

Cooperative design: Prospects for CSCW in design

Kjeld Schmidt

Center for Tele-Information
Technical University of Denmark
DK-2800 Lyngby, Denmark
Email: schmidt@cti.dtu.dk

Abstract. In the contemporary world, engineers and designers face huge challenges as they shift towards novel organizational concepts such as 'concurrent engineering' in order to manage increasing product diversity so as to satisfy customer demands while trying to accelerate the design process to deal with the competitive realities of a global market and decreasing product life cycles. In this environment, the coordination and integration of the myriads of interdependent and yet distributed and concurrent design activities becomes enormously complex. It thus seems as if CSCW technologies may be indispensable if concurrent engineering is to succeed. On the basis of ethnographic studies of cooperative design, the paper attempts to characterize cooperative work in the domain of design and to outline a set of crucial research problems to be addressed if CSCW is to help engineers and designers meet the challenges they are facing. On one hand, designers need highly flexible 'coordination mechanisms' that can support horizontal coordination of large-scale distributed design projects, and on the other hand design organizations require versatile and ubiquitous infrastructures to be able to manage their 'common information spaces' without sacrificing the critical adaptivity and concurrency.

In order to be able to conceptualize and specify the support requirements of cooperative design, it is useful to make an analytical distinction between 'cooperative work' and 'articulation work.' According to this conception, cooperative work is constituted by the interdependence of multiple actors who, in their individual activities, in changing the state of their individual field of work, also change the state of the field of work of others and who thus interact through changing the state of a common field of work. However, since it involves multiple actors, cooperative work is inherently distributed, not only in the usual sense that activities are distributed in time and space, but also — and more importantly — in the sense that actors are semi-autonomous in terms of the different circumstances they are faced with in their work as well as in terms of their strategies, heuristics, perspectives, goals, motives, etc. (Schmidt, 1991a; Schmidt, 1991b). To deal with this source of confusion and disorder, individual and yet interdependent activities must be coordinated, scheduled, aligned, meshed, integrated, etc. — in short: articulated. That is, the orderly accomplishment of cooperative work requires what has aptly been termed articulation work (Strauss et al., 1985; Gerson and Star, 1986; Strauss, 1988; Strauss, 1994). The distinction between cooperative

work and articulation work is recursive; that is, an established arrangement of articulating a cooperative effort may itself be subjected to a cooperative effort of re-arrangement which in turn also may need to be articulated, and so forth.

While cooperative work, as noted, is inherently distributed, the distributed character of cooperative work varies, according to the complexity of the interdependence, that is, depending on factors such the distribution of activities in time and space, the number of participants in the cooperative ensemble, the structural complexity posed by the field of work (interactions, heterogeneity), the degree and scope of specialization among participants, the apperceptive uncertainties posed by the field of work and hence the variety of heuristics involved, and so on. The more distributed the activities of a given cooperative work arrangement, the more complex the articulation of the activities of that arrangement is likely to be.

If ‘more and more design problems are reaching insoluble levels of complexity’, as argued by Christopher Alexander in his classic study (1964, p. 3), design work *conceived of as a cooperative effort* is of an entirely different order of complexity.

In work domains such as process control, the common field of work, i.e., the plant and the ongoing processes, is essentially given in advance and the task is basically to maintain or achieve a certain system state as defined in terms of a set of parameters. Thus, the interdependencies are largely known in advance and the challenge is to coordinate the distributed time-critical processes. Likewise in manufacturing, but here the challenge is to handle interdependencies which for all practical purposes are intractable due to the myriads of parts which have to undergo a variety of processes in different sequences.

By contrast, the common field of work of cooperative design is essentially emerging. Design work is, of course, situated in an environment which is given in advance. In the case of design in manufacturing and similar settings, design work is highly constrained by the manufacturing capabilities of the enterprise (machinery, skills, etc.) and the previous designs and inventories of parts as well as general standards. Likewise, software design is constrained by the interface standards of the target platform and other software systems as well as by existing code libraries. However, the focal object of the work of designers is only taking shape in and through their very work. In fact, the gist of design work can be said to consist in exploring and identifying the interactions between conflicting requirements so as to be able to decide on an acceptable compromise: ‘What does make design a problem in real world cases is that we are trying to make a diagram for forces whose field we do not understand’ (Alexander, 1964, p. 21). That is, design is a ‘wicked problem’, to use the term suggested by Horst Rittel:

‘in order to describe a wicked problem in sufficient detail, one has to develop an exhaustive inventory of all conceivable solutions ahead of time. The reason is that every question asking for additional information depends upon the understanding of the problem — and its resolution — at that time. Problem understanding and problem resolution are concomitant to each other

[...] the process of solving the problem is identical with the process of understanding its nature' (Rittel and Webber, 1973).

These complexities are compounded drastically when solving a 'wicked problem' involves multiple actors, in that different aspects of the problem are addressed by different designers and the interdependencies among these aspects, and hence between the actors, emerge and change as the design project unfolds. If design in general is a 'wicked problem', and if the mischievous nature of the work is compounded when design is done cooperatively, cooperative engineering of software systems is a hostile problem. As pointed out by Frederick Brooks, software is invisible and unvisualizable: 'In spite of progress in restricting and simplifying the structures of software, they remain inherently unvisualizable, thus depriving the mind of some of its most powerful conceptual tools. This lack not only impedes the process of design within one mind, it severely hinders communication among minds.' (Brooks, 1987, pp. 185 f.).

The traditional strategy for managing these complexities consisted in subjecting the cooperative design process to a rigorously hierarchical regime in which 'the problem' was defined and refined in a top-down manner and decomposed into atomic fragments which were then addressed sequentially (Harrington, 1979). Multiple levels of management were then required to handle the unavoidable unforeseen interdependencies among the fragmented design activities. This strategy has now collapsed.

Due to the growth of scientific research activities and the tighter coupling between research and production, the pace of technological change is accelerating in all branches of production. This is further enhanced by the shortening of the incubation period of product design due to the introduction of computer-aided design and engineering technologies. Because of this, product life cycles are being reduced dramatically. For example, in the production of automobiles and trucks major redesigns of engines, cars, axles, and brakes used to occur every six or seven years. Today, however, product life cycles have been cut in half, that is, to about three or four years. In an industry such as the production of engineering workstations, which used to have a product life cycle of three years, it now less than six months. In response to these dynamic conditions of competition, manufacturing enterprises are reducing batch sizes as well as the overall lead time of their operations so as to be able to penetrate markets and recoup investments while the product is still competitive. Furthermore, driven by the development of the means of transportation and communication and the creation of an increasingly integrated world market, global competition is becoming increasingly fierce. In response this development, companies are paying more attention to customers' needs and propensities to the extent that they treat their markets as 'fashion markets.' In order to indulge customers, the shortest possible elapse time from order to delivery is becoming a competitive advantage in its own right, and product life cycles are being cut short as yesterday's models are superseded by tomorrow's. This reinforces the demand on the flexibility and versatility of manufacturing enterprises. Also in order to humor customers, companies are

expanding the list of models and variants in their sales catalogues. As a result, manufacturing has become beset by rampant product diversification. A manufacturing enterprises which in the 60's would produce, say, a few hundred of models and variants of models, will now typically be offering tens or hundreds of thousands of models and variants (Piore and Sabel, 1984; Ohmae, 1985; Gunn, 1987; Aoki, 1988; Best, 1990; Womack et al., 1990).

In such environments, engineers and designers face huge challenges as they fight to manage the increasing product diversity while trying to accelerate the design process to deal with the harsh competitive realities of a global market and decreasing product life cycles. In response to these challenges the work organization of design is shifting towards novel organizational concepts such as 'concurrent engineering'. However, in doing so, the coordination and integration of the myriads of interdependent and yet distributed and concurrent design activities becomes enormously complex.

It thus seems as if CSCW technologies may be indispensable to the design domain if the current transition towards highly responsive work organizations in design such as concurrent engineering is to succeed. Conversely, the domain of design and engineering has indeed attracted much attention from CSCW researchers. Very early in the history of CSCW research related to the problem of design two strategies became dominant: On one hand, CSCW researchers explored different approaches to capturing 'design rationale' and supporting 'organizational memory' and in the course of this line of research a number of experimental systems such as gIBIS (Conklin and Begeman, 1988; Conklin, 1989), Answer Garden (Ackerman and Malone, 1990), and EGRET (Johnson, 1992) were developed and put to test. On the other hand, researchers, especially a team of Japanese researchers around Hiroshi Ishii, addressed the issue of supporting the cooperation among designers over distance in space by means of a series of very refined shared display facilities (e.g., Ishii, 1990; Ishii and Miyake, 1991; Ishii et al., 1992; Ishii et al., 1993).

Since these pioneering activities were undertaken, a large number of in-depth empirical studies have been carried out within the CSCW community (e.g., Anderson et al., 1993; Carstensen et al., 1995; Button and Sharrock, 1996; Carstensen, 1996; Carstensen and Sørensen, 1996; Grinter, 1996; Potts and Catledge, 1996; Robertson, 1996; Grinter, 1997; Robertson, 1997; Tellioglu and Wagner, 1997). In addition to these studies, we are fortunate to have a range of anthropological studies of design which do not primarily address issues of computer-support for cooperative design but contribute significantly to our understanding of design as a cooperative effort (Bucciarelli, 1984; Bucciarelli, 1988b; Bucciarelli, 1988a; Latour, 1993; Bucciarelli, 1994) as well as a large number of studies of design from organizational perspectives (e.g., Adler, 1990) and from the perspectives of software engineering (Curtis et al., 1988) and

software development methodology (Greenbaum and Kyng, 1991; Bødker and Grønbaek, 1996).¹

Thanks to these and other studies of cooperative design we should now be in a better position to understand how CSCW technologies can assist designers and engineers in managing the challenges they are faced with. What comes across in these studies, in all of their methodological variety and the multiplicity of design settings which have been studied, is that one overriding problem is facing contemporary designers, namely the *overwhelming complexity* of handling the myriads of capricious and shifting interdependencies between distributed activities in an orderly and timely fashion.

A simple example will illustrate the point: Foss Electric is a Danish manufacturing company that produces advanced equipment for analytical measurement of quality parameters of agricultural products, e.g., the compositional quality of milk in terms of fat content and the count of protein, lactose, somatic cells, bacteria, etc. At the time of the field study,² the company was engaged in a large design project called S4000 which aimed at building a new instrument for analytical testing of raw milk. The S4000 project was the first project aiming at building an integrated instrument that would offer a range of functionalities that previously had been offered by a number of specialized instruments. In addition, as an innovation compared to previous models, the S4000 system would introduce measurements of new parameters in milk (e.g., urea and citric acid), and the performance was to be radically increased. The instrument would consist of approximately 8,000 components grouped into a number of functional units, such as cabinet, pipette unit, conveyer, flow-system, and measurement unit. Finally, the S4000 was the first Foss instrument to incorporate a personal computer (an Intel-based 486 PC) by means of which the configuration and operation of the instrument were to be controlled (through a Windows interface). Eventually, the first version of the software consisted of approximately 200,000 lines of source code. Altogether more than 50 people were involved in the S4000 project, which lasted approximately 30 months (for the first version).

The design team was faced with quite a challenge: (1) The different subsystems, e.g., the software control system and the mechanical and chemical processes in the flow and measurement system, were intricately interdependent and might interact in unforeseen ways. (2) The S4000 project introduced measurement of new parameters in raw milk for which new technologies had to be developed and mastered. (3) The different subsystems were developed concurrently and the requirements to be satisfied by each subsystem would

¹ In this context one should not overlook journalistic reports which do not profess scientific ambitions but nevertheless provide vivid and often sociologically sophisticated descriptions of cooperative design projects (e.g., Kidder, 1982; Sabbagh, 1989). Nor should one overlook classic studies of large-scale design projects (e.g., Devons, 1950; Devons, 1970; Sapolsky, 1972).

² The field research of the S4000 project was done by Henrik Borstrøm, Peter Carstensen, and Carsten Sørensen.

therefore change as other subsystems were developed. (4) Production facilities at the manufacturing plant were constantly changing as the use of existing machines was optimized and new machines and processes were introduced. Hence, the repertoire of manufacturing processes that the production function could offer to designers and that designers thus had to take into account in their decisions was continually changing. (5) Because of its technological heterogeneity, the S4000 project involved a number of specialisms. The core design team consisted of designers from mechanical engineering, electronics, software, and chemistry. In addition, a handful of draught-persons and several persons from organizational entities such as production, model shop, marketing, quality assurance, quality control, service, and top management were involved to varying degrees at different stages in the course of the project. All in all, the project was significantly more complex than previous projects at Foss.

To survive these challenges, the participants took a number of measures to reduce the complexity of managing the project:

As always at Foss, all project participants from the different technical departments were moved to the same office area to create a shared physical space by means of which participants could develop and maintain shared awareness of the state of the project. Furthermore, of course, a sequence of meetings was scheduled at different intervals and, as the project took its course, a large number of ad-hoc meetings were arranged as well.

However, the amount of detailed information that had to be communicated, aligned, negotiated, etc., required more robust measures. A number of procedures and artifacts were introduced to keep track of the state of affairs and to manage relations and dependencies among actors, tasks, and resources: an 'augmented' bill of materials that identified actors responsible for parts in order to support the coordination of mechanical design, process planning, and production in the construction of prototypes (Sørensen, 1994a); a CEDAC board (Cause and Effect Diagram with the Addition of Cards) for coordinating the diagnosis of faults between mechanical design and process planning (Sørensen, 1994b); and a product classification scheme supporting the distributed classification and retrieval of CAD models (Sørensen, 1994c). Some of these procedures and artifacts were invented for this project, some were redesigns of existing artifacts, and others were merely adopted.

The most dramatic measures were taken with respect to the software design process. In the early phases of the software strand of the S4000 project, the software designers felt that their overview of the state of the project was quite defective and that they needed much greater coordination. At the height of the crisis the software design goals were almost abandoned. To overcome the crisis, the software designers developed a repertoire of procedures and artifacts to ensure the monitoring and control of the integration of software components and modules.

An important component in this repertoire, was the 'software platform' institution. Initially, the 'software platform' was just a point in time at which all

software designers would stop coding in order to integrate their bits and pieces. For each platform integration period, one of the designers was appointed as ‘platform master’ which implied that he or she would be responsible for collecting information on changes made to the software and for ensuring that the software was tested and corrected before it was released. Before the software was released as a ‘platform’ for further development, the project schedule was updated with revised plans and tasks for the next three to six weeks. The establishment of the software platform institution was considered absolutely necessary for the S4000 project.

Moreover, the software design team devised and introduced other procedures and artifacts. Most importantly, a ‘bug report form’ with corresponding procedures for reporting, classifying, and correcting faults were introduced to ensure that bugs were properly registered, that corrected bugs were duly reported, and to make the allocation of responsibilities clear and visible to all members. As a complementary measure, copies of bug forms were collected in a publicly available repository in the form of a simple binder. (For further details, cf. Carstensen, 1996; Carstensen and Sørensen, 1996).

The software designers experienced the hard way that it was practically impossible to handle the distributed testing and bug registration activities of some twenty testers and designers without, *inter alia*, a bug report form and its associated procedures. By devising and introducing these constructs, they managed to alleviate the coordination crisis in the project.

The case of the S4000 project is particularly valuable because we here witness the conception and introduction of specialized artifacts for coordination purposes in response to overwhelming problems encountered in coping with the complexities of articulating the distributed and interdependent activities of cooperative design under conditions that are typical for contemporary design.

To assist cooperating designers and engineers in coping with these complexities, CSCW technologies are needed on two fronts:

Managing task interdependencies: The obvious and fundamental way to coordinate, align, mesh, etc. myriad interdependent and yet distributed activities is to facilitate *mutual awareness* among actors, for instance, by having actors working in the same room or by providing some multi-media emulation of a ‘shared space’ (e.g., Ishii, 1990; Ishii and Miyake, 1991; Ishii et al., 1992; Ishii et al., 1993). However, task interdependencies are often of an order of complexity where the provision of facilities for mutual awareness and ad hoc interactions is insufficient (e.g., Carstensen et al., 1995; Carstensen, 1996; Carstensen and Sørensen, 1996; Grinter, 1996; Grinter, 1997; Tellioglu and Wagner, 1997). Other means are required which make task interdependencies tractable. We have called such means *coordination mechanisms* (Schmidt and Simone, 1996). A coordination mechanism is, simply put, a coordinative protocol with an accompanying artifact, such as, for instance, a standard operating procedure supported by a certain form. It offers a ‘precomputation’ (Norman, 1991) of task interdependencies and is thereby instrumental in reducing the space of

possibilities facing a competent actor in a given situation. However, the distributed nature of cooperative work is not eradicated by coordination mechanisms. They are merely local and temporary closures which assist actors in managing an otherwise overwhelming complexity. They are used in a distributed way and evolve through a process of local adaptations. The challenge is thus to provide a CSCW environment which supports the construction and use of computational coordination mechanisms which are robust in view of the distributed nature of cooperative work.

Managing common information spaces: Cooperating designers and engineers need to keep track of the field of work — the joint design — as it evolves as a result of their distributed activities, in order to be able to see the emerging context of their own contribution, inspect impact of the contributions of others, assess likely interactions, etc. The crucial problem here is that of indexation, that is, the provision of means that allow an individual to assign a publicly visible and permanent ‘pointer’ to each item so as to enable other individuals to locate the items relatively easily and reliably. To cope with this problem, actors use ‘classification schemes’, i.e., a conceptual structure (an ordered arrangement of categories) imprinted upon an artifact (for instance, a catalogue or a graph). In the case of the S4000 project the software designers simply used the directory system of the development environment to manage the database of software modules. The conditions of much larger design projects (e.g., automobiles, power plants, airplanes) require far more sophisticated facilities for keeping track of the vast system of information objects in which the evolving design is expressed. The same applies to the problem of managing the vast heterogeneous and distributed repositories of the wider design environment (previous designs, stocks of components, code libraries). While sophisticated indexation facilities exist and have been in widespread use for decades (e.g., group technology), the problem that is coming to the fore in large scale projects and design environments — and this problem is typically ignored in work on ‘organizational memory — is that the classification schemes required for distributed actors to be able to handle these repositories *evolve over time*, in response to changing conditions and changing conceptualizations and that this evolutionary process is itself distributed. Thus, in large-scale settings actors cannot always negotiate each proposed change to the classification scheme in order to ensure consensus, and even if they are able to negotiate proposed changes the agreed-to categories will typically have local implications and interpretations which are not necessarily known globally within the given community. That is, the classification scheme evolves in a process which is only partially concerted and which involves local innovations which may only gain general acceptance over time (if ever).

In sum, then, in coordinating their cooperative effort designers and engineers are faced with overwhelming complexities. Therefore, for example, the primary role for CSCW in design is not to help managers of design organizations to ‘capture’ the tacit knowledge of their employees so as to avert negative consequences of downsizing, that is, to sack workers while retaining their

knowledge (Conklin and Begeman, 1988; Ackerman and Malone, 1990). Nor is it to provide facilities to capture the ‘design rationale’ for later retrieval and use (Conklin, 1993). While the latter facilities might be beneficial with respect to another set of problems (maintenance, for instance), there are some fundamental conceptual issues that need to be addressed for the idea to be viable in real world design work. For instance, the very notion of ‘design rationale’ is elusive in that a vast and often indeterminate array of motives and reasons are at play on top of the more mundane technical issues (cf., e.g., Anderson et al., 1993), and is not clear, to say the least, how such motives and reasons might be captured by a computer system. Furthermore, in agreeing on a particular design solution, designers do not necessarily agree upon their motives, nor do they necessarily make their individual motives explicit in their negotiations (cf., e.g., Bucciarelli, 1984; Bucciarelli, 1988a; Bucciarelli, 1988b; Bucciarelli, 1994). As long as the agreed-to design is seen as a viable compromise, designers do not need to state — not to mention, *agree to* — their rationales. Thus, to ask designers to state their motives and reasons explicitly as part of constructing the project record will engender controversies that do not improve the quality of the design, nor the efficiency of the design process, and which may even prevent the designers from reaching a workable compromise and getting the job done. That is, the system may easily become an added burden on the designers.

The substantial potential benefits from CSCW to design lie in supporting the *design process*, as opposed to supporting the retention of design rationale or designers’ knowledge, that is, the benefits lie in assisting designers and engineers in managing the overwhelming coordination problems.

References

- Ackerman, Mark S., and Thomas W. Malone: ‘Answer Garden: A tool for growing organizational memory,’ *Proceedings of the Conference on Office Information Systems, Cambridge, Mass., April 1990*, Proceedings of the Conference on Office Information Systems, Cambridge, Mass., April 1990, ACM, New York, 1990, pp. 31-39.
- Adler, Paul S.: ‘Managing high tech processes: The challenge of CAD/CAM,’ in M. A. Von Glinow and S. A. Mohrman (eds.): *Managing Complexity in High Technology Organizations*, Managing Complexity in High Technology Organizations, M. A. Von Glinow and S. A. Mohrman (eds.), vol. , Oxford University Press, New York and Oxford, 1990, pp. 188-215.
- Alexander, Christopher: *Notes on the Synthesis of Form*, Harvard University Press, Cambridge, Mass., 1964.
- Anderson, Bob, Graham Button, and Wes Sharrock: ‘Supporting the design process within an organisational context,’ in G. d. Michelis, C. Simone, and K. Schmidt (eds.): *ECSCW '93. Proceedings of the Third European Conference on Computer-Supported Cooperative Work, Milan*, Kluwer Academic Publishers, 1993, p. 47–59.
- Aoki, Masahiko: *A New Paradigm of Work Organization: The Japanese Experience*, WIDER Working Papers, vol. 36, World Institute for Development Economics Research, Helsinki, Finland, 1988.

- Best, Michael H.: *The New Competition. Institutions of Industrial Restructuring*, Polity Press, Cambridge, 1990.
- Bødker, Susanne, and Kaj Grønbaek: 'Users and designers in mutual activity: An analysis of cooperative activities in systems design,' in Y. Engeström and D. Middleton (eds.): *Cognition and Communication at Work*, Cognition and Communication at Work, Y. Engeström and D. Middleton (eds.), vol. , Cambridge University Press, Cambridge, 1996, pp. 130-158.
- Brooks, Frederick P., Jr.: 'No silver bullet: Essence and accidents of software engineering,' (*Computer*, 1987); in F. P. Brooks Jr.: *The Mythical Man-Month: Essays on Software Engineering*, Addison-Wesley, Reading, Mass., 1995, pp. 177-203.
- Bucciarelli, Louis L.: 'Reflective practice in engineering design,' *Design Studies*, vol. 5, no. 3, July 1984, pp. 185-190.
- Bucciarelli, Louis L.: 'Engineering design process,' in F. A. Dubinskas (ed.): *Making Time. Ethnographies of High-Technology Organizations*, Making Time. Ethnographies of High-Technology Organizations, F. A. Dubinskas (ed.) vol. , Temple University Press, Philadelphia, 1988a, pp. 92-122.
- Bucciarelli, Louis L.: 'An ethnographic perspective on engineering design,' *Design Studies*, vol. 9, no. 3, July 1988b, pp. 159-168.
- Bucciarelli, Louis L.: *Designing Engineers*, MIT Press, Cambridge, Mass., and London, 1994.
- Button, Graham, and Wes Sharrock: 'Project work: The organization of collaborative design and development in software engineering,' *Computer Supported Cooperative Work. The Journal of Collaborative Computing*, vol. 5, no. 4, 1996, pp. 369-386.
- Carstensen, Peter: *Computer Supported Coordination*, Risø National Laboratory, P.O. Box 49, DK-4000 Roskilde, Denmark 1996. [Risø-R-890(EN)].
- Carstensen, Peter H., and Carsten Sørensen: 'From the social to the systematic: Mechanisms supporting coordination in design,' *Computer Supported Cooperative Work. The Journal of Collaborative Computing*, vol. 5, no. 4, 1996, pp. 387-413.
- Carstensen, Peter H., Carsten Sørensen, and Tuomo Tuikka: 'Let's talk about bugs!,' *Scandinavian Journal of Information Systems*, vol. 7, no. 1, 1995, pp. 33-54.
- Conklin, Jeff: 'Design rationale and maintainability,' in B. Shriver (ed.): *Proceedings of 22nd Annual Hawaii International Conference on System Sciences*, Proceedings of 22nd Annual Hawaii International Conference on System Sciences, B. Shriver (ed.) vol. II, IEEE Computer Society, 1989, pp. 533-539.
- Conklin, Jeff: 'Capturing organizational knowledge,' in R. M. Baecker (ed.): *Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration*, Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration, R. M. Baecker (ed.) vol. , Morgan Kaufmann Publishers, San Mateo, Calif., 1993, pp. 561-565. - Originally published in D. Coleman (ed.): *Proceedings of Groupware '92*, Morgan Kaufmann, 1992, pp. 133-137.
- Conklin, Jeff, and Michael L. Begeman: 'gIBIS: A hypertext tool for exploratory policy discussion,' *CSCW '88. Proceedings of the Conference on Computer-Supported Cooperative Work, Portland, Oregon, September 26-28, 1988*, CSCW '88. Proceedings of the Conference on Computer-Supported Cooperative Work, Portland, Oregon, September 26-28, 1988, ACM Press, New York, N.Y., 1988, pp. 140-152.
- Curtis, Bill, Herb Krasner, and Neil Iscoe: 'A field study of the software design process for large systems,' *Communications of the ACM*, vol. 31, no. 11, November 1988, pp. 1268-1287.
- Devons, Ely: *Planning in Practice: Essays in Aircraft Planning in War-Time*, Cambridge University Press, Cambridge, 1950.
- Devons, Ely: *Papers on Planning and Economic Management*, Manchester University Press, Manchester, 1970 (Edited by Sir Alec Cairncross).

- Gerson, Elihu M., and Susan Leigh Star: 'Analyzing due process in the workplace,' *ACM Transactions on Office Information Systems*, vol. 4, no. 3, July 1986, pp. 257-270.
- Greenbaum, Joan, and Morten Kyng (eds.): *Design at Work: Cooperative Design of Computer Systems*, Lawrence Erlbaum, Hillsdale, New Jersey, 1991.
- Grinter, Rebecca E.: 'Supporting articulation work using software configuration management systems,' *Computer Supported Cooperative Work. The Journal of Collaborative Computing*, vol. 5, no. 4, 1996, pp. 447-465.
- Grinter, Rebecca E.: 'Doing software development: Occasions for automation and formalisation,' in J. A. Hughes at al. (eds.): *ECSCW '97. Proceedings of the Fifth European Conference on Computer-Supported Cooperative Work, 7-11 September 1997, Lancaster, U.K.*, ECSCW '97. Proceedings of the Fifth European Conference on Computer-Supported Cooperative Work, 7-11 September 1997, Lancaster, U.K., J. A. Hughes at al. (eds.), vol. , Kluwer Academic Publishers, Dordrecht, 1997, pp. 173-188.
- Gunn, Thomas G.: *Manufacturing for Competitive Advantage. Becoming a World Class Manufacturer*, Ballinger, Cambridge, Mass., 1987.
- Harrington, Joseph: *Computer Integrated Manufacturing*, Krieger, Malabar, Florida, 1979.
- Ishii, Hiroshi: 'TeamWorkStation: Towards a seamless shared workspace,' in *CSCW '90, Proceedings of the Conference on Computer-Supported Cooperative Work, Los Angeles, Calif., 7-10 October 1990*, ACM Press, New York, 1990, pp. 13-26.
- Ishii, Hiroshi, Kazuho Arita, and Takashi Yagi: 'Beyond videophones: TeamWorkStation-2 for narrowband ISDN,' in G. De Michelis, C. Simone, and K. Schmidt (eds.): *ECSCW '93. Proceedings of the Third European Conference on Computer-Supported Cooperative Work, 13-17 September 1993, Milan, Italy*, ECSCW '93. Proceedings of the Third European Conference on Computer-Supported Cooperative Work, 13-17 September 1993, Milan, Italy, G. De Michelis, C. Simone, and K. Schmidt (eds.), vol. , Kluwer Academic Publishers, Dordrecht, 1993, pp. 325-340.
- Ishii, Hiroshi, Minoru Kobayaski, and Jonathan Grudin: 'Integration of inter-personal space and shared workspace: ClearBoard design and experiments,' in J. Turner and R. Kraut (eds.): *CSCW '92. Proceedings of the Conference on Computer-Supported Cooperative Work, Toronto, Canada, 31 October—4 November, 1992*, CSCW '92. Proceedings of the Conference on Computer-Supported Cooperative Work, Toronto, Canada, 31 October—4 November, 1992, J. Turner and R. Kraut (eds.), vol. , ACM Press, New York, 1992, pp. 33-42.
- Ishii, Hiroshi, and Naomi Miyake: 'Torward an open shared workspace: Computer and video fusion approach of TeamWorkStation,' *Communications of the ACM*, vol. 34, no. 12, 1991, pp. 36-50.
- Johnson, Philip: 'Supporting exploratory CSCW with the EGRET framework,' in J. Turner and R. Kraut (eds.): *CSCW '92. Proceedings of the Conference on Computer-Supported Cooperative Work, Toronto, Canada, 31 October—4 November, 1992*, CSCW '92. Proceedings of the Conference on Computer-Supported Cooperative Work, Toronto, Canada, 31 October—4 November, 1992, J. Turner and R. Kraut (eds.), vol. , ACM Press, New York, 1992, pp. 298-305.
- Kidder, Tracy: *The Soul of a New Machine*, Avon Books, New York, 1982. [Avon Books, New York, 1990].
- Latour, Bruno: *Aramis, ou l'amour des techniques*, Éditions la découverte, Paris, 1993.
- Norman, Donald A.: 'Cognitive artifacts,' in J. M. Carroll (ed.): *Designing Interaction. Psychology at the Human-Computer Interface*, Designing Interaction. Psychology at the Human-Computer Interface, J. M. Carroll (ed.) vol. , Cambridge University Press, Cambridge, 1991, pp. 17-38.

- Ohmae, Kenichi: *Triad Power. The Coming Shape of Global Competition*, Free Press, New York, 1985.
- Piore, Michael J., and Charles F. Sabel: *The Second Industrial Divide: Possibilities for Prosperity*, Basic Books, New York, 1984.
- Potts, Colin, and Lara Catledge: 'Collaborative conceptual design: A large software project case study,' *Computer Supported Cooperative Work. The Journal of Collaborative Computing*, vol. 5, no. 4, 1996, pp. 415-445.
- Rittel, Horst W. J., and Melvin M. Webber: 'Dilemmas in a general theory of planning,' *Policy Sciences*, vol. 4, 1973, pp. 155-169.
- Robertson, Toni: 'Embodied actions in time and place: The cooperative design of a multimedia, educational computer game,' *Computer Supported Cooperative Work. The Journal of Collaborative Computing*, vol. 5, no. 4, 1996, pp. 341-367.
- Robertson, Toni Joan: *Designing Over Distance: A Study of Cooperative Work, Embodied Cognition and Technology to Enable remote Collaboration*, Submitted for the Degree of Doctor of Philosophy, School of Computing Sciences, University of Technology, Sydney, 1997. 195 pp.
- Sabbagh, Karl: *Skyscraper: The Making of a Building*, Macmillan, London, 1989. [Penguin Books, New York, 1991].
- Sapolsky, Harvey M.: *The Polaris System Development: Bureaucratic and Programmatic Success in Government*, Harvard University Press, Cambridge, Mass., 1972.
- Schmidt, Kjeld: 'Cooperative work: A conceptual framework,' in J. Rasmussen, B. Brehmer, and J. Leplat (eds.): *Distributed Decision Making. Cognitive Models for Cooperative Work*, Distributed Decision Making. Cognitive Models for Cooperative Work, J. Rasmussen, B. Brehmer, and J. Leplat (eds.), vol. , John Wiley & Sons, Chichester, 1991a, pp. 75-109.
- Schmidt, Kjeld: 'Riding a tiger, or Computer Supported Cooperative Work,' in L. Bannon, M. Robinson, and K. Schmidt (eds.): *ECSCW '91. Proceedings of the Second European Conference on Computer-Supported Cooperative Work, Amsterdam, 24-27 September 1991*, ECSCW '91. Proceedings of the Second European Conference on Computer-Supported Cooperative Work, Amsterdam, 24-27 September 1991, L. Bannon, M. Robinson, and K. Schmidt (eds.), vol. , Kluwer Academic Publishers, Dordrecht, 1991b, pp. 1-16.
- Schmidt, Kjeld, and Carla Simone: 'Coordination mechanisms: Towards a conceptual foundation of CSCW systems design,' *Computer Supported Cooperative Work. The Journal of Collaborative Computing*, vol. 5, no. 2-3, 1996, pp. 155-200.
- Sørensen, Carsten: 'The augmented bill of materials,' in K. Schmidt (ed.): *Social Mechanisms of Interaction*, Social Mechanisms of Interaction, K. Schmidt (ed.) vol. , Computing Department, Lancaster University, Lancaster, UK, 1994a, pp. 221-236. - [COMIC Deliverable 3.2. Available via anonymous FTP from ftp.comp.lancs.ac.uk].
- Sørensen, Carsten: 'The CEDAC board,' in K. Schmidt (ed.): *Social Mechanisms of Interaction*, Social Mechanisms of Interaction, K. Schmidt (ed.) vol. , Computing Department, Lancaster University, Lancaster, UK, 1994b, pp. 237-245. - [COMIC Deliverable 3.2. Available via anonymous FTP from ftp.comp.lancs.ac.uk].
- Sørensen, Carsten: 'The product classification scheme,' in K. Schmidt (ed.): *Social Mechanisms of Interaction*, Social Mechanisms of Interaction, K. Schmidt (ed.) vol. , Computing Department, Lancaster University, Lancaster, UK, 1994c, pp. 247-255. - [COMIC Deliverable 3.2. Available via anonymous FTP from ftp.comp.lancs.ac.uk].
- Strauss, Anselm: 'The articulation of project work: An organizational process,' *The Sociological Quarterly*, vol. 29, no. 2, 1988, pp. 163-178.
- Strauss, Anselm: *Continual Permutations of Action*, Aldine de Gruyter, New York, 1994.
- Strauss, Anselm, Shizuko Y. Fagerhaugh, Barbara Sucek, and Carolyn Wiener: *Social Organization of Medical Work*, University of Chicago Press, Chicago and London, 1985.

- Tellioglu, Hilda, and Ina Wagner: 'Negotiating boundaries: Configuration management in software development teams,' *Computer Supported Cooperative Work. The Journal of Collaborative Computing*, vol. 6, no. 4, 1997, pp. 251-274.
- Womack, James P., Daniel T. Jones, and Daniel Roos: *The Machine that Changed the World: The Story of Lean Production*, Rawson Associates, New York, 1990. [Harper Collins Publishers, New York, 1991].