COOPERATION AND CONFIGURATION WITHIN DISTRIBUTED SYSTEMS MANAGEMENT

G. Dean, D. Hutchison, T. Rodden, I. Sommerville

1. INTRODUCTION

The convergence of workstation technology and local area networks has heralded a change in the nature of systems management. Previously, systems managers were concerned with the maintenance and management of predominantly centralised systems. These systems often had little or no dependency on external system structures. However, the ubiquitous nature of modern personal computers and the widespread acceptance of networking technology has seen systems managers become increasingly concerned with managing systems that are distributed across a number of workstations.

This move toward distributed systems management combined with the increase in the number of machines to be managed has resulted in the need for teams of managers to work closely together to support systems and applications distributed across an organisation. A central aspect of distributed systems management is systems configuration and the associated reconfiguration of the system in response to change. It is already recognised that configuration management of this form requires the cooperation and coordination of a number of systems managers with responsibility resulting from a process of negotiation [1]. However, we would argue that the whole process of distributed systems management is a cooperative one consisting of a variety of independent activities, some of which are open to systems support.

Research into the area of Computer Supported Cooperative Work (CSCW) has examined different forms of communication and cooperation. One of the main themes of this work has been the development of tools which actively support various forms of cooperation among user groups. This paper examines the nature of distributed systems management, highlighting the various forms of cooperation involved, and suggests techniques and tools to support this cooperation.

The paper is structured as follows. Section 2 introduces cooperative systems and highlights the various forms of CSCW system which have emerged to date. Section 3 examines the cooperative nature of distributed systems management by highlighting the various cooperative activities of systems managers which are open to computer support. The role of a shared system model within distributed systems management is examined in section 4 and a configuration language which allows the structure of distributed systems to be modelled is introduced. Finally, section 5 identifies the various tools which we are developing to support cooperation in distributed systems management and the architecture used to integrate them.

2. COOPERATIVE SYSTEMS MANAGEMENT

It is our conjecture that the activities surrounding distributed systems management are cooperative in nature and are open to systems support. The support of groups of users working together on cooperative endeavours has been a growing area of research over the last decade and has seen the emergence of CSCW together with various cooperative (or groupware) systems.

G. Dean, D. Hutchison, T. Rodden and I. Sommerville are all with the Computing Department, Lancaster University, Lancaster LA1 4YR, UK.

The term CSCW was coined by Greif and Cashman in 1984 [2] as a shorthand way of referring to the interests of a number of researchers involved in the use of computers to support user groups. The area has evolved over the last eight years to combine the understanding of the nature of group working with the enabling technologies of computer networking, systems support and applications. Interested readers are referred to [3,4] for more details of groupware and CSCW systems.

Four general classes of cooperative system have emerged from CSCW research activities:

(1) MESSAGE SYSTEMS
Message systems are descendants of early electronic mail programs which allowed a user using a central machine to send textual messages to other users on the same machine. As wide area networks designed to support computer communication became more widespread [5], electronic mail systems increased in complexity and functionality.

This development resulted in the formation of a Message Handling System (MHS) model described in the CCITT-X.400 series of standards documents [6]. Each message system makes use of a particular message format to transfer information (a message format is often defined as a part of most message standards). Structured message systems are based on the principle of extending the amount of machine processible semantic information available by adding structure to the existing message formats. Systems of this class include the COSMOS [7] and AMIGO [8] projects and the Object Lens [9], Strudel [10] and ISM [11] systems.

(2) COMPUTER CONFERENCING
Computer conferencing systems are also a development of the original electronic mail programs. However, structure is imposed in terms of how messages are grouped. A typical computer conferencing system consists of a number of groups called conferences, each of which has a set of members and a sequence of messages. Conferences are often arranged so that they individually address a single topic. A user subscribes to those conferences which are of interest. Usually, the system stores information about how far each conference member has read. This information is normally held along with conference messages in one central database rather than the individual mailbox approach used in messaging systems. Examples of this class of systems include Notepad [12] and COM [13].

The development of reliable high speed communications has lead to the emergence of new *real-time conferencing* systems such as RTCAL [14] which allow conference members to communicate in real-time. *Multimedia or Desktop conferencing* systems such as Rapport [15] and the MERMAID Conferencing system [16] represent the introduction of multimedia technology into conferencing systems. Such systems integrate many forms of media including audio, text and video.

(3) CO-AUTHORING AND ARGUMENTATION SYSTEMS
Co-Authoring and argumentation systems form a general class of system which aim to support and represent the negotiation and argumentation involved in group working. The cooperative authoring of documents is demonstrative of this class of cooperation where the final generation of a document represents the product of a process of negotiation between authors. This class of system is characterised by its widespread use of hypertext technology. Argumentation systems include gIBIS [17] and SIBYL [18], while coAuthoring systems include Quilt [19] and CoAuthor [20].

(4) MEETING ROOMS
Meeting room systems provide computer support for face to face meetings. A typical automated face to face meeting room consists of a conference room furnished with a large screen, projector and a network of computers. Examples of meetings rooms include the CoLab [21] and the Planning Laboratory at the University of Arizona [22].

The cooperative systems highlighted above support various forms of cooperation. These forms are to be found in the activities of distributed systems management. This paper examines the nature of distributed systems management highlighting the cooperation involved and indicating appropriate forms of systems support where applicable.

3. DISTRIBUTED SYSTEMS MANAGEMENT

Distributed systems management researchers, as in many new areas of research, are currently debating both techniques and terminology. This problem is compounded in the case of distributed systems management by the varied history of researchers within the community and the complex nature of key concepts such as management itself. To limit confusion and to emphasise our focus on the cooperative process supporting systems management we adopt the following working definition for distributed systems management:

> *"Distributed systems management is the cooperative process by which a group administers and controls the facilities and services provided by a distributed system."*

A definition of this form begs the question, who exactly manages a system, particularly given the emphasis on the role of the group in the management process. It is our belief that systems management is a potentially open management process where each user of a computing system acts as a system manager over some set of computing resources. Obviously, some users will have a wider range of management tasks to perform and will be responsible for a greater set of resources.

3.1 A Testbed Managed System

The ambiguity involved in distributed systems management reflects many of the characteristics of cooperative systems, and previous research [2] has demonstrated the importance of the examination and investigation of current working practice within system development. Given this impetus our work has been based on the informal examination of current working practice in the management of our local campus network.

Our campus set-up consists of a number of distinct Unix systems connected to a backbone campus Ethernet (see Figure 1). It is a reasonable conjecture that future distributed systems consisting of geographically dispersed LANs joined by high-bandwidth WANs will be comparable with our current campus network. Each of the individual LANs has a substantial system configuration. For example, the Computing Department has approximately 40 workstations connected via two local Ethernets.

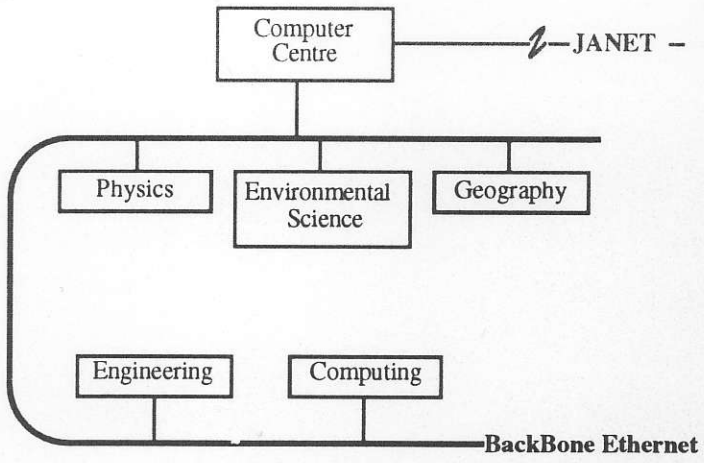

Figure 1: The Testbed Campus Network

Each of the departmental domains within the campus has an "official" systems manager responsible for managing that portion of the system. In the case of the Computer Centre, all

computing system management is the responsibility of a team of managers. For historical reasons management of the Engineering domain has been the responsibility of the Computing Department. However, Engineering is often considered as a separate entity and cannot readily be viewed as a sub-domain of Computing.

Initial investigation of our local system has suggested that one of the most significant problems for our systems managers is the maintenance of system configurations. Currently very few tools exist for recording and maintaining system configurations and most information is kept informally by individual system managers. In addition, a diverse range of systems activities are evident in distributed systems management, and many of these activities are cooperative in nature.

3.2 Supporting Management Activities

A set of activities, often independent of each other, constitutes the process of systems management. To discover some of the activities involved within systems management we have worked alongside systems administrators and other users of the system. This informal study has highlighted that even the simplest of management tasks can sometimes involve the need for cooperation amongst the participants involved. For example, consider the situation where users wish to share some configuration information, as is often the case in shared applications. A common solution is that one user grants access to a personal file containing the configuration information to a number of colleagues. Unfortunately, it is equally common for the original user to forget this some months later and delete the file, causing confusion amongst the other users who share the configuration information.

Systems management involves a wide range of cooperative activities which occur throughout the managed process in an often dynamic and unstructured way. However, the support required for cooperative management activities can be considered under the following five broad categories:

(1) SUPPORT FOR CONFIGURATION

Configuration control is a well recognised problem in distributed systems management [23,24]. At first sight it would appear that this problem is compounded in the situation where management is performed as a group activity. However, by adopting an approach where support is provided for the recording and maintenance of configuration information, many of the problems associated with configuration control can be manually resolved by a group.

This approach of semi-automation where only those parts of an activity most amenable to computer support are automated is characteristic of many of the approaches adopted by CSCW systems [25]. Tools are required to support the recording and maintenance of configuration rather than enforce rigid configuration control policies. These tools should include a configuration browser (see section 5) which will allow configuration and dependency information to be recorded and examined. Cooperative browsers of this form can provide editing facilities akin to those found in co-authoring systems. Other tools may include a simple consistency checker which will examine recorded information to highlight irregularities. However, we envisage the correction of these irregularities will often be undertaken manually by a management team.

(2) SUPPORT FOR QUERIES AND REPORTING

A substantial part of a systems manager's time is taken up with responding to queries and messages concerning problems with the system [26]. Traditionally, the majority of queries were dealt with by a single systems manager who understood and controlled a single system. However, the current reality is that most systems are too complex, particularly when considered in conjunction with the applications they support, to be understood by a single administrator, given that administrators often rely on the knowledge of particular application experts. Unfortunately, little of this acquired knowledge is recorded, and systems managers or users often need to learn new expertise from others when job roles change or individuals move location. Multiuser hypertext systems such as Answer Garden [27] allow knowledge of this form to be recorded and shared by a group of users.

4. MANAGEMENT VIA A SHARED SYSTEM MODEL

The distinct aim of our work is to provide support for the wide variety of cooperative activities involved in distributed systems management. The provision of tools to achieve this aim is greatly aided by the overall focus of the cooperative tasks involved. In general, managers of distributed systems cooperate about and via a shared artifact, "the system", which provides a focus for much of their work. This general style of work allows us clearly to separate the attributes of the system being managed (the system model) and the activities of management (the management process), as illustrated in Figure 2.

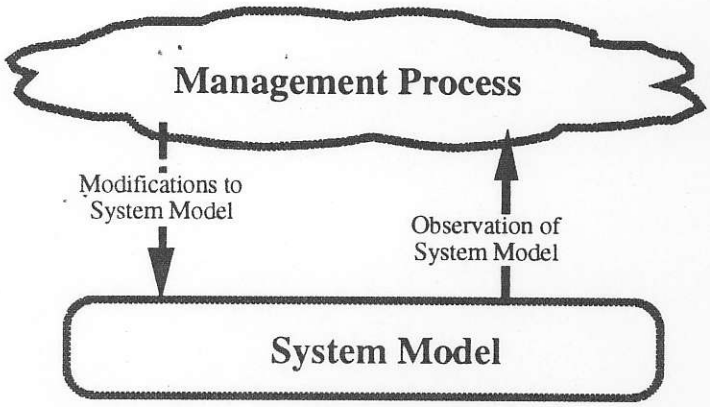

Figure 2: Separating Management from the System Model

While the focus of our work is on the support of the management process through the development of cooperative tools, the system model has a crucial role to play. A model of the system provides a central point of integration for many of the tools being developed in the project. This allows configuration information to be readily recorded and accessed by a number of system managers. In addition, the shared system model provides the focus for the cooperation involved in distributed systems management. The approached we have adopted is to augment the concepts of cooperative authoring systems by providing additional specialist tools to support specific aspects of cooperation amongst a number of managers.

4.1 Different Views and Perspectives

The general system architecture centres on a number of systems managers simultaneously accessing a shared system model (see Figure 3). However, it should be stressed that while these systems managers share this information they may have different perspectives on it. For example, considering a simple user directory, the user to whom the directory belongs would be interested in the details of the files and directories it contains (e.g, name, size, access rights etc.). In contrast, a system administrator in charge of enforcing a strict policy of space usage would be interested only in who owns the directory and how much space the directory consumes. We characterise these alternative perspectives as different managers' views on the same shared entity within the system model.

Each system manager exploits different views of the system's objects and configurations. Views are characterised by the system entities a manager can see and the attributes of these entities that they may access. These different views are controlled by individual manager views on the shared model and are displayed to systems managers graphically (see Figure 4). Each manager can customise this view and will see the system differently depending on which role his view is currently adopting.

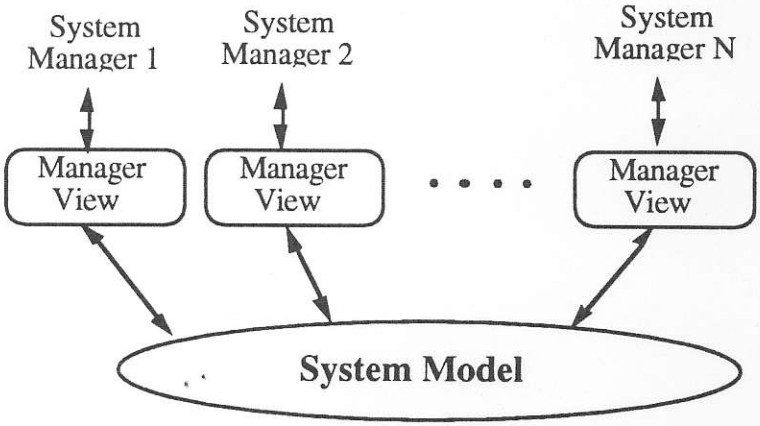

Figure 3: General Framework for Systems Management

The separation of these different perspectives allows us to control access to the shared system model. In addition, the deconstruction of certain features of the system allows the model to concentrate on the configuration of the systems and the interdependencies found within it. Thus, rather than embed much of the semantics of the management process, our systems model contains only information on the structure of the systems and the interdependencies within it. Additional management information, such as expertise and policy, are held within appropriate tools. This decision is reflected within the modelling language used in the project.

4.2 The Modelling Language

The configuration and structure of the distributed system being modelled is described in a language called SySL [31, 24] derived from module interconnection languages such as MIL75 [32] and INTERCOL [33]. SySL allows the configuration of hardware and software systems to be described at varying levels of abstraction.

Systems are described in SySL in terms of abstract entities, which yields many of the benefits of structuring provided by a host of object oriented approaches. Each SySL entity has an associated definition which defines the structure of the entity. For example, a SySL structure to describe a machine and a structure for a network would be of the form:

```
structure MACHINE is
        Name
        Supplier
        Monitor
        Keyboard
        Processor
        Disk
        Memory
        Operating System
end structure
```

```
structure NETWORK is
        Name
        Topology
        Speed
        Medium
end structure
```

These attributes can have values associated with them which describe details specific to each structure. In addition, SySL has been augmented to include a range of relations which can be exploited to describe interdependencies within the system. The addition of these relation constructs was found to be essential for modelling complex distributed systems. For example, consider the campus arrangement described earlier, this requires the definition of a relationship to describe connections to this network structure. The relationship mechanism added to SySL defines a relationship name and the domain and range of the relationship. The general form of the relationship construct is:

```
relation <<NAME>> is
    domain <<CLASSLIST>>
    range  << CLASS LIST >>
end  relation
```

A *connect_to* relationship can be defined as

```
relation CONNECT_TO  is
    domain NETWORK, MACHINE
    range NETWORK
end  relation
```

This relationship can be used to describe the connectivity of machines and networks within the system. For example, the network in Figure 1 would have instantiated SySL objects to describe each of the sub-networks connected to the backbone. These would be instantiated in SySL by the following statement :

```
class NETWORK is
  {backbone, computing,physics, centre, geography, environmental_science, engineering}
```

The SySL object describing the backbone Ethernet would be of the form

```
component backbone : NETWORK is
            Name =>"Lancaster Campus Backbone"
            Topology => "Ethernet Bus"
            Speed => "10Mbits/sec"
            Medium => "Co-axial"
end  component
```

Each of the networks instantiated would have a corresponding component definition. Connections between the networks would be shown as SySL relations as follows:

physics CONNECT_TO *backbone*
computing CONNECT_TO *backbone*
geography CONNECT_TO *backbone*
environmental_science CONNECT_TO *backbone*
engineering CONNECT_TO *backbone*
centre CONNECT_TO *backbone*

Models of this form are presented to managers graphically via a browsing tool (see Figure 4). Depending on the particular role they are playing at any one time, managers can control which relationships and SySL objects they can view. System entities and relations are presented to them graphically, and users can choose which entities and relations are of relevance to their particular activity. Only those entities are displayed to users.

The browser represents one of a set of cooperative tools which are being developed to support the cooperative aspects of distributed systems management within the project. The other tools identified to date and the general architecture to support these tools are described in the following section.

5. ARCHITECTURE

A range of different tools may be provided to support the different forms of cooperation highlighted in section 4. To date, three tools providing different aspects of cooperation support have been identified and are currently being developed. Each of these tools and the form of support provided is briefly examined below.

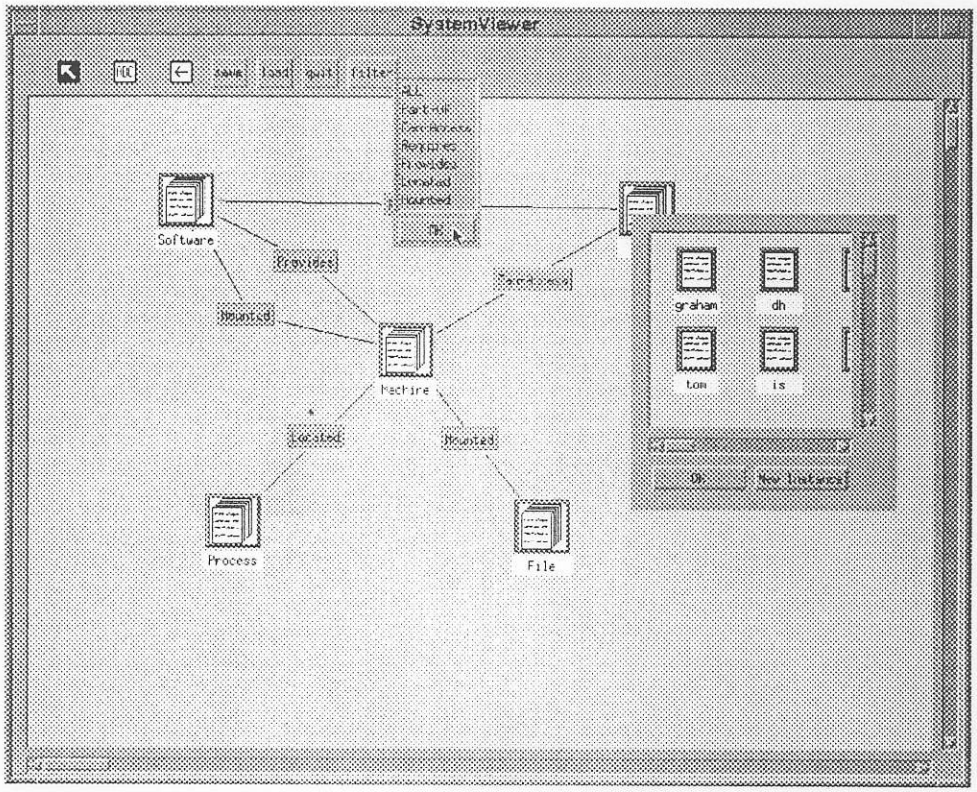

Figure 4: The Initial System Browser

In the previous section we highlighted how the mechanisms used to access and alter a shared system model provide a focus for the different styles of cooperation supported. Consequently a shared system browser plays a central role in our management toolset. It is envisaged that this browser will be the primary means of user interaction and thus of cooperative systems management. A central feature of our approach is the use of "views" upon the systems model. These act both as a filter to alleviate the problems of information overload from the number of system components [34] and as a mechanism to support systems management.

Management roles are supported within the browser by the use of views which designate which SySL objects can be seen and altered when a user is undertaking a particular role. These views are held within the cooperative browser, and managers can move between views depending on which role they are playing at a particular time. For example, a manager playing the role of *smalltalk_manager* would see only those objects associated with managing Smalltalk. However, details of role information including responsibility and access are stored separately within a responsibility and access register.

Two principal modes of cooperation are supported by the browser, namely message oriented cooperation and shared screen cooperation. Shared screen cooperation allow systems views to be shared across workstations. This synchronous sharing allows time critical management to be undertaken by cooperation between managers. In addition, a message based approach to cooperation is exploited to support routine management activities such as error reporting. In message based cooperation the concept of annotation is adopted as an interaction metaphor. Thus users attach comments and queries to system structure which are then passed onto the message system; this routes them to the appropriate user responsible for that component.

The browser also provides the primary means of access to the remaining tools within our toolset, namely the structured message system and the policy rationale system.

In addition to providing routing facilities for the annotations within the shared browser, the message system also provides a direct message system. Users can fill in forms to report errors which are then automatically distributed to the appropriate users. In order to do so, the message system maintains a register of user expertise and interest. These can be stored as simple tuples in a manner akin to the Information Lens [25].

A significant problem in distributed systems management is that the rationale and policy decisions supporting system components are often lost. The policy rationale tool will be based upon an existing general design rationale tool developed at Lancaster [29]. The policy tool is a specialised version of this tool tailored to express the rationale surrounding policy decisions which are attached to different SySL components.

The logical architecture of the current toolset is show below.

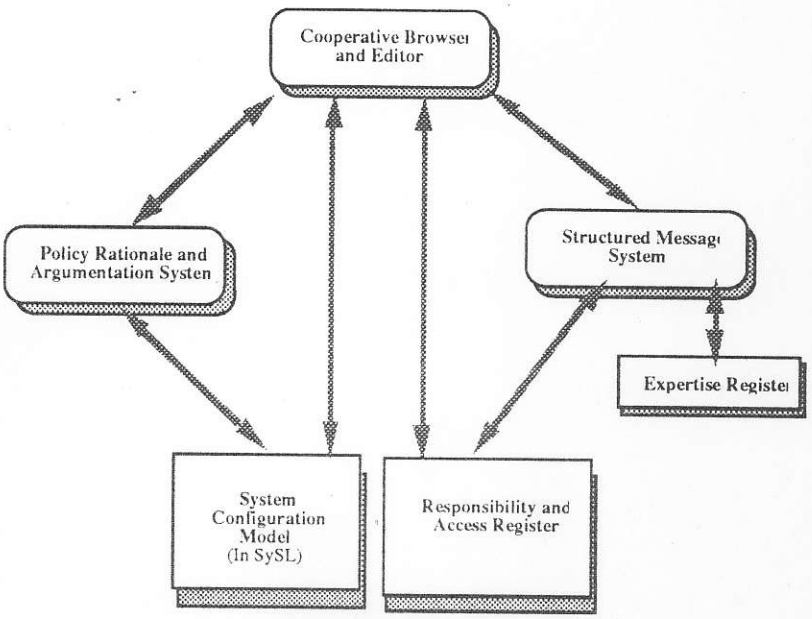

Figure 5: System Toolset Architecture

The cooperative toolset described in this paper is currently under development. To date an initial version of the cooperative browser has been developed. The authors are currently augmenting this browser by developing a version of the structured message system and the design rationale tool which integrates with the system browser.

6. CONCLUSION

This paper has examined the role of cooperation within distributed systems management and has argued that distributed systems management is essentially a cooperative activity. We have described how the dynamic management process may be considered to be separate from a system model in order to highlight the different forms of cooperative support which can be provided. A range of cooperative tools supporting different aspects of distributed system management has been identified and briefly described. These cooperative tools are built around a system model which allows the configuration to be clearly described. Cooperation is supported both through the shared browsing of this model and by communication regarding different components of the model. The toolset is currently under development, and it is planned to test the final system on our local area network.

Acknowledgement

The work reported in this paper is being supported by the UK Science and Engineering Research Council's Specially Promoted Programme in Integrated Multi-Service Communication Networks under grant number GR/F 62674.

References

[1] Moffett, J. , Sloman, M., Twidle, K., "Specifying Discretionary Access Control Policy For Distributed Systems", *Computer Communications*, Vol 13 No 9, 1990.

[2] Irene Greif (ed.): *Computer -Supported Cooperative Work: A book of Readings*, Morgan Kaufman, San Mateo, California.

[3] Ellis, C.A., S.J. Gibbs, and G.L. Rein., "Groupware: Some issues and experiences", *Communications of the ACM* Vol: 34 No.: 1, January, Pages: 38-58.

[4] Rodden T., "A Survey of CSCW Systems", *Interacting with Computers* , Vol 3 No. 2, Dec 1991.

[5 Mortensen E., "Trends in electronic Mail and it's role in office automation", *Electronic Publishing Review* Vol 5 No 4 Dec 1985 pp257-68

[6] CCITT (1987), "Draft Recommendation on Message Handling Systems, X.400, Version 5" , November 1987.

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

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

[9] Malone T.W., Lai K., "Object Lens: A Spreadsheet for Cooperative Work", in *Proceedings of CSCW88*, Portland, Oregon, September 1988.

[10] Sheperd A, Mayer N., Kuchinsky A., "Strudel - An extensible elctronic conversation toolkit", in *Proceedings of CSCW 90*, Los Angeles, CA, October 7-10 1990, ACM press , ISBN 0-89791-402-3.

[11] Rodden T., Sommerville I., "Building conversations using Mailtrays" in *Proceedings of EC-CSCW, the 1st European Conference on CSCW*, Gatwick Hilton September 13th-15th 1989.

[12] Pullinger D.J., "Chit-Chat to Electronic Journals: Computer Conferencing Supports Scientific Communication", *IEEE trans. on Professional Communications*, Vol PC 29, No 1, pp23-29, March 1986.

[13] Palme J., "Survey of Computer Based Message Systems" in B.Shackel (Ed): *,INTERACT-84*.

[14] Sarin S., Greif I., "Computer-Based Real time Conferencing Systems", *IEEE Computer* October 1985, pp 33- 45.

[15] Ahuja S.R., Ensor J.R.,Horn D.N., "The Rapport Mulimedia Conferencing system", in Allen R.B.(ed): *COIS88 Proceeding conference on Office Information Systems*, March 23-25,1988, Palo Alto, California.

[16] Watabe K., Sakata S, Maeno K., et al "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.

[17] Conklin J., "gIBIS: A Hypertext Tool for Team Design Deliberation", in *Proceedings of Hypertext 87,* November 1987,pp 247-251.

[18] Lee, J., "SIBYL: A Tool for Managing Group Decision Rationale" in *Proceedings of CSCW'90*,Los Angeles, CA, October 7-10 1990, ACM press , ISBN 0-89791-402-3.

[19] Fish, R.S., Kraut, R.E., Leland, M.D., Cohen, M., "Quilt: A colloborative tool for cooperative writing", in Allen R.B.(ed): *COIS88 Proceeding conference on Office Information Systems*, March 23-25,1988, Palo Alto, California.

[20] Hahn, U., Jarke, M., Kreplin, K., et al "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.

[21] Stefik M., Foster G., et al "Beyond the chalkboard: computer support for collaboration and problem solving in Meetings", *Communications of the ACM*,Vol 30, No 1, January 1987.

[22] Applegate L., Knonsynski, Nunamaker J.F., "A Group Decision Support System for Idea Generation and Issue analysis in Organisational Planning", in *Proceedings of CSCW 86*, Austin , Texas, December 1986.

[23] Kramer, J., Magee, J., "A Model for Change Management", *Distributed Computer Systems in the 1990s Workshop*, Hong Kong, Sept. 1988, IEEE Computer Society.

[24] Sommerville, I., Thomson, R., "Configuration Specification Using a System Structure Language", *Proc. Int. Workshop on Configurable Distributed Systems*, London, 1992.

[25] Brobst S. A., Malone, T., et al "Toward intelligent message routing systems" in R Uhlig (Ed.): *Computer Message systems -85. Proc. of the 2nd International Symposium on Computer Message Systems*. North Holland, Amsterdam, 1986.

[26] Kolstad, R., "What do System Administrators Really do?", *Communications of the ACM*, December 1991, Vol. 34, No. 12.

[27] Ackerman M.S., Malone T.W., "Answer Garden: A tool for Growing Organisational Memory" in Lochovsky F.H.(ed): *COIS90 Proceeding conference on Office Information Systems*, April 25-27, 1990 , Cambridge, Mass.

[28] Moffett J.D. & Sloman M.S., "The Representation of Policies as System Objects", *Proceedings of the Conference on Organizational ComputerSystems (COCS'91) Atlanta*, GA, 58 November 1991, in SIGOIS Bulletin vol12, nos 2 & 3, pp 171-184.

[29] Haddley, N., Sommerville, I., "Integrated Support for Systems Design", *Software Engineering Journal*, Vol. 5, No. 6, November 1990, pp. 331-338.

[30] Rein, G. L. and Ellis, C. A., "rIBIS: A real-time group hypertext system", *Int J Man Machine Studies*, 34 (3), p349-368, March. In the special edition on CSCW & Groupware. Republished in Greenberg, 1991.

[31] Thomson, R., Sommerville, I., "An Approach to the Support of Software Evolution", *The Computer Journal*, Vol. 32, No 5, 1989.

[32] DeRemer, F., Kron, H.H., "Programming-in-the-large versus Programming-in-the-small", *IEEE Transactions on Software Engineering*, Vol. SE-2, No. 2, June 1976.

[33] Tichy, W.F., "Software Development Control Base on System Structure Description", PhD Thesis, Department of Computer Science, Carnegie-Mellon University, 1980.

[34] Wang, B., "An Architecture for Domain Based Distributed Systems Management", PhD Thesis, Lancaster University, 1989.