including gIBIS (Conklin & Begeman, 1988) and MPCAL (Greif & Sarin, 1987). There were notable exceptions:

Other systems, especially CMC systems (e.g., CRUISER in Root, 1988), were constructed with flexibility and social nuance as critical design factors. That they were partially successful led to considerable additional research.

Social nuance and flexibility were slowly added to all CSCW systems, as the undesirability of being explicit became an assumption within CSCW. For example, the original Answer Garden system (Ackerman, 1994; Ackerman & Malone, 1990) allowed only two basic roles: the information seeker who asked a question and the expert who answered. (There were tiers of experts, but all were explicit roles, such as help desk provider.) In real life, everyone has expertise about some topics, and everyone asks questions. Answer Garden 2 (Ackerman & McDonald, 1996) attempted to close this false dichotomy in roles by providing for ranges of expertise. People were assumed to be both seeking and providing information at different times. Flexibility was provided through the use of CMC components and through escalation among the different CMC components. I claim no particular intelligence here in creating better social fidelity over these several versions. Through the decade, all CSCW systems became more sophisticated, as technical researchers better understood the social requirements.

The understanding that CSCW technical researchers bring to the problem is well shown in Marwood and Greenberg (1994). Their article both demonstrates the social–technical gap again and shows the sophistication that CSCW technical researchers now have in understanding the gap. Marwood and Greenberg authoritatively argued that concurrency control (an aspect of which is floor control) is different for CSCW systems than for standard distributed systems. As they stated,

> In particular, concurrency control problems arise when the software, data, and interface are distributed over several computers. Time delays when exchanging potentially conflicting actions are especially worrisome. … If concurrency control is not established, people may invoke conflicting actions. As a result, the group may become confused because displays are inconsistent, and the groupware document corrupted due to events being handled out of order. (p. 207)

They went on to add, however, "Most concurrency control approaches are designed for non-interactive computer systems. … Groupware is quite different, because the distributed system includes not only computers but people as well. … people can be both more and less tolerant of concurrency problems than computers" (p. 210).

The article discusses locking, serialization, and optimism policies in detail. The authors make it clear that fine-grained locking is difficult to implement well:

> The coarser the lock, the more difficult it is for people to work closely together. … Noticeable delays, however, will interfere with the flow of interaction. For example, selecting a circle and moving it, or moving a text cursor forward and then typing should both be enacted as continuous operations. (p. 211)

Other technical researchers have also argued extensively that aspects of the social–technical gap must be satisfied in CSCW systems. For example, Rodden (1996) argued that systems must consider the ambiguity of awareness and privacy. Kaplan, Tolone, Bogia, and Bignoli (1992) and Dourish, Holmes, MacLean, Marqvardsen, and Zbyslaw (1996) argued that social protocols are considerably more flexible than technical systems. Clearly, CSCW technical researchers are not only aware of the gap but understand its nature.

However, it is not quibbling to question the efficacy of proposed solutions. Kaplan et al.'s (1992) solution was to require writing "social protocols" in Lisp. Rodden (1996) provided a welcome formal evaluation of awareness, but provided for only part of what people would consider peripheral awareness and privacy.

3.2. Arguments Against the Significance of the Gap

Section 3.1. suggested one argument against the significance of the social–technical gap: that this gap resulted merely from ignorance or habit by software designers and researchers. However, as section 3.1. pointed out, CSCW software researchers and designers are indeed aware of the need for nuance, flexibility, and contextualization.

There are other arguments against the importance of any social–technical gap to be examined before a reader should be satisfied. There are two major arguments remaining. First, one could argue that the social–technical gap will be solved shortly by some new technology or software technique. Second, one could argue that the gap is merely a historical circumstance and that we will adapt to the gap in some form. This section examines each argument briefly and shows why neither is a strong argument against suggesting plans of action to ameliorate the gap.

First, it could be that CSCW researchers merely have not found the proper key to solve this social–technical gap, and that such a solution, using existing technologies, will shortly exist. In this view, computer science will learn how to do machine learning, user modeling, or some other technique properly. This article cannot disprove that a technical solution is imminent. It may be. However, I would argue that such a technical solution is unlikely, because computer science, artificial intelligence (AI), information technology, and information science researchers have attempted to bridge the gap without success for at least 20 years. It is time to consider that the gap is likely to endure and that we should consider what to do about it.

A logically similar argument is that the problem is with the entire von Neumann machine as classically developed, and new architectures will ameliorate the gap. As Hutchins (1995a) and others (Clark, 1997) noted, the standard model of the computer over the last 30 years was disembodied, separated from the physical world by ill-defined (if defined) input and output devices. In this view, the von Neumann machine came to be socially inflexible, decontextualized, and explicit. Moreover, in this view, the existing von Neumann architecture led to programming practices that in turn led to explicit and inflexible systems using data that were far too simplified over the natural world. Some proponents of this argument suggest that newer architectures, such as the neural network, may hold more promise. It is believed that neural systems or similar architectures will have greater flexibility, being able to learn. It is hoped that these systems could mimic human activity. However, the efficacy of neural networks or other architectures has not yet been proven. Although it is possible that some future neural network systems could solve the social–technical gap, again this remains unknown. Again, we should consider the gap as enduring until proven otherwise, because the solution may or may not arrive.

A second argument against the significance of the gap is historically based. There are several variants: that we should adapt ourselves to the technology or that we will coevolve with the technology. In its more deterministic and mechanistic form, this argument can be seen as neo-Taylorism–we should adapt ourselves efficiently and effectively to the machine. It has been argued within the software engineering community, for example, that people should fit the necessities of process descriptions. The most famous form of this argument is Osterweil (1987). Osterweil argued that software engineering processes are software, and by extension, software engineers should function according to rule.

The coevolutionary form of this argument is that we adapt resources in the environment to our needs. If the resources are capable of only partial satisfaction, then we slowly create new technical resources to better fit the need. An example in Hutchins (1995a) is the slow evolution of navigational tools. For example, the Mercator projection for maps simplifies navigation calculations, and its creation was an act of tool adoption and coevolution. Moreover, if the resources are important enough, we may slowly change social practice to adapt.

The suggested outcome of the historically based variants is the same: Our culture will adapt itself to the limitations of the technology, so the technical limitations are not important. Admittedly, the variants have differing moral authorities; the neo-Taylorist version argues that we should change to be more rational, explicit, and predictable, whereas the coevolutionary version suggests evolutionary and progressive forces at work. One might even consider the neo-Taylorist to be a peculiar solution to the gap, arguing the gap's inherent benefit to society.

The coevolutionary argument is difficult to dismiss outright. It is hard to imagine that our culture would not adapt to any technology, and if this gap continues to exist, our culture will adapt to it. However, although coevolution will occur, the gap is still important to consider as a CSCW problem. It would be best to "round off the edges" of coevolution. As Postman (1992) argued, technologies have previously affected our culture in profound ways, especially when just assumed. Postman pointed to invisible technologies, or technologies chosen and so adopted as to become invisible to societal members, having profound and long-lasting effects. Grading is one such invisible technology. Grading student papers began with William Farish at Cambridge University in 1792, and Postman argued that numerically ranking individuals' work is a technology so assumed and valued by our society that it is largely invisible. Indeed, merely the invisibility of the technology leads to significant social problems (e.g., overuse of standardized intelligence tests).

As an intellectual discipline, HCI and CSCW should not allow unconscious decisions about technology features and adoptions. As Postman argued in a journalistic piece,

> What I'm interested in is having people take these questions seriously … . Technology is not an unmixed blessing. … we also need for them to talk about what new technologies can undo. … I just don't think we can go into these things anymore with our eyes closed. (as cited in McCreary, 1993, p. 84)

As Heilbroner (1994) and other researchers have argued, technological trajectories are responsive to social direction. I make the case that they may also be responsive to intellectual direction.[1] Indeed, a central premise of HCI is that we should not force users to adapt.

4. WHAT TO DO?

If the social–technical gap is real, important, and likely to remain, then as a field, HCI and CSCW must consider what to do with the gap. We can consider

---

1. Conceptually, coevolutionary effects lend themselves to an overly pessimistic reading of the situation as follows: If coevolutionary effects are inevitable and largely unforeseeable, then what intellectual guidance can be provided? Perhaps it is inevitable that we will merely continue to blunder our way forward with designs no matter what we know. This reading is contrary to the HCI tenet of providing guidance to design (at some level), and most HCI researchers would reject this pessimistic view. Assuming belief in some level of guidance, the gap still needs to be understood and dealt with.

it a black hole in the middle of our discipline, or construe it to be an important finding of our field. The argument here is that CSCW's vitality results from its understanding of the fundamental nature of the gap. Indeed, although the gap is often hazily assumed in the CSCW literature, we should make it an explicit intellectual focus.

Centralizing the social–technical gap as a necessary problematic in CSCW's intellectual mission is a major, first step. However, this is not to say that CSCW should continue to state and restate aspects and results of the gap–it may be time to move on. The HCI and CSCW research communities need to ask what one might do to ameliorate the effects of the gap and to further understand the gap. I believe an answer–and a future HCI challenge–is to reconceptualize CSCW as a science of the artificial. This echoes Simon (1981) but properly updates his work for CSCW's time and intellectual task.[2] In the remainder of this section, I discuss what this would entail.

4.1. A Return to Simon: The Science of CSCW

Thirty-two years ago, Simon (1969/1981) produced his seminal book *The Sciences of the Artificial.* In his autobiography, Simon (1991) admitted that *The Sciences of the Artificial* was thrown together from lectures, and many people feel the book lacks coherence. Yet, *The Sciences of the Artificial* became an anthem call for artificial intelligence and computer science. In the book he argued for a path between the idea for a new science (such as economics or artificial intelligence) and the construction of that new science (perhaps with some backtracking in the creation process). This argument was both characteristically logical and psychologically appealing for the time.

The book's basic premise is that engineering and design are fundamentally different from the sciences. The key to Simon's argument is his distinction between the artificial (as in *artifice*) and the natural. In a famous example, Simon (1969/1981) noted that farms are artificial in that they are created and maintained through human design and agency. Science, then, is about the analysis

---

2. I have found, through talks and reviews, that mentioning Simon is occasionally tantamount to waving a giant red cape in front of some social researchers. Simon is such a large figure, with such strong views about positivist methods, that he is extremely emblematic and problematic for many CSCW researchers. Indeed, until recently, he was for me as well. In the following sections, I caution the reader to try to separate Simon's overall goal from the particulars of his method.

Simon is contemporary, but we commonly do this with prior theorists. No one today would leap to take Vico or Comte at face value; their methods and specific social insights are aged. Yet their overall call to a science of the social is still very important. We should do the same for Simon's work; his call to a new type of science is also very important.

of the natural, but engineering is about the synthesis of the artificial.[3] In this view, engineering and design are synonymous, and new sciences are possible for understanding the nature of design. For Simon, the new sciences of the artificial include economics, organizational science, computer science, and artificial intelligence.

One might expect such an argument would be challenging to existing academic programs and appealing to new intellectual areas. Indeed, for many years, Simon's work was extremely influential, often indirectly, in U.S. programs of computer science and artificial intelligence. Yet, his call to create a science of design per se has gone largely unheeded.

Looking back, one can see a certain naivete about the complexities involved in creating new sciences concerned with the constructed world, whether technical or social. This naivete arose from at least two sources. First, Simon (1969/1981) confused the task of identifying fundamental intellectual issues in his sciences of the artificial with specific technical ideas and implementations. It is clear that he thought his particular methods would lead to new sciences (e.g., he does not separate the intellectual problem of how people examine options from the specific use of his General Problem Solver). Second, Simon (1969/1981) did not confront long-term, systemic incapability as an intellectual possibility. Simon was (and is) a progressive optimist.[4] At a simple level, CSCW's intellectual context is framed by social constructionism and ethnomethodology (e.g., Berger & Luckmann, 1966; Garfinkel, 1967), systems theories (e.g., Hutchins, 1995a), and many large-scale system experiences (e.g., American urban renewal, nuclear power, and Vietnam). All of these pointed to the complexities underlying any social activity, even those felt to be straightforward. Simon's (1969/1981) book does not address the inevitable gaps between the desired outcome and the means of producing that outcome for any large-scale design process, but CSCW researchers see these gaps as unavoidable. The social–technical gap should not have been ignored by Simon.

Yet, CSCW is exactly the type of science Simon envisioned, and CSCW could serve as a reconstruction and renewal of Simon's viewpoint, suitably revised. As much as was AI, CSCW is inherently a science of the artificial, as Simon

---

3. Simon (1969/1981) raised an important caution about engineering education as opposed to engineering practice. In his opinion, academic programs of engineering were not about design. In his view, they were schools of applied mathematics; design had vanished from their curricula and research programs.

4. Progressivism was an American political movement in the early 20th century that optimistically espoused progress through educational, political, and economic reform. It was a sporadic third party at the national level but a very strong political movement in the U.S. Midwest. Simon was raised in Wisconsin in the late 1910s and 1920s, both geographically and temporally the center of Progressivism in the United States.

(1969/1981) meant the term: CSCW is at once an engineering discipline attempting to construct suitable systems for groups, organizations, and other collectivities, and at the same time, CSCW is a social science attempting to understand the basis for that construction in the social world (or everyday experience).

CSCW's science, however, must centralize the necessary gap between what we would prefer to construct and what we can construct. To do this as a practical program of action requires several steps–palliatives to ameliorate the current social conditions, first-order approximations to explore the design space, and fundamental lines of inquiry to create the science. These steps should develop into a new science of the artificial. In any case, the steps are necessary to move forward intellectually within CSCW, given the nature of the social–technical gap.

4.2. Palliatives: Ideological, Political, and Educational

Centralizing the social requirements in a science of the artificial obliges us to address current conditions. Over the last 20 years, people have worked out a number of ideological, political, or educational initiatives in an ad hoc manner. This article has little to add to the actions of these many people; in this section I merely note how centralizing the gap leads to the logical coherence of these ideological, political, and educational initiatives.

Ideological initiatives include those that prioritize the needs of the people using the systems. For example, stakeholder analysis in information technology is a method that brings into a project the voices of all stakeholder parties. Participatory design is a similar method, actually employing important stakeholder parties in the design. Both methods address the inability to solve the social–technical gap by bringing forth a solution that is open and known to all important parties. The so-called Scandinavian approach to information systems design, where trade union participation is mandated, unequivocally addresses the political implications of the social–technical gap. Stakeholder analysis, participatory design, and the Scandinavian approach, as well as standard HCI techniques, provide users with the ability to block troublesome impacts. Knowing that such political initiatives will be logically necessary, as resulting from viewing the gap as an inevitable concern, may be an important step in ameliorating conditions.

Similarly, an educational perspective would argue that programmers and users should understand the fundamental nature of the social requirements. Moving past the naive perspective that additional education or training would bring software engineers the insights for effectively building programs that fit the social world, software engineers could be suitably trained to understand the organizational and social impacts that could result from their designs. If computer science does not know how to build systems that fully support the social world, then a computer science education should teach students what can and cannot be done.

However, palliatives by themselves do not create a science or lead to intellectual coherence in a research area. I next turn to those steps.

4.3. Beginning Systematic Exploration: First-Order Approximations

First-order approximations, to adopt a metaphor from fluid dynamics, are tractable solutions that partially solve specific problems with known trade-offs. They can be constructed from experimentation, although in mature sciences they should result from theoretical extensions. These first-order approximations essentially try to find work-arounds for the social–technical gap, to edge around it in ways that are not extremely odious and to do so with known effects. CSCW needs a set of these approximations, with suitable examples and an understanding of why they succeed. I return later to how these approximations might gather into a science.

CSCW already has a set of potential first-order approximations. One approximation is to provide systems that only partially address the social requirements. Extremely successful CSCW systems, such as electronic mail or chat systems, do not satisfy all social requirements. (Problems with electronic mail often result from contextual problems.) Much CSCW research is centered around knowing which social arrangements need to be satisfied for what tasks and settings; that is, the field is determining the approximation trade-offs. Considerable recent work examined the differences in communication media in providing the necessary social cues for computer-mediated communicative activity (e.g., Ackerman, Hindus, Mainwaring, & Starr, 1997; Kraut, Miller, & Siegel, 1996; J. S. Olson & Teasley, 1996).

Providing CMC components, such as chat, within a system is another approximation. As mentioned earlier, communication through these components allows people to make necessary social adjustments. For example, they can fluidly repair task breakdowns, such as determining which drawing stroke to keep in a shared drawing surface. The use of CMC components allows people to work out arrangements without making their roles or information explicit. They are an approximation, rather than a solution, because they exclude designs that wish to computationally augment communication (e.g., routing), role structures (e.g., workflow), and information processing (e.g., privacy).

Another approximation incorporates new computational mechanisms to substitute adequately for social mechanisms or to provide for new social issues (Hollan & Stornetta, 1992). An intriguing example of this is found in Hudson and Smith (1996). In this article, distorted video images or audio are used to denote presence but not provide the details of activity. In a video link, one can tell that someone is present in the room but not what he or she is doing. In an audio space, one can hear that two people are conversing but not what they are saying. There are, of course, similar mechanisms in natural social activity. For example, muffled voices heard through office walls imply presence. However, similar distortions of visual or aural information would be impossible in normal social activity. The potential privacy disruptions (a form of explicitness) have been ameliorated by a new computational mechanism.

The final first-order approximation is the creation of technical architectures that do not invoke the social–technical gap; these architectures neither require action nor delegate it. Instead, these architectures provide supportive or augmentative facilities, such as advice, to users. If users could judge the quality of the support, the systems would serve as merely another resource in their environment (Hutchins, 1995a), rather than trying to mechanize elements of their social environment. Such architectures include collaborative filtering that provide ratings for services (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994; Shardanand & Maes, 1995), recommender systems that facilitate sharing of information profiles (Starr, Ackerman, & Pazzani, 1996; Terveen, Hill, Amento, McDonald, & Creter, 1997), and critic systems that make suggestions to users about design choices (Fischer, Lemke, Mastaglio, & Morch, 1990).

As an example of an approximation that attempts to address the social–technical gap in an augmentative manner, I return to the P3P example used earlier in the article. One work-around to the social–technical gap with P3P is to avoid the gap itself and merely augment the natural social facilities of the user. In the case of P3P, the approximation is to provide privacy critics, small agents that make suggestions to users about impending privacy problems (Ackerman & Cranor, 1999). These critics do not take action on behalf of the user; instead, they might offer warnings to the user. Furthermore, this architecture has the capability of having hundreds of different critics. There would not need to be one accurate user agent; many critics would work with the user to provide assurances of privacy. Users could, of course, turn these critics off or on.

These critics watch over the shoulder of the user. One such privacy critic could check a third-party database for consumer complaints about a Web site. For example, a Better Business Bureau database might report sites that have privacy or consumer reports against them. Privacy advocacy groups might have databases that report sites participating in known data scams or even nonstated transfers of personal data. Another privacy critic could watch for sites requesting combinations of personal data that could lead to the user being uniquely identifiable.

In more theoretical terms, we are actively exploring critic-based architectures because each critic is a separate resource for the user. If each resource is relatively small in functionality, users can pick and choose the resources they wish to create new ways of engaging in social activity (such as work or social interaction).

In summary, these architectures and the other approximations mentioned explore the dimensions of the social–technical gap in more detail. To create a science, however, it still remains to organize these explorations and demark the fundamental questions and issues.

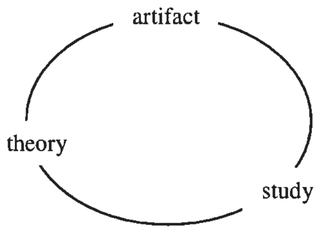
4.4. Toward Making CSCW Into a Science of the Artificial

The most formidable task for CSCW is determining systematic methods for designing around the gap. We do not wish to explore blindly. Yet, at first a fledgling science, such as CSCW, may have only illustrative cases and heuristics for design. It is easy to forget how existing engineering sciences laboriously constructed their repertories of knowledge. Petroski (1994) discussed the Romans' problem of moving enormous blocks of stone over poor roads. One way to do this is to make the columns and slabs into axles for oxen to pull. Trial and error (and suitable reflection) was required to determine how to steer these giant axles. Petroski (1994), citing Vitruvius, described one effort with a single pull line wrapped around the center of the axle. It is obvious to us that this design will continuously wander off the road; yet, this had to be learned by the Romans. Similarly, no less a genius than Galileo determined that the strength of materials was not geometrically proportional to size (Petroski, 1985). That is, if you want to build a ship twice as long as previous ships, you cannot merely use double-sized beams. Again, scientific knowledge in an engineering discipline is slow in coming; yet, failures and successes contribute to a greater understanding over time only if systematically examined.

Nonetheless, determining guiding research principles is difficult when a potential science is still seeking approximations to its problem. This section can make only a general attempt at finding general questions, and it is necessarily preliminary. Nonetheless, several guiding questions are required based on the social–technical gap and its role in any CSCW science of the artificial:

- When can a computational system successfully ignore the need for nuance and context?
- When can a computational system augment human activity with computer technologies suitably to make up for the loss in nuance and context, as argued in the approximation section earlier?
- Can these benefits be systematized so that we know when we are adding benefit rather than creating loss?
- What types of future research will solve some of the gaps between technical capabilities and what people expect in their full range of social and collaborative activities?

Figure 1. **Figure of study-design and construction-theory circle. Adapted from "Research on Computer Supported Cooperative Work" by G. M. Olson and J. S. Olson, in M. Helander (Ed.), *Handbook of Human Computer Interaction,* 1997, p. 1434, with permission from Elsevier.**



These guiding questions must also address evolving technical capabilities and infrastructures, in addition to the standard learning circle (established within HCI; see Figure 1).

CSCW (and perhaps HCI as well) actually has a technical spiral over time. No research group can thoroughly explore all design possibilities (and study them systematically). Moreover, one wishes to redo systems designs as the studies progress, based on the analysis from previous designs and on new technological possibilities. A 5-year program to thoroughly study the design space of the original Answer Garden system (Ackerman, 1994), built with its own underlying hypermedia system, would have been foolhardy, because within that timeframe the Web emerged.

5. CONCLUSION

The title of this article suggests that the social–technical gap is *the* fundamental problem of CSCW. CSCW, like HCI, clearly has multiple intellectual problems. CSCW shares problems of generalizability from small groups to a general population (as do all of the social sciences), prediction of affordances (as does HCI), and the applicability of new technological possibilities (as does the rest of computer science). Nonetheless, it has been argued here that the unique problem of CSCW is the social–technical gap. There is a fundamental mismatch between what is required socially and what we can do technically. Human activity is highly nuanced and contextualized. However, we lack the technical mechanisms to fully support the social world uncovered by the social findings of CSCW. This social–technical gap is unlikely to go away, although it certainly can be better understood and perhaps approached.

The gap is also CSCW's unique contribution. CSCW exists intellectually at the boundary and interaction of technology and social settings. Its unique intellectual importance is at the confluence of technology and the social, and its unique potential lies in its recognition of and attention to both. CSCW has an opportunity to become a true science of the artificial, updating and revising Simon's (1969/1981) classic call so as to be appropriate for CSCW's time and task.

Indeed, an understanding of the social–technical gap lies at the heart of CSCW's intellectual contribution. If CSCW (or HCI) merely contributes "cool toys" to the world, it will have failed its intellectual mission. Our understanding of the gap is driven by technological exploration through artifact creation and deployment, but HCI and CSCW systems need to have at their core a fundamental understanding of how people really work and live in groups, organizations, communities, and other forms of collective life. Otherwise, we will produce unusable systems, badly mechanizing and distorting collaboration and other social activity.

NOTES

Acknowledgments. Conversations with the Human Computer Interaction Consortium workshop, the IRIS Conference, and Platform for Privacy Preferences Project participants were invaluable in developing this argument. Additional conversations with John King and Julian Feldman were helpful in understanding Simon. Comments from the associate editor and reviewers were also extremely valuable.

Support. This work has been funded, in part, by grants from the National Science Foundation (IRI–9702904) and the U.S. Navy (N66001–9–7–M–0157).

Author's Present Address. Mark Ackerman, Information and Computer Science, Computer Science 444, University of California, Irvine, Irvine, CA 92697. E-mail: ackerman@ics.uci.edu.

HCI Editorial Record. First manuscript received December 14, 1999. Revision received May 11, 2000. Accepted by Wendy Kellogg, Peter Polson, and Clayton Lewis. Final manuscript received May 2000. – *Editor*

REFERENCES

Ackerman, M. S. (1994). Augmenting the organizational memory: A field study of answer garden. *Proceedings of the CSCW'94 Conference on Computer Supported Cooperative Work,* 243–252. New York: ACM.

Ackerman, M. S., & Cranor, L. (1999). Privacy critics: UI components to safeguard users' privacy. *Proceedings of the CHI'99 Conference on Human Factors in Computing Systems,* 258–259. New York: ACM

Ackerman, M. S., & Halverson, C. (2000). Re-examining organizational memory. *Communications of the ACM, 43*(1), 58–63.

Ackerman, M. S., Hindus, D., Mainwaring, S. D., & Starr, B. (1997). Hanging on the wire: A field study of an audio-only media space. *Transactions on Computer–Human Interaction, 4*(1), 39–66.

Ackerman, M. S., & Malone, T. W. (1990). Answer Garden: A tool for growing organizational memory. *Proceedings of the ACM Conference on Office Information Systems,* 31–39. New York: ACM.

Ackerman, M. S., & McDonald, D. W. (1996). Answer Garden 2: Merging organizational memory with collective help. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 97–105. New York: ACM.

Ackerman, M. S., & Palen, L. (1996). The zephyr help instance: Promoting ongoing activity in a CSCW system. *Proceedings of the CHI'96 Conference on Human Factors in Computing Systems,* 268–275. New York: ACM.

Bentley, R., Rodden, T., Sawyer, P., Sommerville, I., Hughes, J., Randall, D., & Shapiro, D. (1992). Ethnographically-informed systems design for air traffic control. *Proceedings of the CSCW'92 Conference on Computer Supported Cooperative Work,* 123–129. New York: ACM.

Berger, P. L., & Luckmann, T. (1966). *The social construction of reality: A treatise in the sociology of knowledge.* New York: Anchor.

Boland, R. J., Jr., Tenkasi, R. V., & Te'eni, D. (1994). Designing information technology to support distributed cognition. *Organization Science, 5,* 456–475.

Clark, A. (1997). *Being there: Putting brain, body, and world together again.* Cambridge, MA: MIT Press.

Conklin, J., & Begeman, M. L. (1988). gIBIS: A hypertext tool for exploratory policy discussion. *Proceedings of the CSCW'88,* 140–152. New York: ACM.

Cranor, L., & Reagle, J. (1999). The platform for privacy preferences. *Communications of the ACM, 42*(2), 48–55.

Dourish, P., Holmes, J., MacLean, A., Marqvardsen, P., & Zbyslaw, A. (1996). Freeflow: Mediating between representation and action in workflow systems. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 190–198. New York: ACM.

Erickson, T., Smith, D. N., Kellogg, W. A., Laff, M., Richards, J. T., & Bradner, E. (1999). Socially translucent systems: Social proxies, persistent conversation, and the design of "babble." *Proceedings of the CHI'99 Conference on Human Factors in Computing Systems,* 72–79. New York: ACM.

Fischer, G., Lemke, A. C., Mastaglio, T., & Morch, A. I. (1990). Using critics to empower users. *Proceedings of the CHI'90 Conference on Human Factors in Computing Systems,* 337–347. New York: ACM.

Flores, F., Graves, M., Hartfield, B., & Winograd, T. (1988). Computer systems and the design of organizational interaction. *ACM Transactions on Office Information Systems, 6*(2), 153–172.

Garfinkel, H. (1967). *Studies in ethnomethodology.* Englewood Cliffs, NJ: Prentice Hall.

Goffman, E. (1961). *The presentation of self in everyday life.* New York: Anchor-Doubleday.

Goffman, E. (1971). *Relations in public.* New York: Basic Books.

Greif, I., & Sarin, S. (1987). Data sharing in group work. *ACM Transactions on Office Information Systems, 5*(2), 187–211.

Grudin, J. (1989). Why groupware applications fail: Problems in design and evaluation. *Office: Technology and People, 4,* 245–264.

Heath, C., Jirotka, M., Luff, P., & Hindmarsh, J. (1994). Unpacking collaboration: The interactional organisation of trading in a city dealing room. *Computer Supported Cooperative Work Journal, 3*(2), 147–165.

Heath, C., & Luff, P. (1992). Collaboration and control: Crisis management and multimedia technology in London underground line control rooms. *Computer Supported Cooperative Work Journal, 1*(1), 69–94.

Heath, C., & Luff, P. (1996). Documents and professional practice: "Bad" organizational reasons for "good" clinical records. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 354–363. New York: ACM.

Heilbroner, R. L. (1994). Technological determinism revisited. In L. Marx & M. R. Smith (Eds.), *Does technology drive history?: The dilemma of technological determinism* (pp. 67–78). Cambridge, MA: MIT Press.

Heritage, J. (1984). *Garfinkel and ethnomethodology.* Cambridge, England: Polity.

Hollan, J., & Stornetta, S. (1992). Beyond being there. *Proceedings of the CHI'92 Conference on Human Factors in Computing Systems,* 119–125. New York: ACM.

Hudson, S. E., & Smith, I. (1996). Techniques for addressing fundamental privacy and disruption tradeoffs in awareness support systems. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 248–257. New York: ACM.

Hughes, J., King, V., Rodden, T., & Andersen, H. (1994). Moving out from the control room: Ethnography in system design. *Proceedings of the CSCW'94 Conference on Computer Supported Cooperative Work,* 429–439. New York: ACM.

Hutchins, E. (1995a). *Cognition in the wild.* Cambridge, MA: MIT Press.

Hutchins, E. (1995b). How a cockpit remembers its speeds. *Cognitive Science, 19,* 265–288.

Kaplan, S. M., Tolone, W. J., Bogia, D. P., & Bignoli, C. (1992). Flexible, active support for collaborative work with ConversationBuilder. *Proceedings of the CSCW'92 Conference on Computer Supported Cooperative Work,* 378–385. New York: ACM.

Katzenberg, B., Pickard, F., & McDermott, J. (1996). Computer support for clinical practice: Embedding and evolving protocols of care. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 364–369. New York: ACM.

Kling, R. (1991). Cooperation, coordination and control in computer-supported work. *Communications of the ACM, 34*(12), 83–88.

Kraut, R. E., Miller, M. D., & Siegel, J. (1996). Collaboration in performance of physical tasks: Effects on outcomes and communication. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 57–66. New York: ACM.

Malone, T. W., & Crowston, K. (1994). The interdisciplinary study of coordination. *ACM Computing Surveys, 26*(1), 87–119.

March, J. G., & Simon, H. A. (1958). *Organizations.* New York: Wiley.

Markus, M. L. (1990). Toward a "critical mass" theory of interactive media. In J. Fulk & C. Steinfield (Eds.), *Organizations and communication technology* (pp. 194–218). Newbury Park, CA: Sage.

Marwood, B., & Greenberg, S. (1994). Real time groupware as a distributed system: Concurrency control and its effect on the interface. *Proceedings of the CSCW'94 Conference on Computer Supported Cooperative Work,* 207–217. New York: ACM.

McCreary, L. (1993). Postman's progress. *CIO, 7*(3), 74–84.

O'Day, V. L., Bobrow, D. G., & Shirley, M. (1996). The socio-technical design circle. *Proceedings of the CSCW'96 Conference on Computer-Supported Cooperative Work,* 160–169. New York: ACM.

Olson, G. M., & Olson, J. S. (1997). Research in computer supported cooperative work. In M. Helander (Ed.), *Handbook of human computer interaction* (pp. 1433–1457). Amsterdam: Elsevier.

Olson, J. S., & Teasley, S. (1996). Groupware in the wild: Lessons learned from a year of virtual collocation. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 419–427. New York: ACM.

Orlikowski, W. J. (1992a). The duality of technology: Rethinking the concept of technology in organizations. *Organization Science, 3,* 398–427.

Orlikowski, W. J. (1992b). Learning from notes: Organizational issues in groupware implementation. *Proceedings of the CSCW'92 Computer Supported Cooperative Work,* 362–369. New York: ACM.

Osterweil, L. J. (1987). Software processes are software too. *Proceedings of the ICSE'87 International Conference on Software Engineering,* 2–13. New York: ACM.

Petroski, H. (1985). *To engineer is human: The role of failure in successful design.* New York: St. Martin's.

Petroski, H. (1994). *Design paradigms: Case histories of error and judgment in engineering.* New York: Cambridge University Press.

Postman, N. (1992). *Technopoly.* New York: Vintage.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. *Proceedings of the ACM Conference on Computer Supported Cooperative Work,* 175–186. New York: ACM.

Rodden, T. (1996). Populating the application: A model of awareness for cooperative applications. *Proceedings of the CSCW'96 Conference on Computer Supported Cooperative Work,* 87–96. New York: ACM.

Root, R. W. (1988). Design of a multi-media vehicle for social browsing. *Proceedings of the CSCW'88 Conference on Computer Supported Cooperative Work,* 25–38. New York: ACM.

Schmidt, K., & Simone, C. (1996). Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. *Compuer Supported Cooperative Work Journal, 5*(2/3), 155–200.

Shardanand, U., & Maes, P. (1995). Social information filtering. *Proceedings of the CHI'95 Conference on Human Factors in Computing Systems,* 210–217. New York: ACM.

Simon, H. A. (1957). *Administrative behavior.* New York: Macmillan.

Simon, H. A. (1981). *The sciences of the artificial.* Cambridge, MA: MIT Press. (Original work published 1969)

Simon, H. A. (1991). *Models of my life.* New York: Basic Books.

Sproull, L., & Kiesler, S. (1991). *Connections: New ways of working in the networked organization.* Cambridge, MA: MIT Press.

Star, S. L. (1989). The structure of ill-structured solutions: Boundary objects and heterogeneous distributed problem solving. In L. Gasser & M. Huhns (Eds.), *Distributed artificial intelligence* (pp. 37–54). San Mateo, CA: Kaufmann.

Star, S. L., & Ruhleder, K. (1994). Steps toward an ecology of infrastructure: Complex problems in design and access for large-scale collaborative systems. *Proceed-*

ings of the CSCW'94 Conference on Computer Supported Cooperative Work, 253–264. New York: ACM.

Starr, B., Ackerman, M. S., & Pazzani, M. (1996). Do-I-care: A collaborative web agent. *Proceedings of the CHI'96 Conference on Human Factors in Computing Systems: Short papers,* 268–275. New York: ACM.

Strauss, A. (1991). *Creating sociological awareness: Collective images and symbolic representations*. New Brunswick, NJ: Transaction.

Strauss, A. L. (1993). *Continual permutations of action*. New York: Aldine de Gruyter.

Suchman, L. A. (1987). *Plans and situated actions: The problem of human–computer communication*. New York: Cambridge University Press.

Suchman, L. (1994). Do categories have politics? *Computer Supported Cooperative Work Journal, 2,* 177–190.

Suchman, L., & Wynn, E. (1984). Procedures and problems in the office. *Office: Technology and People, 2,* 133–154.

Terveen, L., Hill, W., Amento, B., McDonald, D., & Creter, J. (1997). PHOAKS: A system for sharing recommendations. *Communications of the ACM, 40*(3), 59–62.