CREWS Report Series 97

A meta-model for goal-driven cooperative work processes

Selmin Nurcan, Colette Rolland

Université Paris 1 Sorbonne

CRI, 17 rue de Tolbiac

75013 Paris - France

{nurcan, rolland}@univ-paris1.fr

Appeared in Proceedings of the Workshop on the many facets of Process Engineering (MFPE'97), Gammarth, Tunis, September 22-23, 1997.

A meta-model for goal-driven cooperative work processes[1]

Selmin Nurcan, Colette Rolland
Université Paris 1 - Panthéon - Sorbonne
Centre de Recherche en Informatique
17, rue de Tolbiac
75013 Paris FRANCE
email : {nurcan, rolland}@univ-paris1.fr

Abstract:
The Computer Supported Cooperative Work (CSCW) discipline examines the possibilities and effects of technological support for humans involved in collaborative group communication and work processes. Organizations are built on the principle that groups of people can carry out tasks which are not feasible individually. For many of them, well-structured and ill-structured procedures coexist in work processes and must be managed in the final solution. Cooperative work techniques become very important in organizations. One can note the emergence of many products dedicated to developing cooperative systems. Therefore, it is necessary to emphasize the specificities of these applications in order to take them into account as soon as possible during design. This paper presents a process meta-model for goal-driven cooperative work processes.

Keywords: *Cooperative work - Meta-modellling - Process modelling*

Résumé:
Le Travail Coopératif Assisté par Ordinateur est la discipline qui étudie les possibilités et les effets du support technologique pour des participants impliqués dans des processus de travail coopératif. Les organisations sont construites sur le principe suivant : des groupes de personnes peuvent accomplir des tâches qui ne sont pas réalisables par les efforts individuels de chacun. Pour la plupart d'entre elles, les processus de travail structurés et les processus de travail ad-hoc co-existent et doivent être gérés dans la solution finale. Les techniques de travail de groupe sont devenues de plus en plus importantes dans les organisations et l'on constate l'émergence d'outils dédiés au développement de systèmes coopératifs. Par conséquent, il est nécessaire de souligner les spécificités de ces applications afin de les prendre en compte aussitôt possible dans la conception. Ce papier présente un méta-modèle pour les processus de travail coopératif dirigés par les objectifs.

Mots-clé: *Travail coopératif - Meta-modélisation - Modélisation de processus*

1. Introduction

The cooperative work or group work is the object of a multidisciplinary research field called Computer Supported Cooperative Work. The growth of connectivity greatly expands opportunities for office workers to cooperate and work together. Most organizations acknowledge that process automation and simplification are essentials to succeed in the present competitive environment where the watchwords are productivity and quality.

Groupware is defined in [Ellis 91] as follows: "*Computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment*". A well-known categorization is the division into synchronous or asynchronous activity and co-located or distributed activity [Ellis 91].

Workflow concerns, at first, an activity of scheduling and coordinating work between actors [Khoshafian 92] [Naffah 94], [OVUM 91]. In workflow applications, cooperative work means that several persons are involved in reaching a common goal, but each of them acts individually in a different step of the work.

The definition given by Peter and Trudy Johnson-Lentz in 1978 shows obviously the aim of groupware: a system which facilitates the group work. Ellis definition precises that the group work involves a common task (or goal) and a shared environment. Many people do not agree that workflow respects these criteria; because only one person executes his own task with his own data at a given time. However, if we take a general view of the procedure, there is a common goal to reach by a group of people which share the same information.

[1] This work is in part funded by the European Community under ESPRIT Reactive Long Term Research 21.903 CREWS

Workflow applications have been divided into two different categories depending on the nature of the processes supported [Palermo 92]. The first concerns well-structured and repetitive procedures having important coordination and automation needs [Nurcan 95a]. The second category of workflow applications deals with occasional and ill-structured (ad-hoc) work processes in organizations. The essential preoccupation with this kind of application is the information and knowledge-sharing in the work group more than the coordination of their tasks.

For many organizations, well-structured and ill-structured procedures coexist in work processes and must be managed in the final solution [Nurcan 95b] [Nurcan 96a]. The integration aims to make the transition between the different types of group activities transparent. This requires homogeneity and coherence of handled concepts.

A workflow application often requires many actors who execute tasks. On the other hand, in cases such as the organization of a meeting, decisions have to be taken before the workflow procedure can continue. This is a typical example for the use of different technologies. The procedural workflow applications contain predictable activities, but these activities can be individual or group oriented. Workflow products can be the thread of the cooperative applications which use individual softwares, ad-hoc workflows and other types of groupware0 [Nurcan 96b].

However usual workflow products and the underlying control flow models require a strict respect of the predefined procedures. Therefore they cannot be used for ad-hoc workflow applications or deal with the dynamic modifiability of predefined scripts. More and more, users ask for adaptive workflow products and models which can provide the robustness and the security of the predefined scripts and the flexibility of ad-hoc applications. In order to deal with a wide range of cooperative work processes, adaptive workflow models allow dynamic modifiability [Swenson 93], the representation of unstructured activities [Ellis 94], or both [Nurcan 96c].

The development of information systems is itself becoming performed in a cooperative manner. In the CREWS[1] project we are developing an approach for supporting cooperative requirements engineering based on scenarios. Therefore, it is necessary to understand the specificities of cooperative work processes in order to take them into account in models and software tools built for supporting their enactment.

Our purpose is to propose a process meta-model, which can deal with both well-defined and wickled work procedures and their interactions, so as to represent a wide range of cooperative work processes.

This paper is organised as follow: In section 2 we brievely present the main concepts of some of the models dealing with workflow representations we studied. In section 3, we present a cooperative process meta-model which provides means to deal with secure and well-structured cooperative work processes and the flexibility to handle ill-structured cooperative work processes.

2. Models dealing with task-role-workflow representations

Workflow application development starts with the modeling of the process to automate. The implementation of this kind of application requires a preliminary analysis phase before the process modeling. For each stage of the work, we have to determine who does what, concerning which task, when, after which task and before which task. We have also to define information holders, types of handled documents, possible locking points…

We have considered 5 models dealing with task-workflow-agent-role representations and have constructed the corresponding meta-models.

Meta-models are represented using an extended binary entity-relationship notation, based on entity-types (large boxes), relationship types (small boxes), cardinalities, "is-a" links and objectified relationship types (an abstraction mechanism allowing us to view a relationship type, at a higher level of abstraction, as an entity-type).

® The **OSSAD** method (Office Support System Analysis and Design) has been developed within the context of an ESPRIT project whose aim was to find appropriate methods for the development of office automation systems [Dumas 90a] [Dumas 90b]. OSSAD is a method which indicates how people coordinate their tasks in order to provide a common result. This method proposes two levels of modelling: the abstract and the descriptive ones.

The abstract model definitively defines stable and durable characteristics of the analysed system that any organization choice must respect. It is based on the division of the organization into *functions*, i.e. into sub-systems having coherent objectives. Each function may be divided into sub-functions, each in turn being subdivisable. At the most detailed level of the analysis, atomic functions are called *activities*. An activity has only one objective. These sub-systems communicate with each other and with the environment exchanging information *packages* (disregarding their physical support). Each activity of the abstract level corresponds to a *procedure* in the descriptive level.

The link between abstract and descriptive levels is made by the activity/role matrix. Rows correspond to activities (abstract concept) and columns to roles (descriptive concept). For each activity, roles which participate by executing a task must be indicated. Each activity of the abstract level corresponds to a *procedure* in the descriptive level.

Descriptive models deal with the organizational, human and technical means implemented to reach of the objectives of the organization. They represent the way the work is done currently or will be done in the future.

The roles descriptive model shows the current organizational structure chosen by the company (or the one which is proposed) to carry out its activities. It uses concepts of *role*, *unit* and *resource*. A unit represents a set of roles assembled for the convenience of modeling. This can correspond to an administrative unit of the analysed organization. Information exchanged between roles, external roles and/or unit appear as *resources*.

The procedures descriptive model shows the functioning of the organization, in other words, current or future work organization. It uses *procedure* and *resource* concepts. This model provides a global view of relationships between procedures.

The operations descriptive model provides the detail corresponding to a procedure. It models the work distribution between roles showing who does what and in which order. This model uses a formalism similar to Petri nets. In addition to the order relationship between operations, this formalism shows three possibilities of flow of operation: parallelism (and), alternative (or) and loop. Certain operations of a procedure may be gathered together to make *macro-operations*. The macro-operation concept allows to represent a cooperation.

Figure 1 shows the meta-model of the OSSAD method.

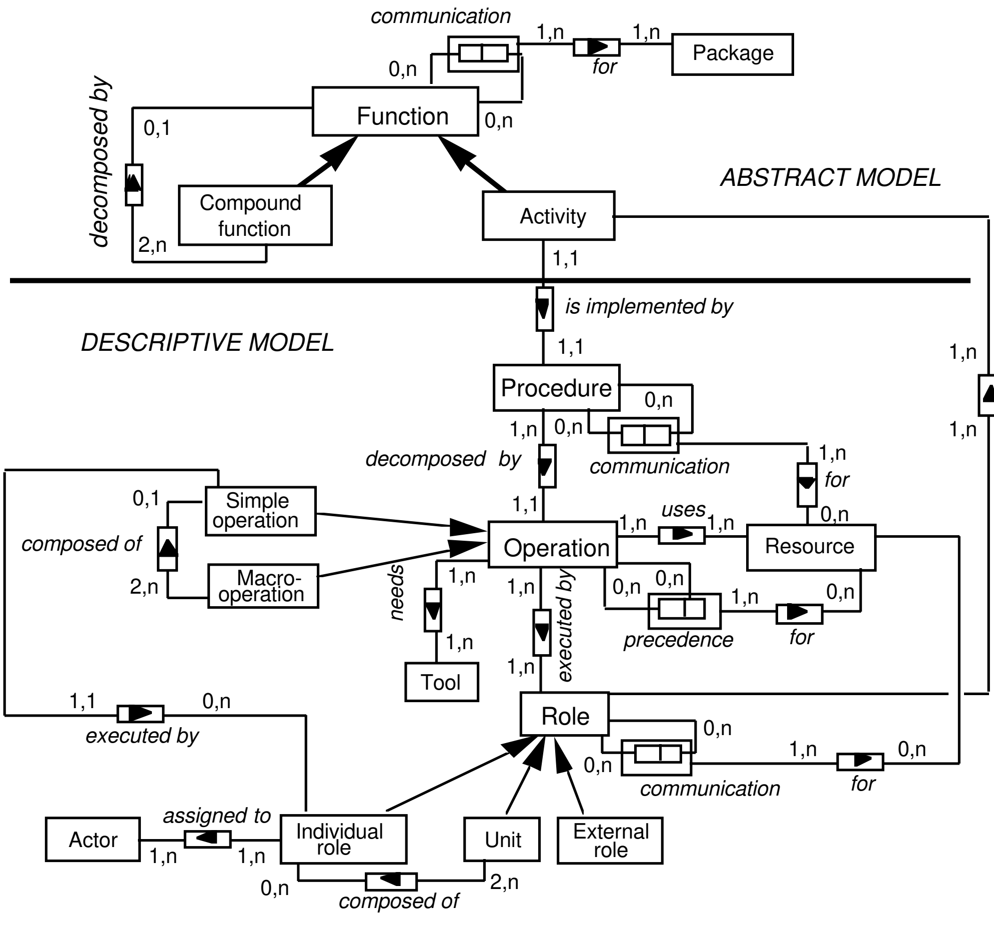

Figure 1 - OSSAD meta-model

® In the **ICN** model (Information Control Net), a *procedure* is a predefined set of work steps and a partial ordering of these steps [Ellis 79] [Ellis 94]. Steps can be related to each other by conjunctive logic or by disjunctive logic. Each procedure has a *responsible* person associated with it. A *job* denotes a particular execution of a procedure.

An *activity* is the body of a work step of a procedure. It is either a compound activity, containing another procedure, or an elementary activity. An *elementary activity* is a basic unit of work which must be a sequential set of primitive actions executed by a single actor. An activity is a reusable unit of work, so one activity may be the body of several work steps.

A *role* may be associated with a group of actors. Also, one *actor* may play many roles within an organization. An actor is a person, program, or an entity that can fulfill roles to execute, to be responsible for, or to be associated in some way with activities and procedures.

An information control net is a set of procedures, steps, activities, roles, and actors with a valid set of relations between these entities. Relations include the *precedence* relation between steps; the *part-of* relation between activities and procedures; the *executor of* relation between activities and roles; and the *player of* relation between roles and actors. The entity diagram presented in [Ellis 94] shows entities and relations of these definitions.

A useful workflow model must capture much more than the steps of procedures. Many social and organizational factors play an important role in the working of any organization. These observations lead the authors to propose the following definition of extended ICN in [Ellis 94]. Instead of choosing procedures and activities as the starting point, they choose people and *goals*.

An organization framework is a tuple $F=[G, H, R]$ where G is a set of *goals*, H is a set of *actors*, and R is a set of *resources*. An extended information control net is a tuple, $S= [F, O, f_S]$ where F is an organizational framework, O is a class of procedural objects (*activities*) and non-procedural objects (*roles*), and f_S is a set of mappings over F and O. O and f_S capture the procedural definition of ICN.

Functional abstraction allows any activity to itself be defined as a procedure or a goal. If an activity is a goal, then there may be multiple procedures which can be invoked to attain the goal. The extended ICN model presented in [Ellis 94] recognizes that an organization comprises resources and goals. The model incorporates the notion of unstructured activity.

® In the **INCONCERT** workflow model [McCarthy 93] a *job* represents a collaborative activity. A job consists of *tasks*, each of which is a unit of work that can be performed by one person. Tasks can be decomposed into subtasks, to obtain a hierarchical breakdown structure. Tasks at the same level may have ordering *dependencies* defined among them: a dependent task cannot be worked on until the precedent task has been completed.

Each task in a job has an assigned *role*, which is a logical placeholder for the *user* (person or program) that will perform the task. Each task in a job may also have any number of references, which are placeholders for *documents* needed in performing the task (for update or as reference material).

A job is activated by instantiating a *template* which provides a predefined task structure for performing a given business process. During the execution of a particular job, the task structure can be modified by adding and deleting tasks and dependencies. This provides users with the flexibility needed to deal with exceptions and deviations from the idealized process descriptions.

® **VPL** [Swenson 93] is a graphical language to support a model for collaborative work process. According this model, work is decomposed into a network of requests for task assignments, which may be recursively decomposed to finer grained tasks.

The model provides a shared collaboration space called *colloquy*, in which to coordinate a set of tasks that are performed to accomplish a specified goal. A colloquy is composed of *stages* and *roles*. Stages each contain a collection of actions. The shared data space can hold *documents* or other *artifacts*. Figure 2 shows the VPL meta-model.

The process is modelled as requests for tasks. *Plans* are composed of network of stages. Each stage represents a task request, commitment or question as a specific step in the process. The person who is responsible for the result of the plan is the *owner of* the plan. The owner is usually the creator of the plan, and is the only person who may make changes in the plan.

Each stage has an assigned *role*. The role that is *assigned to* the stage is said to be responsible for the stage. A stage is a request from one person (the plan *owner*) to another person (the *assignee*). A request is accepted to communicate that the task will be done. Declining a request is a message back to the plan owner that something is wrong. The owner may then choose to assign the request to someone else, or may choose to change the request to make it acceptable. The request may be expressed in any amount of detail; it is not constrained to a set of predefined tasks. This represents the Regatta philosophy of supporting communications, without restriction [Swenson 93].

If the request is not to be completed manually, the *assignee* may create a subplan to accomplish the task. The assignee becomes the *owner* and creator of the new subplan and may make requests to others by creating stages within the subplan. The subplan is ended by activating an exit node, when sends an event to the parent stage (see below).

A *stage* includes one or more user defined actions, called *options*. Each option (with its associated message) represents a declaration that the assignee may take to represent the results of the task or decision. The act of choosing an option changes the state of the process. It does this by sending *events* to activate or terminate other stages. The *event* is an abstract mechanism that is used to coordinate stages.

When two or more event arrows are pointing to a stage it means that the first activated event from any one of them will activate the stage. While stages represent the major steps in a process, there are other kinds of nodes which provide some automated capabilities within the plan: *programmed nodes*, *condition nodes*, *timer nodes, start nodes*, *exit nodes*, *AND-node*s (Swenson, 1993).

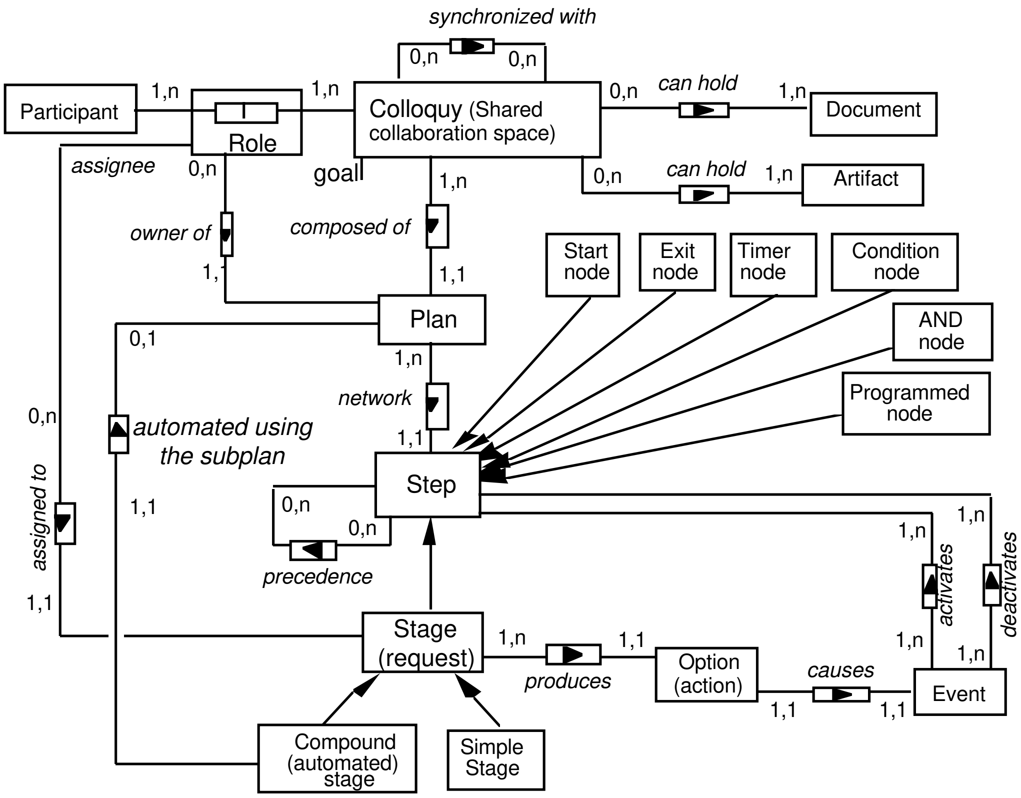

Figure 2 - VPL meta-model

® The **I*** framework [Yu 94] has been developed to help supporting process modelling and reengineering. Processes are taken to involve social actors who depend on each other for goals to be achieved, tasks to be performed, and resources to be furnished. The framework includes a Strategic Dependency model and a Strategic Rationale model. The first describes the network of relationships among actors, the second describes and supports the reasoning that each actor has about its relationships with other actors. The name I* refers to the distributed intentionality perspective offered by the framework.

The Strategic Dependency model is a graph where each node represents an *actor*, and each link between two actors indicates that one actor depends on the other for something in order that the former may attain some goal. In [Yu 94] the depending actor is called the *depender*, and the actor who is depended upon is called the *dependee*. The object around which the dependency relationship centers is called *dependum*. Authors distinguish among four types of dependencies, based on the type of the dependum: *goal dependency*, *task dependency*, *resource dependency*, *softgoal dependency*.

A business process would typically appear as a chain of dependency relationships, rather than as a sequence of input-output flows. A Strategic Dependency model is an intentional model and allows a richer representation of an organization than conventional workflow models that are based on non-intentional entity and activity relationships.

The Strategic Dependency model specializes the concept of actor into roles, positions and agents. A *role* is an abstract actor. Physical *agents* such as human beings or software agents *play* roles. A *position* is a collection of roles that are typically played by a single agent. Agents *occupy* positions; a position *covers* a number of roles; roles are *played* by agents.

The Strategic Rationale model describes actors' reasoning about their external relationships. It shows "how" an actor meets its incoming dependencies (for which the actor is dependee - or internal goals) by modelling actor's "ways of doing things" which are called *tasks*. A task is broken down into its components. Components are broken into sub-components, and so forth. The Strategic Rationale model recognizes the presence of freedom and choice at each level of decomposition. Each component of a task is an *intentional element,* the internal counterpart to the concept of *dependum* in the Strategic Dependency model. An intentional element can be a *goal*, a *task*, a *resource*, or a *softgoal*. Since there can be more than one way to achieve a goal, to perform a task, to produce a resource, or to satisfice a softgoal, [Yu 94] introduces an intervening *means-ends* link between an element (the end) and each way (the means) of decomposing it into sub-elements.

In summary, the Strategic Dependency model allows the modelling of how strategic actors relate to each other intentionally, while the Strategic Rationale model allows modelling of the means-ends reasoning the actors have about different potential ways of relating to each other accomplishing work. Figure 3 shows the meta-model of the I^* framework.

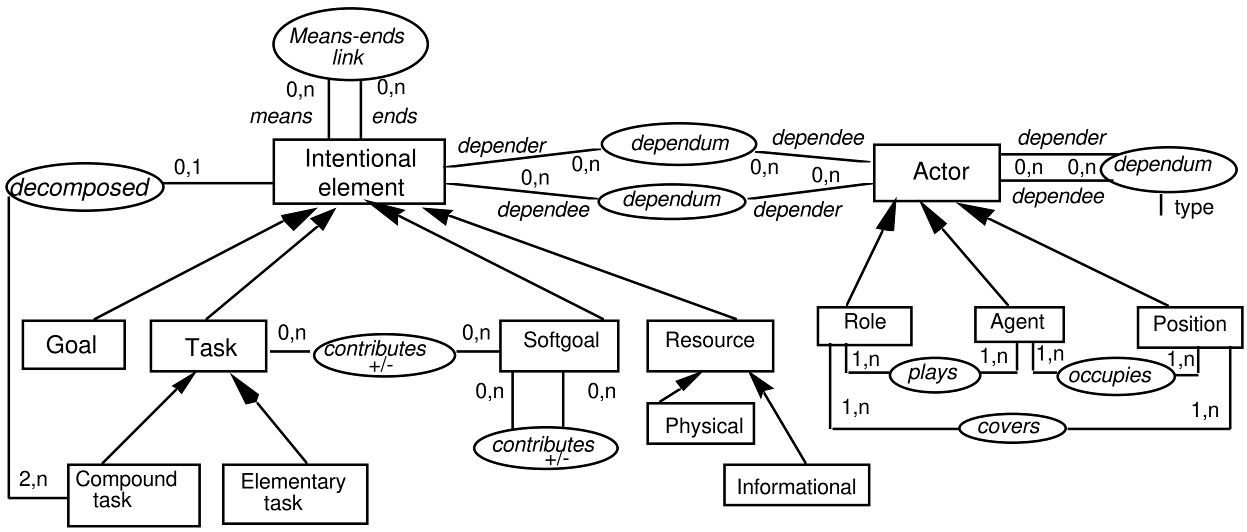

Figure 3 - I^* meta-model

Four of the presented models introduce the notion of goal even if this is made by the use of differents labels. The abstract model of OSSAD [Dumas 90b] defines stable and durable characteristics of the analysed system that any organization choice must respect. It is based on the division of the organization into functions, i.e. into sub-systems having coherent *objectives*. The ICN model [Ellis 94] advocates to choose people and *goals* as the starting point for organization analysis, instead of choosing procedures and activities. The VPL model [Swenson 93] provides a shared collaboration space called colloquy, in which to coordinate a set of tasks that are performed to accomplish a specified *goal*. According to [Yu 94], in a goal dependency, an actor depends on another to bring about a condition in the world. The *goal* is an assertion that the dependee will make true.

The concept of goal expresses an intention, this is what must be achieved. Goals are high level objectives of the organization. They defines stable characteristics of the business that any organization choice must respect. Goals can be decomposed into sub-goals. At the most detailed level, operationalizable goals [Rolland 97] are modelled using the atomic goal concept.

Operationalizable goals are implemented using processes, called respectively, procedure in OSSAD and ICN, job in InConcert and plan in VPL. A process can be structured or unstructured. It contributes to reach one or more goals.

The essential preoccupation of structured processes is the coordination of their component work steps as in OSSAD, ICN, Inconcert and VPL. A structured process is a predefined set of tasks and a partial ordering of these tasks.

A task represents a work step in the process. It can be an elementary task or a compound task defined by another process. The notion of task decomposition is used in ICN (compound activity), VPL (compound stage), Inconcert (compound task) and OSSAD (vertical macro-operation).

An elementary task is defined as a sequential set of primitive actions executed by an individual role which can be human or automated.

The role concept is common to all the presented models. Our understanding about it is the following: A role may describe an individual or an organizational unit, it can be external to the organization. An individual role (specialized in human and automated) can be part of organizational units. It is held by an actor which can be a person, a machine or a program.

Organizations cannot only be described in terms of structured work processes. As advocated in ICN and I^*, and briefly introduced in OSSAD by the use of the macro-operation concept, the process concept can be classified into two sub-types: structured process and unstructured process.

An unstructured process cannot be represented in terms of flow of tasks but using a set of resources that it uses and produces and a set of participating roles. The key concept of unstructured processes is the information and knowledge sharing in the work group. Therefore, models coming from design rationale are more convenient for a more detailed representation of this type of cooperative processes [McLean 93] [Henninger 91] [Kuwana 93].

This study shows a convergence on a set of concepts such as goal, procedure, task, role, actor, resource, decomposition of tasks, etc. However, an appropriate model for goal-driven cooperative work processes must also provide means to represent unstructured activities. We integrate these concepts in one single meta-model that we present in the following section.

3. A process meta-model for cooperative work processes

An approach to generate guidance centered process models is proposed in [Rolland 95]. Authors refer to these models as "ways-of-working" since they are intended to guide application engineers in their way of working to solve a design problem. We argue that the proposed approach is applicable to any process. However the problem of distributed process guidance has not be tackled in [Rolland 95].

We have extended the process meta-model introduced in [Rolland 95] in order to obtain a *cooperative process meta-model* to be used for any cooperative process.

3.1. The cooperative process meta-model

We propose a *meta-model* as a basis for process model definition [Rolland 95] [Nurcan 96] [Nurcan 97]. The process meta-model allows us to deal with many different situations in a flexible, decision-oriented manner. Moreover the meta-model can support different levels of granularity in decision making as well as non determinism in process performance. It identifies a decision in context as the basic building block of ways-of-working and permits their grouping into meaningful modules. Parallelism of decisions and ordering constraints are also supported.

Since a process meta-model carries information about the process model, an instantiation of it shall result in a process model. Our approach introduces three levels of process modelling:

-At the lowest level, process traces are recorded.

-At the second level, ways-of-working are defined. A way-of-working is a process model i.e. a description of process. It has a prescriptive purpose and is similar to the concept of plan. A process is then, an instantiation of a process model which is executed.

-The knowledge required to design such plans is related to the third level of abstraction and takes the form of a process meta-model. A process meta-model provides a set of generic concepts for describing ways-of-working which are therefore, instances of the process meta-model.

The output of a process is a *product*, it can be a requirements specification, a conceptual schema, a service provided to a client in an organization, messages exchanged between members of a group or a set of business goals.

In the presentation of the *cooperative process meta-model* we follow a bottom up approach starting with the concept of *context*, intoducing then the concepts of *role*, *action* and *product*, and ending with an overall view of the concepts progressively introduced.

3.2. The concept of context

The central concept of the process meta-model is the one of *context* which associates a situation with an intention.

A *situation* is a part of the product it makes sense to take a decision on. Situations can be of various granularity levels; they can be either atomic like an attribute of an object class or they can be coarse-grained like the whole product.

A decision reflects a choice that a user can make at a given moment in the process. A decision refers to an *intention.* An *intention* expresses what the user wants to achieve, the goal.

A *context* is the association of a *situation* and an *intention* expressing what the user wants to achieve in this situation. A decision is not sufficient in itself, it needs to be associated with the situation in which it applies. A situation can be associated with several decisions. Acting in a context corresponds to a step in the process: in a given situation, and in order to progress in the process, the user has to take a decision refering to an intention (figure 4).

3.3. The concept of role

In workflow applications tasks are individual and are performed by individual roles. Each task is assigned to a role corresponding to a group of actors (i.e. the collection of the role object).

According to the process meta-model, acting in a context should correspond to a step in the cooperative process. Since there are a number of participating users in cooperative processes who discuss and work with one another, there is need to have specific provision for the conversational action in the process meta-model. Additionally, since users play different roles in organizations and participate in the cooperative process from the point of view of this role, it is necessary to explicitly bring the notion of role in the meta-model.

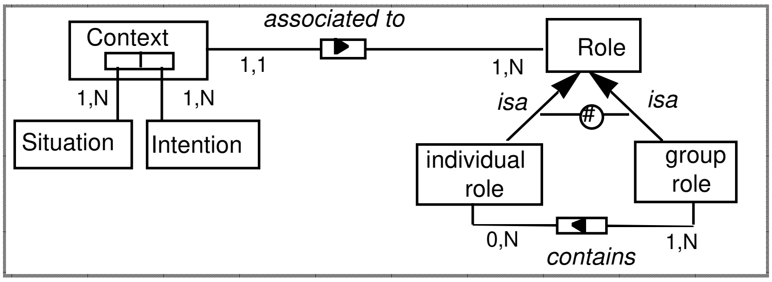

Figure 4 : The context is attached to a role

A role is the definition of an organizational intention shared by a collection of users, all of whom have the same privileges and obligations to a set of work processes in an organization. For example, the role of a reservation service clerk, that of an accounts officer, etc. In a given situation, a user has an intention (because of his/her role in this process), and that makes him/her progress in the cooperative process.

To this end, we introduce the concept of *role*, and then specialise it into *individual role* and *group role* (figure 4). For example, the reservation service clerk is an individual role whereas public relations team is a group role. A group role *contains* several individual roles.

We *associate* the context of the process meta-model to a role. This captures knowledge about which decision can be taken by which role. Therefore, the basic division of responsibility in cooperative processes is imposed on the set of decisions of the meta-model. This helps us in representing co-ordination of roles, providing access control, and in giving more appropriate guidance in the process modelling which is completely tailored to the role.

3.4. The different types of contexts

A situation exists at different levels of granularity. Further, decisions have consequences which differ from one granularity level to another. A complete understanding of the notion of a context can thus be gained by articulating the consequences of making the decision of a context on its situation.

The different contexts are classified (figure 5) according to their consequences in the meta-model into *executable contexts*, *plan contexts*, and *choice contexts*.

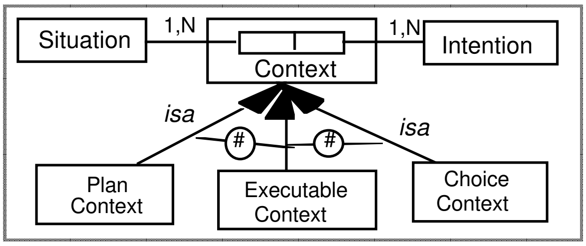

Figure 5 : Different types of contexts

3.4.1. Executable context

At the most detailed level, the execution of any process can be seen as a set of transformations performed on the product, each transformation resulting from the execution of a deterministic action. Such an action is a consequence of a decision made in a certain context. This leads to the introduction of the concept of an *executable context.*

< (Reservation request+resources) ;*Create reservation*; Reservation_service_clerk >

Figure 6 : Example of an executable context

An *executable context* implements a decision, its intention is realised by an action (figure 6). Therefore, in the meta-model (figure 7), an executable context is associated with an action. An *action* performs a transformation of the product, it is the implementation of a decision. Performing an action changes the product and may generate a new situation which is itself, subject to new decisions.

The concept of action

We classify actions into two types (figure 7) according to their characteristics: *individual action* and *conversation action*.

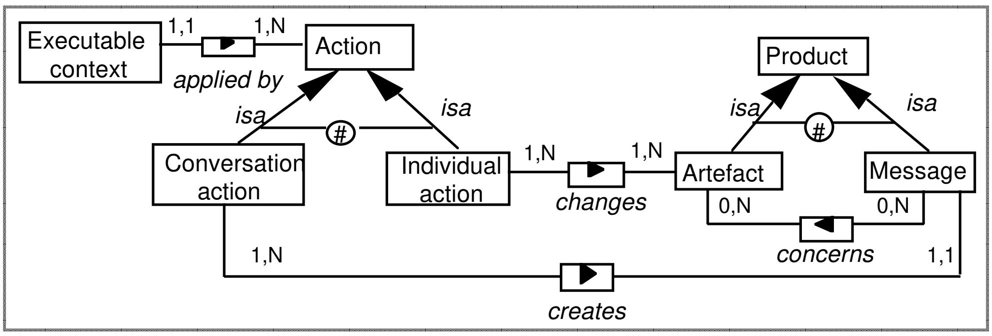

Figure 7 : Actions and products that they transform

Performing an individual action or a conversation action does not change the same kind of product. Individual actions perform transformations of artefacts while conversation actions create messages.

Therefore, we classify the concept of *product* into *artefact* and *message* (figure 7). *Artefact* represents the static component of the information system.

To keep track of conversations, we introduce the *message* concept as the basic component of the conversational activity. A *message* may *concern* several *artefacts.*

The *individual action* can be complex or simple. A *complex individual action* is composed of *individual actions*. A *simple individual action* performs a tranformation of an artefact by creating, updating or deleting it *(changes).* An *individual action* is performed by an *individual role* (figure 8). Figure 6 shows an executable context which is *applied by* an individual action.

We want also to deal with group activities, in the sense that several participants can synchronously act in the same activity by exchanging messages. We represent this type of cooperation by the *conversation action*.

The *conversation action* is *performed by* a *group role*. It creates several messages, each message being *produced by* an individual role *contained by* the previous group role (figure 8).

New *contexts* may *emerge* from any *conversation action* (figure 8). These contexts can be executable and associated to actions, which may themselves be conversational. This, in turn, triggers new contexts and so on. This feature enables the cooperative process meta-model to deal with ill-structured cooperative work processes.

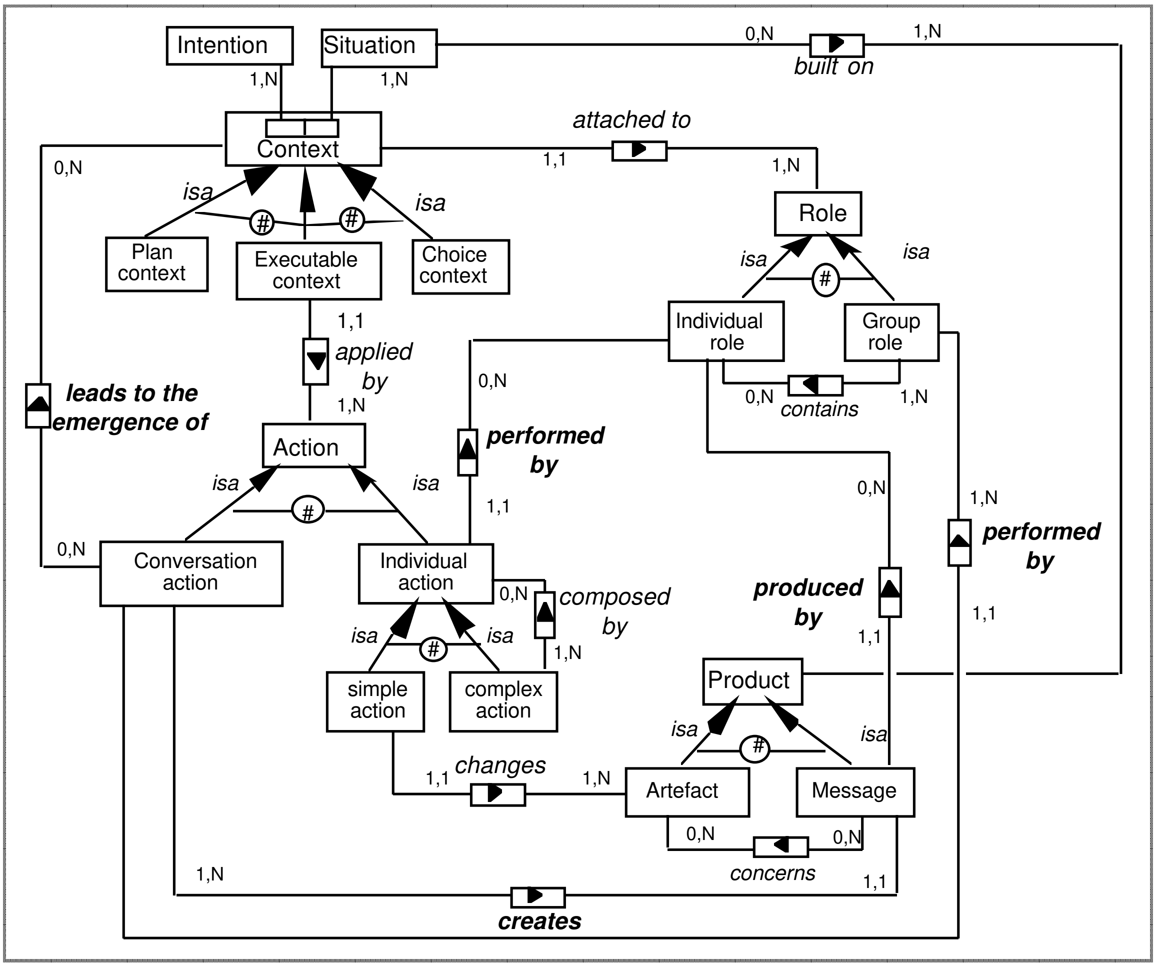

Figure 8 : The cooperative process meta-model

Executable contexts establish situation-based links among contexts, namely *correlation links*. This is modelled in figure 8 by the loop among contexts through action and situation. The term *correlation link* refers to the composition of the three following relationships: *applied by*, *changes/creates,* and *built on*.

3.4.2. Choice context

A user may have several alternative ways to fulfil an intention. Therefore, he/she has to select the most appropriate one among the set of possible choices. In order to model such a piece of process knowledge, we use a second specialisation of the concept of context, namely the *choice context* (figure 9).

A *choice context* corresponds to a situation which requires the exploration of alternatives in decision making. Each alternative is an approach or a strategy for the resolution of the issue being faced by the user in the current situation. By definition a choice context offers a choice among a set of strategies, all of them achieving the same purpose. In this sense, one can look upon the choice context as being *goal oriented.*

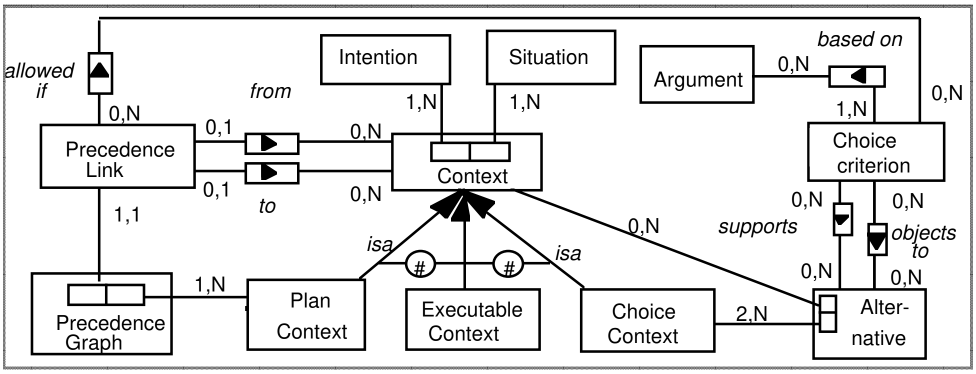

Figure 9 : The representation of the concept of context

There are two major differences between the *choice context* and the *executable context*: the first one lies in the absence of any alternatives in the latter and the second is that a choice context has no direct consequence on the product.

In the process meta-model, the various alternatives of a choice context are represented in the *alternative relationship* (figure 9). They are associated to choice criteria based on arguments.

A choice criterion is a combination of arguments which *supports* or *objects to* an alternative of a choice context. It may provide priority rules to select one *alternative* among several depending on the *arguments.*

Since alternatives of a choice context are also contexts, contexts may share an *alternative* relationship (figure 9), leading to *alternative-based hierarchies of contexts*. The alternative-based relationship among contexts allows the refinement of large-grained decisions into more fine-grained ones. It establishes refinement-based relationships that we refer to as refinement links. This is a means by which the process meta-model handles the granularity problem (figure 10). The refinement link is represented in the meta-model by the alternative relationship.

The notion of *alternatives* and *choice criteria* allow the way-of-working to support the user in exploring possible strategies to resolve an issue and selecting the most appropriate one. This alternative-based guidance leaves freedom to the user who can make a choice which is not even one of the predefined alternatives proposed by the way-of-working.

This non determinism introduced in the meta-model will be useful in order to improve the way-of-working by adding new alternatives in some of its choice-based contexts. This feature enables the cooperative process meta-model to deal with exception handling in workflow applications.

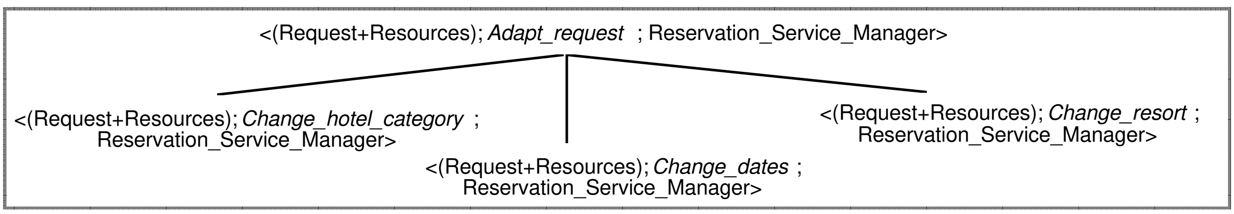

Figure 10 : Example of choice context

3.4.3. Plan context

In order to fulfil an intention associated to a certain situation, a user may be required to take a set of decisions on corresponding situations; he/she has to follow a plan. To this end, a third specialisation of context, namely, *plan context* is introduced.

A *plan context* is an abstraction mechanism by which a context viewed as a complex issue can be decomposed in a number of sub-issues. Each sub-issue corresponds to a sub-decision working on a sub-situation. The decomposition of context is another means provided by the meta-model to solve the granularity problem.

In the process meta-model the decomposition of a plan context into its more elementary contexts is represented (figure 9) by the relationship *precedence graph* between *context* and *plan context*. The component contexts can be of any type i.e. executable, choice or plan contexts.

For example, for the intention named "Satisfy_Request" to be fulfilled, the three decisions "Analyse_Request", "Adapt_Request" and "Create_Reservation" need to be taken. This is modelled (figure 11) by a plan context called "<(Request+Resources), Satisfy_Request, Reservation_Service_Team>" decomposed into three contexts: "<(Request+Resources), Analyse_Request, Reservation_Service_Clerk>", and "<(Analysed/Adapted Request+Resources), Create_Reservation, Reservation_Service_Clerk>" executable contexts, and "<(Analysed Request+Resources), Adapt_Request, Reservation_Service_Manager)>" choice context.

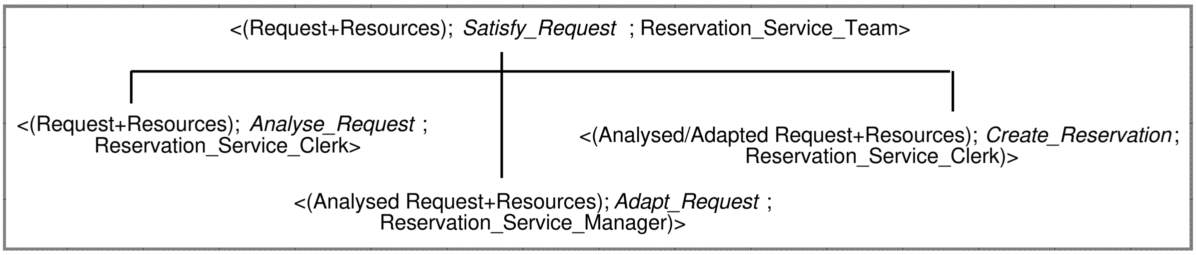

Figure 11 : Example of a plan context

The ordering of the contexts, within a plan, is defined by the *precedence graph*. There is one graph per plan-based context. The nodes of this graph are contexts while the links -called *precedence link*s- define either the possible *ordered transitions* between contexts or their possible *parallel enactment*. Based on arguments, a choice criterion may be assigned to a link. The choice criterion defines when the transition can be performed.

Flexibility is introduced by allowing several sets of possible parallel or ordered transitions to be defined in the same graph. This feature enables the cooperative process meta-model to deal with well-structured workflow applications which require the use of a model in terms of ordered steps.

Flexibility also results from the implicit "abort" feature meaning that the plan can be aborted by the users, during the process, at any moment This is another aspect of the non determinism allowed by the meta-model.

The precedence graph corresponding to the previous plan context is shown by the figure 12.

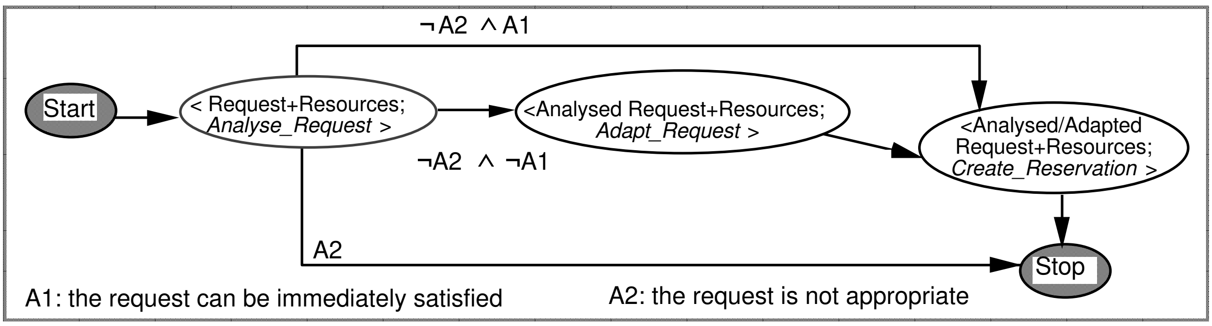

Figure 12 : Example of a precedence graph

Decomposition of contexts can be made iteratively leading to hierarchies of contexts. This hierarchical link is refered to as a *decomposition link.* Notice that this link corresponds in figure 9 to the composition of the *precedence graph* relationship with the *from* and *to* relationships.

Plan contexts provide a different type of guidance than executable and choice contexts do. They support the user in performing long term transactions, providing advice on the ordering of component activities, whereas *choice contexts* help in making the appropriate choice in the situation in hand and *executable contexts* tell how to implement the decision taken.

Each type of context influences the on-going process in a different manner: an executable context affects the product and generates a new situation, which itself becomes the subject of decisions; a choice context does not change the product but helps to further the decision making process through the refinement of an intention; a plan context provides the means to manage the complexity of an intention by providing a decomposition mechanism. Performing decomposition and refinement iteratively allows the users to reach executable intentions and thus, to act on the product.

3.5. The concept of way of working

As introduced in section 3.1, a way-of-working is a process model resulting from the instantiation of the cooperative process meta-model. The basic building block of a way-of working is an instance of context type that we call also context.

Contexts types in the meta-model have hierarchical relationships of two different types, decomposition and refinement. In the way-of-working, we suggest a grouping of contexts (instances) based upon these links. The modules resulting from this grouping are hierarchies of contexts called *trees*. This leads to the final vision of a way-of-working as a *forest* of trees (figure 13).

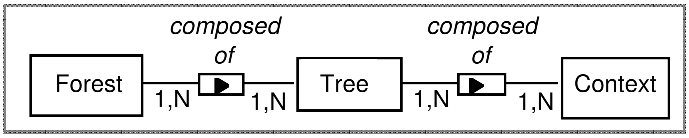

Figure 13 : The way-of-working structure

A non executable context instance can be decomposed/refined into other contexts. Usually, one of these contexts is the subject of interest. However, it may happen that the user wishes to deviate from these. To handle such a case, an "open-choice" is assumed to be associated with every non executable context. This constitutes an open alternative in a choice context and an abort option in a plan context.

Finally, in the meta-model, contexts types have situation based relationships which are reflected in the way-of-working by introducing links between contexts instances of the same tree or of different trees. This leads to the final vision of a way-of-working as a network of contexts.

As an instantiation of a context type, a context is defined by a couple (situation, decision). In the example presented in figure 11, the situation is formed by the customer request and the existing resources in the set of resorts, and the intention is "Satisfy request".

A context instance has a type, namely executable, choice or plan and its description includes its related elements. The description of a choice context will include for example, the list of its alternative contexts, their related arguments and choice criteria whereas an executable context will contain the description of the action and its impact on the product. The context of figure 11 is a plan context composed of two executable contexts for respectively, analysing the customer request and creating reservation, and a choice context for adapting the request. Figure 12 expresses the various possible sequences for these context instances.

This leads us to consider hierarchies of contexts, resulting from the iterative application of either decomposition or refinement link, as meaningful modules of the way-of-working. We call these hierarchies trees. A *tree* has a root which is, either a choice context or a plan context. It has a variable number levels. The more global the root intention is, the more levels are required to support its decomposition and/or refinement. One may notice that the nodes of a tree are contexts of any of the three types: executable, choice or plan. Executable contexts are leaves of trees, and if the way-of-working is completed, all the leaves should be executable contexts. The intermediate nodes can be either choice or plan contexts.

A way-of-working is normally composed of a collection of trees, it is a *forest of trees*. The existence of multiple trees in a unique way-of-working comes from the independence of some contexts due to the independence of either situations or decisions/intentions. For example, a number of executive decisions which can be taken from scratch, participate to contexts which are roots of trees. This can also occur to choice contexts and plan contexts. However, at the same time, such contexts can be embedded in a more global tree as sub-trees.

The guidance provided by a context depends on its type. An executable context tells which action to perform. A plan context, if exclusively composed of executable contexts, plays a similar role, it describes the set of ordered and/or parallel actions to be performed to fulfil the intention of the plan. More generally, a plan context supports the description of a well structured cooperative process. In this case, the component contexts can be executable, plan or choice. The corresponding precedence graph defines the ordering of the component contexts, and consequently, the coordination of the various roles. A choice context supports a user in the deliberation process leading to either the choice of one alternative among a set of possible choices or to the open choice given in every context.

The progression from one context to another one is supported by the knowledge encapsulated in context links. Hierarchical and correlation links play here a different role.

Correlation links are useful after the execution of one (in case of an executable context) or several actions (if an executable plan), to decide on which context to work in the next step. Correlation links offer choices to the user who can follow one of the suggestions or not.

Hierarchical links (refinement or decomposition) help either in the refinement or in the decomposition of an intention expressed at a level which requires to be made more detailed in order to be implemented.

Even in circumstances where the user does not wish to follow the prescriptions of the way-of-working, guidance can still be provided based on a different exploitation of the knowlegde encapsulated in the way-of-working. For example, applying a pattern matching algorithm, it is always possible to select all situations in the current state of the product on which guidance can be provided. Similarly, all situations matching a certain intention can be displayed. Thus the guidance provided is a combination of two guidance modes of the way-of-working :

- the prescriptive mode in which trees and contexts are systematically followed, and
- the user-driven mode in which the user deviates from the prescriptive mode. The process knowledge acquired by the user is capitalized, for example, to take advantage of the "open-choice" referred to earlier.

We claim that any way-of-working related to any cooperative work process can be described as a network of context-trees. A context-tree called a tree instance gathers all associated contexts instances together by either a composition link (through plan context) or refinement link (through choice context), the leaves of the tree being executable contexts. A tree encapsulates an interesting piece of process knowledge. It is a process chunk which provides support to solve the issue raised by the root context. This support comes in two ways:

- by hierarchical links which help to deal with the decomposition or the refinement of a complex decision.
- by the accumulation of a large amount of heuristical knowledge in dependency graphs, arguments and choice criteria. To be able to express heuristics is an important improvement in way-of-working definitions since it allows the capitalisation and sharing of experiences.

The way-of-working is viewed as a network of context-trees because it integrates correlation links among contexts. A correlation link (through executable contexts) provides knowledge to establish situation-based connections in the cooperative process. It is a useful piece of knowledge for guiding the process flow (suggesting what can possibly be done next). Thanks to the meta-model :

(a) ways-of-working (process models) have a good level of genericity,
(b) they are defined in a modular manner with contexts as basic building blocks and trees as modules and are consequently easy to evolve and change, and
(c) they guide participants to a cooperative work process locally (within one context) and globally as well (to monitor the context flow) in a flexible manner.

4. Conclusion

The impact of the use of computer supported cooperative work products on the work organization has to be taken into account in order to develop effective applications. The analysis and the design of cooperative work processes require appropriate methods [Dumas 90a] [Ellis 94] [Loucopoulos 95] [Nurcan 96a] [Yu 94].

The aim of cooperative process analysis is to find the right division of a given work process in tasks in order to implement the flow of tasks when it is possible. An appropriate method has to:

• consider complex organizations whose work processes are not clearly defined,
• discuss objectives (goals) to reach and not different functions of the organization,

• allow to model any work process even when it contains stages which cannot be implemented using a workflow product,
• have a global view of the organization in order to determine roles,
• take charge of the analysis from the identification of work processes to modeling of procedures that must be implemented [Rolland 97].

In this paper, we have presented a *cooperative process meta-model* which combines the representation of well-structured cooperative work processes and the modelling of ill-structured cooperative work processes.

This meta-model allows us:
• to represent cooperative work processes,
• to integrate conversations between agents,
• to guide and keep track of what happened in cooperative brainstorming sessions,
• to model the emergence of new contexts;
all these being made in an homogeneous manner.

An instantiation of the cooperative process meta-model results in a cooperative process model allowing to deal with a large variety of situations in a decision-oriented manner.

The concept of plan context enables the cooperative process meta-model to deal with well-structured cooperative processes which require the use of a control model [Nurcan 96c] [Nurcan 97]. The corresponding precedence graph defines the ordering of the component contexts, and consequently, the coordination of the various roles.

The alternative-based guidance of the choice context leaves freedom to users who can make a choice which may not even be one of the predefined alternatives proposed by the way-of-working. This feature allows the cooperative process meta-model to deal with exception handling in cooperative processes.

The concept of conversation action allows us to represent ad-hoc group activities. It enables the cooperative process meta-model to deal with ill-structured cooperative processes and the unstructured component of globally well-structured cooperative processes.

Our current work consists of building a cooperative environment which supports the definition of cooperative process models (in terms of ways-of-working) and provides the flexible guidance of groups in well-structured and/or ill-structured cooperative work processes. This environment is an extension of the MENTOR process centred environment [Si-Said 96].

Bibliography

[Dumas 90a] : Dumas, P. and Charbonnel, G., *La méthode OSSAD - Pour maîtriser les technologies de l'information - Tome 1: Principes*, Les Editions d'Organisation, Paris, 1990.

[Dumas 90b] : Dumas, P., Charbonnel, G. and Calmes, F., *La méthode OSSAD - Pour maîtriser les technologies de l'information - Tome 2: Guide pratique*, Les Editions d'Organisation, Paris, 1990.

[Ellis 79] : "Information Control Nets, A Mathematical Model of Office Information Flow", In *Proceedings of the ACM conference on Simulation, Measurement and Modelling of Computer Systems*, 1979, p.225-240.

[Ellis 91] : Ellis, C.A., Gibbs, S.J. and Rein, G.L., "Groupware: some issues and experiences", *Communications of the ACM*, 34(1), 1991), p.38-58.

[Ellis 94] : Ellis C.A., Wainer J., "Goal-based models of collaboration", *Collaborative Computing*, Volume 1, Number 1, March, 1994, p.61-86.

[Henninger 91] : Henninger, S., "Computer systems supporting cooperative work: a CSCW'90 trip report", *ACM SIGCHI bulletin* , 1991, 23(3), p. 25-28.

[Khoshafian 92] : Khoshafian, S., Baker, A.B., Abnous, R. and Shepherd, K., "Collaborative work and work flow in intelligent offices",*Intelligent Offices: Object-Oriented Multi-Media Information Management in Client/Serveur Architectures,* Wiley, 1992.

[Kuwana 93] : Kuwana, E. and Sakamoto, Y., "Toward integrated support of synchronous and asynchronous communication in cooperative work: An empirical study of real group communication", *Proceedings of the conference on organizational computing systems, COOCS'93*, ACM,1993, Milpitas, California, p.90-97.

[Loucopoulos 95] : Loucopoulos P., Kavakli E., "Enterprise Modelling and the Teleological approach to Requirements Engineering", *International Journal of Cooperative Information Systems*, 1995, Vol. 4, N° 1, p. 45-79.

[McCarthy 93] : McCarthy D.R, Sarin S.K., "Workflow and transactions in InConcert", *Bulletin of Technical Committe on Data Engineering*, 1993, Vol. 16, N° 2, IEEE, june, *Special Issue on Workflow and Extended Transactions Systems*. p. 53-56.

[McLean 93] : McLean, A., Moran, T. and Young, R.M., "Design Rationale: The argument behind the artifact", *CHI'93 Conference Proceedings*, ACM, 1993, p. 247-256.

[Naffah 94] : Naffah, N., "Workflow: Etat de l'art et évolution", In *Proceedings of the Conference IT FORUM'94,* Télécom Paris, 1994.

[Nurcan 95a] : Nurcan, S. and Trolliet, J.Y., "Une méthode d'analyse et de conception pour les applications workflow", In *proceedings of the 13th INFORSID congress*, May 31-June 2 1995, Grenoble, p.453-472.

[Nurcan 95b] : Nurcan, S. and Chirac, J.L., "Quels modèles choisir pour les applications coopératives mettant en œuvre les technologies de workflow et de groupware ?", In *Proceedings of the AFCET 95 congress*, October 25-27 1995, Toulouse, p.593-602.

[Nurcan 96a] : Nurcan, S., "A method for cooperative information systems analysis and design: CISAD", In *Proceedings of the Second International Conference on the Design of Cooperative Systems (COOP'96)*, 12-14 juin 1996, Juan-Les-Pins, p.681-700.

[Nurcan 96b] : Nurcan, S., "Analyse et conception de systèmes d'information coopératifs", *Numéro thématique "Multimédia et collecticiel" de Techniques et Sciences Informatiques,* 1996, Vol. 15, N° 9.

[Nurcan 96c] : Nurcan, S., Gnaho, C., Rolland, C, "Defining Ways-of-Working for Cooperative Work Processes", In *Proceedings of the First International Conference on Practical Aspects of Knowledge Management (PAKM) Workshop on Adaptive Workflow*, October 30-31,1996, Basel, Switzerland.

[Nurcan 97] : Nurcan, S., Rolland, C : "Meta-modelling for cooperative processes", *7th European-Japaneses Conference on Information Modellling and Knowledge Bases*, May 27-30, 1997, Toulouse, France.

[OVUM 91] : OVUM,*Workflow Management Software*. Ovum Ltd., London, England, 1991.

[Palermo 92] : Palermo, A.M. and McCready, S.C., "Workflow software: A primer", In *Proceedings of the Conference GROUPWARE'92,* 1992, London, p.155-159.

[Rolland 95] : Rolland, C., Souveyet, C., Moreno, M., "An approach for defining ways-of-working", In *Information Systems Journal*, 1995, Vol. 20, N° 4.

[Rolland97] : Rolland C., Nurcan S., Grosz G., "A way of working for change processes", *International research Symposium: Effective Organisations*, September 4-5, 1997, Dorset, UK.

[Si-Said 96] : Si-Said S., Rolland C., Grosz G., "MENTOR : A Computer Aided Requirements Engineering Environment", *in the Proc. of the 8th CAISE Conf. Challenges In Modern Information Systems*, Heraklion, Crete, Greece, May 1996.

[Swenson 93] : Swenson K.D., "Visual Support for Reengineering Work Process", In *Proceedings of the Conference on Organizational Computing Systems*, ACM, Milpitas, California, 1993, p. 130-141.

[Yu 94] : Yu E.S.K., Mylopoulos J., "From E-R to "A-R" - Modelling Strategic Actor Relationships for Business Process Reengineering", In *Proceedings of th 13th Int. Conference on the Entity-Relationship Approach*, December 13-16, 1994, Manchester.