

A guide through the construction of a groupware for efficient knowledge management

Myriam Lewkowicz, Manuel Zacklad

▶ **To cite this version:**

Myriam Lewkowicz, Manuel Zacklad. A guide through the construction of a groupware for efficient knowledge management. COOP, May 2000, Sophia Antipolis, France. hal-02310969

HAL Id: hal-02310969
https://utt.hal.science/hal-02310969

Submitted on 10 Oct 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A guide through the construction of a groupware for efficient knowledge management

Myriam LEWKOWICZ, Manuel ZACKLAD
Laboratoire Tech-CICO
Université de Technologie de Troyes
12, rue Marie Curie BP 2060
10010 Troyes Cédex France

Abstract. The main idea in this article is that groupware benefits organizational memory because it focuses on communication and coordination, but is inadequate for an efficient knowledge management. It has to be completed by a model for exchange structuring in order to improve dialog quality and to enable a conversation classification that would not be simply chronological. Our aim is to build a groupware (MEMO-Net) enriched with such a model. This model (DIPA), which uses and simplifies the concepts of Problem-Solving methods, comes from a review of existing Design Rationale formalisms that gave rise to the ABRICo formalism.

1. Introduction

The result presented in this article belongs to a research project whose aim is to manage knowledge used in design project for capitalization and reusability. In accordance to Zacklad and Grundstein ([24]), knowledge capitalization research can be classified in three categories: social and cooperative approaches, top-down modeling approaches and bottom-up modeling approaches. In the first category, one considers that organizations' critical knowledge comes within a collective competence that is not enough or bad formalized. The development of systematic use of groupware, electronic-mail services, newsgroups, workflow, embodied in Intranets and particularly used in design projects would seem to explain that these tools are indeed considered potential aids for the knowledge capitalization process.

If groupware mediatizes designers' interactions, the best way to locate and protect crucial knowledge (in the sense of M. Grundstein, [13]) exchanged through these nets is to study the interactional structure and to suggest information structuring tools and models for highlighting exchanged knowledge and enabling easier future access. Most of the time in knowledge capitalization projects, there is a lack of quality rather than quantity of information; quality regarding the structure of memorized materials.

In "social and cooperative" knowledge capitalization approaches, there seem to be two types of approach. Some work will aim for *a posteriori* structure information by reconstituting, from the traces of intellectual transactions ([25]), the concept structure previously elaborated collectively. Others will aim for *a priori* structure transactions to guarantee a better quality of both interactions and write track of these interactions that will enable an easier re-exploitation by the "knowledge managers".

This interest in *a priori* structuring of problem-solving processes in order to guarantee a exploitation is not recent. In CSCW (Computer-Supported Cooperative Work) research, several authors ([6], [7]) have already expressed wish to switch from a "object-centered paradigm to a "process-centered" paradigm. In the last one, designers' interaction (that is to say questions, decisions and conversations that form the elaboration environment of the objects) would be memorized as well as objects and design process results.

Following this paradigm, our previous works dealt with Design Rationale, criticized its classical methods and proposed a new formalism, ABRICo ([16]). In this paper, we present DIPA, an evolution of this formalism thanks to our interpretation of knowledge engineering results on problem-solving methods, and its implementation in a groupware, MEMO-Net.

2. CSCW and Groupware

Groupware offers multi-users interfaces to access electronic mail services, forum, and workflow, and to put into practice CSCW (Computer-Supported Cooperative Work) methods, or, as Malone says (quoted in [5]), it is "information technology used to help people work together more effectively". CSCW may be seen as the scientific discipline that guides thoughtful and appropriate design and development of groupware [11]. These technologies represent a changing paradigm in computer science, because they deal more with problems due to human-human coordination and communication than with defining human-computer dialogues for automated procedures.

There are two ways of viewing the variety of groupware, a time/space taxonomy and an application-level taxonomy. They are embodied in this classical 2x2 matrix in [9] or [1]:

	Same time	**Different times**
Same place	*face to face interaction* Meeting Rooms GDSS (Group Decision Support System)	*asynchronous interaction* Project Management tools
Different places	*distributed synchronous interaction* Videoconferences Shared screens	*distributed asynchronous interaction* E-mail Forum Cooperative Writing

As Grudin ([12]) stresses, technology alone cannot provide an efficient introduction of this kind of tools in organizations. It is essential to understand how groups and organizations function and evolve. This comprehension enables the choice of the right tools to fit existing communication flows and to crystallize organizational memory elements without penalizing users [6].

But, if a groupware is capable of recording solution elaboration processes, it is not sufficient for efficient knowledge management ([12]). It has to be completed by a method that structures the exchanges for a better quality of dialogue and for a management of ideas

that would not be simply chronological. These reflections have led to the IBIS method [2], connected afterwards to Design Rationale research.

3. Design Rationale

The IBIS (Issue-Based Information System) method [2] aims to improve the quality of design dialogue processes by structuring discussion in complex problems. It uses several categories: issues, positions that are possible solutions for the issues, and arguments for or against these solutions. This method has been implemented in a graphics tool (gIBIS) and a textual one (itIBIS). Work with IBIS falls within Design Rationale research from the Human-Computer Interaction community. Design Rationale research usually concerns problems appearing in the capture of the rationale followed by the articulation, representation and use of this rationale made explicit. In a first synthesis of these works, edited in a special issue of *Human-Computer Interaction*, Carroll and Moran [3] described the importance of this field by stressing the following points:

According to them, constructing explicit Design Rationale could:

- support reasoning processes in design,
- facilitate communication among the various players in the design processes (designers, implementers, maintainers, users, etc.),
- further the accumulation and development of design knowledge throughout design projects and products.

IBIS and its different implementations have been experimented many times more or less successfully. In a general way, there are two reservations about this method [10] : (a) the model used to represent argumentation is too schematic and (b) its nature is exclusively «dialogal».

The first reservation concerns the non-representation of interdependency between issues. Relations between several options can not be represented; everything is done as if each discussion mediatized by the tool would correspond to an independent sub-problem.

Secondly, the non-deliberated issues are ignored. If a question is not debated, the tool will not stress it, although it could correspond to an important point that could influence the design process.

We made somewhat similar criticisms of the QOC (Questions Options Criteria) method ([17]) and classical Design Rationale formalisms in our previous work ([15], [16]). According to us, formalisms like QOC seem to be relevant for simple, short-lasting design situations where one has to choose between several options and where the shape of the final solution is known. But they are not suitable for collective design situations that we have called complex (unknown solution shape, progressive elaboration of a unique long-lasting solution), where we need to take into account the process dynamics and the participants' roles. We then proposed to represent these complex processes with an original formalism that we called ABRICo (from the French words Accords, Buts, pRopositions, Interprétations en Conception, that mean agreement, goals, propositions, interpretations through design). Its static model is described below (figure 1).

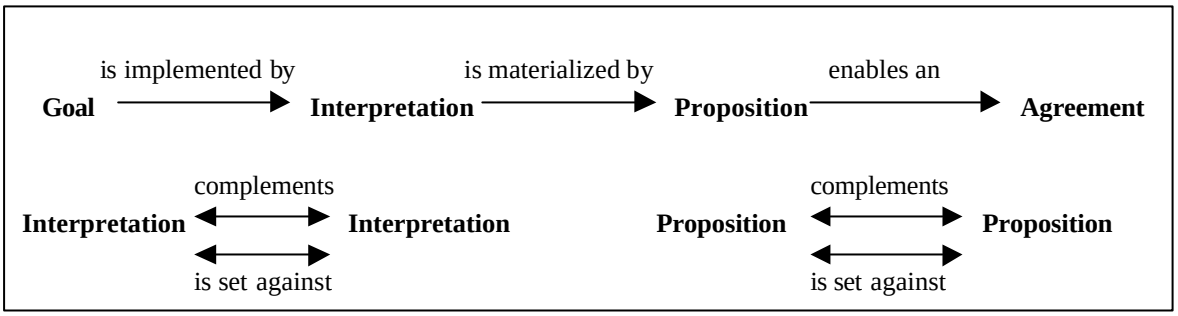


Figure 1 : static model ABRICo

We tested this formalism by representing real complex collective decision-making situations. Seeing that this experimentation was conclusive, we built a first version of a tool based on these concepts, MEMO-Net, and we submitted it to professional groups for a first evaluation. Unfortunately, the model proposed to users seemed to be too abstract for easy comprehension, a conclusion that re-opened the whole question of the ABRICo model and the first version of the tool.

One reason for the difficulty of implementing ABRICo was, from our point of view, that the complex decision process formulation was too far from the design situations which people were confronted with. Where the traditional but simplistic Design Rationale models enable an easy appropriation, the ABRICo model, although more realistic from a cognitive point of view, appeared them both too theoretical and too far from concrete implementation conditions in the work situations for which we built it.

4. DIPA, a collective problem-solving model

In order to approach the cognitive dimension of reasoning, we then enriched ABRICo with problem-solving method concepts from knowledge engineering. This enrichment was translated into a new model, DIPA (from the French words Données, Interprétations, Propositions, Accord, meaning facts, interpretations, propositions, agreement). This model has itself two declinations according to the situations that lead the actors to give priority to either analysis or synthesis processes (for example as in KADS methodology [20]). This link with problem-solving methods seems to us a natural evolution in our researches of more realistic Design Rationale models suited to the complexity of real projects. Below is an outline of the characteristics of the three types of design meeting dialogues modeling that we studied or built.

1) *Description of design meeting dialogues with classical argumentation models from Design Rationale :*

   Actors participate in an argumentative dialogue; they defend contradictory or exclusive options, quoting criteria or arguments in favor of their position.

2) *Description of design meeting dialogues with an argumentation model corresponding to complex situations in ABRICo sense.*

   At the same time as putting forward propositions, the actors debate different possible interpretations that may justify these propositions. Interpretations, like propositions, are not necessarily contradictory or exclusive. An interpretation, regarded as belonging to a more abstract level, may be evoked without necessarily being part of the defence of a point of view. At the same time, the argument semantic is not considered as entirely independent of the roles of the persons who introduce them.

3) Description of design meeting dialogues with an argumentation model inspired by problem-solving models from knowledge engineering (DIPA)

Problem-solving models replace decision-making processes, even complex ones. Each argument is categorized by its role in the problem-solving method. The model comes in two forms, according to the kind of process that it assists (analysis or synthesis). This reference to problem-solving models allows the integration of an important knowledge category that was not taken into account in the two previous models, the "problem data". Actually, we could say that Design Rationale models have neglected the "information" phase of Simon's decision-making process [19], and have only taken into account the solutions selection phase. Models from Artificial Intelligence do not have this fault. In the DIPA model, the reasoning progresses in three major steps:

- i. a problem description step plus collecting of data, considered as symptoms in analysis situations and as needs in synthesis situations;
- ii. an abstraction step going from the collecting of problem data to their interpretation corresponding to a possible cause in analysis situations, and to a functionality in synthesis situations;
- iii. an implementation step that going from an interpretation (cause or functionality) to the elaboration of a proposition that is a corrective action removing the symptom's cause (analysis) or the means suitable for the expressed functionality (synthesis)

5. Relevance of a unique model for analysis and synthesis

The fact that we had to present both analysis and synthesis models to designers teams may seem amazing. Actually, it might appear natural at first glance to propose only synthesis models and their variants (routine design, configuration…). But our practical experience of design meetings showed us that analysis activities are frequent. For example, as soon as a prototype has been developed, its function analysis will give important information that will be reintroduced in the process of solution finding.

These observations are also in accordance with cognitive ergonomic psychology ([8]) results that teach us that design situations in the organizational sense in fact generate two distinct phases of activity: solution generation and then evaluation of these solutions. The first corresponds to synthesis problems in a KADS sense and is close to design models in this method. The second corresponds to analysis problems whose diagnosis models are well-known.

This idea of a unique model (figure 2) to represent the two types of activity is quite close to some interpretations of the heuristic classification [4]. Whereas Clancey compares, for example, methods coming within heuristic *classification* (where one select a solution and then tries to prove that it is the one that fits best), with methods coming within heuristic *construction* (where structure and behavior models are used to construct new solutions), Zacklad and Fontaine ([22]) take a different position.

According to these authors, in both types of situation, there is both exploitation of knowledge from previous solved cases, and construction of an original solution. This solution is a proof or a justification in analysis case and a constraints-compatible approximate solution in synthesis cases. These authors defend the idea that, at a certain abstraction level, it is a matter of the same heuristic reasoning form. This reasoning is characterized (i) by the use of qualitative inferences using the componential semantic of the concepts and (ii) the non-hierarchical contact of concepts coming from different classification hierarchies (this second idea comes directly from Clancey's definition of the

heuristic classification [4], p.294). We feel that this notion of non-hierarchical contact of concepts, that corresponds to the "heuristic matching" in the original heuristic classification model, is at the bottom base of heuristic reasoning used in design meetings, both in analysis and synthesis phases. Several authors have proposed convergent conceptualizations of this kind of reasoning. For example, for Hoc ([14]) it corresponds to exploiting the "implementing" hierarchy. The hierarchical relation does not correspond to running through a class hierarchy, but running through different levels that enables a physical implementation while starting from systematic description of a system's aims via different functional views.

In PAGIC[1]([23]), this reasoning form is described as a navigation between different points of view about the same system that comes into action when designers take into account the "optical" dimension of the distinct "abstraction" of the hierarchical dimension of the "generalization".

Whereas, in the DIPA model, the abstraction and implementation inference steps are two symmetrical aspects of the same heuristic reasoning, applicable both in analysis and synthesis[2]. In abstraction cases, for example, the point of view about any one system will change according to whether the symptoms or their causes are considered most important, or even the requirements of internal system functions.

The formalism used to describe DIPA was inspired by KADS ([20], [18]) but does not strictly follow the KADS conventions as to how to represent inference structures.

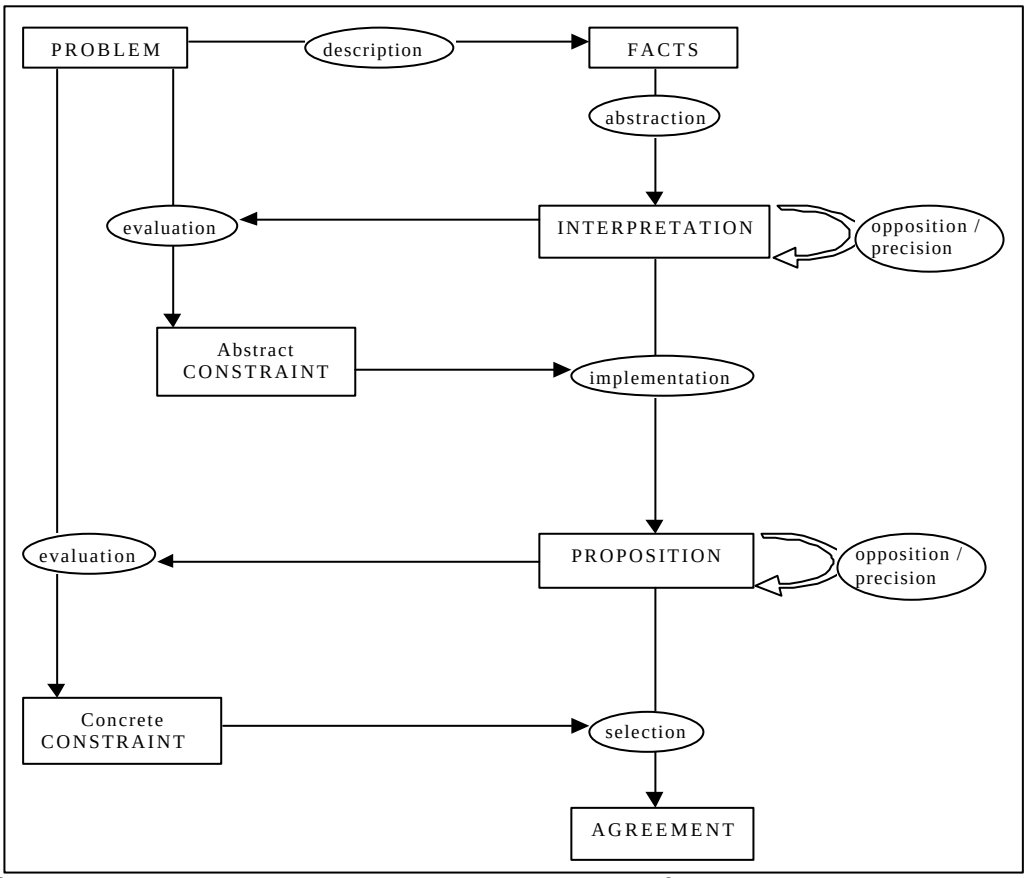


Figure 2 : DIPA, a heuristic model of design reasoning for analysis and synthesis

[1] PAGIC is a French acronym for Partie-tout, Abstraction, Généralisation, Interaction, Cybernétique.

[2] In KAL ontology that formalizes heuristic classification's reasoning ([21]), the abstraction step corresponds to abductive propositions and the implementation step to "constructive-deductive" propositions.

DIPA	DIPA synthesis	DIPA analysis
Problem	Goal	Malfunction
Fact	Requirement	Symptom
Interpretation	Functionality	Cause
Abstract constraint	Constraint	Constraint
Proposition	Means	Corrective action
Concrete constraint	Constraint	Constraint
Agreement	Choice	Choice

Table 1 : Implementation of DIPA model for synthesis and analysis activities

6. Implementation of DIPA model : MEMO-Net groupware

We implemented the DIPA model to build the MEMO-Net groupware. This system consists of two modules, one for synthesis phases (named "design" in the interface), and the other for analysis phases (named "diagnosis" in the interface). Its goal is to allow a project team to solve problems met during design by alternating the two types of activity on a cooperative way. The exchange structure allows both to guide the solution process and to organize the arguments, particularly in argument capitalization aspects.

In the diagnosis module, members of the project team identify a dysfunction and evoke symptoms, causes or corrective actions. In design, once the goal is known, the actors evoke requirements, functionalities and means. To contribute, people click on the following signs (indicating a malfunction, symptom, cause, and corrective action) and then create the corresponding forms:

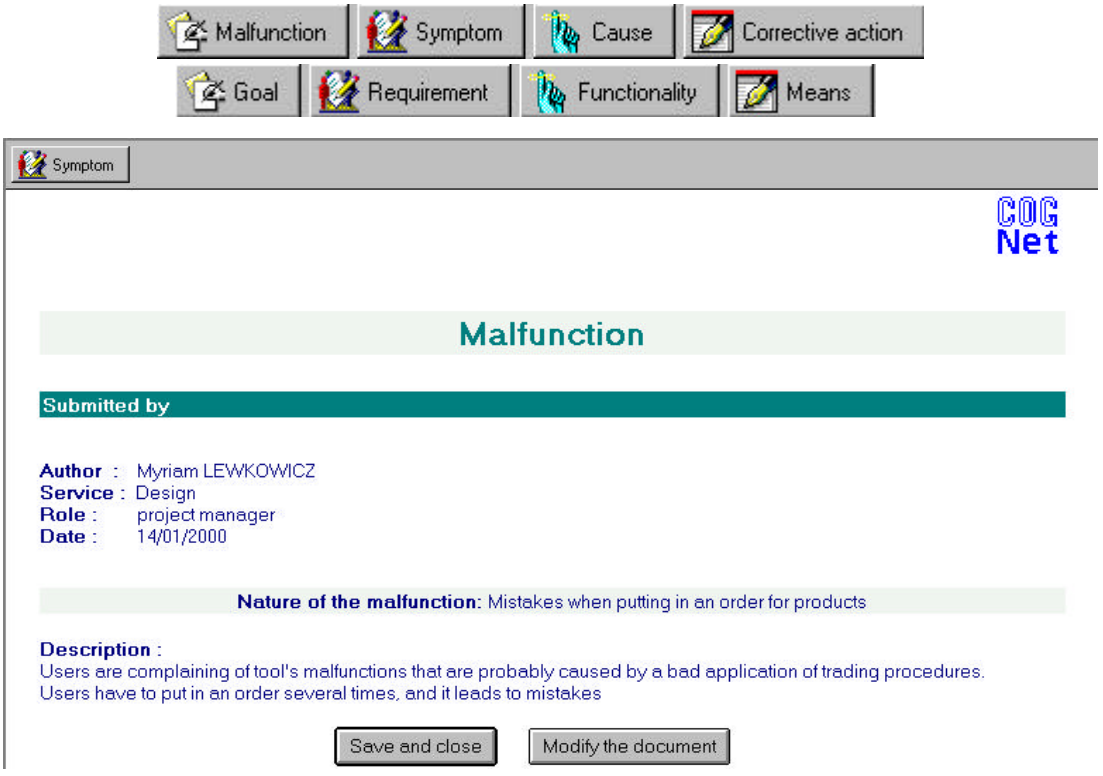


Figure 3 : signs for new forms and a form to describe a malfunction

Contributions are classified chronologically or according to DIPA model categories, or to the authors' names, their roles, or their department.

The following example concerns a team of designers of a trading application. The malfunction noticed is that the users of the application make many mistakes when they put in an order for products. The first symptom seemed to be caused by interface faults, and improvements are suggested, as well as a constraint: the interface has already been modified twice. Another possible cause is lack of training, and two corrective actions have been suggested: training seminars or free books of trading rules. The screen copy below (figure 4) shows the chronological view of contributors' propositions. Figure 5 presents a summarized view in which these contributions are classified by concepts.



Figure 4: Chronological view of a diagnosis problem-solving process

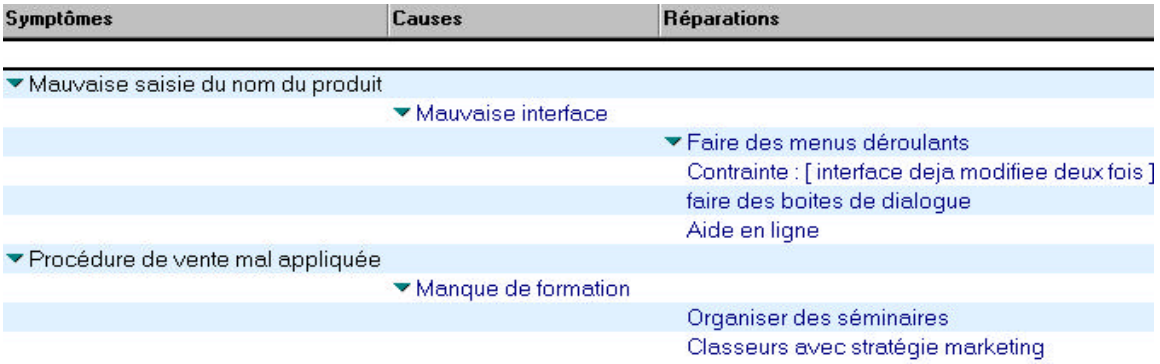


Figure 5 : Summarized view of a diagnosis problem-solving process

The second example that presented below, with, as previously, a chronological view (figure 6) and a summarized view (figure 7) concerns a team of researchers wanting to construct an Intranet for their laboratory, using Lotus Notes. Different needs are mentioned, such as announcing seminars or accessing particular documents. These needs refer to functionalities e.g. managing a documentary fund. Finally, various means to obtain these functionalities are suggested, e.g. a “News module” in a Lotus base or the creation of a Lotus library base.

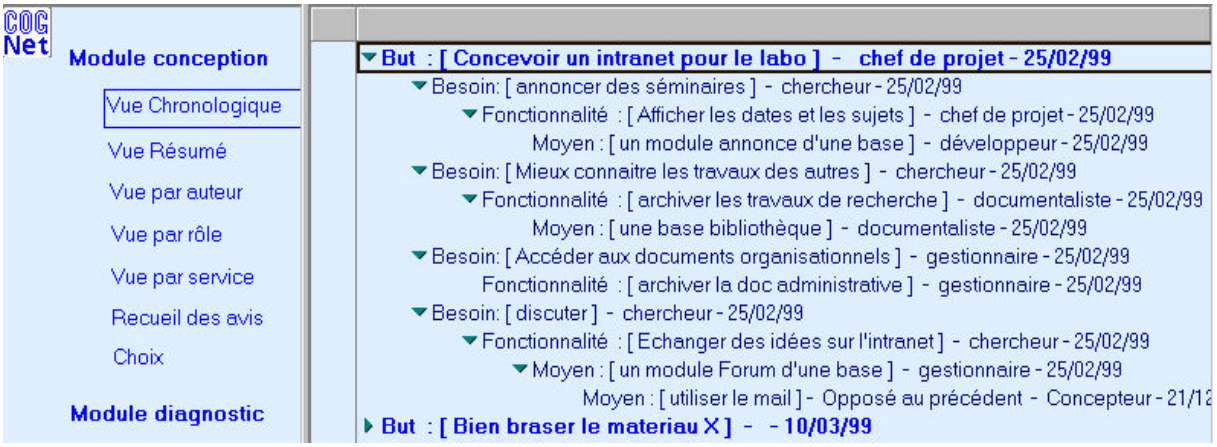


Figure 6 : Chronological view of a design problem-solving process

Besoins	Fonctionnalités	Moyens
Accéder aux documents organisationnels		
	archiver la doc administrative	
annoncer des séminaires		
	Afficher les dates et les sujets	
		un module annonce d'une base
discuter		
	Echanger des idées sur l'intranet	
		un module Forum d'une base
Mieux connaitre les travaux des autres		
	archiver les travaux de recherche	
		une base bibliothèque

Figure 7 : Summarized view of a design problem-solving process

When users have already discussed a problem, one of their propositions may be submitted to others, in order to collect their opinion and take a decision. This last step corresponds to "selection" inference of DIPA model, which enables a definitive agreement on the best possible solution.

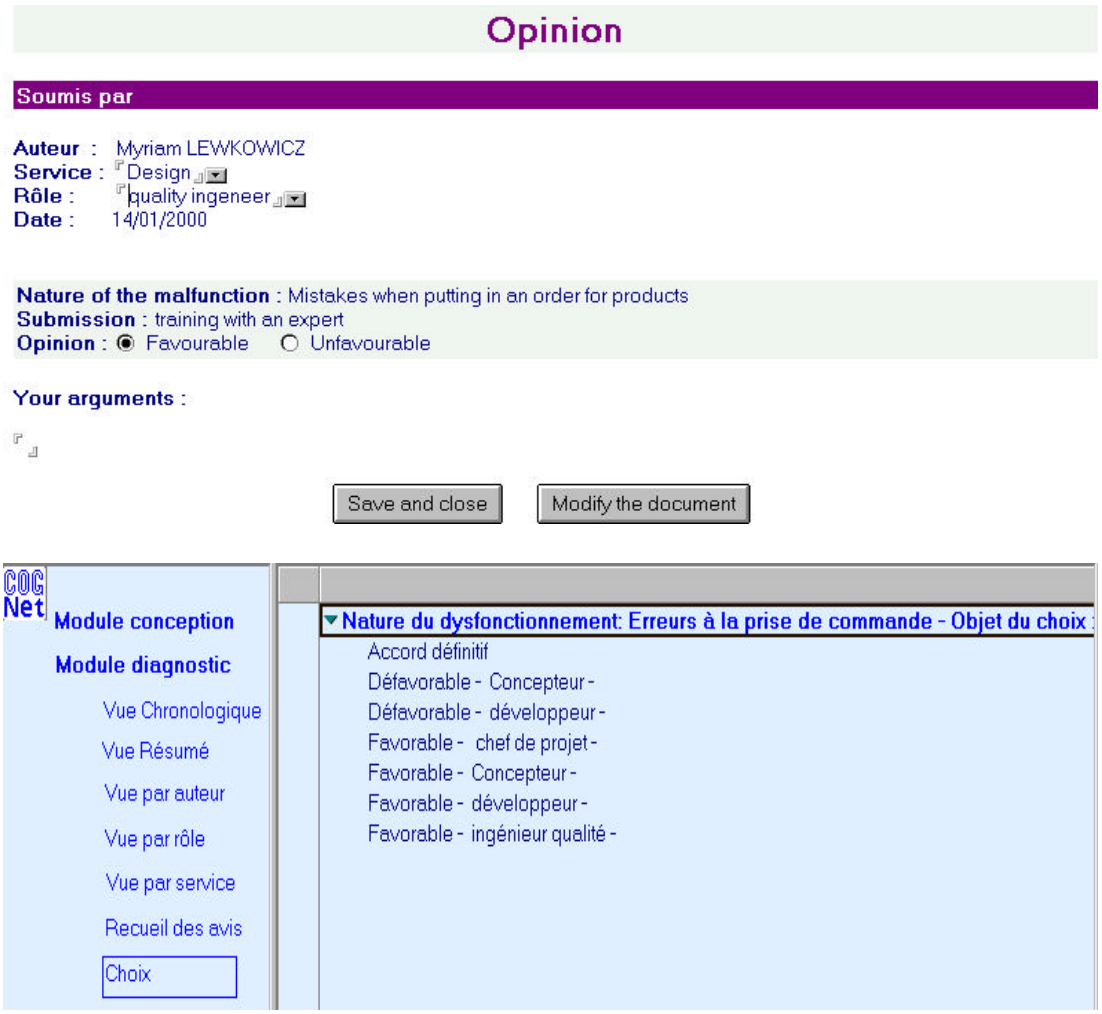


Figure 8 : A form to gather opinion plus the « choice » view visualizing previously opinions gathered

7. Conclusion

Newsgroups between experts working on a project or requests for "help desk" activities are often considered as important for knowledge capitalization. However, we think that they are not sufficient to enable an efficient management of this knowledge because the lack of structuring of the material makes it difficult to exploit.

Our idea is that the enrichment of groupware with knowledge-structuring models could enable both guiding collective work processes and obtaining reusable or exploitable knowledge.

The next steps of our research project will be to test the DIPA model via the MEMO-Net groupware, both in real work situations and with systematic experiments. In our experiments, we will evaluate the MEMO-Net influence in two ways: (a) the speed with which the participants reach an agreement and (b) the quality of the proposed solutions. To achieve this goal, we will ask two groups to solve the same problems several times, sometimes by using low-structured groupware (like a newsgroup) and other times using our tool.

8. Références

[1] Bullen, C.V., Bennett, J.L. (1993). Groupware in practice: An interpretation of work experiences, in Baecker, R. M. (Ed.) *Readings in Groupware and Computer-Supported Cooperative Work*, Morgan Kaufmann Publishers, Inc.

[2] Burgess-Yakemovic, K.C. and Conklin, E.J. (1990). Report on a Development Project Use of an Issue-Based Information System, in Halasz, F. (Ed.) (1990). *CSCW 90: Proceedings of the Conference on Computer-Supported Cooperative Work*. Los Angeles, Oct. 7-10, 1990. Association for Computing Machinery.

[3] Carroll, J.M. and Moran, T.P. (Eds.) (1992), Special Issue on Design Rationale. *Human-Computer Interaction* 6(3-4).

[4] Clancey, W. (1985). *Heuristic Classification*, Artificial Intelligence Journal, 27, pp. 289-350, 1985.

[5] Coleman, D. and Shapiro, R. (1992). Defining Groupware. Special Advertising Section to *Network World*, June 22, 1992

[6] Conklin, E.J. (1993). Capturing Organizational Memory, in Baecker, R. M. (Ed.) *Readings in Groupware and Computer-Supported Cooperative Work*, Morgan Kaufmann Publishers, Inc.

[7] Conklin, E.J. and Burgess-Yakemovic, KC. (1996). A Process-Oriented Approach to Design Rationale, in Moran, T. P. and Carroll, J. M. (Ed.) *Design Rationale Concepts, Techniques and Use*, Lawrence Erlbaum Associates.

[8] Darses, F. (1994). *Gestion des contraintes dans la résolution des problèmes de conception*. Thèse de doctorat, Spécialité Psychologie Cognitive. Paris, Université de Paris 8.

[9] Ellis, C.A., Gibbs, S.J., and Rein, G.L. (1993). Groupware some issues and experiences, in Baecker, R. M. (Ed.) *Readings in Groupware and Computer-Supported Cooperative Work*, Morgan Kaufmann Publishers, Inc.

[10] Fischer, G., Lemke, A.C., McCall, R., Morch, A.I. (1996). Making Argumentation Serve Design, in Moran, T. P. and Carroll, J. M. (Ed.) *Design Rationale Concepts, Techniques and Use*, Lawrence Erlbaum Associates.

[11] Greenberg, S. (Ed.) (1991). *Computer-Supported Work and Groupware*. Academic Press

[12] Grudin, J. (1993). Groupware and Cooperative Work: Problems and Prospects, in Baecker, R. M. (Ed.) *Readings in Groupware and Computer-Supported Cooperative Work*, Morgan Kaufmann Publishers, Inc.

[13] Grundstein, M. (1996), *La capitalisation des œnnaissances de l'entreprise, une problématique de management*, in actes des 5eme Rencontres du Programme MCX, Complexité : la stratégie de la reliance, Aix-en-Provence, 4-5 juillet 1996.

[14] Hoc, J.-M. (1987). *Psychologie cognitive de la planification*, Presses Universitaires de Grenoble.

[15] Lewkowicz, M., Zacklad, M. (1998-a). La capitalisation des connaissances tacites de conception à partir des traces des processus de prise de décision collective, *Actes de la conférence Ingénierie des Connaissances*. Pont-à-Mousson, mai1998.

[16] Lewkowicz, M., Zacklad, M. (1998-b). A formalism for the rationalization of decision-making processes in complex collective design situations, *Procceedings of COOP*. Cannes, may 1998.

[17] MacLean, A., Young, R.M., Bellotti, V.M.E., Moran P. (1996). *Questions, Options and Criteria : Elements of Design Space Analysis*, in Moran, T. P. and Carroll, J. M. (Ed.) *Design Rationale Concepts, Techniques and Use*, Lawrence Erlbaum Associates.

[18] Schreiber, G., Wielinga, B. (1993). *Model Construction*, in Schreiber, G., Wielinga, B., Breuker, J. (Eds.) *KADS a principled approach to knowledge-based system development*, Academic Press.

[19] Simon, H.A., (1947) *Administrative Behavior*, trad. Fçse (1983) Administration et processus de décision, Economica.

[20] Wielinga, B., Schreiber, G., Breuker, J. (1993). *Modelling Expertise*, in Schreiber, G., Wielinga, B., Breuker, J. (Eds.) *KADS a principled approach to knowledge-based system development*, Academic Press.

[21] Zacklad, M. (1993). *Principes de modélisation qualitative pour l'aide à la décision dans les organisations*, Thèse de doctorat, Spécialité Contrôle des Systèmes, Compiègne, Université de Technologie de Compiègne.

[22] Zacklad, M., Fontaine, D. (1996). *L'acquisition des connaissances classificatoires pour les systèmes à bases de connaissances*, in Aussenac-Gilles, N., Laublet, P., Reynaud, C. (Eds.) *Acquisition et Ingenierie des Connaissances*, Cepadues-Editions.

[23] Zacklad, M. (1996). *Cinq dimensions pour la modélisation des interfaces homme-machine : PAGIC, pages 15-25*, in actes du 5eme Colloque Ergo-IA, Octobre 1996.

[24] Zacklad, M., Grundstein, M. (Eds.) (1999). *Système d'Information pour la capitalisation des connaissances : tendances récentes et approches industrielles*, Hermès, à paraître.

[25] Zacklad, M. (1999) *La théorie des transactions intellectuelles : une approche gestionnaire et cognitive pour le traitement du COS,* Intellectica, numéro spécial sur le COS, à paraître.