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

Experiences with the DOMINO Office Procedure System

Thomas Kreifelts, Elke Hinrichs, Karl-Heinz Klein,
Peter Seuffert, Gerd Woetzel
German National Research Center for Computer Science (GMD), Germany

Abstract: The Domino office procedure system has been equipped with a new user interface, and has been put to use for the support of purchasing. In this paper, we describe the system, the user interface, and the experiences we made during the practical use of the system. We also briefly discuss the consequences for our own research.

1 Introduction

DOMINO is an office procedure system for modelling and monitoring structured office processes in organizations. In this paper we report on the first practical use of the system. Our goal was to test the usability and usefulness of the DOMINO system, to evaluate the applicability of the DOMINO office procedure model, and to learn from this experience for future developments and research into group support systems, e.g. in the form of new requirements.

A first DOMINO prototype had been completed by 1984 (Kreifelts et al., 1984), a second and functionally enlarged version by 1987 (Kreifelts & Woetzel, 1987). The user interface of the first prototype was somewhat primitive (an extended text editor for alphanumeric terminals), the user interface for the second system version was an experiment in end user programming and was implemented on a Lisp machine (Spenke & Beilken, 1989). In 1989, we felt that the development of other systems for group support might benefit from experiences with our (by now) rather stable office procedure system. So we looked for a possibility to try out DOMINO in practice. The user interfaces of the existing prototype systems of DOMINO were not suited for an office environment, so an important prerequisite for a practical test of the system was the development of a new user interface for such an environment.

Since we were interested in a rather quick experimental use of the system, we decided to try it out in our own organization. This is usually not the best decision as far as generalizability of the results is concerned, but one is freed from additional overhead in preparing the implementation of the system. Also, this way the question of the technical environment in which the system had to run was settled: a network of personal computers (Apple Macintoshes) connected to some server machines running Unix (SUN's).

First, we will give a brief account of the assumptions underlying the design of DOMINO, its functionality, and its architecture. We will then highlight the characteristics of the new DOMINO interface. The rest of the paper is concerned with the experimental use of DOMINO, the experiences we made, and the conclusions we have drawn for our further research.

2 The DOMINO System

The application domain of DOMINO are well-structured cooperative processes in the office. There are four assumptions underlying the design of DOMINO:

(a) Every office worker has a private working domain; cooperation takes place by exchanging messages between these working domains (rather than by working on common domains, i.e. by information sharing).

(b) The messages exchanged in a cooperation are regarded as "speech acts" of a conversation concerning a certain task in the sense of Winograd and Flores (1986).

(c) Cooperation in an organizational setting concerning groups of people is organized by specifying the input/output relations of the elementary work steps; an autonomous agent then coordinates the performance of these steps via conversations for action.

(d) The specification of the input/output relations of the steps is regarded as an "ideal" procedure; exceptions from this procedure can be handled within the action conversations and by the mediating agent.

DOMINO is a system for the specification and automation of cooperative office procedures. It is capable of controlling a variety of such processes which are specified in a special, application oriented language. A procedure description specifies which steps ("actions") a procedure consists of, and what dependencies exist between these actions in the form of information ("forms") needed and produced during the execution of the actions. The various actions of the procedure are assigned to "roles" responsible for their performance; at run-time, these roles are assigned to persons making use of an organizational data base. The underlying procedure model is based on Petri nets, and allows for alternative and concurrent courses of action. Procedure specifications have a graphical representation. We give an example in figure 1. Procedure specifications are checked for consistency and translated into executable form by the DOMINO office procedure compiler.

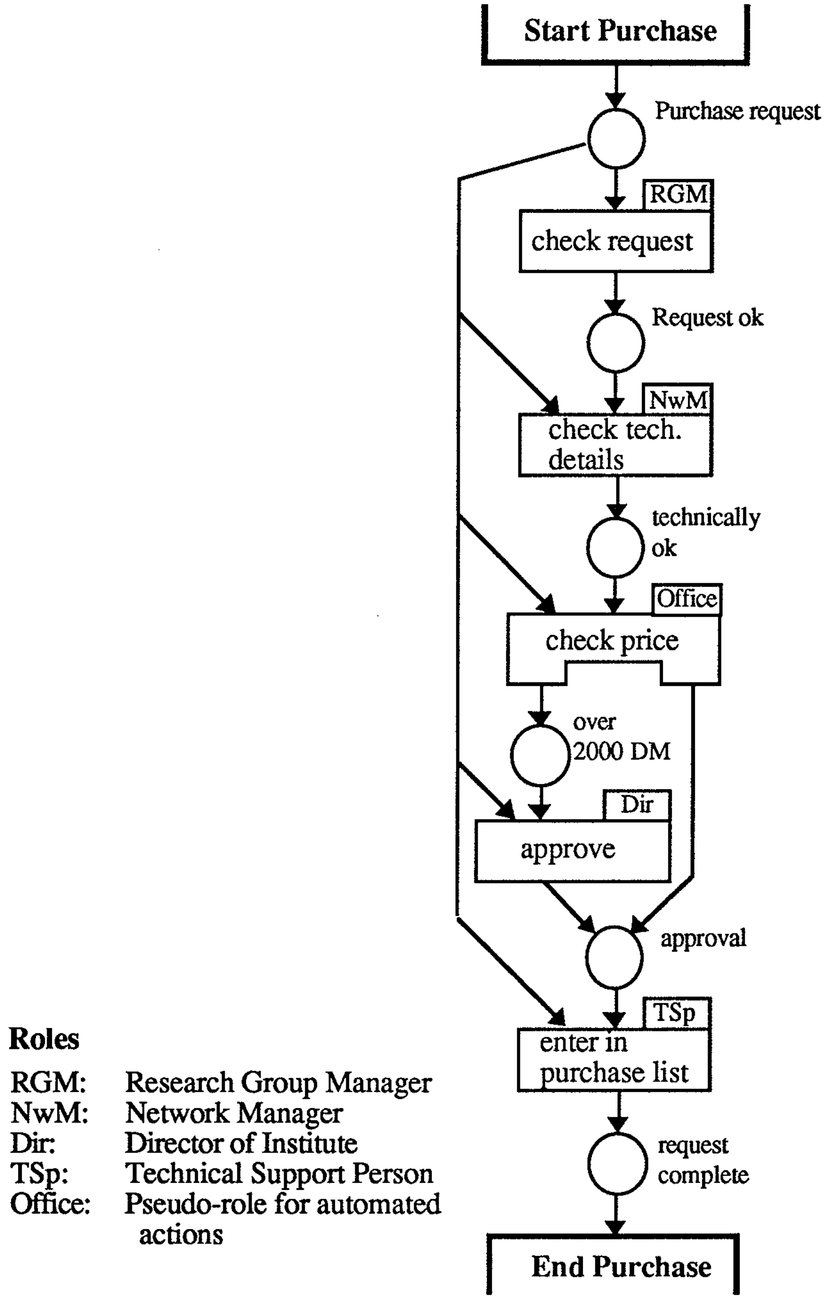


Figure 1. A DOMINO office procedure

DOMINO mediates and controls task related communication by notifying the participants about actions due, by providing them with the information needed, and by routing the results of such actions to the parties responsible. Thus, DOMINO coordinates the activity of a group of persons working on a common task. It is able to inform about the progress of task execution, and provides mechanisms for exception handling in office procedures like delegating an action, or setting back a procedure in case of complaints.

The execution of an office procedure is started on request of a user who becomes the initiator of this procedure instance. The communication between the initiator, the other actors of the procedure and the DOMINO system employs message types which are important in the context of procedure processing. The message types "order", "completion", "confirmation" are used for the straightforward course. "Complaint", "forwarding", "cancellation" (and some more) are used for exception handling. The exchange of these messages follows conventions which are summarized in the CoPlanX protocol. The use of this *conversation for action* ensures a consistent view of the procedure state by all participants.

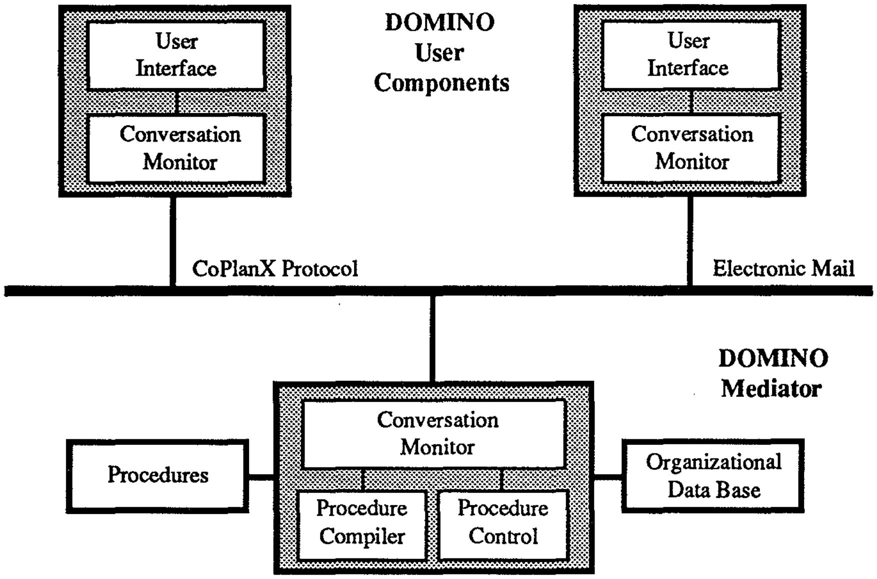


Figure 2. The DOMINO systems architecture

The DOMINO system consists of an automated agent (called "mediator") and user components which communicate via electronic mail using the CoPlanX protocol. The mediator is installed as a fully automated pseudo-user in the mail network. It is responsible for the compilation, installation, and execution of office procedures. It consists of the procedure compiler, the procedure control, and the conversation monitor. All components are implemented in C under UNIX. An experimental organizational data base which is used for role assignment during procedure execution has been realized in Prolog. The overall system architecture is shown in figure 2.

The user components for local user support in procedure processing are installed for every user of the system. They consist of an interface module and the conversation monitor. The implementation depends on the environment in which the DOMINO system is intended to run. In the next section, the user component for our experiment and its development will be described in more detail.

3 The New Interface

3.1 Development

What had to be developed for the new DOMINO version was essentially a way of loosely coupling an office work place equipped with a personal computer (Macintosh in our case) with the central procedure control component. The initial design splitted the user component into a UNIX part and a Mac part and devised a way of communication between the two parts that would ensure a consistent procedure communication even in the absence of a continuous link and with a possible loss of data on the Mac side[1]. The rest of the UNIX components of DOMINO (procedure control, procedure compiler, conversation monitors) remained unchanged.

The initial version of the system concentrated on these technical issues. The user interface design sticked very closely to the original DOMINO concepts, with regard to the procedure model as well as the procedure communication. Although a viable solution for the technical problem of coupling the Macs with the UNIX mediator had been found, this interface was judged as too system-oriented even before completion.

Consequently, the user interface design team was enlarged by a prospective user of the system who is not a computer scientist and who had not been involved in the system development so far. Design work involved many brain storming sessions and screen layouts on paper along with informal descriptions of the functionality of buttons and menus.

The outcome of this phase was a mocked-up user interface. The main characteristics were:

- Form-orientation, i.e. each office procedure type corresponded to an electronic form to be filled in at the various stations it ran through during the procedure execution.
- Inclusion of informal and free format communication, i.e. in addition to the "official" information associated with an office procedure, contained in the form itself, arbitrary enclosures could be added (text documents, drawings, etc.) as well as an informal note sticker (the electronic counterpart to the well known "Post-it" sticker).

[1] With the prospect of even smaller and more portable office computer equipment (lap-tops, handhelds, ...) this design may turn out to be also the right choice for future systems.

- Simplification of the original action conversation, i.e. instead of having to handle 13 different message types in dealing with an office procedure form, the user is given essentially three options: send the form to the next station, send it back to a previous station, or send it to a deputy station. In different contexts the meaning of these options (forward, backward, sideways) would change in an obvious way. This decision, along with the forms orientation, also meant in principle that the original procedure model of a net of actions connected by input and output data was transformed by a "migrating form paradigm" on the user level.

The need for early computerized mock-ups led to the use of HyperCard. Consequently, the decision was made to implement the Mac part of the user component completely in HyperCard. In this form, the system was presented at an information technology fair (Systems '89 in Munich, Germany) with a good general response to the user interface and the potential usefulness of the system. After this successful presentation, the interface was slightly improved with regard to graphic quality, layout, and ease of use, the UNIX-Mac communication was based on faster and more reliable protocols (MacTCP, TCP/IP), and it was then decided to engage in the practical test of the system.

3.2 Interface Description

In the following, we will describe the DOMINO interface as it is currently in use. As the typical Macintosh screen is on the small side, the interface is comprised of several full-screen layouts the user may switch between. A screen layout is divided into a menu bar placed across the top, an information window which takes most of the screen, and a bottom row of function buttons for more frequently used functions.

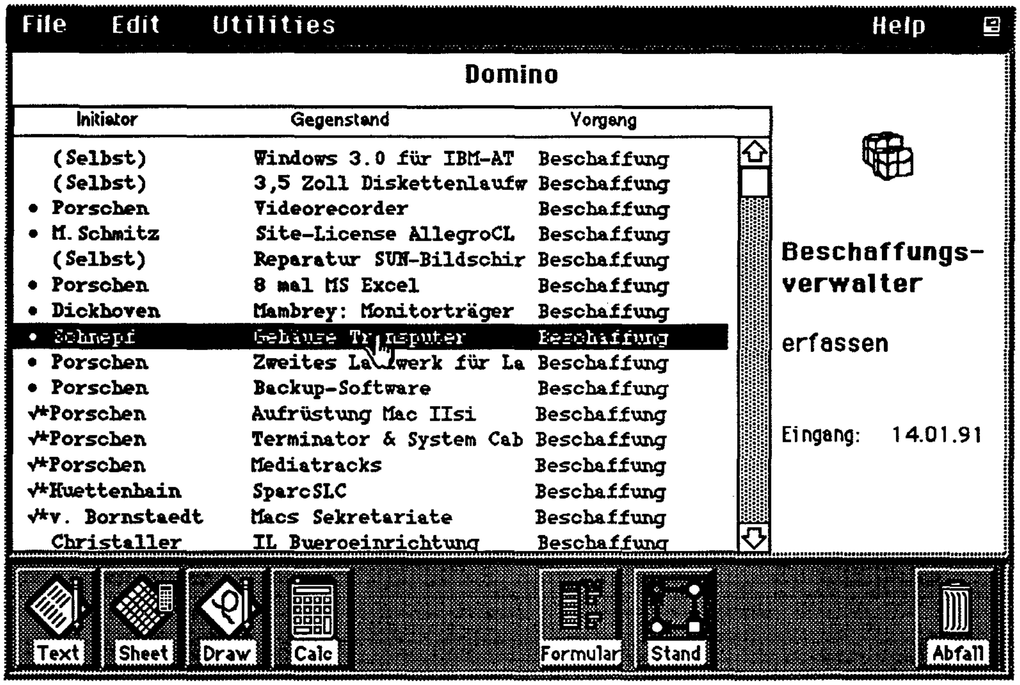
The interface consists of the main screen which gives an overview of the current procedures, a procedure form screen with the data of a single form and several auxiliary screens and dialog boxes, e.g. for tracking procedures, starting a new procedure, entering data in a personal profile, selecting enclosures, etc.

The main screen gives an overview of the procedures the user is currently involved in (see figure 3). The leading character of each entry indicates the state of the procedure from the viewpoint of the user:

(•) form needs to be worked on,
(*) form has changed (new data),
(√) procedure has been successfully terminated,
(†) procedure has been abnormally terminated (e.g. cancelled),
( ) no immediate action is expected from the user, but the procedure is still being processed.

The right part of the screen provides more detailed information when a procedure is selected. The main screen also provides functions such as updating procedure forms when new DOMINO messages have arrived, starting new procedures, saving/

deleting procedures that have terminated,sorting entries and searching for forms. By double-clicking on a procedure entry the associated form is opened.



Domino

Initiator	Gegenstand	Vorgang
(Selbst)	Windows 3.0 für IBM-AT	Beschaffung
(Selbst)	3,5 Zoll Diskettenlaufw	Beschaffung
• Porschen	Videorecorder	Beschaffung
• M. Schmitz	Site-License AllegroCL	Beschaffung
(Selbst)	Reparatur SUN-Bildschir	Beschaffung
• Porschen	8 mal MS Excel	Beschaffung
• Dickhoven	Mambrey: Monitorträger	Beschaffung
• Schnepf	Gehäuse Transputer	Beschaffung
• Porschen	Zweites Laufwerk für La	Beschaffung
• Porschen	Backup-Software	Beschaffung
√*Porschen	Aufrüstung Mac IIsi	Beschaffung
√*Porschen	Terminator & System Cab	Beschaffung
√*Porschen	Mediatracks	Beschaffung
√*Huettenhain	SparcSLC	Beschaffung
√*v. Bornstaedt	Macs Sekretariate	Beschaffung
Christaller	IL Bueroeinrichtung	Beschaffung

Beschaffungs-verwalter

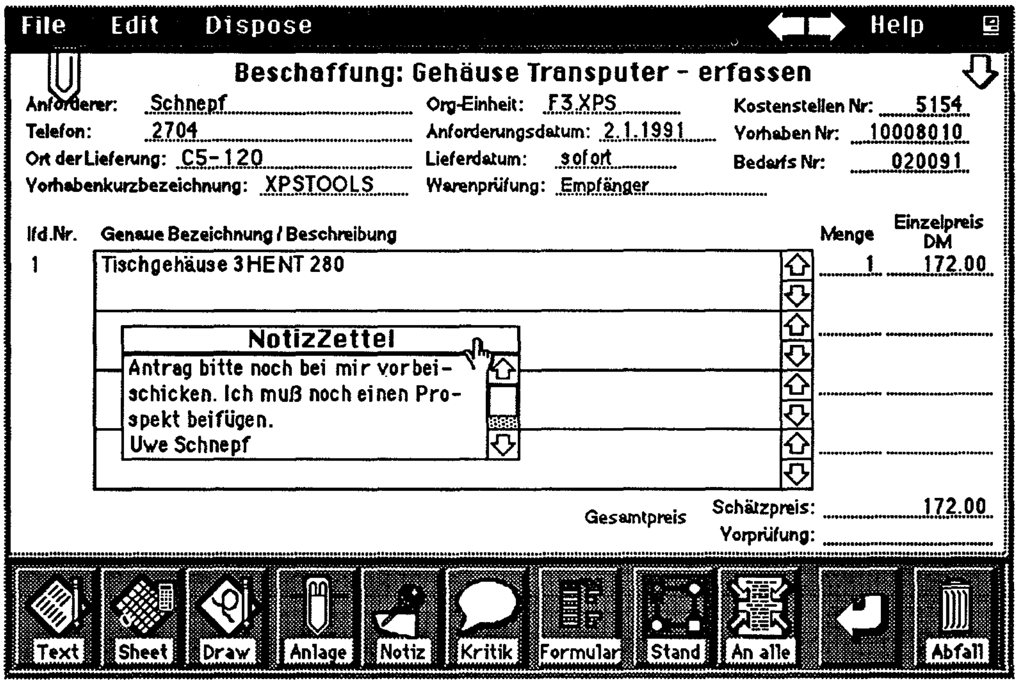
erfassen

Eingang: 14.01.91

Figure 3. DOMINO interface: The procedure overview

The form screen presents the information relevant to a procedure instance in a forms like interface (see figure 4). It may actually consist of several consecutive screens when the data that have to be displayed do not fit onto one (Macintosh) screen, as is the case with the purchase form of figure 4. The filling in of an empty form is facilitated by automatically using defaults from a user profile, pop-up menus with appropriate choices, automatic calculations, and plausibility checks. Arbitrary enclosures as well as informal note stickers can be added to a form. Complaints may be attached to any field of a form.

The "dispose" menu offers the appropriate choice of actions the user can take in order to deal with the form in the current context; the menu commands are dynamically adjusted to the current status of the procedure. E.g., when a form has come in for approval the user can choose from: approve and pass on the form, or complain and send it back. In order to get more information on the status of a procedure, including a local "history" and — more important — the current location of the form, the "Stand (status)" button can be invoked.



Beschaffung: Gehäuse Transputer - erfassen

Anforderer: Schnepf
Telefon: 2704
Ort der Lieferung: C5-120
Vorhabenkurzbezeichnung: XPSTOOLS

Org-Einheit: F3.XPS
Anforderungsdatum: 2.1.1991
Lieferdatum: sofort
Warenprüfung: Empfänger

Kostenstellen Nr: 5154
Vorhaben Nr: 10008010
Bedarfs Nr: 020091

lfd.Nr.	Genaue Bezeichnung / Beschreibung	Menge	Einzelpreis DM
1	Tischgehäuse 3HENT 280	1	172.00

NotizZettel
Antrag bitte noch bei mir vorbeischicken. Ich muß noch einen Prospekt beifügen.
Uwe Schnepf

Gesamtpreis
Schätzpreis: 172.00
Vorprüfung:

Figure 4. DOMINO interface: A purchase form

4 The Practical Test

Our institute comprises approximately 120 researchers and is divided into 5 research groups. The institute is headed by a director, the groups are headed by research group managers. The candidates for DOMINO were the clerical procedures connected with business trips, purchasing, vacations, etc. Our discussions with the management showed that the purchase procedure would be the most suitable test application, because it involved several steps of processing within the institute. Although DOMINO is meant for modelling and monitoring a variety of procedure types at the same time, the other procedures would have only been useful if the administration department had participated in the test. This turned out to be too difficult for technical reasons (incompatible computer networks and systems).

The internal purchase procedure in our institute consists of four steps. After a purchase form has been filled in by the prospective orderer (the "initiator"), the purchase form has first to be signed by the research group manager who checks with his own budget. Then the form is passed on to the network manager. He is responsible for checking the technical details of a proposed purchase and the compliance with the institute's purchasing policy, e.g., compatibility with existing machinery (network of over 150 machines, personal computers, work-stations and servers). The third step, the final approval by the director of the institute, only becomes necessary if the purchase price exceeds 2000 DM. Otherwise, the signature of the research group manager is sufficient. In the following and last step, the technical support person does the necessary bookkeeping, since he is responsible for the institute's purchase budget and the inventory of the technical equipment. Now the purchase form leaves the institute and is passed on to the purchase department which actually carries out the purchase. The (electronic) DOMINO procedure ends at this point, and the technical support person produces a paper equivalent of the electronic form along with the enclosures, if any. This paper form is then signed by a manager and sent off to the purchase department.

After installation of the procedure itself, a small group for an initial test was selected in which the system would be used only for fictitious purchases. This group consisted of five people who played the roles of the procedure (partly their real life role) with the development team as initiators of purchases. This test phase took approximately three months from June to August, 1990, and resulted in a number of minor modifications concerning technical and organizational details (apart from the identification and elimination of quite a number of bugs).

The official introduction of DOMINO for purchasing in our institute was then discussed with the research groups and the research group managers. Since the majority of employees place orders very seldom or never at all, it was decided to have two to five "purchasers" per research group who could act as initiator on behalf of others. This resulted in a user community of 22 people: six "officials" and 16 "purchasers". This group had the opportunity to play with the system for three weeks, and since mid October, 1990, DOMINO is in official use for purchasing in our institute.

5 The Experiences

In this section, we present our first experiences with the DOMINO system after three months of operation. This does not come near a systematic evaluation of the system (and was not intended as such) since our experimental basis is too narrow: the user community was very limited, we had only one type of procedure and not a variety of procedures, the time period was rather short, and the intensity of usage was quite inhomogeneous within the user group.

But apart from these limitations, we think that the qualitative judgements we have derived from both positive and negative experiences during our current experiment are still quite valuable. They should not be overestimated for the reasons given above, but they still represent first answers to a number of questions we had about the DOMINO system: How good is the user interface? Do the potential advantages of an office procedure system show in practice? Are the DOMINO procedure and process model adequate for practical use? We did not take a systematic quantitative approach to get these questions answered, but rather an informal approach with individual conversations and user meetings. During system installation members of the development team gave a short introduction to the system. We had discussions with users on various occasions (a user had difficulties with the system, a bug turned up, a new system version had to be installed, etc.). Additionally, we had two user meetings where questions, suggestions and experiences were discussed.

Some of the observations we made are DOMINO specific, some are groupware specific, and still others could have been observed with the introduction of any software system in an organization, but are of a specific quality since the software system is a multi-user application.

User interface. In general, the user interface was judged as easy to use and mainly self-explanatory. Sometimes, missing context was criticized: "Who will be at the next station in the procedure?", "Whom does a complaint go to?", "What have I already done with a procedure form?" (the latter especially with people frequently interrupted during their work with the system).

Keeping track of purchases. The better trackability of purchases was appreciated: The person currently processing the purchase order can be looked up any time by the people involved in the procedure. The usefulness of this feature would have been even higher had we succeeded in including the purchase department which is responsible for later stages of a purchase.

Unified treatment of purchases. One of the main benefits of an office procedure system is the unified treatment of all purchases according to the rules laid down in the procedure definition. This also results in a more complete, consistent, and up-to-date budgeting for the institute as a whole as well as for the research groups. However, this can only be achieved if the system is used throughout. In the first months, the use of DOMINO was not strictly enforced so that there were still a number of procedures run with paper forms. The main reason was that paper forms can be run quicker through our institute than their electronic counterparts if the initiator takes the trouble of running around and collecting the necessary signatures on the fly. This mixed mode of operation resulted in extra work (see below) and hampered the advantages of using DOMINO .

Suitability of the DOMINO procedure model. First, the DOMINO procedure model with its net of actions linked together by input/output relations and its potential of mixing parallel and alternative courses of action turned out to be too complex. For the task at hand — purchase procedure processing — which could be characterized as a hierarchical signing process, a simple sequential procedure model would have been sufficient. Secondly, the strict input/output relations between the actions of a procedure do not allow the data produced in one action to be changed in a subsequent action. While this safeguards against unauthorized changing of procedure data, the mechanism is rather rigid in that it requests the procedure to be set back to the person who produced the data to be changed. This was a source of much discussion and we had to employ a work-around for some cases. Of course, this will be different in different application environments.

Suitability of the DOMINO processing model. The conversation-based DOMINO model of procedure processing offers some provisions for exception handling. In addition to the normal course, the processing of a procedure instance can vary in the sense that it may be set back by the procedure control system when one participant complains about, e.g., missing prices, reasons given for purchase, or when the initiator cancels the procedure altogether; however, this exception handling facilities were considered not flexible enough, and there was felt a need for

Integration of informal communication. Especially officials felt themselves "fenced in" by the system features they had to use. The most common complaint was the missing possibility for letting another (arbitrary) person have a look at the purchase form and get it back with her or his comments in an informal way (ad-hoc comment feature). In general, a smooth transition from office procedure processing to more informal ways of communication with respect to the procedure form was missed. This ranged from the possibility of sending forms to other persons just for their information to the before mentioned comment feature.

Grouping procedures. In the DOMINO system, each procedure instance is treated separately. There was, however, the requirement — especially during later stages of the procedure — for grouping procedure forms for further processing: saving them jointly in an archive, dealing with them the same way (e.g., same complaint), or simply creating a local context important for budgeting or tracking purposes.

Lack of integration of other tools. While this seems to be a general problem with CSCW applications, DOMINO users complained particularly about two issues in this area:

(a) A spreadsheet program popular with those concerned with purchasing in our institute could only be "integrated" with purchase procedure processing via a not too comfortable copy and paste mechanism (same would have been true for other programs or procedures).

(b) DOMINO messages are treated separately from ordinary electronic mail — the user has to switch tools to send an e-mail message concerning an office procedure.

Media specific communication problems. The interleaved use of the paper and computer medium turned out to be some problem ("media clash"). The necessity for this mixture arose partly from the limited organizational domain in which DOMINO was in use, and partly from the fact that the provision of paper documents in addition to the purchase form is quite common. When the purchase procedure leaves our institute for the purchase department, a paper version of the form has to be generated, possibly along with enclosures. At the moment, this paper form has still to be signed, and sometimes paper documents, mostly advertisements have to be added. This results in overhead and diminishes the potential benefits of a computerized office procedure system. Possible solutions to this type of problem are the acceptance of computer generated signatures, a more complete coverage of the organization, and simple-to-use scanners.

A different type of communication problem arises through the use of the computer medium itself. Communication gets more indirect and more explicit at the same time. This tends to result in a certain uneasiness with some users using the system, especially with "negative" communication acts (e.g., complaints). Even in the initial testing phase, users did not very much playfully explore the system as is quite common when trying out other (single user) Macintosh applications. The main reason seems to be that the current system does not give too much cooperative context (no explicit representation of procedures, of people involved) so that users may be in doubt which is the next or previous station, who exactly is receiving their comment or complaint.

Overhead-benefit relation. The question of who is paying for the overhead of a computer group support system and who is going to receive the benefits is crucial for its success (Grudin, 1988). The filling in of the electronic forms was in general not regarded as too much overhead by the initiators when compared with the benefit, i.e. the better trackability of the purchase forms. This may be attributed to the filling-in helps implemented in the user interface (defaults, pop-up menus, automatic calculation). The technical support person, however, complained about an additional workload. The reason was mainly the parallel use of paper forms for purchasing which created two different work modes at his desk. The potential benefit he anticipated — a more accurate budget since no paper form would slip through to the purchase department without passing his desk — could have been only realized by exclusively using DOMINO for purchasing.

6 Conclusion and outlook

We have presented a new version of our office procedure system DOMINO with a form-oriented interface which is running on a network of personal computers and Unix server machines (Macintoshes and SUN's). We have put this system to use for the purchasing process in our institute (as a candidate of a clerical procedure involving more than one step of processing). Comparing the efforts — the development of the new DOMINO interface and the introduction of the system to a user community — with the feed-back we received, we think that the experiment was (and is) a worthwhile exercise.

Our first experiences show that we have succeeded in building an easy-to-use procedure system for an office environment, which is able to demonstrate the potential benefits of such a system. The experiences also exhibited a number of problems. While some of these problems might be attributed to the somewhat limited organizational domain in which DOMINO was used or the initial mixed mode of paper and electronic forms, some of these problems indicate weaknesses of the system, which may be summarized as follows:

- The DOMINO procedure and processing model with its pre-structured net of actions and its given exception handling facilities turned out to be too rigid or ineffective in some respects.
- Easy transition to more informal or simple ways of communication and cooperation was felt to be missing.
- The environment for working on office procedures was lacking tailorability, e.g. individual grouping of procedures, or integration of personal tools.
- The organizational or group context was not or not adequately represented.

Some of these weaknesses may be overcome within or around DOMINO, e.g. the CoPlanX conversation could be extended to include the ad hoc comment feature mentioned above, or e-mail and simpler cooperation support tools could be integrated into the DOMINO environment. We have indeed begun to implement such simple tools like circulation folders and information requests with replies. Circulation folders are sent around in sequential way to a number of persons and can carry any kind of documents[2]. With information requests with replies, the request is posed in parallel to a number of people and the arrival of answers is monitored by the tool. We will integrate these new tools, e-mail and the DOMINO system and put them to use in the same organizational environment.

As a second consequence, we think that more flexible group support tools are needed which complement office procedure systems like DOMINO and lend themselves more easily to serve as a medium for groups as well as individuals to organize their work in areas which are not dominated by pre-structured procedures. Our current research aims at such tools for the coordination of distributed work which are to address the above key issues: flexibility and configurability, easy transition to informal communication, better overview of individual work and its group context. So, our DOMINO experiences have contributed to our direction in CSCW research: from pre-structured cooperation to unstructured or user configurable/modifiable cooperation patterns, from processing of "official" procedures to coordination of day-to-day work in a rich environment allowing for different views on tasks and representing the group context more explicitly, and from a coordination model governed to a large extent by the formalized conversation paradigm to a coordination model where there is still structured interaction but also nonformalized communication, conferencing, and some simple ways of information sharing.

Acknowledgements

We are indebted to our DOMINO users for their readiness to participate in our experiment and their criticism, and especially Peter Hoschka for many valuable suggestions concerning the user interface.

[2] We favor an even simpler processing model than that of the circulation folders of Karbe and Ramsperger (1990).

References

Grudin, J. L. (1988): "Why CSCW applications fail: Problems in the design and evaluation of organizational interfaces," in Proc. CSCW '88, ACM, New York, NY, 1988, pp. 85-93.

Karbe, B. H. and Ramsperger, N. G. (1990): "Influence of exception handling on the support of cooperative office work," in S. Gibbs, A. A. Verrijn-Stuart (eds.): *Multi-User Interfaces and Applications*, Proc. IFIP WG 8.4 Conf. on Multi-User Interfaces and Appl., Heraklion, Crete, Greece, Sept. 24 - 26, 1990, North-Holland, Amsterdam, 1990, pp. 355-370.

Kreifelts, Th., Licht, U., Seuffert, P. and Woetzel, G. (1984): "DOMINO: A system for the specification and automation of cooperative office processes," in B. Myhrhaug, D. R. Wilson (eds.): *Proc. EUROMICRO '84,* North-Holland, Amsterdam, 1984, pp. 33-41.

Kreifelts, Th. and Woetzel, G. (1987): "Distribution and exception handling in an office procedure system," in G. Bracchi, D. Tsichritzis (eds.): *Office Systems: Methods and Tools,* Proc. IFIP WG 8.4 Work. Conf. on Methods and Tools for Office Systems, Pisa, Italy, Oct. 22 - 24, 1986, North-Holland, Amsterdam, 1987, pp. 197-208.

Spenke, M. and Beilken, Chr. (1989): "A spreadsheet interface for logic programming," in *Proc. Conf. on Human Factors in Computing Systems (CHI '89),* Apr. 30 - May 5 1989, Austin TX, ACM, New York NY, 1989.

Winograd, T. and Flores, F. (1986): *Understanding Computers and Cognition: A New Foundation for Design*, Ablex, Norwood NJ, 1986.