Conflict Management in Concurrent Engineering: Modelling Guides

Nada Matta
INRIA (projet ACACIA)
2004 route des Lucioles BP. 93
06902 Sophia-Antipolis Cedex
e-mail: {nmatta}@sophia.inria.fr

Abstract. In Concurrent Engineering, several designers (participants) aim at designing a system (artifact), given requirements. Conflict Management in Concurrent Engineering is a complex task in which it is difficult to identify conflicts and to solve them. Reuse of generic components is generally considered as a good help to guide complex task modelling. An overview of our study to define a library of generic components to guide conflict management modelling is presented below. Conflict management methods, proposed in the literature, are classified and associated to each concurrent engineering subtask. This classification provides help to index these methods, considered as generic components in a library.

Key Words: Concurrent Engineering, conflict, negotiation, generic components.

INTRODUCTION

Concurrent Engineering (CE) aims at facilitating a product development process requiring different specialities, and at achieving better product quality [15]. So, it is important to facilitate teamwork and collaboration between participants in the development process. The main problems in CE ensue in communication between participants and collaborative decision making.

Conflict management remains a complex task in Concurrent Engineering (CE) [10]. On the one hand, it is difficult to detect conflicts and their nature precisely, and on another hand, it is complex to determine appropriate methods to solve them.

Our main objective is to define generic models for the conflict management task in CE. These models will provide generic components to be used as building blocks in conflict management task modelling for a specific application. This work is a part of the Genie[1] project, in which the CE task is analysed in order to provide guides for modelling this task and for defining corporate memory.

Our generic models are represented in CML formalism [14] and organized in the same way as the generic components in the Common-KADS library [4], which offer models to guide several task types modelling. So we have represented all studied methods and models from the literature in a single formalism. This allows to analyse their characteristics and to reveal their respective contributions. The generic models, provided by CommonKADS library for tasks such as diagnosis, design, etc., are described in CML. Therefore, it seemed natural to exploit CML to represent our generic models.

Before describing guides to model conflict management (II, III and IV), let us define some conflict types handled in CE (Cf. I).

1. Genie project is a common project between INRIA and DASSAULT AVIATION.

I. Conflict in Concurrent Engineering

At least, two Classes of conflicts can be handled in CE:

- First, *conflicts between design and requirements*. In other words, the design made by a participant doesn't satisfy corresponding requirements. This type of conflicts is handled in the same way as in mono-expertise design, when a designer tries to generate a design in order to satisfy requirements. CommonKADS library [4] and Brazier in [3] offer a number of generic models to assess the design respecting the requirements. We don't study this class of conflicts.

- Second, *disagreements between some participants* in the designers group, according to their design propositions. Such conflicts arise from problems caused by strategies used and propositions made by designers: 1) *Strategic conflicts* ensue from the inconsistency in methods and tools used by designers and in the allocation of tasks to each designer. Divergence between the participants responsibilities and failure in their cooperation [3] cause also strategic conflicts. 2) *Conflicts about propositions* made can appear from: a) misunderstanding of the participants' terminology and their points of view, b) unacceptance of the conditions under which a proposition is made and of consequences by which a proposition constrain the design cycle. Preconditions problems can be revealed from preferences like needs [13], from the use of the same resources and from difference in the requirements evaluation by the designers. Constraints imposed by the proposition and its interaction with other propositions can generate consequences problems. Note also that the quality of a proposition can cause its unacceptance (Figure 1.) shows a typology of such conflicts. In this typology objects about which problems can appear and the nature of conflicts revealed are emphasized.

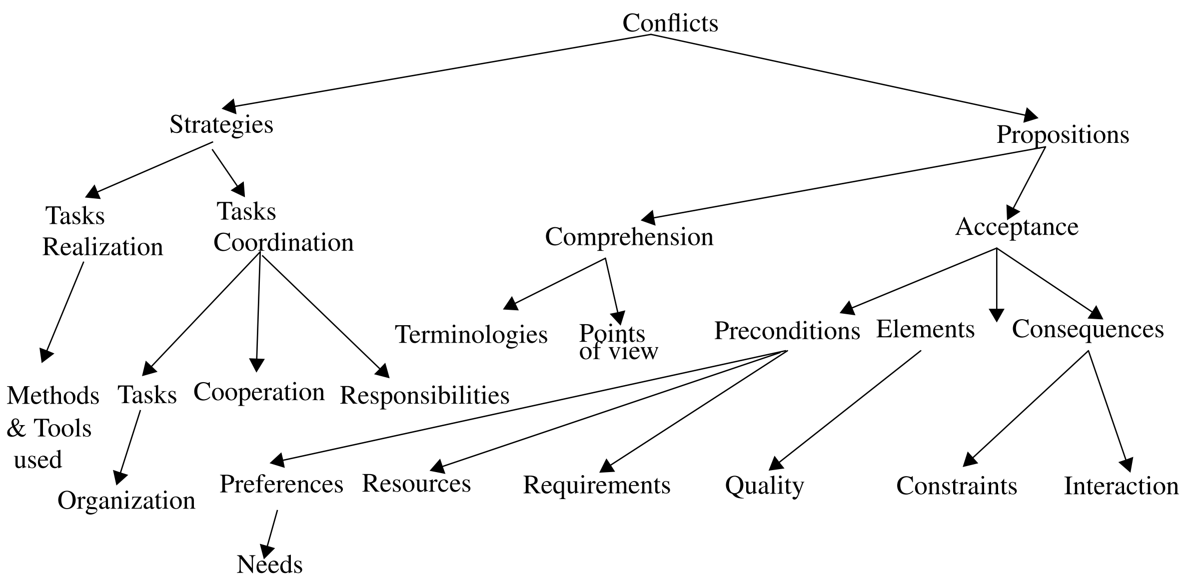


Figure 1. A Typology of Conflicts.

It is necessary to define the nature of such conflicts and explicit their contributors, to determine appropriate methods to solve them. We offer in a library a number of methods presented like generic components to manage these types of conflicts between participants.

II. Management conflict in Concurrent Engineering

To determine in which step in CE task conflicts can appear and how to manage them, we first studied a model for the CE task itself. A number of models for this task are presented in the literature like Bond's model [2] and Brazier et al's model [3]. We can distinguish some CE particularities from these models like:

- The existence of a shared model and private models [2] in CE. Each one of the private models belongs to a participant of the CE task and describes his knowledge. In fact, a private model is the expertise model (an explicitation of the model of his knowledge) of a participant. These expertise models are private and they are not shared. However, shared model describes shared knowledge of the system to be designed.

- Modifications can be made in requirements as well as in artifact [3]. So conflicts can be revealed and managed in requirements as well as in artifact model modifications.

After studying these models, we propose a generic model for concurrent engineering task, in which private propositions are combined with coordination (Figure 2).

After studying such models of CE task, we propose a generic model for the concurrent engineering task (Figure 2.). We distinguish three main subtasks in this model:

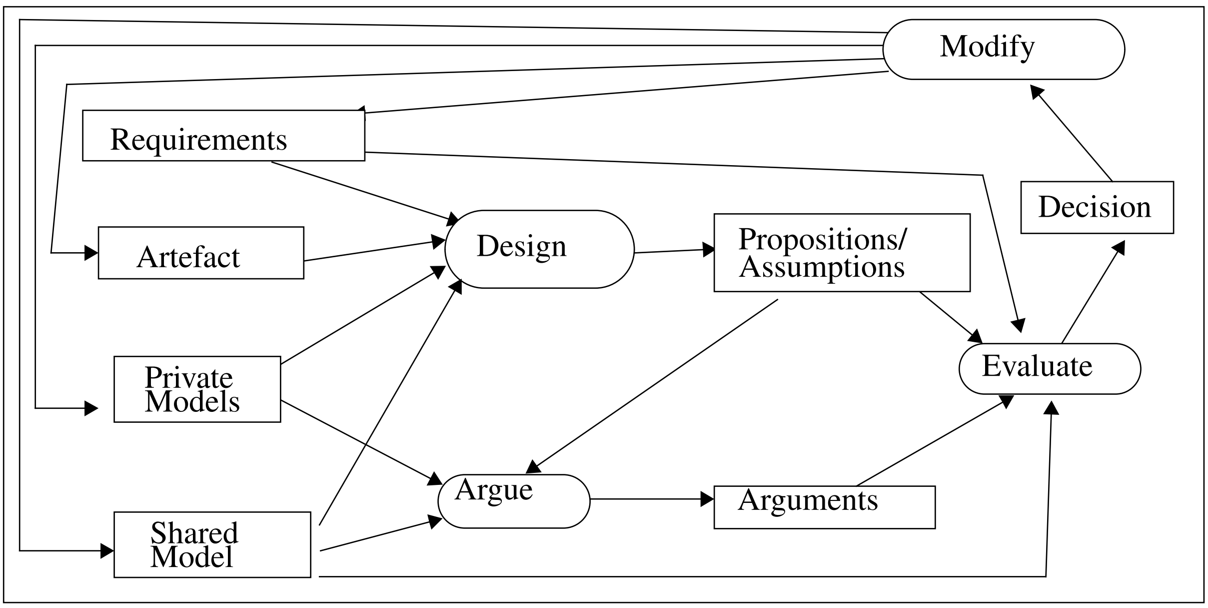


Figure 2. Concurrent Engineering data flow.

- *Design task*: Relying on his private knowledge (Private model), each designer generates some propositions to satisfy given requirements. His task is similar to a design task. Generic design model represented in the CommonKADS library [4] can guide the modelling of this task.

- *Argue task*: To promote the acceptance of his propositions by the group, a participant justifies them with a number of arguments. Assumptions made in the design task are used to determine arguments and to define them. Argumentation tries to change other participant's opinion by justifying the utility and the necessity of a proposition [5]. It forms an important part of negotiation which aims at solving a conflict.

- *Evaluate task*: The group evaluates the integration of propositions in the artefact. Propositions may not satisfy participants' needs and conflicts can appear. So, the principal subtask in this evaluation consists of detecting and solving conflicts.

Propositions can concern the requirements or the artifact model. Decisions about which object (requirements or artifact model) can be modified in the next process cycle and in which part modifications can be made, are taken in the design control (a subtask of «Evaluate»)

Shared model can be reorganized and modified, because other shared knowledge can be explicited at each cycle.

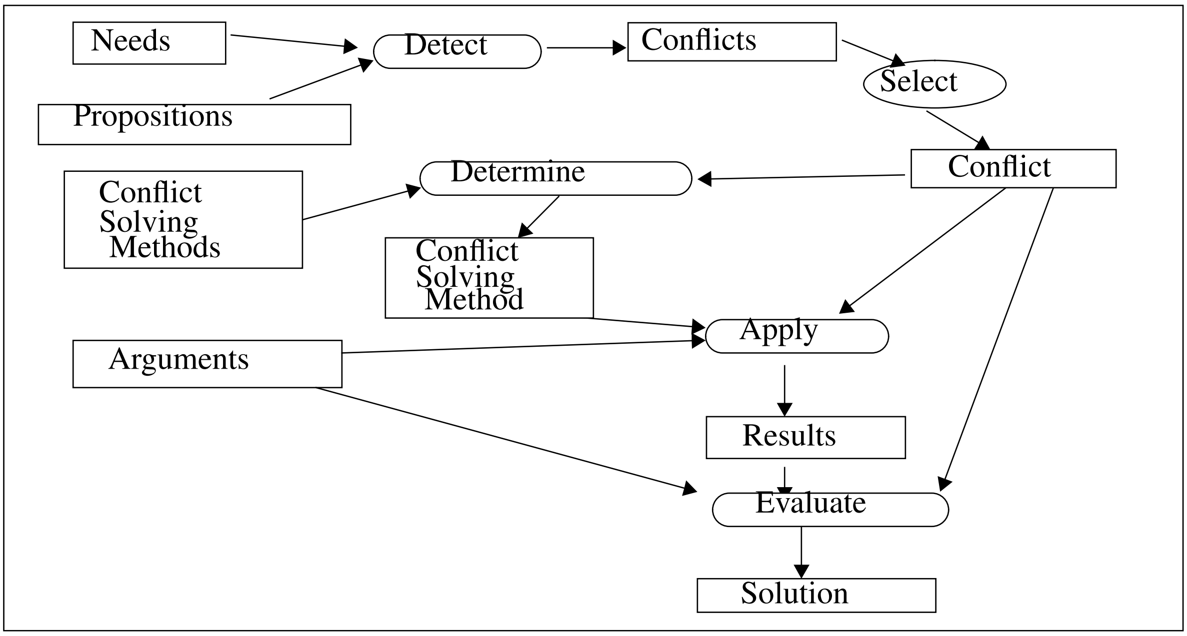


Figure 3. Conflict Management data flow.

Conflict management task consists of first detecting a conflict (its nature) and then solving it. Detection task can be assimilated to a diagnosis task in which a default (i.e: the conflict) is located (well defined). Once detected, to solve a conflict, a solving method is selected and applied. The result is then evaluated to determine if the conflict is solved or not. In the last case, another method can be selected and applied and so on. (Figure 3).

Emphasizing the characteristics of the tasks performed in the CE, specially the players in each task allowed us to express the nature of conflicts which can be detected in CE and the types of methods which can be used to solve them.

III. Methods to manage conflicts for each CE subtask

We associated several conflict management methods proposed in the literature, to CE subtasks as generic components in the library.

III. 1. «Design»: Prevention methods

In CE, several designers in different specialities collaborate together. Each designer has a specific viewpoint due to his speciality. B. Ramesh proposes to share part of private knowledge to avoid terminology and interpretation conflicts [11]. S. Easterbrook relates potential conflicts to group member characteristics [9]. He recommends some strategies to form a group respecting members' characteristics and relationships. Such methods can be associated to the Concurrent Engineering «Design» subtask to avoid potential conflicts if possible.

III. 2. «Argue»: Argumentation methods

Other methods are proposed to help a participant to argue his proposition. Argumentation is used in CE to defend a proposition and to persuade the group to choose it. Arguments must be explicit either to avoid conflicts if possible or to determine their nature precisely. K. Sycara [13] proposes some strategies using general principles and heuristic rules to define arguments. B. Ramesh [11] favours interdependency sharing to detect potential differences.

The important step in the «Evaluate» subtask of CE (Figure 2.) is to detect and solve conflicts. Detection and negotiation methods can be used to (respectively) detect and solve conflicts.

III. 3. «Evaluate Propositions»: Conflict detection methods

Methods to detect conflicts and to determine their nature are not well explicited in the literature. Klein considers a conflict as an exception in the CE cycle and provides a typology of the nature of exceptions [10]. He recommends also some strategies to handle exceptions depending on their nature. Conflict detection can be assimilated to a diagnosis task in which conflicts (faults) are to be determined (located). So, diagnosis methods like those presented in [1] can be used to detect conflicts. Conflict symptoms can be first detected and contributors (opposed propositions and arguments) are analysed as hypothesis in order to define explicitly conflict causes. Our conflict typology (Figure 1) helps to determine the nature of a conflict and to detect elements which cause it.

III. 4. «Evaluate Propositions»: Negotiation methods

A large number of negotiation methods can be used to solve detected conflicts. Some of them like «Locate a consensus», where a consensus is located and a solution is then chosen, can be applied to solve mixed-motive conflicts [11]. Another method like «Introduction of a Third party», where a third party imposes a solution [9] is defined to solve resource conflicts and also strategic conflicts.

Other methods are proposed in some conditions. For example, to apply Case-Based Reasoning (CBR) [13], first a number of past cases must be stored and second, at least one case similar to the current one must be selected. CBR recommends to retrieve conflict solution from an appropriate previous case and to adjust it as a solution to the current conflict. Another method proposes a Goal Graph Search to add goals or to substitute rejected goals in order to modify tasks decomposition and attribution [13]. To be applied, this method needs goal graphs organized as influence trees. Defining a «Counter-proposition» or modifying a rejected proposition [13] are also recommended.

IV. Representing conflict management methods in the library

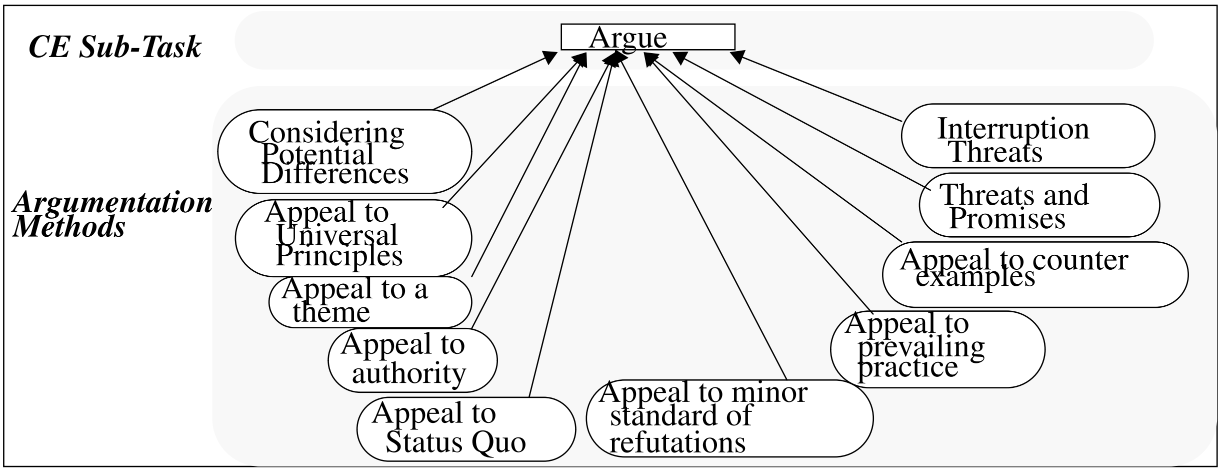


Figure 4. Argumentation methods.

A global view of the associations between CE subtasks and conflict management methods can guide a knowledge engineer to determine appropriate methods to manage conflicts in a particular application. So, trees of associations CE-subtasks/corresponding methods are presented in the library. For example, representing, a number of argumentation methods (Figure 4), shows a global view of possible strategies which can be used to justify a proposition. These trees are used also to mark out the description of a method.

Method «Interruption Threats»:

Competence: Guide to generate threats. If a conflicted proposition is not considered in the solution, the overall design process can be interrupted.
Input: Factors (Project Information), Conflicts
Output: Interruption Threats
Sub-tasks: Calculate interruption cost, Generate threats
Additional Roles:
Control: Calculate (Factors -> Interruption Cost)
Generate (Conflict, Proposition Payoff,Interruption Cost -> Interruption Threats)
Acceptance: Information about project and company's design and economic policies.

Method «Interruption Threats»: Data Flow

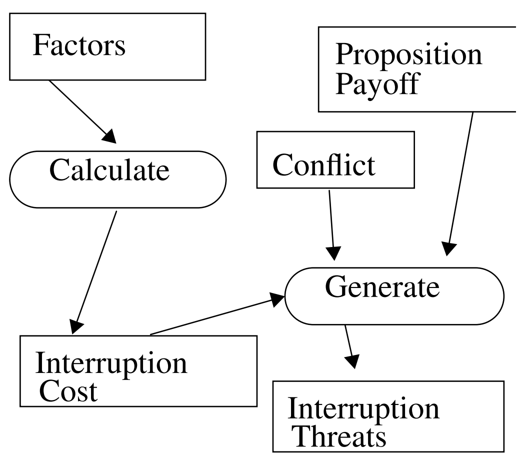


Figure 5. A Method Description.

The competence and the acceptance described in a method inform the knowledge engineer what the method is able to perform and in which conditions can be applied. For example, the «Interruption Threats» method guide to formulate arguments by using interruption threats in order to incite designers to accept a proposition.

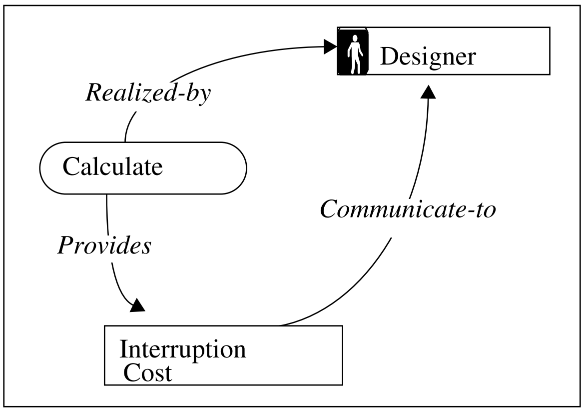
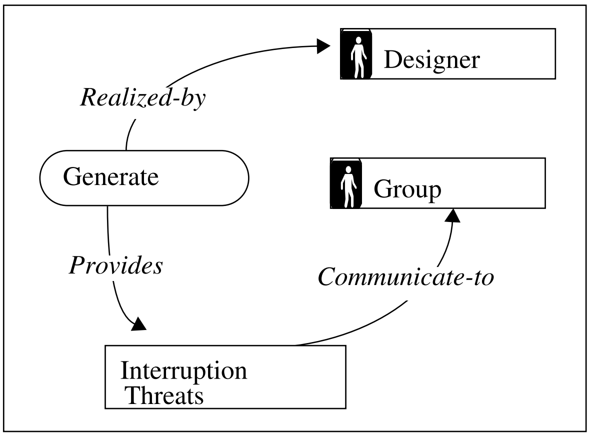
Sub-Task «Calculate»: ***Definition***: **Goal**: Determine the Interruption Cost. **Specification**: To calculate the interruption cost, a number of factors like information about economic situation of the company and of the industry overall, about the project itself and about the specific design sector are considered. **Input**: Factors **Output**: Interruption Cost ***End definition.***	Allocation and Communication 
Sub-Task «Generate»: ***Definition*****:** **Goal**: Generate Interruption Threats. **Specification**: To determine a number of Threats as arguments. These arguments emphasize that the corresponding proposition is a necessary condition to the progress of the design process and it is costly to interrupt the design process. **Input**: Interruption Cost **Output**: Interruption Threats ***End definition***.	Allocation and Communication 

Figure 6. Tasks Description.

To apply this method, a designer needs to know information about the economic state and the politics of the company. He will use these information to calculate the interruption cost of the design process. This cost helps to formulate threats as arguments in which acceptance of the corresponding proposition as a necessary condition of the design process progress, is emphasized. Data flow shows the interaction between tasks and roles which can be played by the application static knowledge (Figure 5).

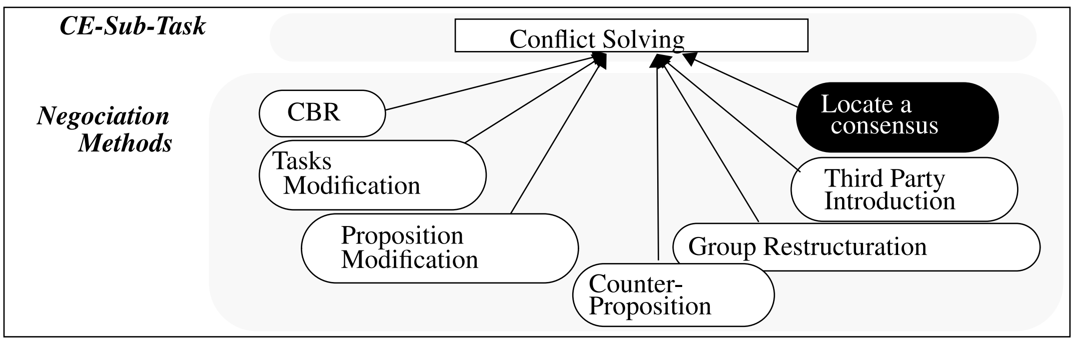


Figure 7. Negotiation methods.

Each task is then described, with its goal, I/O roles,... (Figure 6). For example, the task «Calculate» allows to determine the design process «Interruption Cost» using the company's economic information as factors. A definition of agents which will realize the task and to which task results must be communicated, is also made (Figure 6). This definition emphasizes players in each level in CE task (personal, inter-personal or group level). For example, the task «Calculate» will be realized by one designer. The result of this task is used by the same designer to generate arguments which will be communicated to the group. So, the «Interruption Threats» methods is performed at the inter-personal level, in which a designer executes all the method subtasks, and the results are then communicated to others (the designers group).

<u>Method «Locate Consensus»:</u>

Competence: This method allows to locate a consensus from all propositions provided and to define a corresponding solution of the conflict.
Input: Arguments, Conflict
Output: Conflict Solution
Sub-tasks: Locate, Calculate
Additional Roles:
Control: Locate (Arguments,Conflict -> Consensus)
Calculate (Consensus -> Solution)
Acceptance:

<u>Method «Locate Consensus»: Data Flow</u>

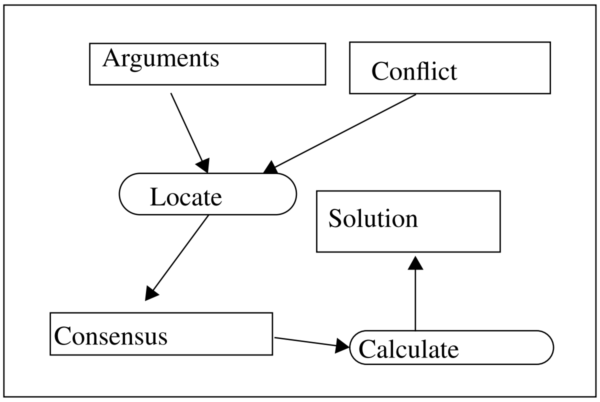


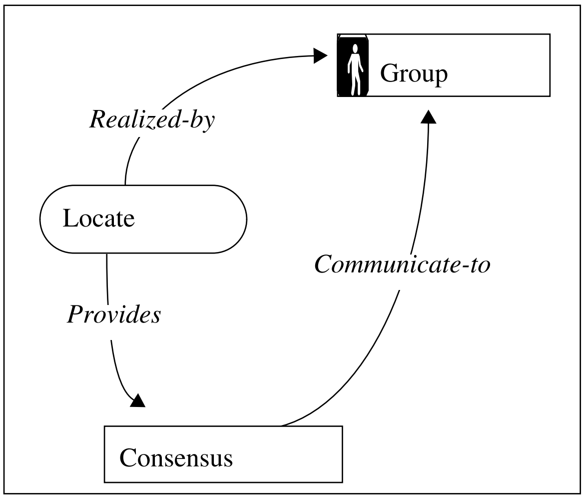
Figure 8. The «Locate Consensus» method description.

Let-us show another example of the description of methods in the library. A number of negotiation methods are presented as guides to help conflicts solving (Figure 7).

<u>Sub-Task «Locate»:</u>
Definition:
Goal: Locate a consensus.
Specification: Find an area of agreement, with evaluating valid propositions and determining norms which define the guidelines of a solution as a potential consensus.
Input: Conflict, Arguments
Output: Consensus
End definition.

Body:
Type: composite.
Sub-Tasks: Evaluate, Determine
Additional Roles:
Control:
Evaluate (Arguments,Conflict -> Valid Propositions)
Determine (Valid Propositions -> Consensus)
Assumptions:
End body.

<u>Allocation and Communication</u>



Roles (I/O):
Conflict: Conflict to solve.
Arguments: allowing to defend propositions.
Consensus: a potential consensus.

Figure 9. The «Locate» task description.

The «Locate Consensus» method, presented in [11] can be described as (Figure 8 and 9) in the library. The «Locate» task recommended by this method are described as in Figure 9.

These associations are not exhaustive. We plan to study other methods and enrich these associations.

We expressed a formal representation of this library in the CML language, using Cokace, a CML dedicated environment [7], in order to facilitate its exploitation. An HTML format is also generated automatically using this tool. The library is accessible via a knowledge server called "Web-Cokace", defined in our team [8], at the URL: `http://www.inria.fr/acacia/Cokace`.

V. Conflict types as another library index

To facilitate the selection of methods from the library, we use the conflict typology (illustrated in Figure 1) as another index. The associations of prevention, argumentation and negotiation methods to conflict types which can be managed by these methods, guided us to determine a number of selection criteria as index.

V. 1. Strategic Conflicts

To manage strategic conflicts, the appeal to the authority, to Threats or Promises and to Interruption Threats, can be used in the argumentation. Group Restructuring, or The Introduction of a Third Party will help conflict solving. Conflicts came out from tasks realization like conflicts about methods and tools used or from missing in the responsibilities to execute tasks, can be managed with an appeal of a minor standard of refutations rules or an appeal of rules relative to a particular theme, as argumentation methods. Extracting solution from past cases, or locating a consensus between designers will help also in the negotiation. However, The modification of tasks decomposition and attribution help to solve coordination conflicts (like conflicts about task organization and cooperation problems) (Figure 10).

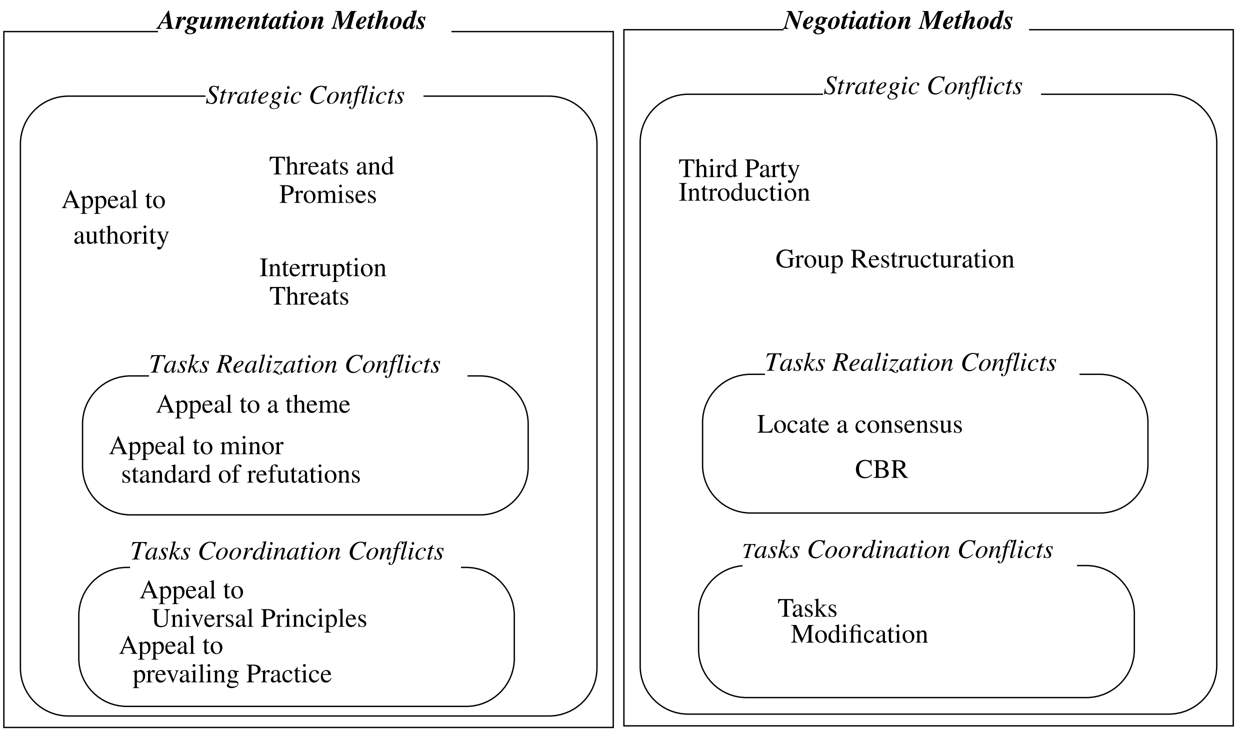


Figure 10. An example of Conflict types/Generic methods association: Methods to manage strategies conflict.

V. 2. Conflicts about propositions

Knowledge sharing is necessary to prevent problems caused from a misunderstanding of propositions provided [11]. Considering Potential differences and Appealing to a particular theme's rules can be used in this case as argumentation methods. Propositions Modification is an interesting method to be used in the negotiation.

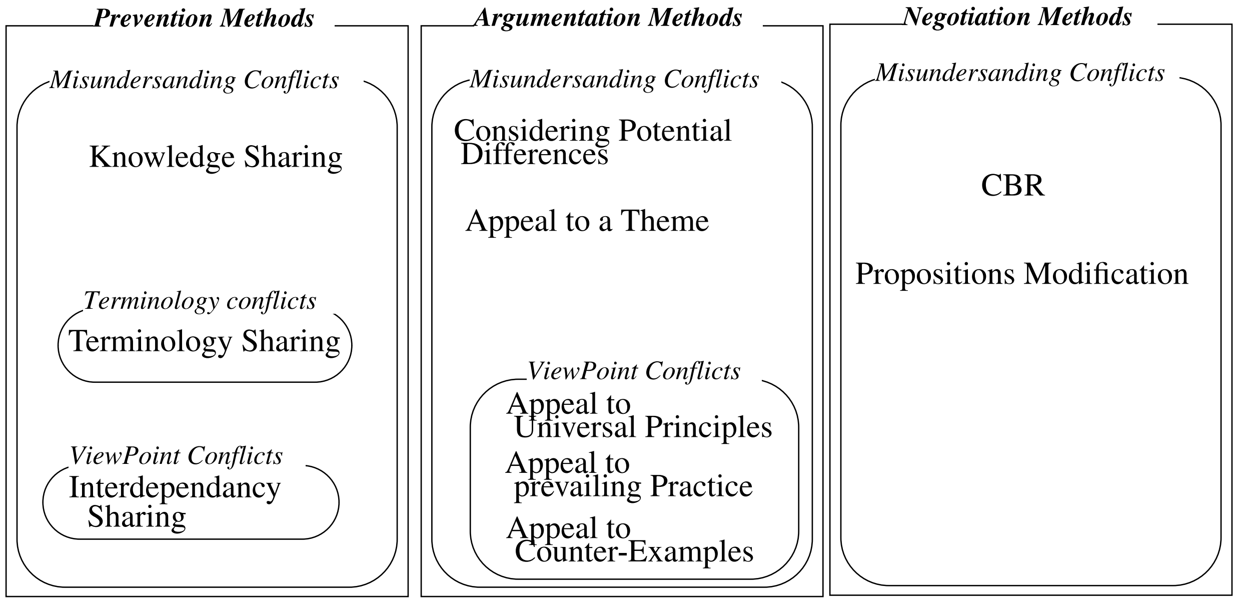


Figure 11. An example of Conflict types/Generic methods association: Methods to manage conflicts that appeared from misunderstanding of propositions.

To manage acceptance conflicts, the negotiation can be oriented in order to locate a consensus. To solve those caused from different evaluation of preferences or of requirements, negotiation methods like Propositions Modifications and Counter-Proposition will be used. To solve resources problems, the Introduction of a Third Party Decision is noticeable. The Appeal to authority, to Threats or Promises or to Interruption Threats can be also used as argumentation methods in this case. Problems about the acceptance of the quality of propositions can be solved using the Appeal to Prevailing Practice, to Minor Standard Refutations rules, to Threats or Promises or to Interruption Threats as argumentation methods. Negotiation methods like the modification of tasks decomposition and attribution, Proposition modification, Counter-Proposition and Introduction of a Third Party decision will help also to deal with this type of conflicts.

CONCLUSION

Conflicts ensue essentially from divergence of different fields implicated in a CE project. So, the management of this type of conflicts requires to detect first the discordance between proposals of different designers and to negotiate modifications to be done in order to find a coherence between design propositions and to integrate them in a single product.

In negotiation, a designer tries to change with arguments, the opinion of others in order to minimize modifications on his propositions. The main objective of this paper is to provide guides which help to detect this type of conflicts and to negotiate a solution.

Determining generic components as guides to model CE task and specially Conflict Management task is an innovative aspect of the research in CE. Representing conflict management as generic components allowed us to characterize the nature of different types of conflicts. Prevention, argumentation and negotiation methods are organized to manage these types of conflicts.

We analyse conflict types as conditions of conflict management methods selection and application. This helps to index these methods considered as generic components in the library. Other work in indexation is also studied for this purpose.

We plan also to a use real application in aeronautics to validate components selection from the library and their adaptation to specific applications.

We plan to study the representation of shared knowledge in CE applications using multi-perspective structures [12]. A multi-agent architecture is in development in our team [6]. This architecture aims at helping a group of participants to design an artefact. It guides participants to share their propositions by structuring them in a single formalism and to detect and solve conflicts by facilitating propositions comparison and by recommending conflict management methods extracted from our library. This architecture is based on the generic CE task model.

Acknowledgements

We deeply thank the French «Ministère de l'Enseignement Supérieur et de la Recherche» and DASSAULT-AVIATION (Genie Project) that funded this research for their partial funding. We thank also Dr. R. Dieng and Dr. O. Corby for their remark

REFERENCES

[1] R.Benjamins, Problem Solving Methods of Diagnosis, *Thesis Univeriteit van Amsterdam* 1993.

[2] A.H. Bond, A Computational Model for organizations of cooperating intelligent agents, *Proceedings of the Conference on Office Information Systems*, Cambridge April, 1990.

[3] F.M.T. Brazier, P. H.G. Van Langen, J. Treur, Modelling conflict management in design: An explicit approach, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol.9, N.4, Cambridge University Press, USA 1995, pp.353-366.

[4] *Common-KADS Library for expertise modelling Reusable problem solving components, Frontiers in Artificial Intelligence and Applications,* J. Breuker and W. Van de Velde (EDS), Amsterdam: IOS.Press 1994.

[5] C. Castelfranchi, Conflict Ontology, Proceedings of ECAI'96 Workshop on Modelling conflicts in AI, H.J. Muller, R. Dieng (Eds), Budapest August 1996.

[6] C. Cointe, Guide to Manage Conflicts in Concurrent Engineering: A Multi-Agent Architecture, Proceedings of CEE'97, Building Tomorrow's Virtual Enterprise, K. Reger (Editor), SCS (Publisher), Germany April 1997.

[7] O. Corby, R. Dieng, Cokace: A Centaur-based environment for CommonKADS Conceptual Modelling Language, *Proceedings of ECAI'96*, Budapest August 1996.

[8] O. Corby, R. Dieng, A CommonKADS Expertise Model Web Server, Proceedings of ISMICK'97, Management of Industrial and Corporate Knowledge, Compiegne, October,

1997.

[9] S. M. Easterbrook, E. E. Beck, J. S. Goodlet, L. Plowman, M. Sharples, C. C. Wood, A Survey of Empirical Studies of Conflict, *CSCW: Cooperation or Conflict?* S. Easterbrook (ed), Springer-Verlag 93

[10] M. Klein, Conflict management as part of an integrated exception handling approach, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* Vol.9, p.259-267, Cambridge University Press USA 1995.

[11] B. Ramesh, K.Sengupta, Managing Cognitive and Mixed-motive Conflicts in Concurrent Engineering, *Concurrent Engineering: Research and Applications* Vol.2, N.3, 1994, pp.223-236.

[12] M. Ribière, N. Matta, C.Cointe, A proposition for managing project memory in concurrent engineering, To appear in International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'98), Churchill, Australia, February 1998.

[13] K. P. Sycara, Cooperative Negotiation in Concurrent Engineering Design, *Computer aided cooperative product development. Proceedings of MIT-JSME workshop*. D. Sriram, R. Logcher, S. Fukuda (Eds), Cambridge, MA 1991.

[14] G. Schreiber, B. Wielinga, W. Van de Velde, A. Anjewierden, CML: The CommonKADS Conceptual Language, *Proceedings of EKAW'94, Lecture Notes in AI N.867,* L.Steels, G. Schreiber, W.Van de Velde (Eds), Bonn: SpringerVerlag, September 1994, pp.1-25.

[15] T. Tomiyama, Concurrent Engineering: A successful Example for Engineering Design Research, 1st International Engineering Design Debate, Glasgow, September, 1996.