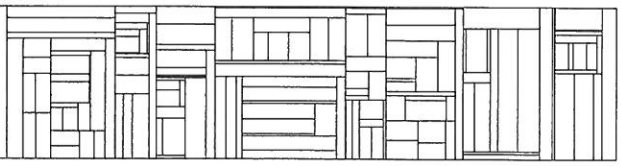
ISSN 0105-8517

A cooperative work perspective on use and development of computer artifacts

Pål Sørgaard

DAIMI PB – 234
November 1987

AARHUS UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Ny Munkegade 116 – DK 8000 Aarhus C – DENMARK
Telephone: + 45 6 12 71 88 *Telex: 64767 aausci dk*



A cooperative work perspective on use and development of computer artifacts

Pål Sørgaard
Computer Science Department
Aarhus University

Paper presented at
the 10th Information Systems Research Seminar in Scandinavia
Vaskivesi, August 10-12 1987

Abstract

This paper presents a cooperative work perspective on the role of computer artifacts in organisations. Traditionally the issue of cooperative work has been given little attention in the design of computer artifacts, and in fact much cooperative work has been disrupted rather than supported by computers.

Cooperative work is seen as a specific, prototypical kind of work with many desirable properties. Cooperative work has an informal nature and often it will not be reflected in the formal organisation of the work, it's importance often being neglected. Cooperative work will often be found in health care teams, in office work, in many project groups, and also in many kinds of mechanical work.

Two ways of coordinating cooperative work can be identified. Coordination by explicit communication and coordination through the shared material of the working process. In parallel to this there are two main forms of computer support for cooperative work, computerised media for explicit communication and computer implemented shared material.

1 Introduction

The topic computer supported cooperative work has recently started to receive attention. The first conference on computer supported cooperative work was held in December 1986[1] [10].

Cooperative work is nothing new. It has been there all the time, but it has usually been ignored in the design of computer artifacts. Instead of computer supported cooperative work we have often seen computer disrupted cooperative work. Notable exceptions do exist. Some revision and configuration control systems provide explicit support for cooperative development processes.

The main problem in system development has been shifting, now the main problem is to to understand the work and organisation which is the object of the development process [1, p. 43]. This paper focuses on cooperative work, its organisation, and on how it can be supported by computers.

The relationship between the computer and the organisation has often been seen as a purely technical question. More recently it has become common to see the role of the computer as that of a tool. When dealing with cooperative work we need to see the role of the computer in a new way. The role of the computer could be compared to the role of architecture. The architecture of an office certainly has influence on the work patterns in that office, but it can be hard to pinpoint the exact nature of this influence. Care must also be taken not to assume that the relationship between the computer artifact and the work organisation is deterministic. Blomberg describes a case where the use of the same computer artifact in two different settings has resulted in very different changes in the work organisation [5].

The research area of computer supported cooperative work is now emerging, but the question of what cooperative work actually is has so far not received much attention. The absence of this discussion in most of the papers in [10] is not motivated by an obvious shared agreement on the issue. The typical "CSCW"-applications support very different kinds of work. One of the aims of this paper is to contribute to the discussion of what computer supported cooperative work actually is. A second aim is to provide a theoretical introduction to the field. Therefore

[1] The next will be held in September 1988.

much emphasis is put on discussing the nature of cooperative work in order to provide a basis for a discussion of concrete applications for the support of this kind of work.

The paper proceeds as follows. Section 2 discusses the specific nature of cooperative work, which here is seen as distinct from the more general term collaborative work. Elements from organisation theory are used to discuss cooperative work. In section 3 the current state of the art is commented upon. Section 4 discusses the nature of computer supported cooperative work. A distinction is made between use of computers as media and use of computers to implement the shared material which is manipulated in the work process. In section 5 a number of typical computer applications for cooperative work are discussed and commented upon. Finally section 6 concludes by discussing some of the potentials and perils of computer supported cooperative work.

2 What is cooperative work?

By the term cooperative work I try to characterise a kind of work typically found in much office work, in health care, in system development projects, and in many scientific collaborations. Cooperative work is also common in blue-collar work.

Cooperative work is found where individuals work together *due to the nature of their task.* It may be by pure necessity, as when two mechanics manipulate a nut and a bolt from each side of some physical barrier, or it may be in order to increase efficiency or the quality of a product, as when system developers cooperate in a project group. I will reserve the word cooperation to the situations where people work together due to the intrinsic characteristics of the work process. I exclude work performed jointly only because of external compulsion; assembly lines have nothing to do with cooperative work.

People involved in cooperative work have a *shared goal*, part of which is the fulfilment of their shared task. Therefore cooperative work is *non-competitive.*

Cooperative work is *not hierarchically organised.* The organisation of the cooperative work is relatively flat and has an informal character. It relies heavily on horizontal communication. Cooperative work is normally performed in small groups, for example in project groups. Cooperative

work typically takes place independently of the formal hierarchy in the organisation. Authority relations, if present, need not be based on the positions of the individuals in the formal hierarchy. Communication in cooperative work may be very specific to the kind of work. We can observe specific work languages, with special words, phrases, and communication patterns. It may be hard to understand this language and its importance for the work process for an external observer.

Cooperative work is relatively *autonomous*. External planning, control, and distribution of work tasks reduce the cooperative nature of the work. Division of work may be present, but will normally be established internally in the cooperative work setting. The degree of specialisation can be very high, the participants in cooperative work may have different background and competence, but the individual tasks are not rigidly described beforehand. Flexibility in the distribution of the tasks is typical.

To sum up: I have stated a number of criteria for a work situation to be cooperative: (1) People work together due to the nature of the task, (2) they share goals and do not compete, (3) the work is done in an informal, normally flat organisation, and (4) the work is relatively autonomous. These criteria correlate, but are not completely dependent. For example, some tasks requiring people to work together may still be performed by a traditional hierarchy. But a more autonomous group could perhaps do the task much more efficiently.

The definition of cooperative work given here is strict, it defines a specific kind of work rather than a way of organising a company or agency. The definition applies to an ideal or prototypical situation. Pure cooperative work is hard to find. Cooperative work can be an aspect in many organisations, an aspect which is present concurrently with, for example, a hierarchy.

Cooperative work as defined here is a less general term than collaborative work. To collaborate is to work together or with someone else, whereas to cooperate is to work or act together for a shared purpose [25]. The distinction can also be motivated by Roget's Thesaurus [14] where cooperation and collaboration are said to have related meanings, but where collaborator otherwise is related to co-worker, team-mate etc., whereas cooperation is related to words like coagency, symbiosis, duet, and even clannishness. The distinction is also used by Dunham et al. [13, p. 343]: "Theories of human collaboration vary widely. Some emphasize cooperation. Others observe the inevitability of conflict."

The definition of cooperative work given here has some parallels in organisation theory.

Galbraith [23] uses a simple information processing model of organisations. He sees the role of the hierarchy as that of coordinating the efforts of the organisation as efficiently as possible. In this model a hierarchy can be efficient because it reduces the number of communication links in the organisation. Galbraith uses *task uncertainty* as a key concept to the design of organisations. When task uncertainty is high, functionally divided, i.e. strictly hierarchical, organisations break down because the hierarchy becomes overloaded with coordination efforts. Several design alternatives are available. These are increased slack, establishment of independent tasks, investment in vertical information systems, and establishment of horizontal contacts. Galbraith focuses on the use of horizontal contact in for example project groups, teams, and matrix organisations. This corresponds closely to the settings where cooperative work is typical. Using Galbraith's terminology we can characterise cooperative work with high task uncertainty. Therefore we often see cooperative work in single unit production or where the series are short. Likewise we see cooperative work where the technology changes rapidly.

In organisation theory the transaction cost school makes a distinction between markets, bureaucracies, and clans [31,42,7]. Markets are characterised by maximum opportunistic behaviour and low task uncertainty, prices are used to control behaviour. The typical contract in the market is the spot contract. Bureaucracies have medium task uncertainty and medium opportunistic behaviour, and rules are used to control behaviour. The typical contract is the employment contract. Clans have high task uncertainty and no opportunistic behaviour, tradition being the behaviour-controlling mechanism. Contracts in clans are totally unstructured and impossible to describe. These three kinds are prototypical organisations, seldom found in their pure form. Actual organisations exhibit a mixture of them. Thus there may be economical and formal relations between the members of an actual clan, but the social ties are more important. The theory is a kind of organisational-theoretic darwinism, where the most efficient type of organisation for a kind of activity is assumed to become predominant. Thus clans evolve in areas where the uncertainty is high and where opportunistic behaviour would be fatal. Clans appear to have many of the essential properties of groups performing cooperative work.

In an approach totally different from organisation theory, Axelrod [3] uses a game as a simple model for cooperation. In Axelrod's model cooperation may evolve between players without any verbal communication, real life examples can, for example, be found in biological symbiosis. This may serve to stress that communication is not the only, nor the central, aspect of cooperative work, as argued in, for example, [13]. In this paper it is maintained that *shared material* is an equally important aspect of cooperative work.

3 Current practices, methods and theories

Technically it ought to be feasible to support cooperative work with computers. Classical mainframe and minicomputer configurations with many terminals and the newer networks of workstations both satisfy the obvious need for connectedness.

The technical potential has not, however, been used to support cooperative work. Typically, computers have been used to support economic transactions, flow of documents, administration of students, tax payers, or items in a warehouse. Certainly these systems have been used in collaborative work in the sense that the files or records are manipulated by several people in a more or less predetermined sequence. The result of one piece of work is mediated through the system to be used later by some other or by the same worker. Such systems do not, however, lend themselves to be understood as joint facilities, but are instead more easily understood as a collection of personal information systems [18]. These systems tend to freeze the division of work, and the fact that people work on the same task or manipulate the same substance is hidden. Often horizontal communication is destroyed by such systems, thus improving management's control of planning and performance of work. Such situations can often be characterised as *computer disrupted cooperative work.*

We can also observe that people — in spite of the obstacles — use computers cooperatively. In Århus we have recently started a research programme in Computer Support in Cooperative Design and Communication. The description of the programme [19] was written by approximately 10 persons. Although not entirely without conflicts of interests, the writing process was an example of a cooperative work process. We

did it using Macintoshes interchanging floppy discs. And it worked. But we lacked support for version and configuration control since updates were frequent and since different versions of the same document for different research councils were to be prepared. We also lacked facilities for commenting each others contributions. The lesson to learn is that this piece of work was cooperative although our computer support was not developed with this in mind.

System development methods seldom focus on cooperative work. Many methods have been written on the basis of their author's experience from some successful projects. As turnaround in this process is slow, methods tend to reflect old ways of doing things. As an example, by its focus on data-flow, Yourdon's method [12,46] appears to be best applicable in the development of batch systems.

System development methods can be characterised by their *perspective* [27,30]. Methods seldom state their perspective explicitly, but their perspective is implicitly described by the tools and techniques proposed, and by their advice in questions like: What to look for? Whom to talk to? What should be described and how? How should work be organised? Should experiments be made? etc. Methods typically focus on data or information which can become data. To my knowledge there is no method which would "see" the importance of a coffee machine in the user organisation. In terms of the concepts mentioned in the previous section methods typically address systems supporting vertical rather than horizontal communication, markets and bureaucracies rather than clans, formal organisation rather that the actual, and collaboration rather than cooperation. This critique does not make methods useless, but care should be taken to ensure that the object area of the development process falls within the method's area of application. I do not know any system development method addressing the issue of cooperative work.

So far I have criticised current practices and methods. I will now discuss how the issue of computer supported cooperative work has been dealt with in the literature within the field of system development.

As mentioned in the previous section, Galbraith emphasises that task uncertainty is the primary design parameter for organisations. In his terminology increased task uncertainty may result in an overload of the information processing hierarchy in the organisation, for example due to the introduction of shorter planning intervals. Investment in vertical information systems is a strategy based on the use of computers, or other

means, to improve the information processing capacity of the organisation without changing its structure. Management information systems, real time planning systems, etc., are examples of computer applications used in this strategy. This is the only design strategy where Galbraith mentions the use of computers, so his view on the role of computer support in organisations is relatively traditional. Galbraith is aware of, and is in favour of, issues related to cooperative work, but he does not seem to be aware of any potential for computer support for the design strategy of establishing horizontal contacts.

Among the references to the transaction cost approach, Ciborra [6,7] explicitly addresses the role of computers in organisations. But Ciborra is very short on computer support for semistructured and unstructured exchanges — the kind of transactions characteristic to clans. He briefly mentions systems handling commitments[2] and for signalling interests. The rest of the discussion he explicitly restricts to markets and bureaucracies[3]. This is because Ciborra's main purpose with his paper is to use the framework of transaction cost to criticise the naive — conflict-free — understanding of the role of information expressed in many system development methods. Recently Malone et al. [26] have used the transaction cost approach to discuss the impact of computer technology on organisations. They do, however, only address the relative profitability of markets and hierarchies, claiming that we will see both electronic markets and electronic hierarchies, but that information technology will lead to a shift in relative profitability favouring market organisation. They do not, however, take clan organisation into account, and they implicitly assume that the set of transactions is constant. One may conjecture that the typical clan transaction, the transaction characterised by high task uncertainty and vulnerability to opportunistic behaviour, will become more and more important in organisations where routine work becomes automated and where the technology changes rapidly. Thus Malone et al.'s analysis, if at all correct, does not reduce the importance of cooperative work and its need for computer support.

Much effort has been laid down in documenting the negative effects of information technology on the skill of workers, on work organisation etc. (see, for example, the NJMF [29], the DUE [32], and the DEMOS [16] projects). These projects have, however, mainly aimed at developing

[2]Like the coordinator, see next section.

[3][7, footnote 4, page 64.]

strategies for the workers' influence on the system development process. They have not developed alternative technical solutions in line with their view on work organisation. The UTOPIA project [33] faced the challenge of showing that it was possible to design for skill — rather than against it. In UTOPIA, design was performed in a work-process or tool perspective [15], with primary focus on tools and materials. As presented by Ehn and Kyng [15] the tool perspective primarily addresses individual work. The experimental design strategy recommended by UTOPIA, based on the use of mock-ups, and prototyping in general, are best applicable for the design of computer applications for individuals. I do not argue that tools are "out", instead I argue that we must emphasise that tools and the material they can manipulate must be developed together, and that much human cooperation relies heavily on the properties of this material. The notion of shared material introduced in this paper is directly inspired by the UTOPIA project.

Another perspective related to the tool perspective is the human-scale information system approach [18,28]. In this approach an information system can be seen as a collection of individual — tool-like — systems which may communicate by the exchange of messages. This view is an example of the *tool/media dichotomy* prevalent in the discussion of these topics, where all features of computer systems are characterised as either tools for individual manipulation of substance or as media for the exchange of messages. This is in line with the, also prevalent, view that the central aspect of cooperation is communication [13,44,45], but it ignores cooperative manipulation of shared substance.

On a theoretical level many works point at the relationship between computer systems and organisations. In [1], for example, it is stated that one should also plan the necessary organisational changes, but, unfortunately, nothing is said about what these changes are or how they are related to the properties of the computer system.

4 Computer Supported Cooperative Work

Imagine two persons working on a car. They are mounting a new headlight. One of the workers is outside the car, positioning the headlight and pushing the bolts through the appropriate holes; the other worker is in the car directing the positioning of the headlight and attaching and

screwing the nuts to the bolts. When adjusting and screwing the nuts and the bolts, the workers use different tools to manipulate the headlight, the nuts, and the bolts. Through the materials they can feel the action of each other, thus adjusting the leverage and otherwise coordinating their work. During the mounting process they also communicate in their own, specialised, car-repair-shop language [2].

The work is coordinated in two distinct ways: By explicit communication and through the material manipulated in the process. This is schematically illustrated in figure 1. In analogy with this we can identify two different kinds of computer support for cooperative work: media for explicit communication and shared material (or substance) through which the implicit coordination can be manipulated.

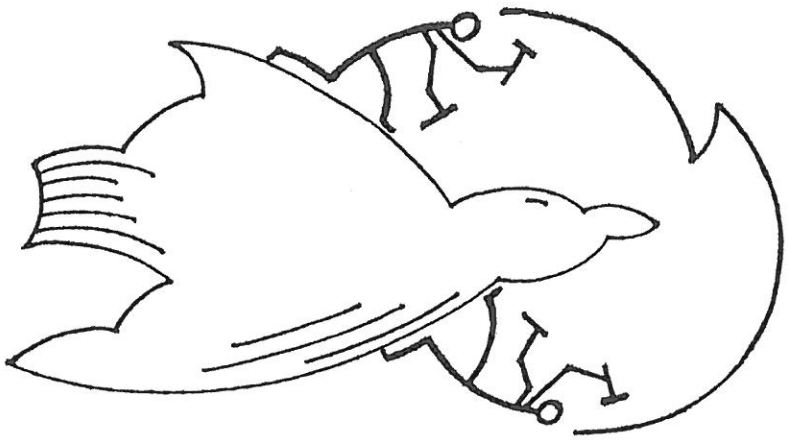


Figure 1: Tools, explicit communication, and shared material

Examples of the first kind are electronic mail, bulletin boards etc. Examples of the second kind are harder to find, although some databases, software configuration systems, and Colab, which is mentioned in section 5, do provide shared material. Finally there will be tools, tools to manipulate the shared material, and tools to access the media. The tools, however, are individual.

Shared material is a metaphor in the same sense as the tool metaphor. When we apply the tool metaphor to a text processor, we address its ability to let the user have his/her primary attention directed to the document, not to the text processor. Similarly the shared material metaphor addresses the implemented material's capability to reflect that it is used

and manipulated by several people. This issue has in some sense been addressed in the database field, but here the focus has been on protecting the system from inconsistencies and deadlocks, and the ideal has been to give the users the illusion that they are working alone on their own database independently of the others. The shared material metaphor addresses issues like:

- some actions require other people to take compensating or completing actions, and some actions have to be performed as a result of other peoples' actions, thus it should be visible that there are several users, and some of their actions should be visible or otherwise observable in a carefully thought out way,
- some actions can only take place when the others have given their explicit permission, therefore facilities for explicit communication should be integrated with the application, and
- some actions require exclusive access to parts of the shared material, where a part of the shared material should not be the smallest unit allowing consistent update, instead it should be a natural unit for the coordination of work.

Inspiration for the design of shared material can be obtained from the example with the nut and the bolt, from the situation where two persons carry a piano, or from the distinction between original documents and carbon-copies. This last example deserves a few additional words. Traditionally there was a clear difference between originals and copies, a difference which certainly was inconvenient at times, but which also served to coordinate some actions, like serialising updates, ensuring unique archiving, etc. A careful analysis of such a case could result in a design providing the best of the two worlds.

Tools and materials are always developed together. The shared material metaphor supplements the tool metaphor by stressing that this material is often shared and that much cooperation relies on certain properties of this shared material. In practice we do, however, often see problems in the integration of several tools, their materials are not the same, and we fall back to seeing the material only as files. Tools are inherently individual,[4] hence computer based tools tend to ignore issues of cooperative work. People will have to resort to working on individual copies

[4]The only exception I can think of is the 2-handled saw.

of the material in question, with no support for the issues of coordination and cooperation. The writing process described in section 2 was an example of this.

One field where the shared material metaphor should be applied is in programming environments and system development environments. These environments have had a tendency to degenerate to collections of tools, and the material has typically been sequential text files; UNIX seen as a programming environment is a good example of this.

In system development environments it is not enough to reimplement or copy the properties of existing materials like documents or paper-based files. One has to address the question of what, for example, a program is: a text-file, an abstract syntax tree, a prescription for an information process, etc. In [24] sentential forms,[5] or fragments, are proposed as basic building blocks. The chosen representation also has to reflect the way programs are used cooperatively. Programs are reused in different ways in other programs. Thus dependencies between programs, restrictions on updates, responsibility, status, maintenance of different versions and variants, etc., are important issues.

There are also other kinds of computer support for cooperative work than those that can be classified as shared material and support for explicit communication. Any computer application supporting the formation and maintenance of cooperative teams can be said to support cooperative work. It is therefore important that computer applications can be used by the group of users in the way they want to, also for "non-work" like a computerised coffee-kitty or the dart scores.

5 Cases

In this section I will briefly describe some typical computer applications for cooperative work.

Electronic mail is probably the best known computer application for the support of cooperative work. Electronic mail allows for explicit asynchronous user to user communication. Mail is only transmitted to those explicitly addressed. Normally the contents of electronic letters is restricted to reasonably sized text-only files. Electronic mail is now avail-

[5] A sentential form in a language is anything derivable from a non-terminal in the syntax for the language. Thus sentential forms may contain a mixture of terminals and non-terminals.

able in most of the capitalist, industrialised world, although its use outside the computer science community is still relatively low. Reliability and delivery times vary from network to network.

Electronic mail does not by itself address cooperative work. It is just as useful for communication where the parties are not so tightly coupled. Feldman [21] does, for example, argue that electronic mail can have an important effect by supporting exchange of so-called weak-tie messages. It turns out, however, that people primarily use electronic mail to communicate to people they work with, and they use it also when they are physically close [20]. Most operating systems also supply facilities for synchronous communication between users; these are, however, much less popular. In addition to the papers already cited, Eklundh [17] provides an extensive analysis of the kinds of dialogue and kinds of language used in an electronic mail system. An overview of the different networks available, their characteristics, how to address them, etc., can be found in [34].

Facilities for electronic mail are often poorly integrated in the user's computing environment, although a counterexample is described in [41]. In the Mjølner project work is under way to integrate electronic mail in a programming environment, thus making a little step from a programming environment to a system development environment [37]. Ciborra and Lanzara [8] report on an example where a mail facility was the main key to the success of a system development environment. This was more important than the attempts to make people follow specific system development methods.

Mail systems often allow for system defined and user defined mailing lists. Such groups may appear similar to newsgroups (see below), but they are still closed, each recipient getting his or her personal copy. It was in an environment using mailing lists Feldman [21] made her observations on weak-tie messages.

Another well known computer application in the field is electronic bulletin-boards, newsgroups etc. They differ from electronic mail in that the addressee is a newsgroup, to which those who want to may subscribe. Some newsgroups are local to a single institution, some are national, and some have a world-wide distribution. Since the group of addressees is open and typically very large, newsgroups are of little relevance to cooperative work. Cooperations can manage just as well by system maintained or privately maintained mailing lists, although establishing a mailing list requires a minimum of formal organisation. There is a notable difference

between moderated and non-moderated newsgroups. In the latter anybody may drop any message in the newsgroup, the result often being a flood of garbage. In the former all entries go via an appointed moderator who may act as anything from a serious editor to a censor removing obviously irrelevant material.

At the Computer Science Department, Aarhus University, we use a number of local newsgroups to provide information about our computer installations. Recently a course was taught where the students had to subscribe to a specific newsgroup to get course information, exercises, etc. Although it was possible, and encouraged, the students in this course put almost nothing in the newsgroup themselves. Almost all news came from the lecturer and the programmer responsible for the software used in the course. The reason for this relatively low success is hard to explain. The first explanation coming to mind is that there were too few terminals available. Experiences from other universities, with the same or a lower availability of terminals, are quite different, however. I now believe the main bottleneck was that the students were new to the medium, there was no established social code around the use of the system, like there is around the paper-based communication system at the university. The norm of emptying one's pigeon hole every day is well established, it takes some time to establish similar norms for electronic media.

A number of computer applications for cooperative work have been developed at XEROX PARC's Colab [40]. Colab is an experimental meeting room equipped with a number of interconnected personal workstations supplemented with a "liveboard", a whiteboard sized computer display. Several applications have been developed for the Colab, one of these is Cognoter [22]. Cognoter is designed to support activities like idea generation, idea organisation, writing of outlines etc. Cognoter imitates many functions of the blackboard, but it does allow its users to preserve what is on the "board" for later use, to add longer comments, and the contents of the board may be printed as an outline of a paper. The Colab group has used Cognoter in the early steps of writing their own papers. Personally I think Cognoter would have been useful in the early stages of the writing of [1].

Cognoter is supposed to be used by a closed group of people, concurrently, in the same room. Voice and other sorts of communication between the participants is expected to take place while the system is being used. Friendly behaviour is assumed in the locking algorithms. Thus

Cognoter is a system with a specialised area of application — a certain kind of meeting, and the emphasis is on shared material rather than on supporting explicit communication.

The strategy used to implement shared material in Colab is called relaxed WYSIWIS (What You See Is What I See). In strict WYSIWIS all displays are identical, a strategy which would result in chaos; imagine five moving cursors on the screen. In Colab investigations have been undertaken in different ways to relax WYSIWIS [39,38] in order to allow for private parts of the screen, etc.

There are of course situations where WYSIWIS will turn out to be appropriate, but in my opinion it is an unfortunate ideal for a multi-user interface. The metaphor underlying the XEROX STAR user interface is the desktop [36], and this metaphor has also been used in Colab. A desktop is normally used by one individual, and one person's desktop need not have any meaning to anybody else. If people try to communicate with me by dropping messages on my desktop there is a high risk that their messages never reach me. A shared material should not be described by an individual metaphor like the desktop metaphor. Other metaphors should be searched for. For a system like Cognoter, where one might want to distinguish between individual notetaking and what is written on the board, a suitable design ideal could be "What You See Is What I Show You". If the focus is on meeting support, however, concepts like agenda, floor, etc., must find some counterparts in the metaphor.

A final example of computer applications with the potential of supporting cooperative work is the coordinator [9]. The coordinator aims at maintaining the state of a certain kind of conversation: conversation for action [44,45]. The coordinator attempts to make speech acts like request, promise, declare complete, etc., explicit. Use of the coordinator will therefore lead to changes in organisational practice. To the coordinator users are not equal, for each conversation the system maintains different roles for the different parties in the conversation. One important result of using the coordinator is that it is easy to get an overview of one's commitments to others as well as other peoples' commitments to oneself.

The coordinator runs on personal computers which communicate via electronic mail. Thus it shares many characteristics with electronic mail, for example asynchronous use, but the users will typically have some common work that ties them together.

In terms of the concepts in this paper the coordinator focuses exclusively on explicit communication, the notion of shared material being entirely absent. A comparison of Colab and the coordinator can be found in [43]. Experience from use of the coordinator is documented in [13].

6 Conclusion

In this paper I have given a prototypical definition of cooperative work, and stated that shared material and explicit communication are the most interesting areas of computer support for this kind of work.

Classical computer applications only support market or bureaucratic aspects of organisations, thus causing a shift in favour of these kinds of organisation. Media like electronic mail, and applications of electronic mail like the coordinator, are primarily useful for horizontal communication. One may also suspect that they are of more use to internal communication and coordination in clans than in markets or bureaucracies where the communication is of a more formal nature. In this way we should expect a shift in favour of clan organisation, a shift opposite to the shift predicted by Ciborra [6,7] and by Malone et al. [26]. This is not the only tendency we will observe, however. It is also possible that the efficiency and obvious advantages of electronic media may lead to ignorance of the importance of physical proximity between and a milieu for cooperators. This will typically be the case if electronic communication is seen as a substitute for travelling, meetings, etc. Thus we may also observe a shift from physically close cooperations to electronically connected collaborations.

The contradiction between these different predictions is primarily due to the different ways of conceiving and using electronic media. If they are used as a *supplement* to existing media and working practices they may favour cooperative work, if, on the contrary, they are used as a *replacement* they may have the opposite effect.

Computerised shared material gives us a rich set of possibilities. We can construct materials which to a certain extent simulate the properties of their former, physical counterparts. It would, for example, be easy to distinguish original documents from copies, drafts from final versions, etc. More importantly we can construct entirely new sorts of materials with properties desirable for cooperative work. Hypertext systems can,

for example, allow for writing comments and annotations in documents without painting it with red ink, this could be used to collect comments from many people in one hypertext document. The people giving comments might be allowed to see each others comments, thus relieving them from pointing at issues already commented upon. Facilities for version and variant control for texts or programs may allow a document to be "taken" by one person but still letting it be readable to others. Using techniques from robotics even the physical coordination between the two auto mechanics mentioned in section 4 can be mediated through computers. This sort of coordination can thus be extended to environments hostile to humans or to tasks below or above the human powerrange.

When changing the material used in a working process there is a risk that some aspects get lost. Examples from text processing are the absence of difference between original and copies when using laserprinters or good photocopiers, the use of deceptively nice typography for very incomplete drafts of a paper, the possibility for small changes resulting in a flood of almost identical versions etc. Another example is the use of circulation lists: a document or a magazine is circulated in a group, each person in the group acknowledging that he/she has read it by checking his/her name on the list. This property may get lost when each person gets his own copy, photocopy, or electronic letter. In general it is hard to know how a traditional material supports cooperation, its properties in this respect may be hidden to the observer because they only work due to the idiosyncrasies of the work process in question.

Acknowledgements

I would like to thank Morten Kyng for insightful ideas and comments. I would also like to thank my working group at the 10th Information Systems Research Seminar in Scandinavia for stimulating discussions. Finally Riitta Hellman gave the illustration of tools, media, and shared material an artistic look.

References

[1] Niels Erik Andersen, Finn Kensing, Monika Lassen, Jette Lundin, Lars Mathiassen, Andreas Munk-Madsen, and Pål Sørgaard. *Pro-*

fessionel systemudvikling. Teknisk Forlag, København, 1986.

[2] Peter Bøgh Andersen. *Sproget på arbejdet.* GMT, Kongerslev-Grenå, Denmark, 1977.

[3] Robert Axelrod. *The Evolution of Cooperation.* Basic Books, New York, 1984.

[4] David R. Barstow, Howard E. Shrobe, and Erik Sandewall, editors. *Interactive Programming Environments.* McGraw-Hill Book Company, New York, 1984.

[5] Jeanette Blomberg. The variable impact of computer technologies in the organization of work activities. In [10].

[6] Claudio U. Ciborra. Information systems and transactions architecture. *International Journal of Policy Analysis and Information Systems*, 5(4), 1981.

[7] Claudio U. Ciborra. Reframing the role of computers in organizations: the transaction costs approach. In Lynn Gallegos, Richard Welke, and James Wetherbe, editors, *Proceedings of the Sixth International Conference on Information Systems*, Indianapolis, December 1985.

[8] Claudio U. Ciborra and Giovan Francesco Lanzara. True stories and formative contexts in information systems development. In *Information Systems Development for Human Progress in Organizations*, Atlanta, May 1987.

[9] *The Coordinator Workgroup Productivity System, Workbook and Tutorial Guide.* Technical Report, Action Technologies, San Fransisco, 1986.

[10] *Conference on Computer Supported Cooperative Work*, MCC Software Technology Program, Austin, Texas, December 1986. Proceedings.

[11] Pierpaolo Degano and Erik Sandewall, editors. *Integrated Interactive Computing Systems*, North-Holland, Amsterdam, 1983.

[12] Tom DeMarco. *Structured Analysis and System Specification.* Yourdon Press, New York, 1979.

[13] Robert Dunham, Bonnie M. Johnson, Grady McGonagill, Margarethe Olson, and Geraldine M. Weaver. Using a computer based tool to support collaboration: a field experiment. In [10].

[14] Robert A. Dutch, editor. *Roget's Thesaurus.* Penguin, 1966.

[15] Pelle Ehn and Morten Kyng. *A Tool Perspective on Design of Interactive Computer Support for Skilled Workers.* Publication PB-190, Computer Science Department, Aarhus University, Århus, January 1985. Also in [35].

[16] Pelle Ehn and Åke Sandberg. *Företagsstyrning och löntagarmakt.* Prisma, Stockholm, 1979.

[17] Kerstin Severinson Eklundh. *Dialogue Processes in Computer-Mediated Communication.* Volume 6 of *Linköping Studies in Arts and Science*, Liber, Malmö, 1986. Ph.D. thesis.

[18] Inger Eriksson and Riitta Kalmi. Interpretation of large information systems as a set of personal information systems and enrichening the users' knowledge with this view. In Hans-Erik Nissen and Gunhild Sandström, editors, *Quality of work versus quality of information systems. Report of the ninth Scandinavian research seminar on systemeering*, University of Lund, Department of Information and Computer Sciences, Lund, December 1986.

[19] Peter Bøgh Andersen et al. *Research Programme on Computer Support in Cooperative Design and Communication.* Internal Report IR-70, Computer Science Department, Aarhus University, Århus, May 1987.

[20] John D. Eveland and Tora Bikson. Evolving electronic communication networks: an empirical assessment. In [10].

[21] Martha S. Feldman. Constraints on communication and electronic mail. In [10].

[22] Gregg Foster and Mark Stefik. Cognoter, theory and practice of a colab-orative tool. In [10].

[23] Jay Galbraith. *Planlægning af organisationer.* Inter European Editions, Amsterdam, 1979.

[24] Bent Bruun Kristensen, Ole Lehrmann Madsen, Birger Møller-Pedersen, and Kristen Nygaard. Syntax directed program modularization. Pages 207–219 in [11].

[25] *Longman Dictionary of Contemporary English.* Longman, Burnt Mill, Harlow, Essex, 1978.

[26] Thomas W. Malone, Joanne Yates, and Robert I. Benjamin. Electronic markets and electronic hierarchies. *Communications of the ACM*, 30(6):484–497, June 1987.

[27] Lars Mathiassen. *Systemudvikling og systemudviklingsmetode.* Publication PB-136, Computer Science Department, Aarhus University, Århus, 1981. Also DUE-report nr. 5.

[28] Markku Nurminen. *Human-scale Information Systems.* Technical Report I.821, Institutt for informasjonsvitenskap, Universitetet i Bergen, Bergen, 1982.

[29] Kristen Nygaard and Olav Terje Bergo. *Planlegging, styring og databehandling. Grunnbok for fagbevegelsen.* Tiden, Oslo, 1972.

[30] Kristen Nygaard and Pål Sørgaard. The perspective concept in informatics. In Gro Bjerknes, Pelle Ehn, and Morten Kyng, editors, *Computers and Democracy - A Scandinavian Challenge*, Gover, Aldershot, England, 1987.

[31] William G. Ouchi. Markets, bureaucracies, and clans. *Administrative Science Quaterly*, 25, March 1980.

[32] The DUE project group. *Klubarbejde og EDB.* Fremad, København, 1981.

[33] The UTOPIA project group. *An Alternative in Text and Images.* GRAFITTI 7, Swedish Center for Working Life, Stockholm, 1985.

[34] John S. Quarterman and Josiah C. Hoskins. Notable computer networks. *Communications of the ACM*, 29(10), October 1986.

[35] M. Sääksjärvi, editor. *Report of the Seventh Scandinavian Research Seminar on Systemeering.* Helsinki School of Economics, Helsinki, 1984.

[36] David Canfield Smith, Charles Irby, Ralph Kimball, Bill Verplank, and Eric Harslem. Designing the STAR user interface. *BYTE*, 7(4), April 1982. Also in [11].

[37] Pål Sørgaard. *HEIMDAL: a Mjølner mail handler.* Mjølner report DK-15.1, Computer Science Department, Aarhus University, November 1986.

[38] Mark Stefik, Daniel G. Bobrow, Gregg Foster, Stan Lanning, and Deborah Tatar. WYSIWIS revised: early experiences with multiuser interfaces. *ACM Transactions on Office Information Systems*, 5(2):147–167, April 1987.

[39] Mark Stefik, Daniel G. Bobrow, Stan Lanning, and Deborah Tatar. WYSIWIS revised: early experiences with multi-user interfaces. In [10].

[40] Mark Stefik, Gregg Foster, Daniel G. Bobrow, Kenneth Kahn, Stan Lanning, and Lucy Suchman. Beyond the chalkboard: computer support for collaboration and problem solving in meetings. *Communications of the ACM*, 30(1), January 1987.

[41] Warren Teitelman. A display-oriented programmer's assistant. In [4].

[42] Oliver E. Williamson. The economics of organization: the transaction cost approach. *American Journal of Sociology*, 87(3), 1981.

[43] Terry Winograd. The design of collaboration: a contrastive case study of two tools for conversation. 1987. Draft of paper.

[44] Terry Winograd. A language/action perspective on the design of cooperative work. In [10].

[45] Terry Winograd and Fernando Flores. *Understanding Computers and Cognition.* Ablex Publishing Corp., Norwood, New Jersey, 1986.

[46] Edward Yourdon. *Managing the System Life Cycle.* Yourdon Press, New York, 1982.