Available online at www.sciencedirect.com

ScienceDirect

European Journal of Operational Research 177 (2007) 1385–1399

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/ejor

Information management in distributed collaborative systems: The case of collaboration studio

Francisco Antunes [a], Paulo Melo [b], João Paulo Costa [b,*]

[a] *Management and Economics Department, Beira Interior University, 6200-209 Covilhã, Portugal*
[b] *Economics Faculty, Coimbra University, 3004-512 Coimbra, Portugal*

Available online 23 June 2005

Abstract

This paper presents the Collaboration Studio (CS) system, its argumentation and data-structuring models and gives some insights for dealing with information divergence. The system allows discussions among a group of participants that includes a coordinator. The working mechanisms implemented within CS are perfectly transparent to the user, hiding implementation details, giving an appealing and user-friendly environment, and so users do not have to worry about patterns of data distribution, or the details of distribution management. CS shares characteristics with other collaboration computational tools, such as synchronous and asynchronous support and both group working spaces and a local working space. However, its main purpose differs in that, instead of trying to achieve a single document as the outcome of the joint work of several users, CS aims to achieve a broader objective, which is to register (and to demonstrate) the "path" used to obtain certain knowledge.
© 2005 Elsevier B.V. All rights reserved.

Keywords: Collaborative systems; Groupware; Divergence management; Knowledge management

1. Introduction

Information systems and information technology are leading to a global organizational change. Some of the more dramatic examples are those where entrepreneurs have created radical new business structures supported by the power of information technology and the arrival of new networked or virtual organizations (Boddy et al., 2002, p. 167). Information becomes the organization's main power source, its main fuel and most important resource or input (Chiavenato, 2000), and the knowledge workers (Drucker, 1988) gain vital importance, as they constitute a scarce resource in the economy, moving freely within and between organizations, presenting new challenges, such as knowledge management and organizational memory (Conklin, 2001).

* Corresponding author.
E-mail addresses: fantunes@ubista.ubi.pt (F. Antunes), pmelo@fe.uc.pt (P. Melo), jpaulo@fe.uc.pt (J.P. Costa).

0377-2217/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2005.04.010

However, it is known that organization knowledge does not lie within some sort of technological information storage, the importance of humans in the process is fundamental. Nevertheless, they cannot be simply detached. Technological artefacts (such as collaborative systems, databases, knowledge bases, etc.) can be useful for capturing explicit knowledge (for more details on explicit and tacit knowledge see Nonaka and Takeuchi, 1995) and maintaining it in proper repositories, which should be available to all the appropriate people in an organization (Courtney et al., 2000). The major drawback is that less expensive data storage mechanisms in conjunction with increased processing capabilities and networking can generate information overload (Nunamaker et al., 2001), hampering information retrieval appropriate to teams' needs. Mere file or document storage (usually known as formal knowledge objects, Conklin, 2001) becomes inefficient. These records are kept in filing cabinets, folders, in the memory of personal computers, on computer discs and in various databases using a variety of information technologies. The problem with this knowledge is that it is easily lost if the information is stored in a manner that makes it difficult to retrieve, and so this knowledge can become implicit. The storing of information on the hard disk of personal computers by individual employees is a classic example of information that may well be totally lost when the employees leave the organization (Gunnlaugsdottir, 2003). Another level of information (meta-information or information about information) is required to structure and categorize the information repositories for an enhanced utilization of the information objects.

Technically, there are exciting possibilities for the use of hypertext, groupware, intelligent agents, neural networks, advanced search techniques and other computational technologies to provide "relevance retrieval" access in large databases—retrieval which respects the meaning relationships among the stored items (Conklin, 2001). Data manipulation and knowledge mapping representational tools are often designed to create meta-information according to the organization's needs. Such tools include basic office tools, such as spreadsheets and outliners, annotation tools (Ginsburg and Kambil, 1999), information retrieval systems such as collaborative filtering and browsing (Romano et al., 1999a; Hilmer and Dennis, 2000), or "knowledge mapping" software.[1] Sharing and exchanging knowledge will not be a fruitful process unless it has been structured (Gunnlaugsdottir, 2003).

The globalisation process and the spread of enterprise managers all over the world has led not only to new organization structures (Drucker, 1988), but also to virtual project management and associated knowledge management problems (for further details, see Katzy et al., 2000), and to the increasing need for distributed collaboration and decision-making support systems, driving a move towards business teams distributed along the dimensions of space, time, and technology. Large groups using group support systems (GSS) appear to benefit more than smaller groups (Vreede et al., 2003), and research findings suggest that group support systems can facilitate knowledge acquisition (Kwok et al., 2000), improve decision quality and quantity, enhance participant satisfaction (Dennis et al., 1996), and reduce the cost and length of meetings. All of this helps to boost efficiency and productivity (some clear examples of this can be found in Vreede et al., 2003).

GSS research is moving towards providing anytime/any-place/any-technology support for teams of any size (Romano et al., 1999b). Distributed collaboration systems demand specific tools to support both space and time distributed groups. Different tools are needed from the ones used in regular synchronous meetings, whether they are held face-to-face (F2F), or not. During an asynchronous meeting, group members do not interact at the same time, even if they are at the same location. This line of work is extremely important to organizations with dispersed managers and workers, and vital to virtual project management. However, some problems with lower group satisfaction, decreased socialization among the group, and meeting coordination/facilitation problems have been pointed out as process drawbacks (Vreede

[1] See http://www.thebrain.com, for example.

et al., 2003; Lowry et al., 2002; Massey et al., 2002; Dufner et al., 2002).

Research has presented many examples of systems that deal with this kind of working environment (for a detailed review, see Bafoutsou and Mentzas, 2002). Yet such systems normally use an active connection to a broader collaboration network (some kind of private network, or the internet, for example). Despite the fact that little research has been done in this area, it is the authors' opinion that embedding new features that can enhance asynchronous usage allows work to proceed even without an active connection to the physical network, to the advantage of worker mobility. Therefore, groupware/collaboration tools may well bridge the gap by combining the two strategies. They create a platform where people can share codified, systematic knowledge and at the same time co-operate creatively in an organized environment. These systems even offer the opportunity to participate in discussions, to present and challenge ideas, and to vote anonymously, thus creating an environment where the participants can be frank and open without having to worry about any consequences (Abell and Oxbrow, 2001, p. 52). This concern was embedded in the building of an argumentation model (as presented in Section 2) and the collaboration tool to implement it (the Collaboration Studio).

The Collaboration Studio (CS) argumentation model deals with four collaboration elements or document types, whose information and relationship network constitute a collaboration discussion. That set of documents consists of: a theme—the discussion generic description; attributes—used to define certain aspects that may require a more specific characterization within the theme; attribute characterizations—implemented on simple sentences, or on a more complex structure like a data matrix; and messages—whose objective is to comment, question or give some sort of explanation, and not to establish attribute characterizations. Each theme can contain multiple attributes, which in turn, can contain other attributes. Every attribute can also accept multiple characterizations.

The CS software system (described in Section 3) shares characteristics with other collaboration tools, such as synchronous and asynchronous support and both group working spaces and a local working space. However, its main purpose differs substantially from those tools due to the fact that instead of trying to achieve a sole document as the outcome of the joint work of several users, CS has a broader objective, which is to register (and to demonstrate) the "path" used to obtain a certain knowledge, in such a way that, when a collaboration document is finally drawn up, users are able to retrace all the steps leading to its achievement.

CS as an integral system comprises two elements: (i) a central memory that ensures global information persistence and server tasks; (ii) different workstations that ensure local information persistence. Central memory and workstations interact through a distributed architecture. The system allows discussions among a group of participants that include a coordinator. The discussion group is created by means of submissions, which are sent to the current coordinator for approval. The working mechanisms implemented within CS are perfectly transparent to the user, with implementation details remaining hidden, giving an appealing and user-friendly environment. This means that users do not have to worry about patterns of data distribution, or details of distribution management.

The system field-tests (Guerra, 2004) led to some important insights that were the major motivation for directing research towards divergence analysis in collaborative systems. The idea is to find a robust, but flexible, way to remove the "locking" approach, when multiple users asynchronously (with or without connection to a central memory) access and manipulate information objects, using divergence management, without losing consistency and structuring capabilities.

Though there are a number of different approaches to deal with different information versions or document replicates, none of them aims directly at knowledge management or organizational memory creation (Section 4). Our purpose is therefore to extend the previous research on divergence by embedding data structuring capabilities and an information linking structure with a high degree of expressiveness, in order to fill in the gap between divergence support features and divergence capabilities to help/improve knowledge creation and retrieval. We argue that this line of work can especially benefit decision agents in retrieving and recognizing the reasons that led users to diverge in their work, helping to better understand past decisions. We therefore aim at bringing closer GDSS to knowledge management theory, using divergence management.

2. Structuring information

The goal of a collaborative discourse structure is to provide a template for a group debate so that the greater part of the discussion can be captured, categorized and classified into meaningful categories that structure their relevance and significance according to the objectives of the discussion and the characteristics of the group (Turoff et al., 1999). This leads to a structured information network that can both promote explicit knowledge transference while also contributing to the knowledge creation process, individual and organizational knowledge acquisition, and the dissemination, sharing and confrontation of knowledge (Serrano and Fialho, 2003). This ability to structure group discussions is directly related to the fact that people do not come up with ideas in complete isolation, and therefore idea generation will benefit from information association. It is not only new information that leads to new ideas, but a recombination of old information can also have a similar result (Vandenbosch et al., 2001).

Tools that provide representational guidance help learners to see patterns, express abstractions in concrete form, and discover new relationships (Suthers, 1999), thereby fostering the process of exposing tacit to explicit knowledge, and enhancing the ability to deal with less structured problems. Moreover, structuring information objects and their interrelationships can also improve the capability to build and develop an effective organizational memory. This does not solely aim to accumulate formal knowledge, but should preserve context information (or informal knowledge), and therefore becomes the "glue" that holds the formal knowledge documents together and preserves their meaning (Conklin, 2001).

Whilst information technology is primarily intended to obtain and process more information, it may also lead to information overload (real or perceived), especially in distributed teams (Franz, 1999). It is the authors' opinion that collaborative discourse structures can also help organizations to filter, browse and retrieve information objects by establishing a relationship network, which would be an important protection against information overload. A multi-criteria decision-making process (MCDM) also requires advanced structuring to identify the problem, to explore alternatives, and to develop and rank the criteria of the problem (Cao and Burstein, 1999); collaboration features therefore seem right for guiding the problem-structuring process to the point where MCDM models can be loaded and used to evaluate different alternatives.

2.1. Argumentation models

Changing work structures into virtual project management and distributed or global teaming leads to new technological approaches that go beyond classic decision support systems. They mix systems such as Computer Supported Collaborative Work—CSCW, Group Support Systems—GSS, Computer-Mediated Communication System—CMCS, or Group Decision Support Systems—GDSS. It is hard to establish concrete boundaries for such systems, and so hybrid approaches such as Collaborative Decision Making Systems emerge. These can be defined as interactive computer-based systems, which help a set of decision makers to solve ill-structured problems by working together as a team, where the main objective is to augment the effectiveness of decision groups through interactive information-sharing between the group members and the computer (Karacapilidis et al., 1999).

The need to interpret and reason in relation to knowledge, during a discourse that is a structured conversation with categorized speech acts and procedural rules (Gordon et al., 2001), is substantially addressed by the above-mentioned systems. With that idea in mind, the development of an argumentation or interaction model becomes more important since argumentation theory has shown that discussion-structuring goes beyond the simple question–reply pattern of newsgroups, and that formalizing debates with argumentation models both gives speech acts a logical structure, and a specific grammar (Rinner, 2001). The benefits of these models include structuring a discussion procedure, to make complex contributions clearer, recording, organizing, regulating and coordinating argumentation processes in organizations, and enabling managers to argue about decisions, issues, and goals (Sillince and Saeedi, 1999). An argumentation model allows for better interfaces, and also (and perhaps more importantly) it is a valuable aid for developing flexible software structures, according to organizations' requirements. Such ideas are expressed in (among others) Toulmin's argumentation model (Toulmin, 1958), Questmap software using the IBIS system (Conklin, 2001, na; Rinner, 2001), ZENO argumentation framework (Gordon and Karacapilidis, 1996; Karacapilidis et al., 1999; Gordon et al., 2001), SEPIA (Streitz et al., 1989), and EUCLID (Singh et al., 1995).

According to Sillince and Saeedi (1999), conventional CMCS increase the productivity of meetings through human parallel processing and are particularly successful in those meetings whose purpose is to extract and record ideas from participants. GroupSystems[2] and MeetingWorks[3] are two of a number of such systems. They contain features such as brainstorming, voting, and issue analysis; but argumentation systems go beyond such tasks, improving collaborative reasoning processes, knowledge management, organizational memory and recording the reasoning behind decisions, for later retrieval.

Such issues are the motivation for developing an argumentation model for use within a collaboration tool to deal with the need to structure (and register) distributed collaboration discussions, with the possibility of retrieving the "knowledge path" when achieving desired collaboration results, thus enhancing problem-structuring in decision making. The CS argumentation model (Fig. 1) deals with four collaboration elements or document types (or issues, in IBIS terminology), whose information and relationship network constitute a collaboration discussion. That set of documents consists of: a theme—the discussion generic description; attributes—used to define certain aspects that may require a more specific characterization within the theme; attribute characterizations—implemented on simple sentences, or on a more complex structure like a data matrix; and messages—whose objective is to comment, to question or to give some sort of explanation, but not to establish attribute characterizations. Each theme can contain multiple attributes, which in turn, can contain other attributes. Every attribute can also accept multiple characterizations. Messages can be associated to any document as shown in Fig. 1.

The relationships among the documents may not be directly represented by a hierarchical structure (though it can assume such structure, just like other thread scheme tools such as electronic mail or discussion forums). In fact, by supporting the associations of several documents at the same time it broadens its usefulness for expressing information linking within the information net.

2.2. Data structuring model

GDSS group supporting mechanisms frequently handle a great deal of diverse information. The direct manipulation of that information by the end user will often be difficult and unrewarding (Melo and Costa, 1999). To help limit the need to refer to stored data directly, a semantic level was created over the data stored in the databases. This semantic level acts by encapsulating related information in a "bundle" with a unique identification and some significance. This stops short of the usual object-oriented encapsulation, as no support for procedural knowledge is present. The general DOC model is presented in Fig. 2.

A DOC that is to be explicitly shared between users is transmitted to a central unit or memory, where it is stored and placed on outbound queues, awaiting the connection of their designated users. A DOC that is to be shared, but without an explicitly designated destination user, is stored in a common storage area on the central unit, where the

[2] http://www.groupsystems.com.
[3] http://www.entsol.com.

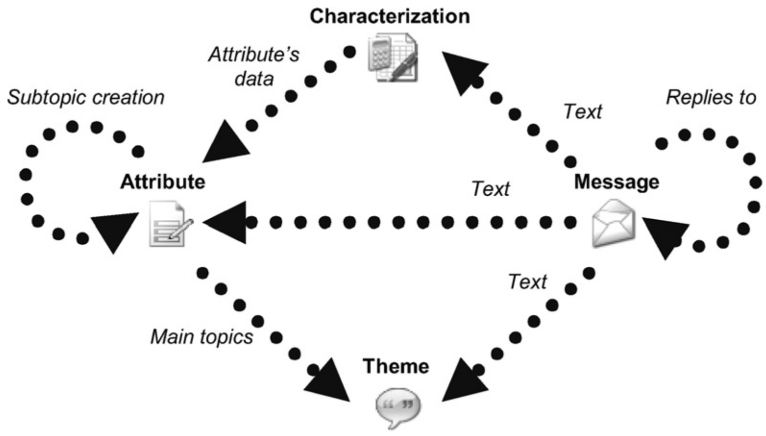

Fig. 1. Collaboration Studio argumentation model.

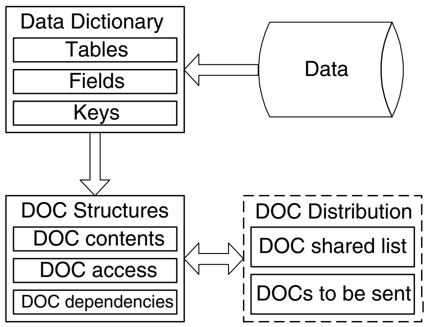

Fig. 2. DOC general model.

users will be able to check it and request it if needed, according to its access permission. The central unit always checks the access permission, as it is understood that a particular workstation may be compromised, and as such should never have access to information its owner is not allowed to possess.

As an example of DOC usage, Fig. 3 presents two DOCs, and related information. DOC1 refers to the data regarding a particular plant expansion project. It contains links to the project authors, and scenario information (including the project duration and the Value Added Tax rate to be used). It also refers another DOC (DOC2, case information associated with that project). The reference of another DOC implies that the information referred to in that DOC may be needed and must be transferred when that DOC is used.

The DOC model was the basis of the development of AGAP—Aid to Groups of Analysis and evaluation of Projects—software system (for more details see Costa et al., 2003). In spite of achieving a large amount of the objectives initially proposed, this approach was not broad enough to cover all the applications aimed at, or to deal with complex information chain management. It was also found that the information model was not suitable for both synchronous control and complex group manipulation procedures, which made information access quite slow (Melo and Costa, 2001, 1999). The DOC hierarchical structure did not prove to be the best solution for coping with more complex interactions, especially where the same information included contributions from several users, since the information involved tended to lose contributors' identities. Automatic actualisation mechanisms within the system were also limited. The model presented conflicting situations regarding access permission that could result in information corruption, due to the fact that each DOC was treated individually, even if it belonged to a broader information structure. The graphical group interaction was not very flexible, because documents were also excessively attached to database persistence.

The problems that this caused with the DOC model directed the research towards distributed object systems analysis, in search of a more reliable and broader system, leading to a new

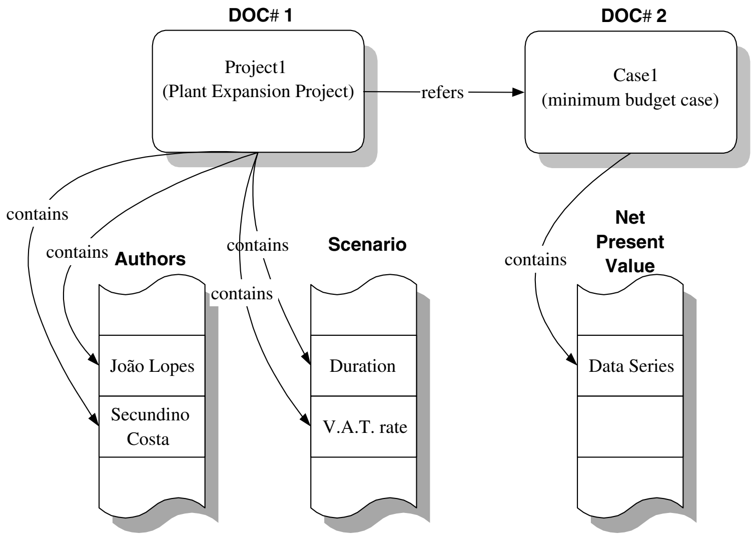

Fig. 3. A DOC usage example.

theoretical model supported by object-oriented programming and a distributed object infrastructure. This approach generated a new information model that uses distributed objects for information storage (Melo and Costa, 2001). Each of these objects, named DOC2 objects, performs information package management and controls its access, retaining common control actions from the DOC model, but replacing its infrastructure. A particular DOC2 object cannot move between workstations, but it can be replicated for those who need it. Distributed objects and their supporting infrastructure are responsible for managing all information access operations. Each DOC2 object allows more complex information than that provided by DOC objects, and may also contain procedural information (Melo and Costa, 1999, 2001). DOC2 persistence, and that of its dependencies, was independently modelled from the real storage mechanism. However, the solution used for a first test of the DOC2 model was based on relational databases (Aleixo, 2000).

The basic DOC2 model ensures object association through a dependency object. Further research (Antunes, 2002), however, eliminated this element by embedding that mechanism in the DOC2 object itself (see Fig. 4). This led to the storage differing from the type established by the original model. With this change a DOC2 object can be used to represent different types of information, including an association, or relationship, between two objects. It also allowed merging in the DOC2 object definition and reservation management methods.

Objects that may exist in a central memory system, or in the workstations, keep the DOC2 infrastructure. However, to increase the system usefulness in both synchronous and asynchronous utilisation, the necessary services to support individual decision-making must be available even when a connection to the central memory system, or to other workstations, is not possible. The proposed DOC2 communication mechanism between different workstations, or between a workstation and the central memory (which ensures global information persistence), is thus the simple calling up of distributed objects. In this way it is possible to use the distributed object infrastructure without having to duplicate it.

The data structuring and argumentation model research together provided the basic elements for a collaborative software system, with a distributed work platform and multi-synchronous support, allowing group-supporting mechanisms from basic interaction to complex model analysis possibility.

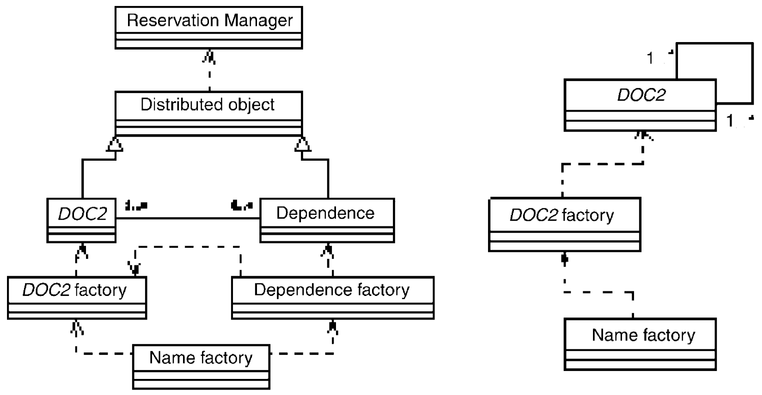

Fig. 4. DOC2: Basic and simplified models (Aleixo, 2000; Antunes, 2002).

3. Collaboration studio

As a collaboration tool, Collaboration Studio[4] (CS) shares characteristics with others, such as synchronous and asynchronous support and both group working spaces and a local working space. However, its main purpose differs substantially from those tools since, instead of trying to achieve a sole document as the outcome of the joint work of several users, CS is targeting a broader objective. It aims to register (and to demonstrate) the "path" used to obtain certain knowledge, in such a way that, when a collaboration document is finally set up, users are able to reconstruct all the steps in its achievement. Though other tools (such as ShrEdit, Dourish, 1995) allow several users to work simultaneously (with the possibility of dispersed locations and different synchronization modes), and consequently have document edition/locking management mechanisms (Dourish and Bellotti, 1992), they are not normally capable of keeping records of all the information produced. This means that information can be deleted at some point of a collaboration discussion (because either there is new information or the previous one is thought to be wrong or obsolete). However, the deleted information could have been an untested hypothesis, a hunch or gut feeling, which, though not relevant to the final document, could have triggered research tasks, or acting guidelines. Being erased, this information becomes useless and can only be retrieved if the users happen to remember it.

The CS, as a complete system, consists of two elements: (i) a central memory that ensures global information persistence and server tasks; (ii) different workstations that ensure local information persistence. Central memory and workstations interact through a distributed architecture supported by CORBA[5] services. When the system is used for the first time, users need to register, a procedure that can be based at any workstation. However, as central services are in charge of user-management features, the registration procedure requires a connection between the workstation and the central memory in order to avoid duplications. The system allows discussions among a group of participants that includes a coordinator. Coordination tasks can be transferred among participants during the discussion process. The discussion group is created by means of submissions that are sent to the current coordinator for approval.

[4] The system is a non-commercial research software prototype. The current version of the system and its source code are both available, free of charge, upon request to the authors.

[5] CORBA—Common Object Request Broker Architecture—The architecture and specifications described in the Common Object Request Broker are aimed at software designers and developers who want to produce applications that comply with OMG standards for the Object Request Broker (ORB). The benefit of compliance is, in general, the ability to produce interoperable applications (which can be of heterogeneous nature) that are based on distributed, interoperating objects.

The working mechanisms implemented within CS are perfectly transparent to the user, with implementation details being hidden, creating an appealing and user-friendly environment, and so users do not have to worry about patterns of data distribution or details of distribution management. Though not a major concern, since CS was designed to be used by small decision groups, scalability is provided by the use of a distributed architecture, relying on a CORBA infrastructure to manage communications within the system, and object-oriented programming (OOP).

The following sub-topics will briefly describe the main features of CS (for more details about the implementation see Antunes et al., 2004).

3.1. Creating discussion themes

Any user can create a new discussion theme. Such a process can be held while workstations are connected or disconnected from the central memory. By creating a new discussion theme, CS automatically associates the theme (its description and search topics) with a group of participants and to a group of coordinators. All these elements are stored locally until the central registration is performed and the information is passed from the workstation to the central memory (Fig. 5), becoming visible to the other users. A discussion process can only begin when the discussion theme is registered at the central memory.

3.2. Discussion area

The set of working documents is represented in the interface as a thread scheme (Fig. 6), like the one used by electronic mail (due to the fact that it is very easy to understand), but is capable of representing an information network that is not hierarchical. For this, some information elements can be repeated visually. The structure of the documents can be rearranged without information loss. This operation is reserved solely for the coordinator, who is also responsible for closing (and reopening) attributes, as well as the discussion itself.

When working with CS in synchronous mode, information changes may arise from different users at any time. Such changes are first sent to the central memory and then propagated to the workstation interface. The visual interface updating process is customisable, as it can be automatic (where the refresh time rate is user defined), or manual (requiring user activation). Besides that, data management routines and interface updates do not interfere with the system interactivity, making data access and manipulation a swift, easy and reliable process.

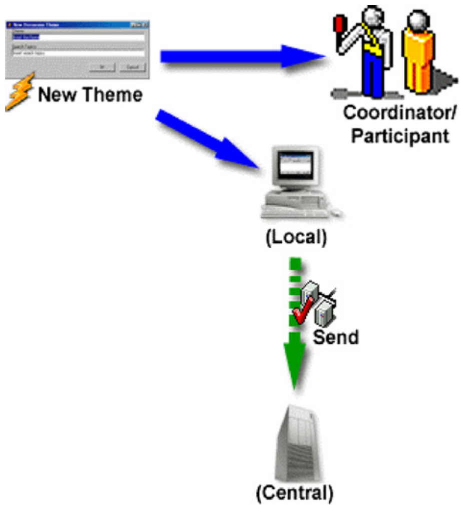

Fig. 5. New discussion.

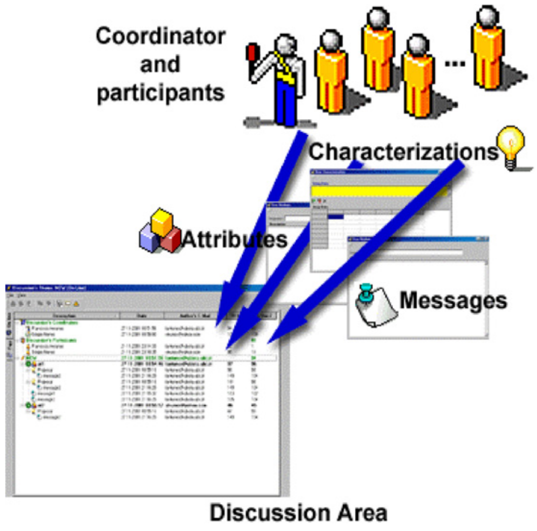

Fig. 6. Discussion elements.

3.3. Replication and information reservation

As CS can be used asynchronously, without the need for an active connection to the central memory, participants can replicate discussions from the central memory to local workstations, enhancing data availability for distributed working conditions, where mobility is an important issue. This process copies the information elements, and all the linking structure of a given discussion, and stores it on a private database, to be accessed locally (Fig. 7). Though every participant is able to replicate discussions, the process differs according to whether the participant is the coordinator. Since the coordinator alone is allowed to perform structural changes to the discussion (re-arranging information nodes, selecting attribute characterization, closing or re-opening attributes for discussion, as well as the discussion itself), she/he has extended replication options.

A non-coordinator participant has only two options for performing a replication: the "see only", where the replica is a mere "photography" of a discussion at a certain moment, and therefore information cannot be added; and the "adding replication", where it is possible to incorporate new contributions into the discussion. The "see only" option has no consequences in terms of data integrity preservation, since there is no information manipulation. The "adding replication" requires the user to perform an "adding reservation". This "reservation" implies that the structure of a discussion cannot be changed during the reservation period or otherwise it might compromise the context and the linking of new information. There can be as many "adding reservations" related to a particular discussion, as the number of participants in that discussion.

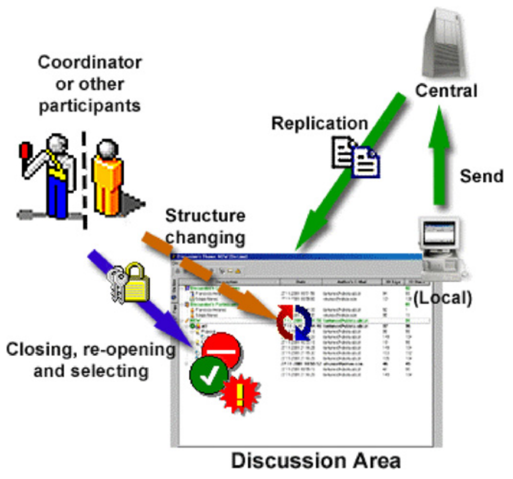
Coordinator or other participants
Central
Replication
Send
Structure changing
(Local)
Closing, re-opening and selecting
Discussion Area

Fig. 7. Discussion replication and coordination tasks.

In addition to the previous options, the coordinator has a third choice for discussion replication: "structure changing". This enables him/her to carry out structure changes to the discussion, when the process needs to be held asynchronously. A "changing reservation" of the discussion is registered at the central memory and the discussion is replicated to the coordinator's workstation. There can be only one "changing reservation" at a time for each discussion.

"Adding reservations" and "changing reservations" are mutually exclusive. Both "reservation" types are withdrawn when a participant (coordinator or not) connects its workstation to the central memory and starts the synchronisation process.

The DOC2 data-structuring model supports the linking of the documents, working with both the transaction-based persistence and the distributed architecture of the system, maintaining integrity. Because users should see identical (or, at least, consistent) views of the shared data, a "reservation" tool had to be implemented. "Reservations" act as locking mechanisms, with two separate functions: avoiding document structure rearrangement when performing an "adding reservation", and preventing information being added when performing a "changing reservation".

3.4. Experimentation insights

The system was first tested in a classroom environment in the Economics Faculty of Coimbra University (Guerra, 2004), with a view to seeing if there were any technological problems that could jeopardize real-test scenarios. Further tests were conducted at a research institute (INESC Coimbra), not to perform a quantitative analysis, but mainly to determine if both DOC2 and CS argumentation models would suit their purposes, according to the users. These tests yielded some important insights.

The technological infrastructure of the tool seems to handle the distributed collaboration environment pretty well. Connection management, "reservation" processes, transaction based persistence, synchronous and asynchronous behaviour (connected, or disconnected from central memory) and error handling are some of the issues that were automatically, and successfully, dealt with by the software. The information structuring process was also correctly performed and integrated into the argumentation model, using the thread scheme to represent the information structure (in a hierarchical or non-hierarchical fashion).

Despite the fact that its main objectives were achieved, the CS software was not without problems. Most of them were related to its graphical environment, and, although relevant to collaborative usage and effort, they were not found to be of major concern. However, two other more profound issues arose. Firstly, since all documents are shown to the users in a thread scheme, information overload can occur with "small-talk" ("ok", "I agree", "alright", etc.), meaning that the linking structure needs enhanced features to accommodate this kind of discourse, to improve information browsing and retrieval, to express relevance patterns (system or user defined) and to allow a more solid basis for the development/adjustment of better awareness tools/interface. Secondly, although asynchronous work without an active connection to the central memory was correctly supported, "reservation" mechanisms were able to delay the collaboration process in the absence of a defined strategy (such as expiry dates for "reservations") for freeing never-ending "reservations", especially if such "reserves" are "structure changing". Therefore, this solution did not prove to be as appropriate as intended.

This last finding was the major motivation for directing research towards divergence analysis in collaborative systems, so as to find a robust, but flexible, way of removing the "locking" approach, when multiple users asynchronously (with or without connection to a central memory) access and manipulate information objects, without losing consistency and structuring capabilities.

4. Divergence

The divergence notion and divergence management strategies in our research closely follow the work of Dourish (1994, 1995, 1996) and try to enhance it by developing a model that should be able to deal with divergent replicated documents, with or without guarantees of resynchronisation consistency, allowing the users to freely manipulate information, without the risk of compromising information reliability or relationships.

Though there are a number of different approaches to deal with different information versions or document replicates, none of them aims directly at knowledge management or organizational memory creation. Versioning (Haake and Haake, 2003; Munson and Dewan, 1994), operational transformation (Ellis and Gibbs, 1989; Beaudouin-Lafon and Karsenty, 1992), and replicated databases (such as used in Lotus Notes) emphasise the creation and management of parallel versions rather than the subsequent integration of different versions. Consistency guarantees (Dourish, 1995) seems to open a door for providing an effective means for divergence resolution and information integration, but does not support the possibility of retrieving the "knowledge path" when achieving information convergence. Such approaches seem to concern much more about answering the question on how to deal with information divergence rather than how to enhance knowledge management or organizational memory with the use of the information divergence model.

A CSCW model based on multiple, parallel streams of activity, developed for both synchronous and asynchronous usage, such as the DOC2 model, usually means that there can be replicated information objects. The manipulation of such objects is only a natural consequence of distributed work. However, this situation generates different versions of a base object and is a likely cause of divergence. According to Dourish (1994) there are two crucial moments within the replication process: (i) a divergence point, where the information replica of some base object is altered; (ii) a synchronization point, where the elements are brought together to form a single replication group (see Fig. 8).

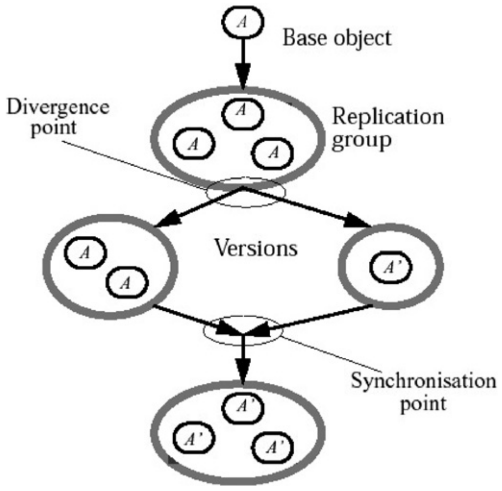

Fig. 8. Divergence process (Dourish, 1994).

It can be seen that, after the synchronisation point, the information contained in object A is transformed into A'. Though it means that there is a consistent view of the information after synchronisation, it can also mean that the earlier information may no longer be available. But we maintain that the possibility of retaining the persistence for both information objects increases the system's capability for retrieving the information's sources and relationships. This surely contributes to a better understanding of information creation and association, thus enhancing the process of knowledge internalisation, transforming explicit knowledge into tacit knowledge, as defined by Nonaka and Takeuchi (1995). A feature of this type assumes that information objects are neither replaced by new information nor deleted; a structuring support is needed that can both ensure information persistence, and (and perhaps more important) perform the linking structure, taking care of information consistency. This mechanism means that information is not lost, even if it becomes irrelevant or inappropriate to current discussions, leaving it open to possible use in the future.

In the current CS version none of the working documents is available for editing, therefore divergence resolution mechanisms are not needed, even when there are replicated documents at the workstations. Instead, there is a chain of new information that can be built, relying on document association and on attribute/characterization selection tools to indicate which are relevant to the discussion in hand. Though there are drawbacks, as it restricts the group discussion process, hinders information awareness, and limits the software itself, it is the authors' belief that the data structuring capabilities of the DOC2 model may be both relevant and useful to deal with divergent information.

4.1. Improving object-relation expressiveness

By setting different types of links among information objects it is possible to build a much easier and more solid way of retrieving the information

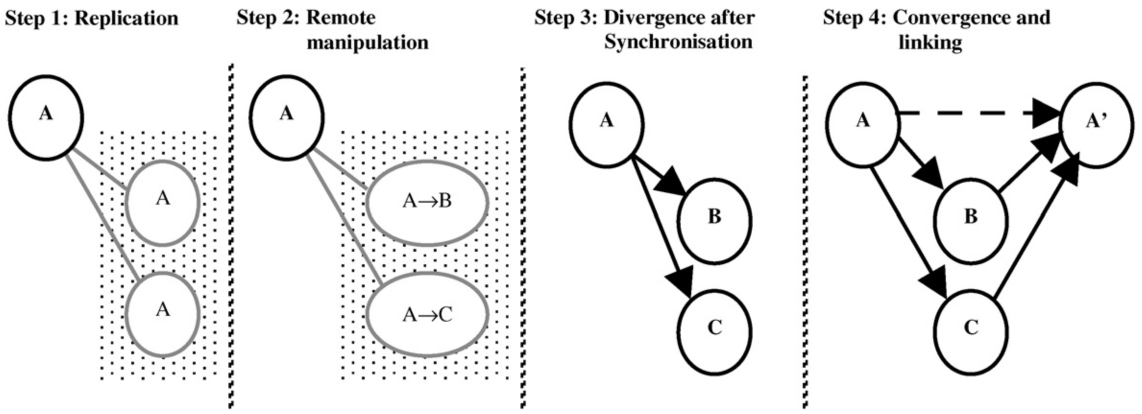
Step 1: Replication
Step 2: Remote manipulation
Step 3: Divergence after Synchronisation
Step 4: Convergence and linking
A
A→B
A→C
B
C
A'

Fig. 9. Information divergence and convergence process using object structuring.

network. Fig. 9 illustrates these ideas by using a small example.

In *Step 1* an information object (A) is replicated on different workstations (dotted area). At the synchronisation point the manipulated objects ($A \rightarrow B$ and $A \rightarrow C$ in *Step 2*) represent a divergent situation. Therefore two new objects are created, and information that allows an easy source association of objects B and C with object A is linked. Having identified the existence of divergence, it is now possible to generate a new information object (A') that represents the updated version of the information contained in previous object A.

The linking network will not only allow the retrieval of the information sources that led to the final information, but also enable the association of current information with older versions. The complexity of the linking structure may vary according to the actual situation, from a simple hierarchical or threaded structure, to a network of relationship layers (such as simple association, versioning, quantitative or qualitative rankings, etc.), which may increase linking expressiveness. All these considerations need to be embedded into the DOC2 model and into the CS itself.

5. Final remarks

This paper has described the Collaboration Studio (CS) system. It is made up of two elements: (i) a central memory that ensures global information persistence and server tasks; (ii) different workstations that ensure local information persistence. The system allows discussions among a group of participants, one of whom is a coordinator. The CS argumentation model deals with four collaboration elements, whose information and relationship network constitutes a collaboration discussion: a theme, attributes, attribute characterizations and messages. Each theme can contain multiple attributes, and these in turn can contain other attributes. Every attribute can also accept multiple characterizations.

The system field-tests led to some important insights and these were the major motivation for directing research towards divergence analysis in collaborative systems. The idea is to find a robust, but flexible, way to remove the "locking" approach, when multiple users asynchronously access and manipulate information objects, without losing consistency and structuring capabilities. However, it was found that previous research on divergence management does not exploit this flexible solution, towards knowledge management. So, it is the author's opinion that embedding data structuring capabilities and an information linking structure with a high degree of expressiveness may help explain the followed steps towards an achieved decision and to bridge the gap between classic GDSS support and knowledge management.

Embedding the referred features into the CS software and its field testing, using the same methodology as described in Section 3.4, will constitute the main lines for future research.

Acknowledgements

We would like to thank the reviewers of this paper as they improved the quality of the exposition.

This work was partially funded by FCT and FEDER, project POCTI/32405/GES/2000 project.

References

Abell, A., Oxbrow, N., 2001. Competing with knowledge: The Information Professional in the Knowledge Management Age. Library Association Publishing, London.

Aleixo, J., 2000. An object model for distributed asynchronous applications (original in Portuguese). MSc Thesis, Organizational Information Management. Faculty of Economics, University of Coimbra.

Antunes, F., 2002. Collaboration Studio: A synchronous and asynchronous collaborative system for distributed data structuring (original in Portuguese). MSc Thesis, Organizational Information Management. Faculty of Economics, University of Coimbra.

Antunes, F., Melo, P., Costa, J., 2004. Collaboration Studio as a CSCW tool (original in Portuguese). Research Report No. 19, INESC, http://www.inescc.pt/pubinter.php.

Bafoutsou, G., Mentzas, G., 2002. Review and functional classification of collaborative systems. International Journal of Information Management 22, 281–305.

Beaudouin-Lafon, M., Karsenty, A., 1992. Transparency and Awareness in Real-Time Groupware Systems. Proceedings of the ACM Conference on User Interface Software and Technology UIST'92, Monterey, November.

Boddy, D., Boonstra, A., Kennedy, G., 2002. Managing information systems. Financial Times—Prentice Hall, Pearson Education Limited.

Cao, P., Burstein, F., 1999. An asynchronous group decision support system study for intelligent multicriteria decision making. In: Proceedings of the 32nd Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Chiavenato, I., 2000. Introduction to general administration theory (original in Portuguese). Campus Editor, sixth ed., Rio de Janeiro, pp. 655–656.

Conklin, J., na. The IBIS manual, a short course in IBIS methodology, <http://www.touchstone.com/tr/wp/IBIS.html> (accessed on 2004-04-08).

Conklin, J., 2001. Designing organizational memory: Preserving intellectual assets in a knowledge economy. <http://www.cognexus.org/dom.pdf>, 2001 (accessed on 2004-04-08).

Costa, J., Melo, P., Godinho, P., Dias, L., 2003. The AGAP system: A GDSS for project analysis and evaluation. European Journal of Operations Research 145, 287–303.

Courtney, J., Chae, B., Hall, D., 2000. Developing inquiring organizations. Knowledge and Innovation: Journal of the KMCI 1, 1.

Dennis, A., Haley, B., Vandenberg, R., 1996. A meta-analysis of effectiveness, efficiency, and participant satisfaction in GSS research. Proceedings of ICIS, 1996, Cleveland.

Dourish, P., 1994. A divergence-based model of synchrony and distribution in collaborative systems. Technical Report EPC-1994-102, Rank Xerox Ltd.

Dourish, P., 1995. The parting of the ways: Divergence, data management and collaborative work. In: Proceedings of the Fourth European Conference on Computer-Supported Cooperative Work ECSCW'95 (Stockholm, Sweden). Kluwer, pp. 215–230.

Dourish, P., 1996. Open implementation and flexibility in CSCW toolkits, PhD Thesis, University of London.

Dourish, P., Bellotti, V., 1992. Awareness and coordination in shared work spaces. ACM Conference on Computer-Supported Cooperative Work CSCW'92, Toronto, Canada.

Drucker, P., 1988. The coming of the new organization. Harvard Business Review (Jan–Feb), 45–53.

Dufner, D., Park, Y.-T., Kwon, O., Peng, Q., 2002. Asynchronous team support: Perceptions of the group problem solving process when using a CyberCollaboratory. In: Proceedings of the 35th Hawaii International Conference on System Sciences (CD/ROM), January 4–7. Computer Society Press.

Ellis, C., Gibbs, S., 1989. Concurrency Control in a groupware system. Proceedings of ACM conference on Management of Data SIGMOD'89, Seattle, Washington, pp. 399–407.

Franz, H., 1999. The impact of computer mediated communication on information overload in distributed teams. In: Proceedings of the 32nd Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Ginsburg, M., Kambil, A., 1999. Annotate: A Web-based knowledge management support system for document collection. In: Proceedings of the 32nd Hawaii International Conference on SystemSciences (CD/ROM), January 3–6. Computer Society Press.

Gordon, T., Karacapilidis, N., 1996. The Zeno argumentation framework. Proceedings of the Workshop on Computational Dialectics: Models of Argumentation, Negotiation and Decision Making, International Conference on Formal and Applied Practical Reasoning (FAPR'96), June 3–7, Bonn, Germany.

Gordon, T., Voss, A., Richter, G., Märker, O., 2001. Zeno: groupware for discussions on the Internet. Künstliche Intelligenz, Heft 2/01, Verlag, Bremen, <http://www.ais.fraunhofer.de/~maerker/paper/gordon_et_al.pdf>, 2001 (accessed on 2004-04-12).

Guerra, A., 2004. Collaborative systems evaluation: The case of Collaboration Studio (original in Portuguese). MSc Thesis, Organizational Information Management, Faculty of Economics, University of Coimbra.

Gunnlaugsdottir, J., 2003. Seek and you will find, share and you will benefit: Organising knowledge using groupware systems. International Journal of Information Management 23, 363–380.

Haake, A., Haake, J., 2003. Take CoVer: Exploiting Version Support in Cooperative systems. Proceedings of the INTERCHI, Amsterdam, Netherlands, 24–29 April, pp. 406–413.

Hilmer, K., Dennis, A., 2000. Stimulating thinking in group decision making. In: Proceedings of the 33rd Hawaii International Conference on System Sciences (CD/ROM), January 4–7. Computer Society Press.

Karacapilidis, N., Papadias, D., Pappis, C., 1999. Computer-mediated collaborative decision making: Theoretical and implementation issues. In: Proceedings of the 32nd Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Katzy, B., Evaristo, R., Zigurs, I., 2000. Knowledge management in virtual projects: A research agenda. In: Proceedings of the 33rd Hawaii International Conference on System Sciences (CD/ROM), January 4–7. Computer Society Press.

Kwok, R., Ma, J., Vogel, D., 2000. Effect of GSS and facilitation on knowledge restructuring. In: Proceedings of the 33rd Hawaii International Conference on System Sciences (CD/ROM), January 4–7. Computer Society Press.

Lowry, P., Albrecht, C., Lee, J., Nunamaker, J., 2002. User experiences in collaborative writing using Collaboratus, an Internet-based collaborative work. In: Proceedings of the 35th Hawaii International Conference on System Sciences (CD/ROM), January 4–7. Computer Society Press.

Massey, A., Weiss, M., Hung, Y.-T., 2002. Synchronizing pace in asynchronous global virtual project teams. In: Proceedings of the 35th Hawaii International Conference on System Sciences (CD/ROM), January 4–7. Computer Society Press.

Melo, P., Costa, J., 1999. Some design options for persistent distributed information within group decision support systems. Proceedings Vol. 1—Information Systems. In: Callaos, N., Nada, N., Cherif, A., Aveledo, M. (Eds.), International Institute of Informatics, pp. 178–184.

Melo, P., Costa, J., 2001. A couple of models for distributed group decision support systems data management (original in Portuguese). In: Évora, A. Serrano, Zorrinho, C. (Eds.), Proceedings of Second Conference of the Portuguese Association of Information Systems. November 21–23.

Munson, J., Dewan, P., 1994. A Flexible Object Merging Framework. In: Proceedings of the Computer Supported Cooperative Work'94. ACM, Chapel Hill, North Carolina, pp. 231–242.

Nonaka, I., Takeuchi, H., 1995. The Knowledge Creating Company: How Japanese Companies Create the Dynamics of Innovation. Oxford University Press.

Nunamaker, J., Romano, N., Briggs, R., 2001. A framework for collaboration and knowledge management. In: Proceedings of the 34th Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Rinner, C., 2001. Argumentation maps: GIS-based decision support for on-line planning. Environment and Planning B: Planning and Design 28, 847–863.

Romano, N., Nunamaker, J., Roussinov, D., Chen, H., 1999a. Collaborative information retrieval environment: Integration of information retrieval with group support systems. In: Proceedings of the 32nd Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Romano, N., Nunamaker, J., Briggs, R., Mittleman, D., 1999b. Distributed GSS facilitation and participation: Field action research. In: Proceedings of the 32nd Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Serrano, A., Fialho, C., 2003. Knowledge Management, the New Organizational Paradigm. FCA Lda., Lisboa, pp. 69.

Sillince, J., Saeedi, M., 1999. Computer-mediated communication: Problems and potentials of argumentation support systems. Decision Support Systems 26, 287–306.

Singh, N., Genesereth, M., Syed, M., 1995. A distributed and anonymous knowledge sharing approach to software intercooperation. International Journal of Cooperative Information Systems 4, 339–367.

Streitz, N., Hannemann, J., Thüring, M., 1989. From ideas and arguments to hyperdocuments: Travelling through activity spaces. In: Proceedings of the Hypertext'89 Conference. ACM Press, New York, pp. 343–364.

Suthers, D., 1999. Representational support for collaborative inquiry. In: Proceedings of the 32nd Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Toulmin, S., 1958. The Uses of Argument. Cambridge University Press, Cambridge.

Turoff, M., Hiltz, Starr R., Bieber, M., Rana, A., 1999. Collaborative discourse structures in computer mediated group communications. In: Proceedings of the 32nd Hawaii International Conference on System Sciences (CD/ROM), January 3–6. Computer Society Press.

Vandenbosch, B., Fay, S., Saatçioglu, A., 2001. Where ideas come from: A systematic view of inquiry. Sprouts: Working Papers on Information Environments, Systems and Organizations, 1, Fall.

Vreede, G.-J., Vogel, D., Kolfschoten, G., Wien, J., 2003. Fifteen years of GSS in the field: A comparison across time and national boundaries. In: Proceedings of the 36th Hawaii International Conference on System Sciences (CD/ROM), January 6–9. Computer Society Press.