A MODEL AND ARCHITECTURE TO SUPPORT ASYNCHRONOUS COLLABORATION MODES IN INTRANETS

Rui Moreira da Luz[1,3], Adérito Fernandes Marcos[1,3]
Christoph Hornung[1], Manuel Próspero dos Santos[2]

[1]Fraunhofer Institute for Computer Graphics, Rundeturmstr. 6, 64283 Darmstadt, Germany
[2]Universidade Nova de Lisboa, Dept. de Informática, Quinta da Torre, 2825 Monte de Caparica
[3]Centro de Computação Gráfica, R. Rodrigues Gusmão, 21, 3000 Coimbra

Abstract

The typical synchronous collaboration solutions have been proved to be rather unacceptable when work conditions avoid group members to work on-line. This can be the case when people are geographically distributed among different time zones with no (or few hours of) overlapping work-time or when difficult network (transmission) problems occur. For those cases, the support of asynchronous collaboration environments would be highly desirable. Through them, people have the possibility to work locally and to exchange cyclically within each other their contributions. A process of annotation and explicit authoring of each private editing shall be kept while the shared documents circulate.

This paper examines the importance of supporting asynchronous forms of collaboration and presents a global model and architecture for a related environment. Some fundamental concepts are established along with implementation considerations and strategies we have adopted during the development phase of a prototype system. An example of application in the area of distributed CAD is also included.

Keywords: CSCW (Computer Supported Cooperative Work), Workflow, Asynchronous Collaboration, Distributed CAD, Multimedia Co-Authoring Process.

0. Introduction

Computer Supported Cooperative Work has emerged in the last decade as an identifiable research area which focuses on the role of computers in group work and has been involving researchers across various disciplines. It is not easy to find a well-defined and overall definition of CSCW. In the literature, the term has been referred to address the whole variety of research areas related to computer-mediated groupwork.

The concept of Computer Supported Cooperative Work was pioneered by Douglas Engelbart, who demonstrated as early as 1968 a prototype called NLS/Augment [ENG88]. In the early 1980s, intensive research was carried out by Winograd and Flores [WIN86], who have developed a theoretical perspective for analysing group action based on ideas from linguistics. This perspective emphasises different kinds of speech acts, such as requests and commitments. The authors analysed a generic conversation for action with the possible modes and transitions involved when one actor performs a task at the request of another.

Since the mid-80s, the CSCW area has been investigated more intensively. The first

workshop took place at MIT in August 1984. More official recognition of CSCW was achieved by the first ACM-sponsored Computer Supported Cooperative Work (CSCW) conference held in Austin, Texas, in December 1986 [GRE91].

Cooperative Work (cp. teamwork, groupwork) is a topic which different research fields have been dealing with, some of them for a rather long time like sociology or psychology [MCG66]. Dix [DIX94] describes the term cooperative work as a process, where two or more participants are involved. They are engaged in the same common work, and to do so, interact with various tools and products. Some of them are physically shared, where others are shared in the sense that they contribute to a cooperative purpose. The participants communicate with each other as they work. In real life this may be by speech or letter. Part of the purpose of communication is to establish a common understanding of the task they are engaged in. This understanding may be implicit in the conversation, or may be made explicit in diagrams or text.

Cooperative work can be also examined under the dimension of time (synchronous/asynchronous) and space (centralised/decentralised) [JOH88], i.e. when and where the participants are performing cooperative work. This can be summarised in a time/space matrix (see Fig. 1).

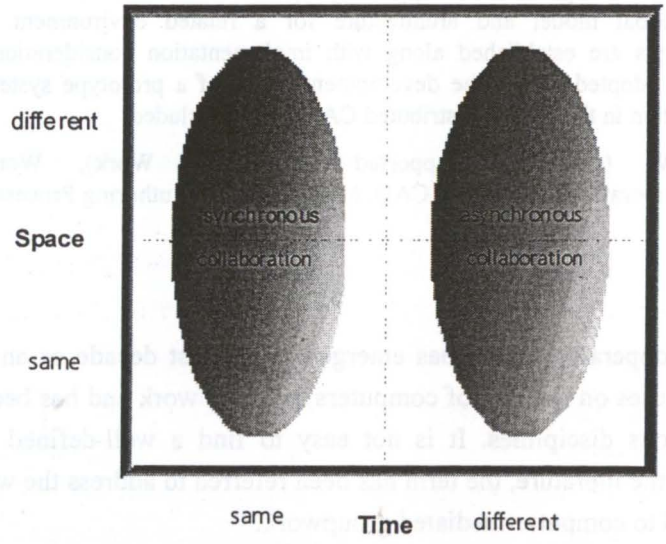

Fig. 1 - The well-known Joahnsen Matrix.

The time/space matrix is a very useful shorthand to refer to the particular circumstances a groupware system aims to address. Basically, we look at the participants and ask whether they are in the same place or not, and whether they are operating at the same time or not.

As we can see exposed in the Fig. 1, synchronous collaboration can be defined as the form of computer-supported groupwork where all the participants share the same common workspace during the same period of time. This type of collaboration has been the main

focus of most of the CSCW researchers in this area. It is supposed that people engaged in working together would online share a same common work-domain, being this last, files, editors or databases.

However, synchronous collaboration solutions have been proved to be rather unacceptable in context of offline work conditions. This can be the case when people are geographically distributed among different time zones without the acceptable overlapping work-time. For those cases, the support of asynchronous collaboration environments would be highly desirable. These are defined as enabling offline forms of computer-mediated groupwork, where individual contributions can be completed during different periods of time and integrated together in a final common version.

Asynchronous collaboration requires a high level of coordination mechanisms in order to avoid merging conflicts between individual contributions. Such mechanisms shall establish a common framework based on the division of tasks among the participants and their complete synchronisation. Also the support of intra-group awareness is here highly required thought no explicit means of direct user-user online communication are considered.

In this paper we examine the importance of supporting asynchronous forms of collaboration, considering especially Intranet environments while presenting a global model and architecture for a related system. We conclude the paper by exposing an example of application of the results achieved in the area of distributed CAD (Computer-Aided Design).

1. The Model's Basics

Many aspects of a process have to be described by its modelling, e.g., what is going to be done, who is going to do it, how and why will it be done, who is dependent on its being done, etc. Process modelling languages differ in the extend to which they highlight the answers to these questions. Process modelling languages and representations usually present one or more different perspectives related to these questions [CUR92].

Process models can have different goals and objectives, depending on the intended use. When automation is involved, process representation becomes a vital issue in redesigning work and allocating responsibilities between human and computers. Process modelling is distinguished from other types of modelling in computer science because a human rather than a machine must enact many of the phenomena modelled. Two possible categories of process modelling reflect the objectives to:

- *facilitate human understanding and communication* which requires that a group is able to share a common representational format;
- *support process improvement* which requires a basis for defining and analysing processes;

The fundamental unit for getting things done (products and services) in a work process is a workflow. A workflow is an unit of work that happens repeatedly in an organisation of

work. It has a beginning, middle, and an end. And, most importantly, in a workflow things get completed in order to producing satisfaction for customers. In fact, every workflow has a customer, who may be a customer or client of the organisation, or may be another work group or an individual in the organisation (an internal customer) [SCH93].

Relating now to our prototype tool, it is based on workflow, as a set of formal rules and specific process(es) which are automated to improve efficiency. Most workflow systems share a basic action protocol. This protocol (see Fig. 2) comprehends four phases:

- opening;
- commitment;
- execution;
- closing.

Opening phase represents the period of time when the customer asks for something to be performed by a service provider, which in his/her turn, agrees (or not) to do it, thus concluding the commitment phase. The execution of the requested service, and report it done, characterises the third phase, which is naturally followed by the declaration of satisfaction (or not) on side of the customer, closing this way, the basic workflow cycle.

In the model that we have adopted, it is assumed that the commitment phase is previously achieved. This decision takes in consideration that when a person joins a company, he/she not only accepts a *modus operandi* imposed by the institution but also automatically commits him/herself to fulfil future tasks when working in the corporation context.

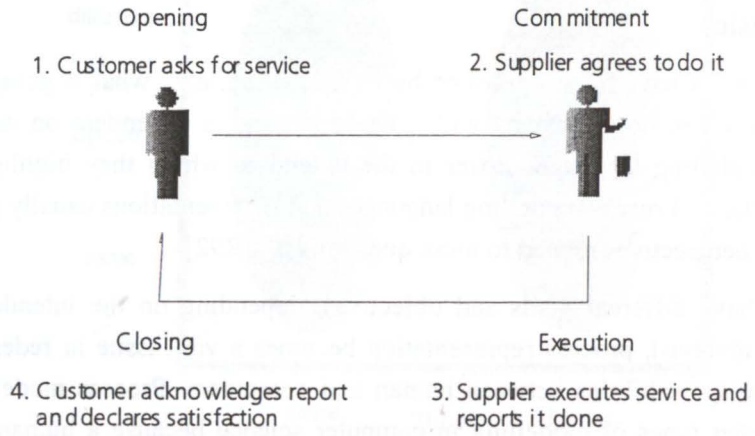

Fig. 2 - Basic action workflow protocol.

This action can also then be interpreted as part of the commitment phase of the basic action workflow protocol, which guides us to a simplified version of the protocol (see Fig. 3). Accordingly, we have now only three phases:

- opening, where the customer asks for a service;
- execution, where the supplier executes the service and reports it done;
- closing, when the customer acknowledges report and declares satisfaction.

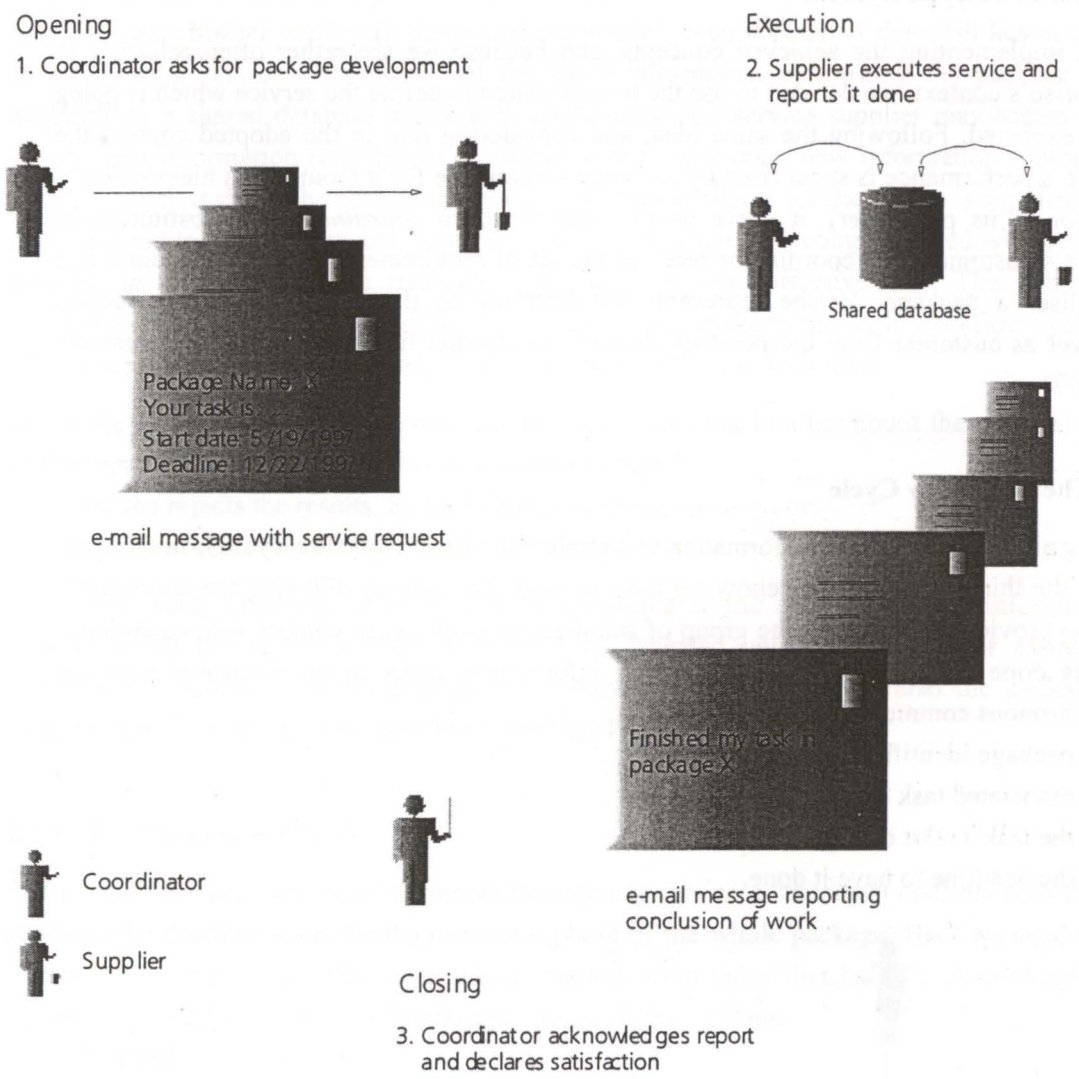

Fig. 3 - Basic workflow cycle and supporting technologies.

The information transmitted along the basic workflow cycle includes the following items:

- service identification – what is going to be done?
- service's customer identification – who requested the service?
- service's supplier identification – who is going to do it?
- service description – how and why will it be done?
- service start date – when shall be the work begin?
- service deadline – when shall be the work end?

Positively, these items model a piece of work or workflow. They can be employed for the concrete fulfilment of tasks, i.e. labour.

2. The Prototype System

When implementing the achieved concepts, and because we are rather often referring to enterprise's context, we decided to use the term *package* to define the service which is going to be executed. Following the same idea, and considering that in the adopted context the service's performance is supervised by someone responsible for it (sometimes hierarchically superior to its performer), it drove us to adopt the term *coordinator* in substitution of service's customer. The coordinator receives the set of requirements from the customer and formalises a package. He/she represents the interface to the external customer, acting however as customer from the point of view of the internal fulfilment of the requirements [LUZ97].

2.1 The Workflow Cycle

Having all the indispensable information to complete the basic workflow cycle is not enough to put the things running. Somehow we have to start the cycle by allowing the coordinator and the provider (who set up the group of suppliers or workers) to contact with each other. This is done by condensing the following information items in an electronic message (asynchronous communication as intended):

- package identification
- associated task description
- the task's start date;
- the deadline to have it done.

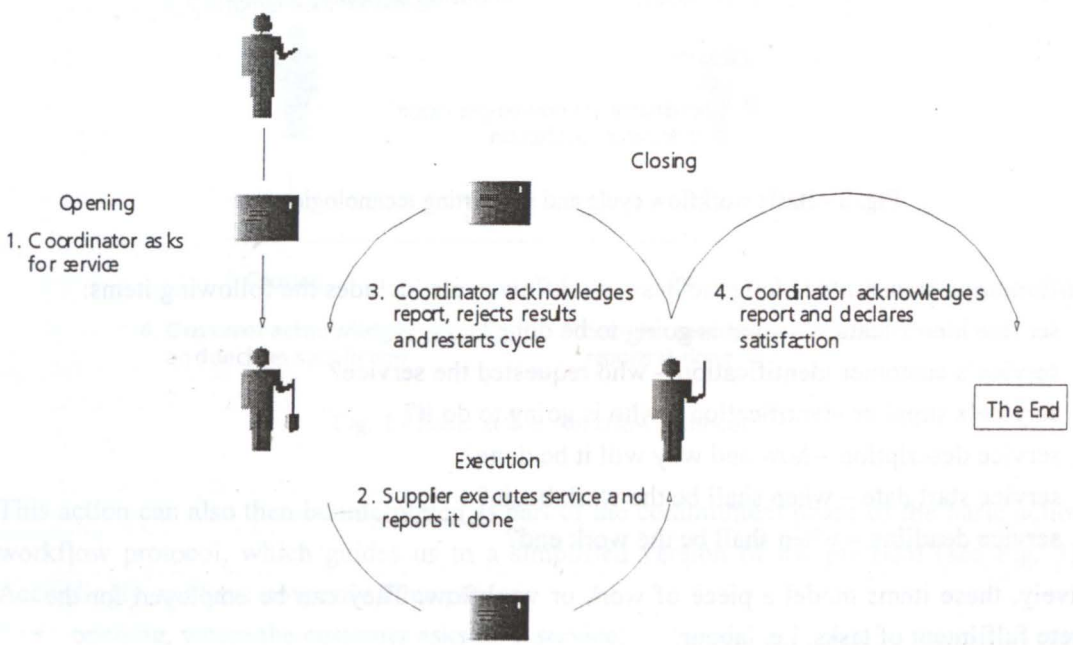

Fig. 4 - Detailed workflow cycle.

After receiving the message that contains a request for a task execution, the service supplier may then start his/her work with the available material, even if the start date still lies in the future. During service's execution, all the basic information concerning the package is available in a shared database where both coordinator and service supplier may access or supply new information referring to the package (e.g. new data, new information sources, clarifications). It should be noticed that the information available for the service's supplier is not restricted to this sources and, when necessary, they should be complemented with other sources, in order to ease the realisation of task in a more effective way. This can be encouraged through the contents of the E-mail message that opens the basic workflow cycle, or by references in the available set of information, like links to web sites.

When the coordinator receives a warning message informing him/her about the conclusion of the service execution, one of two options may succeed:

- he/she rejects the results, by finding them non-satisfactory, or;
- he/she accepts them.

In the first case, the cycle is restarted with a warning to the service supplier, informing him/her about the rejection. The cycle must be repeated until the coordinator express satisfaction (second situation) and then, not only the basic cycle but also the package execution, will be declared as completed (see Fig. 4).

2.1.1 The Execution Phases

The package coordinator (who represents the internal service's customer) controls both the start and the deadline dates for the execution phase of the whole package. Here we need to differentiate the packages that have already started, from those that haven't. Accordingly, we have divided the execution of packages into six different states:

- Planned;
- In Production;
- Completed;
- Deadlined;
- Archive;
- Trash.

Planned packages are those, whose start date still lies in the future. The packages that have already started and are not completed or *deadlined*, are identified as being under the *in Production* state. Both states (*planned* and *in production*) depend on the start date defined for the package. The deadline date may also determine a third state called *Deadlined*. Any package that is not completed and whose deadline date is already over will be catalogued as deadlined. Finally, if the execution phase is completed and the package coordinator accepts the results within the period defined by the start date, then the package is said to be in a *Completed* state.

Both *archive* and *trash* states were introduced to ease the manipulation of packages and provide a complete set of possible states. The packages will be in *archive* or *trash* state only by decision of the package coordinator. They exist for purposes of global control and private use of the coordinator. They are not accessible to suppliers.

A package can then present a sequence of several states, which are not necessarily in the traditional predefined order: *planned*, *in production*, and *completed*. Since the variables that dictate the package state may change, following not only the coordinator's will or requirements but also the suppliers activity, then states may evolve along with an unexpected order.

2.1.2 The Execution Structures

Most of the times performing a service involves more than one person. In a work organisation domain, it is rather usual to take advantage of the distribution of labour, looking for complementation and synergy between workers or groups of workers (tailoring activities), and having in return better and faster results in certain tasks. This guides us to consider in our workflow model the supplier as being a group and not a single person. The supplier group can be built of several individuals, or must have at least one member.

By having groups at his/her disposal to develop a package, the coordinator needs to provide each member of the group with a specific task description along with start and deadline dates.

Aiming to provide some more flexibility to end-users, on the one hand, and reflect what actually happens in real life situations, on the other hand, we decided that package coordinators should define not only the start and deadline dates for each group member, but also for the whole package, the start and deadline dates. These dates determine the interval where all the other dates, concerning the package execution (member's specific dates), have to be confined in.

If each group member has specific start and deadline dates (see Fig. 5) then, by combining them with the different work sequences within a group, the workflow may assume three different types of structure. These are:

- Parallel;
- Serial;
- Hybrid.

The parallel structure is characterised, despite different start and deadline dates among each member's task execution, by the fact that the results achieved will not serve as input to other members. Considering the case of the (strictly) serial sequences, each member, except the one who starts the package execution, receives as input the results obtained by the precedent member and may proceed from it. Hybrid workflow structures are those that include both parallel and serial structures. Both serial and hybrid workflow structures require that the

start and deadline dates for each group member shall be compatible among them. Compatible dates are those that prevent overlapping execution phases between those elements that are working in a *serial* mode.

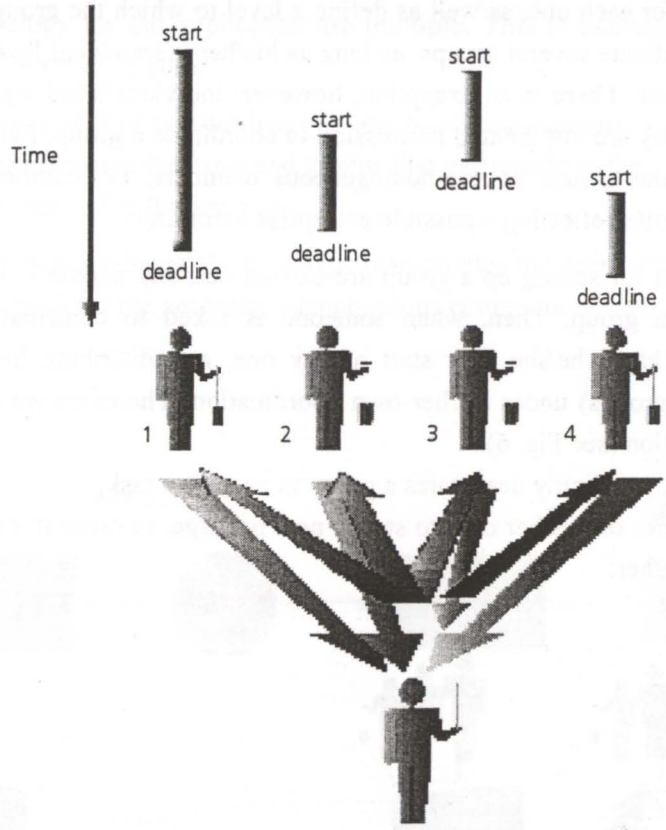

Fig. 5 - Basic Execution Structures.

2.2 Group Management

Within large organisations, it is crucial the existence of a hierarchy. It brings advantages such as providing people with the sense of working together in a team, but also helps to manage labour activities towards effectiveness. Notwithstanding each corporation adopt its own hierarchy model, we have observed that essentially the well-known tree model (and inherent existence of hierarchic levels) is being overall applied. Taking that into account, as far as possible, we aimed to implement an asynchronous generic collaboration tool we included in our prototype a component that allows users to define a hierarchy. Relations between people, as individuals, groups or groups of groups, may be configured and further used in asynchronous cooperative processes.

Besides the pointed advantages of a hierarchical organisation, this social structure will help users to define the group that will best suit his/her needs (or corporation's needs),

concerning the service's execution. The solution offered to the user allows him/her to register new persons to the system and define, for each one, a level within the hierarchy. From the whole set of persons registered in the system, he/she may set up groups of people and assign a coordinator for each one, as well as define a level to which the group belongs. A single person may coordinate several groups, as long as his/her hierarchical level is equal or higher to the group level. There is an exception, however: individuals belonging to the lowest level in the hierarchy are not granted permission to coordinate a group. Furthermore, it is also possible to create groups of non-homogeneous members, *i.e.* combinations of people and groups, thus better reflecting a possible enterprise hierarchy.

As long as the stated rules for setting up a group are carried out, any person is virtually a candidate to coordinate a group. Then, when someone is asked to contribute for the accomplishment of a package, he/she may start a new one, and distribute his/her task execution to a group (or groups) under his/her own coordination. Therefore we may have two types of task ramification (see Fig. 6):

- when the coordinator explicitly designates a group to execute a task;
- when a person decides on his/her own to start a new package, in order to execute the task assigned to him/her.

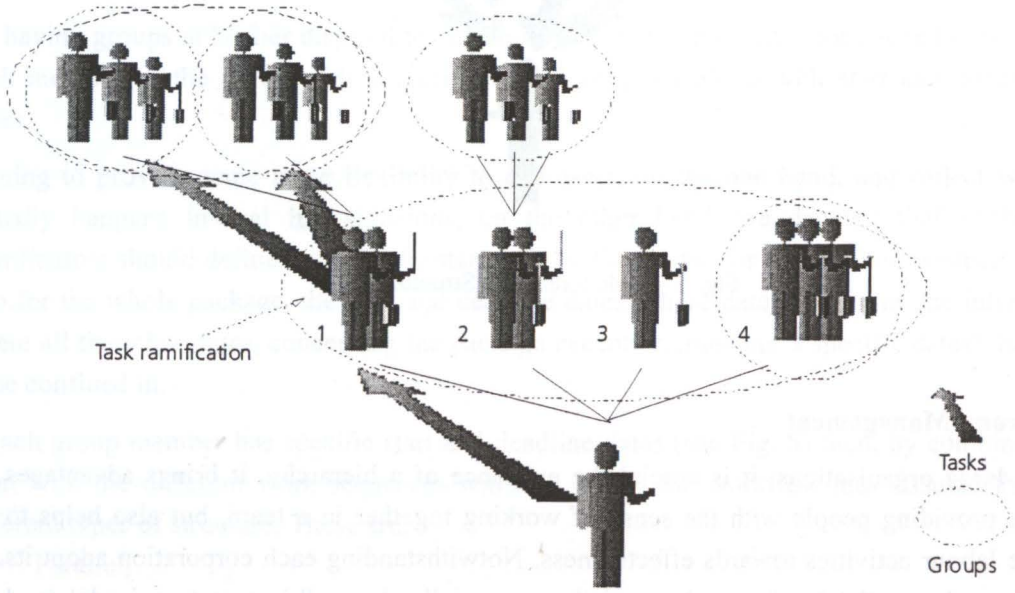

Fig. 6 - Task ramification.

In corporations characterised by a hierarchical structure, it is common to see the second type of task ramification and, in most cases, following a top-down structure (*e.g.*, from higher to lower levels in the hierarchy). This was fully implemented in our prototype system.

2.3 Cooperative Architecture

Our Asynchronous Computer Based Work prototype tool was entirely developed under the Lotus Notes (client and server) environment. The possibilities offered by Lotus Notes to define a topology for our application are multiple. This is essentially due to the database replication feature it supports.

Our prototype is built of two databases (Lotus Notes applications):

- one for managing the users and groups that participate in the system;
- one for managing the packages.

Besides these two databases, the prototype manipulates the users' mailboxes (which are also databases) to perform the necessary asynchronous communication between the participants.

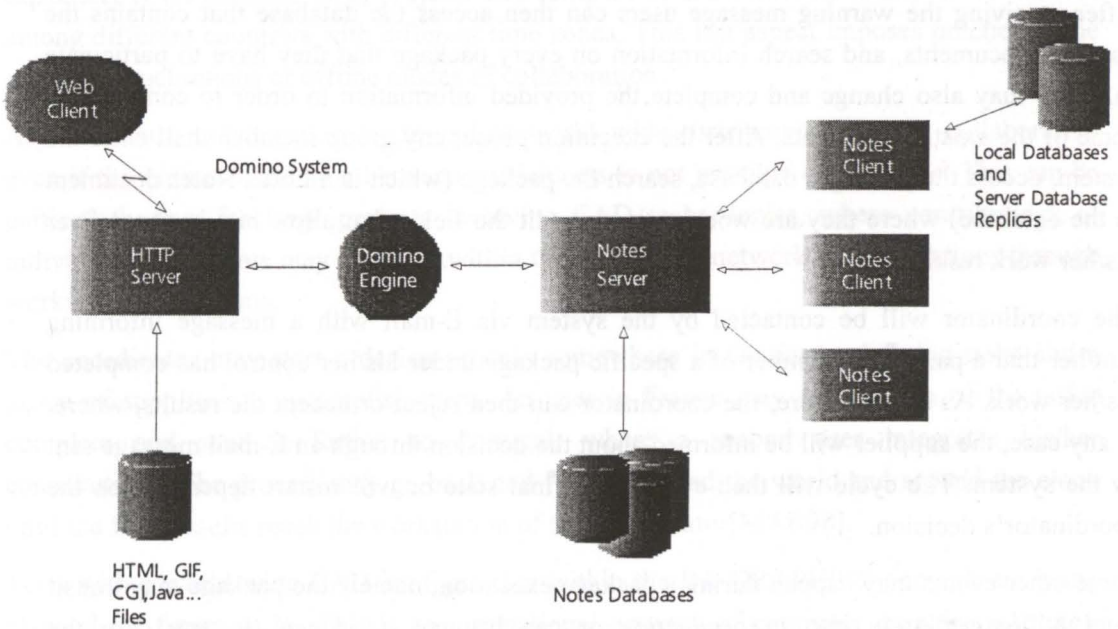

Fig. 7 - The Lotus Notes based system architecture.

The prototype was tested in a LAN with Windows NT4.0 and three different machines (see Fig. 7). One was to set up the server part (essentially for mail routing) and the others were the terminals from where users could access the databases and interact with them. All the databases were in the server side although they could be replicated in client machines, thus enhancing performance and functionality.

Recently, Lotus has released a new component (Domino) which can be attached to Lotus Notes. It gives a new dimension to the system by allowing Internet users to access database contents. This certainly is an important improvement, which could be explored by our prototype in order to enforce its flexibility in terms of accessibility through the web.

2.4 A session of Asynchronous Collaboration

Users of the system may play two different roles in the system (or both). He/she may be a simple member of the suppliers' group who only performs tasks or he/she can be a package coordinator who defines the parameters of the tasks execution (or he/she can play both).

When an user starts a new package, by defining the package start and deadline dates as well as assigning tasks and their completion sequence to a group of suppliers, then the system will automatically send to the group members a specific message alerting each one of his/her task. This message may contain text, HTML links, graphics or even attached files. It has been designed to be a more complete tool in giving useful information and help people to fulfil their tasks more efficiently.

After receiving the warning message users can then access the database that contains the package documents, and search information on every package that they have to participate in. They may also change and complete the provided information in order to complement some of the existing elements. After the execution phase, any group member shall enter the system, access the packages database, search the package (which is a Lotus Notes document in the database) where they are working in and edit the fields that allow him/her to deliver his/her work results.

The coordinator will be contacted by the system via E-mail with a message informing him/her that a particular member of a specific package under his/her control has completed his/her work. As stated before, the coordinator can then reject or accept the results, where, in any case, the supplier will be informed about the decision through an E-mail message sent by the system. The cycle will then evolve to a final state or will restart depending on the coordinator's decision.

Some other events may happen during a package execution, namely the package may transit from a *planned* state into *in production* or can become *deadlined* (in any case the coordinator is informed by the system about the state changes). The coordinator may also decide to move the package into *archive* or *trash*.

This session example can be complemented with the description of the group management interaction where the users (with adequate accessing permissions) register new system members and compose new groups that will take part on the packages' execution. In fact, the user will choose from the Lotus Notes system the names of the persons he/she wants to have involved in the execution of the package, by following the rules that we have exposed before. Group management actions related to people who are intended to be involved in a package execution, shall always be taken before the definition of the package itself. In fact, if a user is not registered as coordinator he/she will not be able to start a package. On the other hand, if there are no groups created, then there will be no potential suppliers to realise the package.

3. Potential Applications in Distributed CAD Environments

Computer-Aided Design (CAD) has been intensively employed in the industry for product design and development. Efficient CAD processes becomes an international factor in competition and put forth an important impact on company success. Here automation can be adopted for the storage and re-utilization of product information as well as for self-regulation of the production processes. Efficient structurisation and intuitive access to data for the fast communication of information are an essential factor for economic success [KEH97].

In order to obtain the best results, qualified designers, engineers and experts of different areas shall be involved. These people are usually not located in the same geographical place, especially if we consider large projects and companies, and most important, are distributed among different countries with different time zones. This last aspect imposes practically the need of asynchronous or offline modes of collaboration.

Asynchronous collaboration modes allow world wide distributed teams to collaboration even if the online connection between participants is not possible or desirable. This can be particularly true for large projects involving CAD technologies, where contributions of individual participants may circulate within the distributed network of workstations through workflow mechanisms.

The coordinator may start a design process somewhere in Germany, defining tasks and a group of suppliers to accomplish them. An user in France may receive one of the tasks, complete and send it further to Portugal, where a second user integrates his/her contributions. The process may go back and further around the world and around the clock until the final results reach the workstation of the coordinator[MAR98].

Tests with a world wide CAD configuration within the INI-GRAPHICS network are being planned for future work. This is an international network of computer graphics institutes, namely Fraunhofer IGD in Darmstadt, Germany, Centro de Computação Gráfica - CCG in Portugal, Fraunhofer CRCG in Providence, USA and Centre for Advanced Media Technology – CamTech in Singapura. Here an extension of our Lotes Notes based arquitecture in order to integrate an Internet/ISDN infrastructure needs to be conceptualised. Also more specialised forms of definition and management of groups and their hierarchies are required. With such an asynchronous cooperative platform, people from these different institutions may joint efforts and promote synergies in order to collaborate and produce better and more efficient results, without leaving their desks.

4. Conclusions

In this paper we have discussed the importance of supporting modes of asynchronous collaboration. These modes can me modelled though the principles and technologies of workflow. We have exposed the main aspects related with workflow that are relevant to the

support of asynchronous collaboration.

A prototype has been described along with concepts of package, tasks, coordinator of a package, suppliers and their management, workflow cycle, states of the workflow cycle, serialisation of tasks, etc. Also the implementation of the prototype based on the Lotus Notes platform has been presented. We have seen that the implementation of specialised forms of E-mail communication and their management is crucial in any asynchronous collaborative environment.

On the other hand, we have presented a typical asynchronous collaborative session along with the most important steps involved a their correlation.

Finally, an example of application of the concepts and results achieved within the INI-Graphics network has been outlined. This network represents an ideal testbed for the results of this work.

Acknowledgements

We thank Prof. José L. Encarnação for the opportunities given. This work is partially funded by a PRAXIS XXI-JNICT grant (CIENCIA BD/2663/93-IA).

References

[COL97] Coleman D., "Groupware - Collaborative Strategies for Corporate LANs and Intranets", Prentice Hall PTR, London, Sydney, Toronto, Rio de Janeiro, ISBN 0-13-727728-8, June 1997.

[CUR92] Curtis B., Kellner M., Over J., "Process Modelling", in Communications of the ACM, Sept., 1992, pp.35-39.

[DIX94] Dix A., "Computer-supported Cooperative Work: A FrameWork", in Design Issues in CSCW, Rosenber&Hutchison (Eds), Springer-Verlag, London, pp.9-26.

[ENG88] Engelbart D., Lethman H., "Working Together", in Byte, pp. 245-252.

[GRE91] Greenberg S., "Computer-supported Cooperative Work and Groupware", Academic Press, San Diego, CA.

[KEH97] Kehrer B., "Qualification Project "Multimedia Publishing and Design"", in TOPICS, vol.9, n. 5, 1997, pp.18-19, ISSN 0936-2770.

[HUG97] Ch.E. Hughes Ch. E., Moshell J.M., "Shared virtual worlds for education: the ExploreNet experiment", ACM Journal Multimedia Systems, Vol.5, No.2, pp. 145-154, March 1997.

[JOH88] Johansen R., "Groupware: Computer Support for Business Teams", The Free Press, New York Johnson.

[LUZ97] Moreira Da Luz, R..M., "ASYNCHRONOUS COMPUTER-BASED WORK", Graduation report, Darmstadt University of Technology, Nova University of Lisbon, published as IGD technical report, FIGD-9709, Darmstadt, Germany, Aug. 1997.

[LOT96] Lotus White Paper, "Lotus Workflow Strategy", http://www.lotus.com/ntsdoc96/28d6.htm.

[MAR97] Marcos A., Hornung Ch., "Towards the process of transforming traditional stand-alone environments into cooperative ones", in Proc. of Online Cooperation Berlin - International Conference on Teleworking, Berlin, June 23-24, 1997.

[MAR98] Marcos A., "Modelling Cooperative Multimedia Support for Software Development and Stand-Alone Environments", PhD thesis, Darmstadt University of Technology, to be published by Shakar Verlag, Germany.

[MCG66] McGrath J., Altman I., "Samll Group Research: a Synthesis and Critique of the Field", Holt, Rinehart and Wiston, New York.

[SCH93] Schäl T., Zeller B., "Supporting Cooperative Process with Workflow Management Technology", in Tutorial H3, ECSCW'93, Milan, Sept., 1993.

[SCH94] Schmidt K., "Modes and Mechanisms of Interaction in Cooperative Work - Outline of a Conceptual Framework", Technical Report, RISØ National Laboratory, Roskilde, Denmark, 1994.

[WIN86] Winograd T., Flores F., "Understanding Computer and Cognition: A New Foundation for Design", Addinson-Wesley, Reading et al.