Proceedings of the Second European Conference on Computer-Supported Cooperative Work
Bannon, L., Robinson, M. & Schmidt, K. (Editors)
September 25-27, 1991, Amsterdam, The Netherlands

CSCW and Distributed Systems: The Problem of Control

Tom Rodden Gordon Blair
Lancaster University, U.K.

The user-centred philosophy of CSCW challenges the established principles of many existing technologies but the development of CSCW is dependent on the facilities provided by these technologies. It is therefore important to examine and understand this inter-relationship. This paper focuses on distributed computing, a technology central to the development of CSCW systems. The nature of both CSCW and distribution are compared by using a common framework. In this discussion, control emerges as the major problem in supporting CSCW systems. It is argued that existing approaches to control in distributed systems are inadequate given the rich patterns of cooperation found in CSCW. A number of recommendations are made for improving distributed support for CSCW.

1. Introduction

Computer support for cooperative working (CSCW) has emerged over the last five years as a research discipline in its own right (Bannon, 1991). The growing interest in CSCW reflects the demands of industry for improved tools to aid the coordination and control of group activities. The majority of CSCW applications are fundamentally distributed and are dependent on the facilities provided by existing distributed systems platforms. It is therefore important to assess the support that such systems provide.

The aim of this paper is to evaluate distributed system support for CSCW. In particular we wish to consider the particular requirements of CSCW and the interaction between distributed systems and CSCW. To achieve this two dimensions of CSCW are introduced in section 2. These dimensions provide a basis for the our discussion. This is followed in section 3 with an examination of distributed system support for CSCW based on the above dimensions. Control, an additional and important feature of both CSCW and distributed systems, is introduced in section 4. The impact of control on distribution is examined and techniques to support control are also discussed. Distributed transactions are presented in section 5 as an illustrative case study of the problem of control. Finally some concluding remarks are presented in section 6.

2. Dimensions of CSCW

A wide variety of CSCW systems have been developed reflecting the many different views of cooperation. The nature of cooperation has been an on-going debate within the CSCW community (Schmidt, 1989). Two principal characteristics have emerged from this debate *the form of cooperation* and *the geographical nature.* We shall use these characteristics as the basis for examining both CSCW systems and the underlying support provided by distributed systems.

2.1 The Form of Cooperation

CSCW systems are primarily concerned with supporting a number of users cooperating to address a particular problem, or range of problems. People cooperate in a variety of ways depending on a range of circumstances. The nature of this cooperation can be distinguished by the way in which the group members interact. People can either interact and cooperate *synchronously* or *asynchronously.* Synchronous interaction requires the presence of all cooperating users while asynchronous cooperation occurs over a longer time period and does not require the simultaneous interaction of all users.

Figure 1 shows how a number of classes of CSCW systems fit into this division.

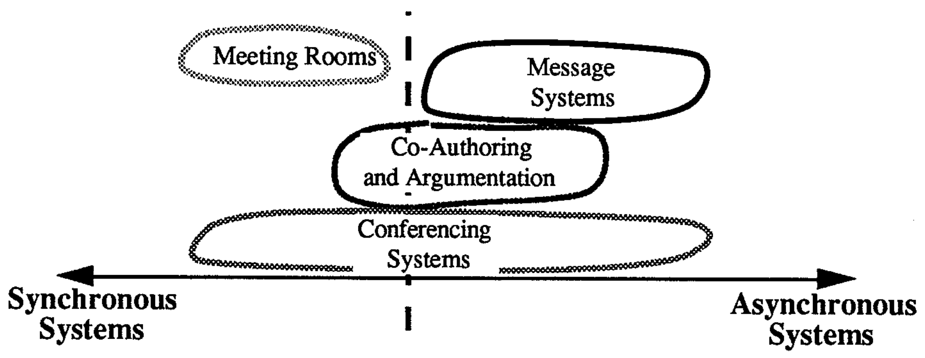

Figure 1 Forms of cooperation in CSCW systems

From this classification three general classes of CSCW system can be highlighted.

i) Purely synchronous systems

Purely synchronous systems need the simultaneous presence of all users. This general class of system is used for investigative and creative problems. Systems which typify this approach include real-time conferencing systems (Lauwers, 1990) using shared screen techniques (Stefik, 1987b) and the brainstorming tools found in meeting rooms (Stefik, 1987a)

ii) Purely Asynchronous systems

Asynchronous systems are designed to allow cooperation without the simultaneous presence of all group members. Cooperative message systems are a primary example of this type of system where users take on independent roles which produce and consume messages. Similarly, traditional conferencing systems assume an asynchronous mode of cooperation with users reading and adding articles to conferences independently of other users.

iii) Mixed Systems

Mixed systems contain elements of support for both synchronous and asynchronous cooperation. They allow real-time synchronous cooperation to take place within the same framework as time-independent asynchronous working. The primary examples of this type of systems are computer conferencing and co-authoring and argumentation systems. Modern computer conferencing systems provide a central asynchronous conferencing systems often augmented with facilities such as real-time conferencing (Sarin, 1985).

2.2. Geographical Nature

Computer support for group interaction has traditionally considered the case of geographically distributed groups who work asynchronously to each other. More recent research (Stefik, 1987a) has complemented this emphasis by considering the support of face to face meetings. As a result cooperative systems can be considered as being either *remote* or *co-located.* In this classification the division between remote and co-located is as much a logical as a physical one and is concerned with the accessibility of users to each other rather than their physical proximity. A range of CSCW systems are divided in terms of their geographical nature in figure 2.

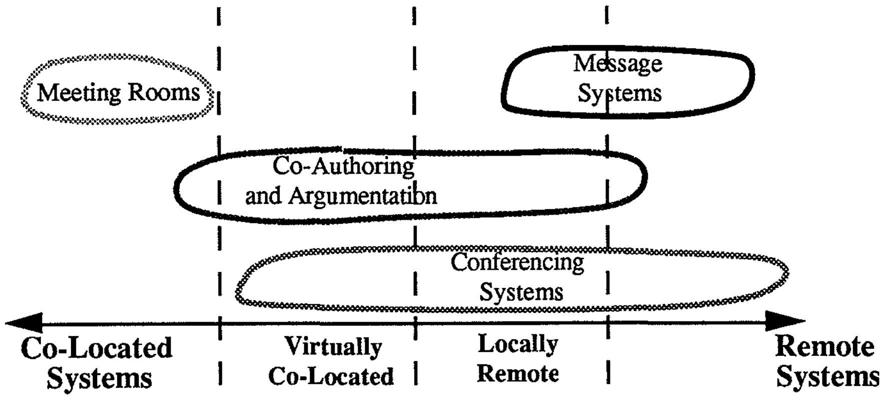

Figure 2 Geographical nature in CSCW systems

Four general divisions can be highlighted.

i). Co-Located.

Purely co-located systems require the local presence of all users. This class of system normally takes the form of a purpose built meeting room with a large projected computer screen and a number of personal computer linked by a local area network (Kraemer, 1988).

ii) Virtually co-located

Systems which are virtually co-located are similar to co-located systems but do not have the requirement that participants need to be in one room. This is often achieved by the use of Multimedia technology, allowing real-time audio and video links to be maintained. Systems in this class include real-time multimedia conferencing systems such as those been developed for MMConf (Crowley, 1990) and Mermaid (Watabe, 1990).

iii) Locally Remote

Locally remote systems are those systems which provide high-bandwidth real time accessibility between users often using shared screen techniques.. Argumentation and co-authoring systems such as CoAuthor (Hahn, 1991) or gIBIS (Conklin, 1987) and real-time conferencing facilities such as RTCAL (Sarin, 1985) can be considered in this way.

iv) Remote

Remote cooperative systems are those that assume the existence of only minimal accessibility between users. These include message systems which assume only the simplest of communication systems and computer conferencing systems which assume only rudimentary "dial-in" mechanisms.

3. CSCW and Distributed Systems

Distributed systems have been one of the major growth areas in computing over the past decade. Products such as ETHERNET (Shoch, 1982) and MACH (Jones, 1986) are available in the market place and standards to provide open communications are generally agreed.

It can be argued that distributed systems are entering a period of consolidation with techniques for implementing distributed systems relatively well understood, and that emphasis should be placed on issues such as promotion of standards, large scale experiments, and gaining of experience. However, a major problem in distributed systems is a lack of existing applications of the technology leading to technological solutions to technological problems.
Until recently, this feature of distributed computing has not posed many problems. However, the emergence of CSCW has led to more sophisticated demands on the underlying technology. This section reviews the ability of existing distributed system technologies to support the wide range of CSCW systems.

3.1. The Form of cooperation

Distributed systems have traditionally being viewed in terms of support for cooperation between a number of computers connected by a network. It is important to note that the term cooperation is used in this context to refer to how closely related the computers within the distributed systems are to each other, rather than the more general application of the term in section 2. This interpretation is used throughout this section.

The nature of support for cooperation varies greatly from system to system. A traditional problem with cooperation in distributed systems is the need to recognise autonomy of individual sites in a network. Indeed, full cooperation and full autonomy are actually two extremes in a spectrum of possibilities with most practical systems found between these two extremes.

Increasing the autonomy of a system inevitably decreases the support for cooperation and vice-versa. Much of the research in distributed systems has been concerned with resolving this design tension and establishing a compromise between the two extremes. A number of distinct classes of system, each taking a particular approach to this issue, have been developed:-

i) autonomous systems with mailing capabilities

This is an important class of system where personal computer environments are interconnected by electronic mail allowing users to interact asynchronously via (usually) text based messages.

ii) resource sharing systems

Resource sharing systems allow resources to be accessed whether they are local or remote to a workstation. The motivation for resource sharing systems is that many resources are expensive and hence it is economical to share such resources across a network.

iii) distributed operating systems

Distributed operating systems are operating systems which manage resources across a distributed environment (Tanenbaum, 1985). They provide global management of such resources with the consequent loss of node autonomy. Most distributed operating systems are based on the *client-server* model of interaction with clients requesting remote operations from *servers* on other nodes.

More recent distributed operating systems have also tended to provide more sophisticated support for interaction. For example, several systems have developed protocols to provide general group interaction (e.g. ISIS (Birman, 1989)) as opposed to the one to one patterns encouraged by the traditional client-server model.

The need to support autonomy has proved to be more important than the support for cooperation. Most commercially available computer systems support a mailing capability and this has become accepted as a standard means of cooperation in a distributed system and provides adequate support for the development of asynchronous cooperative systems.

Systems with resource sharing capabilities provide access to networked resources such as printers and remote files. More sophisticated resource sharing systems will provide the user with a global file system accessible from anywhere in the network. Resource sharing systems provide an ideal platform for developing *mixed cooperative systems* with asynchronous cooperative working as the norm and rudimentary synchronous support being provided as an expensive shared resource.

There has been much less commercial interest/ exploitation of distributed operating systems. Distributed operating systems have been an area of intense research activity (Mullender, 1986); however, this has yet to be mapped on to a real demand for the technology.

Distributed operating systems represent the maximum support available for cooperation. They support reasonably sophisticated modes of interaction and often mask out the problems of distribution (e.g. locating objects and handling failure). However, most distributed operating systems allow some recognition of the autonomy of individual nodes and the cooperative end of the spectrum has not been explored fully. There are a few notable exceptions , for example, the work of the ISIS project on group interaction could be viewed as supporting more sophisticated levels of cooperation.

This spectrum of support corresponds quite closely to the forms of cooperation described in section 2.1. Synchronous systems require highly cooperative distributed systems while asynchronous systems tend to be much more autonomous in nature. Basic asynchronous cooperative systems need only the facilities provided by electronic mail systems, the most autonomous of distributed systems. Fully synchronous systems test the facilities provided by the most cooperative of distributed operating systems. Most distributed systems are found at the lower end of the spectrum, i.e. supporting a high degree of autonomy. This may account for the lack of highly interactive synchronous cooperative systems. The two views of a cooperative system are shown together in figure 3

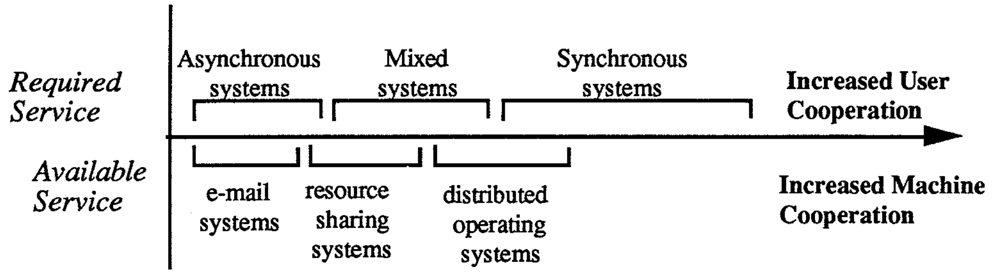

Figure 3 Comparison of Models of Cooperation

A strong correspondence can be seen between the two categories in figure 3. However, there appears to be a gap when considering more sophisticated forms of synchronous working. It is not clear how best to support highly synchronous cooperative systems such as meeting rooms and real time multimedia conferencing with existing distributed technology.

3.2. Geographical Nature

As discussed in section 2.2, the geographic nature of CSCW systems is concerned with the *logical* concept of co-location. In contrast, distributed computing has been solely concerned with the *physical* transmission and processing of specialised, computer-oriented, media such as numerical and textual data. Most distributed computing environments have been connected by a range of local or wide area networks providing a reasonable handling of such media types. The characteristics of each type of network is summarised in figure 4.

Network	Throughput	Classification
Ethernet	10 Mbits/Sec	Local Area Network
1Base5 CSMA/CD	1 Mbits/Sec	Local Area Network
PSS (UK)	64 KBits/Sec	Wide Area Network
Arpanet (USA)	64 KBits/Sec	Wide Area Network

Figure 4 Local vs Wide Area Networks

The performance characteristics listed above have proved sufficient to support a range of distributed computing environments. The table also highlights the quantitative difference that currently exists between local and wide area technologies. Local area networks have been used to implement the full range of systems described in section 4.1 including distributed operating systems supporting varying degrees of cooperation. Wide area networks have generally been restricted to mailing systems and, occasionally, resource sharing systems.

Recently, there has been great interest in high speed networks and, in particular, their capability to handle a greater variety of media types. The capabilities of these multimedia networks are summarised in figure 5.

Network	**Throughput**	**Classification**
FDDI	100 Mbits/Sec	Local Area Network
DQDB	upto 100 Mbits/Sec	Metropolitan Area Network
Basic Rate ISDN	64 KBits/Sec	Wide Area Network
Primary Rate ISDN	2 MBits/Sec	Wide Area Network
Broadband ISDN	155 MBits/Sec	Wide Area Network

Figure 5 High Performance Networks

Networking technology is increasing at a rapid rate but there is still a long way to go before such networks can provide support for sophisticated forms of multimedia cooperation. The problems are most acute when considering the group interactions demanded by CSCW. Considerable research is also required in issues such as synchronisation of different multimedia channels and the integration of high performance protocols into multimedia workstations (Hopper, 1990).

The existing spectrum of communications technologies in distributed systems can be compared directly with the geographical nature CSCW systems (figure 6)

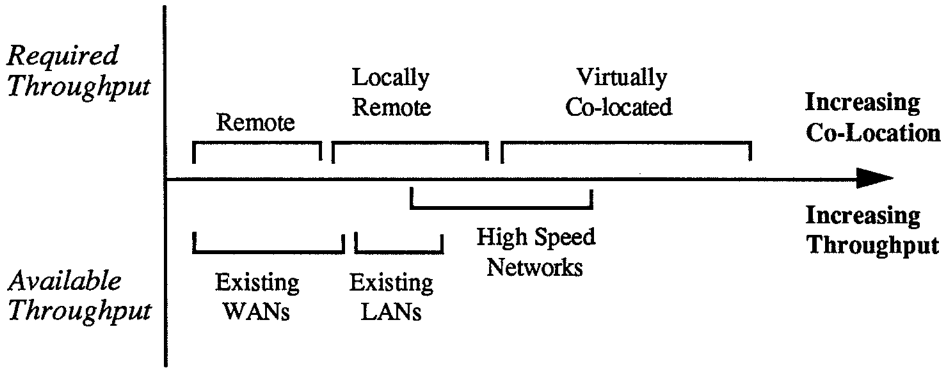

Figure 6 Geographic Dispersion Comparison

As in the case of cooperation distributed systems seem to provide good support for asynchronous cooperative systems. However, there are limitations with existing technology in supporting more synchronous styles of work. This is particularly true in CSCW applications supporting a high degree of co-location. Communication networks simply cannot cope with the logical 'bandwidth' demanded by this class of application. It is likely that high performance multimedia networks will have some impact on CSCW systems. The extent of this impact will depend on the development of protocols suitable for CSCW systems.

4. The Importance of Control

The previous sections have examined the styles of interaction and the geographical nature of both CSCW and distributed systems. However, a critical element is still missing from our discussion. The distinguishing feature of CSCW systems is their approach to representing and controlling cooperation. This section examines this issue in more depth. In effect, the authors see this issue as crucial to future success of CSCW systems.

4.1 CSCW and Control

People work together to solve a wide variety of problems using different forms of cooperation for each class of problem. Cooperative problems can be though of as existing at some point on a spectrum ranging from *unstructured* problems at one end to *prescriptive* tasks at the other. Unstructured problems are those requiring creative input from a number of users which often cannot be detailed or described in advance; software design is a good example of such an activity. Prescriptive tasks, on the other hand, represent the routine procedural cooperative mechanisms used to solve problems which have existing group solutions. Prescriptive tasks respond well to detailed control of cooperation while unstructured problems require a significant degree of freedom to be exercised by the cooperative system.

The amount of control provided by cooperative systems is an additional means of classifying cooperative systems. This classification is significant in that it highlights the level of automation each cooperative system provides.

CSCW systems exhibit two major forms of control, *explicit* or *implicit* control. In systems which provide explicit control users may both view and tailor group interaction and cooperation. In contrast, systems exhibiting implicit control provide no techniques for representing or coordinating group interaction. These systems dictate cooperation by the styles of interaction they allow.

A simple classification of the representation and control of cooperation in CSCW systems yields five classes of system.

i) Speech act or conversation based systems

Speech act systems apply a linguistic approach to computer supported cooperation based on speech act theory which considers language as a series of actions. Cooperation is represented and controlled within this class of system using some form of network structure detailing the patterns of message exchange. Speech Act theory has been forms the basis of several computer systems including the Coordinator system (Winograd, 1987) and the CHAOS project (De Cindio, 1986).

ii) Office procedure systems

Office procedures describe tasks performed within an office in terms of the combined effect of a number of small sub-tasks or procedures. Research has concentrated on developing languages which allow the specification of office procedures and a description of their interaction. This class of system is characterised by the use of a procedural language to describe and control cooperation by defining roles and activities.Approaches of this form include the AMIGO (Danielson, 1986) and COSMOS (Wilbur, 1988) projects.

iii) Semi-formal active Message systems

Semi-formal or active message systems provide supportive mechanisms for automatic message handling including the concepts of roles and autonomous agents. Systems of this form include the OBJECT LENS (Malone, 1988), the Strudel project (Sheperd 90), and the ISM system (Rodden, 1991)

iv) Conferencing systems

Conferencing systems provide basic control mechanisms which are minimal and fixed within applications. In traditional conferencing systems this takes the form conferences and moderators who control the addition of information to these topics. In real time conferencing systems control centres around the floor control mechanism imbedded in the conferencing application which dictates who has access to a shared conference space at any given time.

v) Peer- group meeting or Control free systems

Peer meeting systems such as the Colab system (Stefik, 1987a) deliberately do not provide any control mechanisms and rely on the meeting participants to formulate their own meeting protocols. All users have equal status and may amend and use the systems freely. In turn the systems keeps no track of the nature or form of group work being undertaken and provide limited support for these work processes.

The first three classes listed above are all examples of systems which exhibit *explicit* control allowing the representation and editing of control information. In contrast, conferencing and control free systems are *implicit* control systems which contain no representation of control.

4.2 Control requirements for CSCW

CSCW encompasses a wide range of control techniques. In many ways this is to be expected; CSCW is essentially about supporting the rich patterns of inter-personal cooperation. This richness should be reflected in the provision of control within CSCW systems, and the underlying technology should support rather than constrain this process.This latter point highlights the importance of the relationship between CSCW and distributed systems design. It is difficult to derive precise requirement from the list of control techniques presented above. However, some important observations can be made:

- The organisational context of the work needs to be captured.
- The many different forms of cooperation need to co-exist.
- The structure and organisation of groups need to be explicitly recognised.
- Groups work in dynamic and unexpected ways and are themselves dynamic
- Control should be enabling rather than constraining

Collectively these issues demand a *user-centred* approach to the control of cooperation within CSCW systems. This poses fairly fundamental questions for distributed system designers and highlights significant deficiencies in existing technology.

4.3 Supporting Control in Distributed Systems

Traditionally, distributed systems have taken a *systems-oriented* approach to control. They view control as dealing with the problems of distribution and masking such problems from applications (*distribution transparency*). Unfortunately this focus on transparency has tended to re-inforced the bottom-up development of distributed systems. For example, consider the problem of shared access to resources. In most distributed systems this is dealt with by masking out the existence of other users. Hence sharing is transparent with each user unaware of the activity of others. This clearly contradicts the needs of CSCW.

Recent work on distributed systems has clarified the meaning of the term distribution transparency (ANSA, 1989). Distribution transparency is now seen as a collective name for the masking out of various features of a distributed computation. In effect, there are a number of individual transparencies corresponding to each of these features(figure 7).

Transparency	**Central Issue**	**Result of Transparency**
Location	The location of an object in a distributed environment	User unaware of the location of services
Access	The method of access to objects in a distributed system	All objects are accessed in the same way
Migration	The re-location of an object in a distributed environment	Objects may move without the user being aware
Concurrency	Shared access to objects in a distributed environment	Users do not have to deal with problems of concurrent access
Replication	Maintaining copies of an object in a distributed environment	System deals with the consistency of copies of data
Failure	Partial failure in a distributed environment	Problems of failure are masked from the user

Figure 7 The Forms of Transparency in Distributed Systems

The prevalent view in distributed computing is to implement each of these transparencies to mask out *all* the problems of a distributed system. This is particularly true in the distributed operating system community. The problem with this approach is that presumed control decisions are embedded into the system and hence cannot be avoided or tailored for specific classes of application. This is the root of the problem in supporting CSCW. Because of the dynamic requirements of CSCW applications, it is very unlikely that such prescribed solutions will be suitable.

It is important to consider alternatives to this complete distribution transparency. System designers are currently aware of the problems that can be caused by full distribution transparency. Consequentially, a number of alternative approaches have already been explored:

i) Non-transparency

In non-transparent systems, all the features of distribution are visible to the programmer. They must therefore deal directly with issues such as failure and migration. This allows more flexibility since individual applications can deal with the management of objects in a distributed environment. However, the handling of distribution can become an intolerable burden on the programmer.

ii) Selective Transparency

Selective transparency allows the application developer to opt for transparency or non-transparency for each of the issues in distributed computing (figure 7). It is therefore possible to have location and access transparency, for example, but request non-transparency for the other issues. This approach provides some of the flexibility required for CSCW applications, however, existing solutions do not include user selection.

None of these provide complete solutions to the problem of controlling CSCW applications. The option of transparent control is too prescriptive for the needs of CSCW applications. However, the alternative of non-transparency imposes too high a burden on application developers. Selective transparency does appear more promising but does not address the fundamental user issues within control in cooperative working. CSCW demands a fresh approach to control which is specifically tailored for cooperative working. There has been very little work in this area. However, it is possible to identify a number of features of such an approach.

i) Clean separation of mechanisms and policies

The first requirement for control in CSCW applications is that there should be a clean separation between the mechanisms required for distribution management and the policies which govern the use of these mechanisms. To appreciate this distinction, consider the case of migration. There is a clear distinction between the ability to move an object (the mechanism) and the decisions about when the object should be moved and to which site (the policy). Distributed systems can provide the mechanisms required to manage distribution leaving higher level authorities to impose the policy. This separation of concerns is implicit in both non-transparency and selective transparency.

ii) Tailored Mechanisms

Current mechanisms have been developed in the classical bottom-up tradition of distributed systems. Such mechanisms may not be suitable for the particular semantics of CSCW applications. It is therefore important to consider existing mechanisms for the various transparencies and whether they are suitable for the demands of CSCW. Returning to the discussion of section 4.1 mechanisms delimit the *implicit control* exhibited by CSCW systems. A single set of mechanisms is unlikely to be suitable for all manifestations of implicit control in CSCW applications and the co-existence of a range of mechanisms needs to be considered.

iii) Tailored Policies

Distribution policies provide the representation necessary for *explicit control* in CSCW systems. It is important that these policies meet the control requirements identified in 4.2. It is equally important that policies can be tailored to allow support across the range of explicit control techniques identified in section 4.1. The provision of the policies will require input from all areas of CSCW. It is important to avoid these policies overly inhibiting the cooperation of users. As described in (Armstrong, 1990) when considering good practice in management science: "Policies are both restrictive and permissive at once. They spell out the limits to actions, but at the same time they give freedom to act within the limits specified".

5. A Case Study in Control: Distributed Transactions

Transaction mechanisms are concerned with the maintenance of consistency in a distributed system (Spector, 1989). In particular, they deal with concurrent access to data and partial failure of the system. Traditional approaches to transactions typify the *transparent* approach to distributed computing. More specifically, transaction mechanisms realise both concurrency and failure transparency, masking out problems associated with these features of a distributed system. Transparency is achieved by *prescribing* the following principles:-

i) serialisability

Transactions handle concurrent access to shared information by enforcing a regime where concurrent operations are allowed only if their combined effect is equivalent to a serial sequence of operations.

ii) recoverability

Systems recoverability is supported by the creation of a set of consistent *snapshops* which can be returned to in the event of failure. Effectively, this allows transaction to be undone if an error occurs.

The provision of both serialisability and recoverability has been examined in detail and a wide range of algorithms have been proposed (Kohler, 1981). The general approach adopted is to restrict access to data by *locking* out other operations. This gives the impression of shared access being carried out in isolation. The problem with this approach is that it embeds one particular view of cooperation. This is unacceptable for CSCW giving the rich patterns of cooperation identified above (section 4.1). For example, consider the case of a co-authoring system. If a group member is updating a section of text, then it might make sense for an interested colleague to "read over their shoulder". This would not be supported by a simple locking strategy.

Several researchers have started to focus on transactions for group working (Ellis, 1989). This research is still at an early stage. However, some interesting results are starting to emerge. For example, a paper by Skarra (Skarra, 1988) challenges traditional transaction models and proposes an alternative approach more closely tailored for group work. They explicitly identify the notion of a *transaction group* which co-ordinates access to shared data for a number of co-operating members. Within a transaction group, the notion of serialisability is replaced by access rules based on the semantics of the cooperation. Access rules provide the *policy* of cooperation as discussed in the previous section. Policies can thus be *tailored* for a particular application by amending access rules.

Transaction are symptomatic of the mismatch between distributed systems platforms and CSCW systems. It is clear that traditional approaches to transactions are not well suited to group work and hence many group applications have chosen to by-pass the system support. This places unacceptable burdens on developers of CSCW systems. It is therefore important to continue the work on group transactions and to identify suitable user-centred mechanisms and policies. Similar examinations are required across the field of distributed systems.

6. Concluding Remarks

Existing distribution technology currently has shortfalls in supporting CSCW systems both in terms of the cooperation between users and the geographic nature of these users. However, it is possible to see how particular shortfalls can be overcome by current developments in technology, e.g. high speed networks. More seriously, the traditional approach to control in distributed systems seems to be inadequate. It is difficult to foresee how distribution transparency can provide the highly flexible and tailorable facilities needed to represent the process of cooperation within CSCW applications.

The provision of appropriate facilities will almost certainly require a careful re-examination of distributed systems architectures and the provision of control within these architectures. It is important to avoid prescriptive and often unsuitable solutions to issues such as migration, concurrency and failure. Rather, both the mechanisms and policies of distribution should be tailored more closely to the demands of group working. This raises some fundamental and, as yet, unresolved questions:-

i) what are the most suitable *mechanisms* to support group working,
ii) what are the appropriate control *policies* for CSCW, and
iii) how are cooperation and control *represented* in CSCW systems?

A solution to these problems will require a detailed understanding of *both* the behaviour of distributed systems and the behaviour of interacting user groups. This problem therefore illustrates the inherently cross-disciplinary nature of CSCW research. The problem compounded further by the fact that existing distributed systems already provide adequate support for a range of applications. It is important that distributed systems continue to support these applications and any mechanisms for supporting CSCW systems need to smoothly integrate with these existing distributed applications.

7. References

ANSA (1989): *ANSA: An Engineer's Introduction*, Release TR.03.02, Architecture Projects Management Limited. November 1989.

Armstrong, M (1990): "Management Processes and Functions", in Armstrong, M., Farnham, D. (eds): *Management Studies Series*, ISBN 0 85292 438 0.

Bannon, L., Schmidt, K (1991): "CSCW: Four Characters in Search of Context", in J.M. Bowers and S.D. Benford (eds): *Studies in Computer Supported Cooperative Work. Theory, Practice and Design*, North-Holland, Amsterdam, 1991, pp 3-17.

Birman, K., and K. Marzullo.(1989): "ISIS and the META Project." *Sun Technology* No.: Summer, Pages: 90-104.

Conklin, J (1987): "gIBIS: A Hypertext Tool for Team Design Deliberation", *Proceeding of Hypertext 87*, November 1987,pp 247-251.

Crowley, T., Milazzo, P., Baker E., Forsdick H., Tomlinson R.(1990):"MMConf: An Infrastructure for Building Shared Multimedia Applications", *in proceedings of CSCW* 90, Los Angeles, CA, October 7-10 1990, ACM press , ISBN 0-89791-402-3.

Danielson, T., Panoke-Babatz, U., et al. (1986): "The AMIGO project: Advanced Group Communication Model for Computer-based Communication Environment", *in proceedings of CSCW 86*,Austin,Texas,December 1986.

De Cindio, F., De Michelis, G., et al (1986): " CHAOS as a Coordinating Technology", *in proceedings of CSCW 86*, Austin, Texas, December 1986.

Ellis, C.A., Gibbs.S.J. (1989): "Concurrency Control in Groupware Systems." *ACM SIGMOD International Conference on the Management of Data*, SIGMOD Record, Pages: 399-407.

Hahn, U., Jarke, M., Kreplin, K.et al.(1991): "CoAuthor: A Hypermedia Group Authoring Environment", in J.M. Bowers and S.D. Benford (eds): *Studies in Computer Supported Cooperative Work. Theory, Practice and Design*, North-Holland, 1991, pp 79-100.

Hopper, A. (1990): "Pandora - An Experimental System for Multimedia Applications", *ACM Operating Systems Review*, Vol. 24, No. 2, April 1990.

Jones, M.B., and R.F. Rashid. (1986): "Mach and Matchmaker: Kernel and Language Support for Object-Oriented Distributed Systems." *Proceedings of the Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA '86):*, Portland, Oregon, 1986. Editor: N. Meyrowitz, Special Issue of ACM SIGPLAN Notices, Vol: 21, Pages: 67-77.

Kohler, W.H (1981): "A Survey of Techniques for Synchronisation and Recovery in Decentralsied Computer Systems", *ACM Computer Surveys*, Vol. 13, No. 2, June 1981.

Kraemer, K L, Kling, J L (1988): "Computer Based Systems for Cooperative Work and Group Decision Making" , *ACM Computing Surveys*, Vol 20, No 2, June 1988.

Lauwers, J.C., Lantz, K.A. (1990): "Collaboration Awareness in Support of Collaboration Transparency: Requirements for the Next Generation of Shared Window Systems", *Proceedings of CHI '90* Seattle, Washington April 1-5, 1990, ACM press, ISBN-0-201-50932-6.

Malone, T W, Lai, K (1988): "Object Lens: A Spreadsheet for Cooperative Work", in *proceedings of CSCW'88*, Portland, Oregon, September 1988.

Mullender, S.J., and A.S. Tanenbaum. (1986): "The Design of a Capability-Based Distributed Operating System." *The Computer Journal* Vol: 29 No.: 4, pp 289-299.

Rodden T., Sommerville I. (1991): "Building Conversations using Mailtrays", in J.M. Bowers and S.D. Benford (eds): *Studies in Computer Supported Cooperative Work. Theory, Practice and Design*, North-Holland, Amsterdam, 1991, pp 79-100.

Sarin, S., Grief, I.(1985): "Computer-Based Real time Conferencing Systems", *IEEE Computer* October 1985, pp 33- 45.

Schmidt, K. (1989): "Cooperative Work: A Conceptual Framework", FCI Publication #89-1, The Informatics Centre of the Danish Trade Union Movement, June 1989, ISBN 87-89369-00-9.

Sheperd A, Mayer N., Kuchinsky A. (1990): "Strudel- An Extensible Electronic Conversation Toolkit", *in proceedings of CSCW 90*, Los Angeles, CA, October 7-10 1990, ACM press , ISBN 0-89791-402-3.

Shoch, J.F., Y.K. Dalal, D.D. Redell, and R.C. Crane.(1982): "Evolution of the Ethernet Local Computer Network." *IEEE Computer*, August 1982. Pages: 10-26.

Skarra, A.H. (1988): "Concurrency Control for Cooperating Transactions in an Object-Oriented Database." *Proceedings of the ACM SIGPLAN Workshop on Object-Based Concurrent Programming*, San Diego, September. Editor: Gul Agha, Peter Wegner and Akinori Yonezawa, SIGPLAN Notices, Pages: 145-147.

Spector, A. (1989):"Distributed Transaction Processing Facilities", in Mullender, S. (ed):, *Distributed Systems,* Addison-Wesley, New York, 1989.

Stefik M., Bobrow D.G., et al. (1987b): "WYSIWIS Revised: Early Experiences with Multiuser Interfaces", *ACM transactions. on Office Information Systems*, Vol 5, No 2, April 1987, pp 147-168.

Stefik M., Foster G., et al. (1987a): "Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings", *Communications of the ACM* Vol 30, No 1, January 1987.

Tanenbaum, A.S., and R.V. Renesse. (1985): "Distributed Operating Systems." *ACM Computer Surveys* Vol: 17 No.: 4, December 1985, Pages: 419-470.

Watabe K., Sakata S, Maeno K et al. (1990): ' Distributed Multiparty Desktop Conferencing System: MERMAID', *in proceedings of CSCW 90,* Los Angeles, CA, October 7-10 1990, ACM press , ISBN 0-89791-402-3.

Wilbur S.B., Young R.E.(1988): "The COSMOS Project : A Multi-Disciplinary Approach to Design of Computer Supported Group Working", in R. Speth(ed): *EUTECO 88: Research into Networks and Distributed Applications,* Vienna, Austria, April 20-22,1988.

Winograd T. (1987): "A Language/Action Perspective on the Design of Cooperative Work", Stanford University Department of Computer Science Technical Report, STAN-CS-87-1158.