A Case-Based Reasoning Support System in Network Traffic Control

Patrice Caulier
LAMIH - URA CNRS 1775
University of Valenciennes
Le Mont Houy, BP 311
F-59304 Valenciennes Cedex
FRANCE

Abstract: Case-based reasoning offers special advantages for continuous control applications, but these applications also have special requirements that demand extensions to the case-based reasoning paradigm. This article describes a case-based reasoning support system for control of telephone traffic and examines issues it raises. We present different stages of the methodological approach, based on knowledge acquisition and case-based reasoning, developed to design the diagnosis support system and trace their motivation to the requirements of a continuous control application. The specific domain which we are applying knowledge acquisition and case-based reasoning to is the management of traffic flow in the standard urban public switched telephone network of the *Ile de France*. Main research objective is to place at the network managers disposal an intelligent system able to capitalize and re-use past incidents, in order to assist them in their diagnosis and decision -making activities. This system is developed within an interdisciplinary research co-operation supported by the CNET (the France Telecom research centre) and the CNRS (the French National Board for Scientific Research).

Keywords: Design methodology, activity analysis, knowledge modelling, case-based reasoning, incident case, diagnosis support system, network traffic control.

Patrice Caulier has received his MS in data processing applied to industry and computer science, in 1991, and he prepares his Ph.D thesis in automatic control and computer science, all at the University of Valenciennes, France. He is a member of AFIA, the French Association for Artificial Intelligence. His current research interests include knowledge modelling and the relevance of case-based reasoning to continuous control problems, more specially the management of traffic flow in the standard public switched telephone network.

1. Introduction

France Telecom is in charge of the installation and management of all network and routing structures and has decided to create, few years ago, new network management centres for long distance and urban networks. At the present time, there are several supervision centres, and these centres control the transit switches. Network traffic management is aimed at maintaining in optimal quality of service in case of abnormal situations. This task can be divided into two steps:

1. *supervision*: using the huge amount of data periodically collected in the network, analyze the status and traffic performance of the network in order to detect any abnormal decrease of the grade of service resulting from *traffic overloads* and/or *equipment failures*;

2. *control*: activate the consistent traffic controls in order to remove or minimize the effect of the disturbance.

This task becomes more and more complex due, on the one hand, to the size of the network under consideration and, on the other hand, to the sophistication of new services which are continuously introduced and provided to the customer. In such a context, a disturbance may appear and spread all over the network elements (resources and services) and the large number of data makes the diagnosis difficult to perform. It is therefore necessary to provide network managers with advanced computerized tools. Within this context, the aim of our research project is to incorporate case-based reasoning techniques into network management. Our case-based reasoning support system is designed to advise network managers on problems, and to recommend sets of controls that would alleviate those problems. In order to develop an effective support system, a

specific "user-centred" methodological approach has been defined [1]. This approach is based on three successive stages: (i) expertise data elicitation and network manager's activity analysis, (ii) domain knowledge modelling and (iii) case-based reasoning to develop the diagnosis support system. From the first stage of this approach, we have found that network managers seem to perform their control tasks by drawing on experience of previous control episodes. Case-based reasoning offers an appealing way to capture that episodic knowledge. In this article, we present the methodological approach we are using to deal with the diagnosis support system.

The article is organized as follows. The next section briefly overviews the network traffic management domain, and focuses on the monitoring and control aspects that have posed the greatest challenge to case representation. Then, our methodological approach to design the case-based network management assistant is presented in Section 3. The first knowledge acquisition (KA) stage is described in Section 4. Presentation of the architecture of the case-based reasoning (CBR) support system is considered in Section 5.

2. Network Traffic Management Domain

The specific problem to which we are applying CBR to is the management of traffic flow in the standard public switched telephone network of the Ile de France. Controlling such traffic is a problem of allocating a changing set of network resources to satisfy demands from a fluctuating pattern of calls. This control is exercised by a small group of experienced traffic management personnel, located at a centralized site, who modify the network's call processing in response to a continuous stream of network performance data. Network management is a very complex task, mastery of which requires extensive training and years of apprenticeship.

Lots of things can go wrong with a telephone network, and usually the general public is unaware of these problems. The principal objective of network management, in fact, is to minimize the effect network problems have on callers. A network manager's job is to optimize the traffic flow in the network, in order to increase the proportion of completed calls, and to maintain this transparency, even when network facilities fail or when local traffic demand temporarily exceeds the network's designed capacity for handling calls.

The sources of network data and the machines that execute the traffic controls are the automatic telephone switching systems (or just "switches") that are the distributed nodes of the telephone network, which are very large, specialized computers that process and route calls, each capable of handling thousands of connections simultaneously. These switches, and the trunk groups which interconnect them and over which calls are transmitted between nodes, are the finite-capacity network resources being managed. An individual phone is connected to just one switch, over a "line"; these individual lines are not part of network management, because there are no actions short of repair that can affect their performance. When a switch fails, either partially or completely, or a trunk group is severed, the network's total call processing capacity is diminished, and traffic managers search for the best available ways to complete calls despite the lost capacity. Even when the entire network capacity is available, it may be exceeded by demand, especially during holidays such as Mother's Day, when many calls are generated, during radio calls-in, or just on a particularly busy Monday morning. These exceptional demands are usually localized, and traffic managers can often find idle capacity in the rest of the network to satisfy the local demand.

Switches generally know over which trunk group to route an individual call, based upon the number dialed. Sometimes, though, abnormal demand fills up the trunk group (or groups) over which traffic for a particular destination is usually routed. The same problem occurs when trunk group capacity is reduced due to a facility failure such as a severed cable. When this happens, calls start to overflow and the caller hears (usually) a fast-busy tone. Network managers try to find temporary alternate routes for these calls, installing controls on the switch to re-route any calls that would otherwise overflow.

Because call routing requires processing by the switch, excessive demand can also interfere with a switch's own ability to handle all the calls that are routed to (or through) it. A common trigger for this situation is the radio call-in giveaway, which can generate a huge burst of traffic, focussed on the single switch that serves the radio station. The vast majority of these calls are "doomed" - they will fail anyway because the line is busy - but they must all be processed by the focus switch. This phenomenon is called focused overload, and can result in the focus switch performing less and less useful processing, instead of devoting its time to handling these doomed calls and performing more and more overhead operations, until it is essentially "thrashing". This is a very serious problem because neighbouring switches are affected as they spend more time trying to communicate with the overloaded switch, until they get themselves overloaded. In this way, this acts by installing "gap" controls around the network that cancel some share of the focus-destined calls.

This traffic management application is an instance of a class of *continuous control* problems, in which a stream of data is analyzed to diagnose the situation in the controlled system, an appropriate plan of action is devised for treating the situation, and the effects of control actions are continuously monitored and adjusted. Such a system, then, includes aspects of diagnosis and planning - classic artificial intelligence domains; indeed, planning is one well-known application of CBR [2]. As discussed below, other aspects of the CBR approach are of value for this class of application, but extensions to CBR are necessary to satisfy special requirements of these applications, and are proposed as part of this research project.

3. Design Methodology

The aim of our research project is to place at the network managers' disposal an intelligent tool designed to capitalize and re-use past solved incident cases in order to assist them in their traffic management activities.

Design of such a support system requires artificial intelligence methods but also methods and techniques of the cognitive sciences and especially , using of a CBR proposed. The principal difficulties in the design and development of a CBR system lie in the case definition, in the indexing algorithm definition, in the similarity measure to retrieve a similar incident case and in the adaptation rules definition. Which are the relevant attributes to define data of a new problem? and How to retrieve and adapt a solution for this problem?

In order to achieve these objectives a methodological "user-centred" approach has been defined. This approach is based on KA, to model domain knowledge, and CBR, to develop expertise (represented at different knowledge levels of the expertise model) into a decision support system. Figure 1 gives a representation of this approach, which is described in this article.

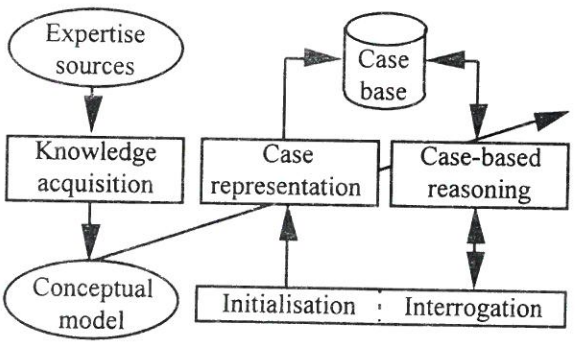

Figure 1. The Methodological Approach

The first stage of the approach consists in expertise model-driven knowledge acquisition. The *expertise model*, as a result of the KA stage, is the most important aspect used in our project. It consists in a symbolic representation of the manager's knowledge. All the knowledge about an entity (description and use) is collected in a single component of the model. This expertise model allows to represent two knowledge sets: specific entities of the domain (universe of domain concepts), specially the structure of an incident case, shown as the doublet {symptom, solution}, and the universe of problem -solving methods used by managers (e.g. information retrieval). Finally, the expertise model is the interface between the two stages of the approach. The expertise model is used in the second stage to:

- specify the incident case definition;
- initialize case base of the CBR support system;
- define the different problem- solving methods (retrieve, re-use, revise and retain) used by this cognitive system to find the incident solution.

The second stage consists in using results supplied by the first one, in order to define and exploit an incident case base with a system based on CBR. CBR is a form of analogical reasoning. It consists in solving a new problem, or in illustrating, explaining or criticizing new situations, from (by re-using) previous similar cases. CBR systems are composed of a case base, a retrieval mechanism to select and extract similar cases and an adaptation mechanism of selected solutions in order to solve the initial problem [3]. Studies on this new reasoning form, very frequently used, were initialized, in the USA, in the early 80's, with works of R.C. Schank [4].

4. Knowledge Acquisition

Knowledge acquisition is the first stage of our design methodology. Knowledge engineering has traditionally been viewed as a process of extracting knowledge from a human expert, and transferring it in computational form to the machine [5]. This conventional view was also reflected in the popularity of rapid prototyping using rule-based shells. In contrast today knowledge engineering is approached as a modelling activity: the heart of the work of the knowledge engineer lies in the actual construction models. Indeed, the growing interest in the specification and development of user -friendly and easy to maintain decision- making support systems, has urged knowledge engineers to develop and use problem- solving knowledge acquisition and modelling methodologies. In this section, we present the application of the *CommonKADS* knowledge level modelling framework [6] in the field of the network traffic management. This first KA stage takes place in our global methodological approach to design a CBR support system for continuous control applications and includes two parts: knowledge elicitation and activity analysis and using of the CommonKADS knowledge modelling methodology. In a first part, the objective is to model the expertise used in network traffic management domain. In this modelling context, which reveals an activity based on experiences (i.e. cases) re-using, the second objective is to (each of these results is provided by the expertise model of the application):

- locate the incident reusing activity into the global control activity;
- describe this activity in the information retrieval to solve the current problem;
- identify relevant descriptors of the incident case model;
- identify best index to organize the case base;
- define a similarity measure for matching;
- record knowledge necessary to adapt solution part of the selected case, in order to solve the current problem.

This section provides an overview of the KA stage. In the first subsection the knowledge elicitation and activity analysis part is presented. In the following subsections the CommonKADS

knowledge level modelling framework is put forward. We describe the CommonKADS expertise model and the main categories of knowledge are being discussed: domain knowledge, inference knowledge and task knowledge.

4.1 Knowledge Elicitation and Activity Analysis

Activity analysis of traffic managers allows to identify domain knowledge used in different stages of their reasoning. As discussed below, we have found that the expertise needed for a diagnosis or decision- making is widely made up of experiences. Elicitation sessions of the expertise, in the control room, are based on different methods: document analysis, interviews, repertory grids, traffic manager's activities analysis and results of the activity (problem reports). Each method presents its own goal and allows, generally, to obtain a particular type of knowledge. Therefore, it is necessary to use these methods concurrently, one cancelling out the drawbacks of others taken apart, benefiting the qualities of each one [7, 8].

From interviews with traffic management experts, we have found that a large part of their knowledge is *episodic*. That is, the expert solves a new problem by relating the current network situation to his previous experiences. These experiences are sometimes specific incidents, with real dates and places, and sometimes general classes of similar occasions. This body of traffic management experience is transferred from expert to apprentice as "war stories" of illustrious and (sometimes) ignominious traffic controls of the past. This "case-based" approach to the teaching of traffic management recommends cases to represent knowledge in this domain. Certainly, knowledge acquisition naturally results in traffic management cases.

The appropriateness of cases is justified by more than this anecdotal evidence. From our understanding of the domain, it appears that past experience is the best available guide to decision-making. Although there is some knowledge of general causal relationships between control actions and network responses, no complete model of the domain exists; traffic managers often act in accordance with previous success, with an incomplete understanding of the reason why that success occurred. This is understandable, given the intractable scale of the network dynamics and the time constraints on responses. CBR offers the best available means for taking advantage of this shallow, episodic knowledge [9].

Traffic management, as well as many other applications - especially control of high technology hardware - are in a state of continuous, gradual change. In the traffic management domain, these changes occur because the telephone companies increase the number of switches under traffic management, add new types of switches to the operation, add new types of services or add new hardware/software components that gradually change the behaviour of the network. Human experts are often unaware of such changes, gradually adapting and applying a modified version of their previous experience to new situations. In contrast, conventional non-CBR-based expert systems would exhibit a continuous and mysterious degradation in performance in the face of such change.

The CBR approach is a natural fit for such evolving domains: previous experiences (cases) are applied to new situations, and new cases are created to address situations for which the existing cases are inadequate.

4.2 Knowledge Modelling

Second part of the first KA stage is the knowledge modelling part. Among the available general -purpose knowledge modelling methodologies (e.g. *CommonKADS*, *KOD*, *MERISE*, *SADT*, etc.), we have selected the *CommonKADS* methodology [10]. Indeed, CommonKADS offers an open representation formalism, works on a computerized tool, and results from eight years of European project work. We present in this second subsection the knowledge modelling activity in our research project.

4.2.1 The CommonKADS Methodology

The CommonKADS methodology, *Common Knowledge Acquisition and Design Support*, is issued from research works developed into the KADS-II European ESPRIT project (P5248). CommonKADS methodology is certainly the most complete and the most known and used in the knowledge acquisition community [11]. The development of a KBS is seen as the construction of a set of models of problem-solving behaviour, viewed in its concrete organizational and application context. A KBS is a computational realization associated with these models.

Among modelling activities identified in CommonKADS, the main one is the application's know-how modelling. Indeed, a central model in the CommonKADS methodology is the *expertise model*, which models the problem- solving behaviour of an agent in terms of knowledge that is being applied in carrying out a certain task.

Figure 2 summarizes the suite of models involved in the CommonKADS methodology (see [12]) for a comprehensive description of the models and of the dependencies between them). The expertise model is only one of many models that are relevant to KBS development. Other models capture relevant aspects of reality such as the task that an application supports, the organizational environment within which it takes place, the assignment of tasks to agents, their capabilities and communication, and the computational design of the KBS. It is noted that these models are engineering-type models and serve engineering purposes. An additional important point is that in the CommonKADS methodology the models are considered not only as "steps along the way", but as independent products in their own right to play an important role during the entire KBS life-cycle.

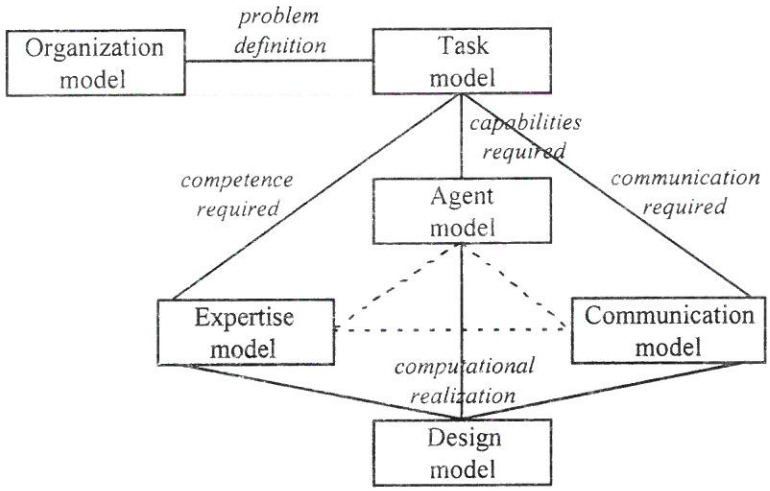

Figure 2. Position of the Expertise Model in the CommonKADSSsuite of Models

The organization model is the tool to analyse an organization with. The task model captures the global task within the organization that deals with a certain function that needs support, including the assignment of tasks to agents. The agent models describe agent capabilities, the expertise model agent competence involved in realizing the overall task. The communication model describes the communication behaviour among the agents. The design model describes the structure and mechanisms of the artefacts - usually KBS - involved in the task. The lines indicate direct dependencies between elements of the models.

We have used the CommonKADS methodology in this first stage of our approach, to specify the expertise model of the network traffic management activities.

4.2.2 The Expertise Model

As specified in the previous paragraph it is the CommonKADS expertise model, developed through the cognitive engineering workbench *KADS-Tool*, from *Ilog* company [13], that we have used to model domain knowledge and manager's activities. The expertise model for the KBS then reflects the required expertise of the system rather than that of the human expert. Of course, a model of human expertise may serve as an important input for modelling the expertise of a KBS, but it is not necessarily identical to it. The CommonKADS expertise model of application is made up of three nested level of models [14]:

- the *domain level* represents entities of the application domain used by the reasoning mechanisms during problem- solving;

- the *inference level* represents the operations performed during the reasoning process, and the roles played by the domain components in these operations;

- the *task level* defines the precise control structure that links together the operations (i.e. the inference structures) defined at the inference level and necessary to achieve a goal.

In order to describe an application at these knowledge levels, it is necessary to provide knowledge for each of these categories. An application system cannot be described by, for example, domain knowledge alone. The knowledge levels of the expertise model are always linked. Thus, task level *controls* the inference level and the inference level *describes* the domain level (see Figure 3).

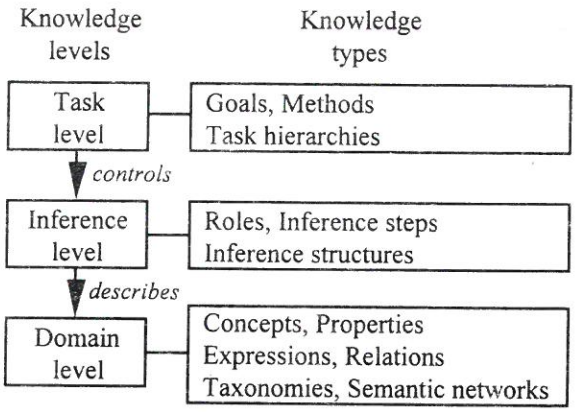

Figure 3. Hierarchical Levels of the Expertise Model

4.2.3 The Domain Knowledge Level

This modelling level expresses and organizes the components relevant to the application domain. Domain components are *concepts* (i.e. physical entities or abstractions of the reasoning process), *properties*, *expressions* (i.e. states or conditions about the concept properties) and *semantic relations* between these components. These domain components are linked together by *taxonomies* and/or *semantic networks*. Hierarchy, presented in Figure 4, describes a part of the typological classification of the application concepts.

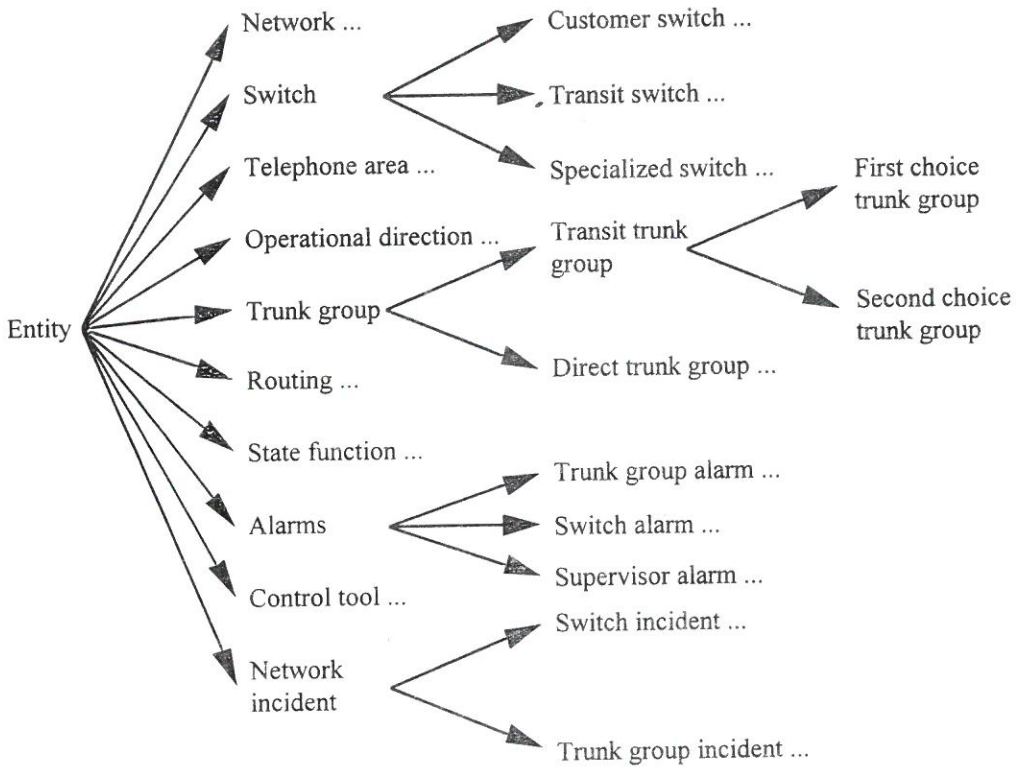

Figure 4. Typological Classification of the Domain Concepts

At the domain level, the expertise model brings some information to define an incident case. Indeed, the incident case concept definition is a great problem for the assistance system development. A traffic management case must include descriptors allowing, on the one hand, to index and retrieve a (or several) case(s) in the memory and, on the other hand, to give the user information for diagnosing and solving the current incident. These descriptors, or attributes, are given by the compositional description of the incident concept and the properties of this concept. Descriptors are organized according to four generic categories:

- *incident detection*. This first part concerns information about the context when an incident is detected in the network (i.e. symptoms): alarm indicators, incident type, date, detection tool, network state, etc.;

- *incident diagnosis*. From symptoms, descriptors of this part inform the user on the causal relationships, on the diagnosis strategy and give some explanations;

- *incident correction*. This part list the actions that should be taken to correct the problem, describe the expected effect of those actions, describe the situation when the problem has diminished and the actions should be undone, which is followed by some descriptors describing the plan for undoing the actions and the situation that is expected after all actions are undone;

- *incident information*. We find here some "informative descriptors" to inform the operator on the incident; as for example: code SGTQS of a network object (switch or circuit group), characteristics of this object, number of lost calls, comments, etc.

4.2.4 The Inference Knowledge Level

Inference knowledge is knowledge about how domain knowledge can be applied in the reasoning process. An inference operates on some input data and has the capability of producing a new piece of information as its output. The major components of the inference knowledge are:

- *Inference steps* as functional components that define the elementary reasoning steps operating on restricted parts of the domain knowledge.

- *Static roles* pointing to domain knowledge elements that are used in the problem -solving process but not affected by it. Static roles represent the domain dependent information used by the inference steps to realise their functions. A static role may, for example, be associated with a set of rules.

- *Dynamic roles* pointing to domain knowledge elements that are manipulated in the problem -solving process. Dynamic roles represent the information flow (i.e. the input and the output of the inference steps).

- *Inference structures*, diagrams showing the dependencies between inference steps and roles.

The inference structures are the backbone of CommonKADS. They specify the operations (or inference steps) used to solve the problem, as well as the information used or produced during the reasoning process. The formalism derives from the data-flow notation, which it extends. An inference structure represents the organisation of a set of inference steps linked by roles. It is the inference structure which describes the operations. It shows the different operations, or actions realized by the manager to transform domain components into other domain components. An inference structure is described by a set of inference steps and roles. The inference step defines a basic operation realized during the reasoning process. The role characterizes a set of data which plays an actor role in the problem solving. Role defines how a domain component is used at the inference level. Association of a role with a domain component allows to apply the reasoning to the application domain.

Inference structure of Figure 5 describes the "evaluation reasoning" developed by the network managers. This Figure contains a part of notations used to define an inference structure. An inference structure represents the organisation of a set of operations: the inference steps represented by ellipsis. These operations consume and produce data. This information named dynamic roles is represented by simple rectangles attached to the ellipsis by links that may be oriented. If the connection mode between a role and an inference step is not known, the rectangle is attached to the ellipsis by a non-oriented link. In this case, it is the control of the reasoning that determines dynamically the nature (input/output) of the role. The inference steps may need domain dependent knowledge to be implemented. This information, named static roles, is represented by bold rectangles.

The operations as well as the roles may be described at different levels of abstraction. An inference step may be "primitive". In this case, it models a terminal operation which is not further described, and is represented by a single ellipsis. It may also be "decomposed". In this case, it represents an abstract operation which is further specified by the inference structures that represent its decomposition and is represented by a double ellipsis. The decomposition of the inference step will be executed by one or many inference structures. Finally, within an inference structure that represents the decomposition of an inference step, the inherited roles defined at the upper level of the decomposition, are represented by hashed rectangles. The binding of the inference structure, that describes the problem independently of the domain, to the domain model is specified on the roles. For each role, one describes the domain elements that it represents.

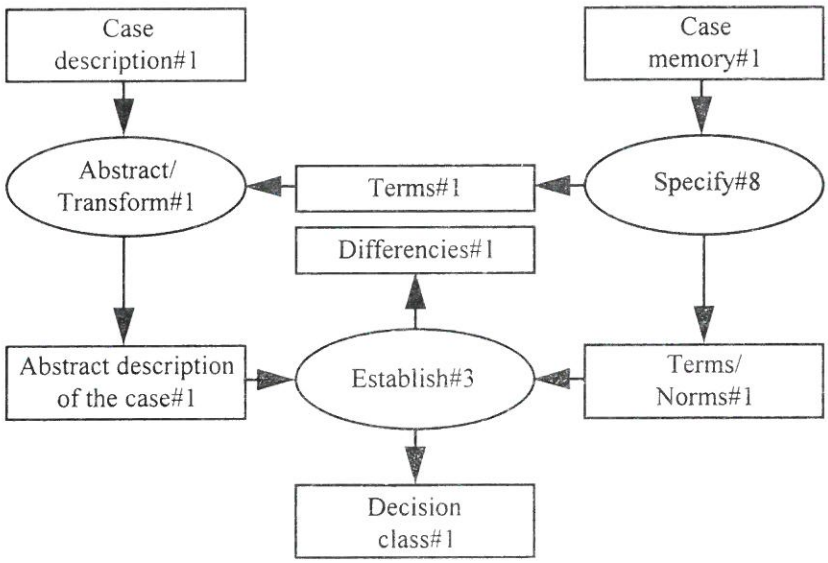

Figure 5. Structure of the "Evaluation Reasoning"

4.2.5 The Task Knowledge Level

The task knowledge describes through a *hierarchy of tasks*, the tasks that must be carried out in order to achieve a particular problem-solving goal as well as the control over execution of these tasks. It consists of two parts:

- *Task definition*, a declarative specification of the goal of the task, or of what the task needs to achieve. It consists of a goal, input/output roles and a task specification.

- *Task body*, a procedural specification of the activities contained within the task, or how the task achieves its goal.

A hierarchy of tasks represents a tree of *goals* (task definition) and *methods* (task body). A goal is typically an intermediate objective of the application and equal to an inference step. Thus, methods specify how the inference structures are organized to achieve an application goal. A method corresponds to a specific control algorithm applied to describe the dynamic aspect of an inference structure. There are three different types of tasks body or methods:

- *Composite tasks*: tasks that are further decomposed into sub-tasks (e.g. diagnosis is decomposed into generate and test).

- *Primitive tasks*: tasks that are directly related to inferences (i.e. not to be further decomposed).

- *Transfer tasks*: tasks of interaction with the world (e.g. the user). These tasks are not further specified in the expertise model but are part of the communication model.

5. Case-based Reasoning Support System

Knowledge used by a KBS is represented by a base where there are defined all the behaviours to apply to facts, which describe the problem to solve. This organization requires the reasoning to be clearly defined and the possibility of applying

a set of basic reasonings (rules). In our case it is not like that: manager's reasoning can be obtained only from an analysis of the knowledge used and produced to solve a current traffic incident.

An alternative to this classical vision of the knowledge base of a KBS consists in representing each solved problem in the system, by a real case base and not by a classical knowledge base. Then, resolution of a new problem consists of retrieving, in the case base, a similar incident. The incident solving method allows to infer the solution for the initial problem. This principle defines the CBR [15].

Before presenting the main functionalities of our CBR support system, we present, in the two next paragraphs, the analogical reasoning and the CBR.

5.1 Analogical Reasoning

Analogical reasoning is a powerful mechanism for exploiting past experience in planning and problem- solving [16]. Analogical reasoning is a basic problem -solving technique. An analogy is often defined as "a problem of the form A is to B as C is to D ($A : B :: C : D$), where, in most situations, the last term is omitted and must be filled in, selected from among answer options, or confirmed in a true-false situation". Analogical reasoning is a process by which D is determined by some known properties among A, B and C. It is a core process of thinking and has been studied extensively in artificial intelligence and cognitive science [17]. A major reason for using analogical reasoning is that it reduces the cognitive load in sophisticated problem solving. *Modelling by analogy* (so- called *analogical modelling*) is a process thereby analogical reasoning is adopted for model construction. In other words, a model is constructed for a problem based on the similarity of the problem to previously solved problems. As illustrated in Figure 6, solution B is developed based on a known solution for problem A and the similarity between problem A and B. A combination of problem A and solution A serves as a source for analogical reasoning, whereas problem B and solution B are the target.

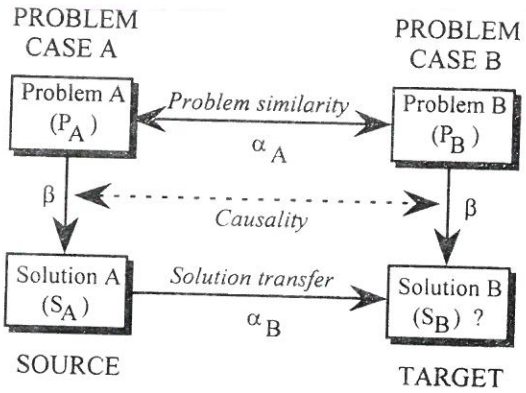

Figure 6. Analogical Paradigm

The researches in artificial intelligence about analogy concern its use for problem- solving and machine learning. In this case, the analogical reasoning is used to build a solution as Figure 6 shows. The analogical reasoning uses dependent and similarity relations between a target object, partially known, and a reference source object in order to transpose knowledge from an application universe to another, according to a point of view.

An analogical reasoning in which source and target belong to the same semantic domain, source being an example for the target, is named case-based reasoning. A universe represents the set of knowledge in relation with a given domain. For example, the continuous control universe contains explicit knowledge such as incident case descriptions and mechanisms in the universe theory. They allow to infer new knowledge.

There is dependent relation β between the structured elements of description P_A (Problem A) and S_A (Solution A) of the continuous control universe, if and only if P_A is necessary in the deduction of S_A in the knowledge domain. By definition, we define as dependent description all sets (P_A, β, S_A) where P_A and S_A are in dependent relation β. In the analogical reasoning, depending relations exist within the source and must be transferred to the target by similarity relations. Let (P_A, β, S_A) and (P_B, β, S_B) two dependent descriptions from the same universe, their similarity is defined by: (P_A, β, S_A) α (P_B, β, S_B). This similarity may be projected on each argument: $P_A \alpha_A P_B$ and $S_A \alpha_B S_B$ in writing α_A and α_B the restrictions of the α relation of each same nature component of both dependent descriptions. We then have the general process of Figure 6 which represents the analogical paradigm [18]. We have introduced different notions allowing to formalize the static aspect of an analogy (description of an analogical situation). From these notions, the analogical reasoning, that is to say the dynamic aspect (analogical process), concerns:

- research of dependent relations;
- define similarities;
- global strategy to control the analogical inference;
- transfer of the explained knowledge from the source (Solution A) to the target (Solution B).

To conclude on these notions which are the base of the analogy, we can say that the solution of an incident is built on the basis of a real similarity with an incident solved in the past. As Figure 6 shows, the development of solution B is based on the knowledge of a solution A to problem A and on the similarity between problems A and B. The arrangement {Problem A, Solution A} is used as a source for the analogical reasoning of which the couple {Problem B, Solution B} is the target. Within the framework, the case-based reasoning can be defined as an analogical reasoning for which partial knowledge of the target is known and, also, a similarity. In our situation, the incident definition has two parts: the first one is known (the alarm description and its environment) and the other to infer (the incident solving). We now define the different stages of the case-based reasoning and show for each one the pieces of knowledge used.

5.2 Case-based Reasoning

CBR is a particular form of analogical reasoning, as it is defined in artificial intelligence [18], in which the source case (the solved problem) and the target case (description of the problem to solve) belong to the same semantic domain - network traffic management here; source is then an example for the target. The CBR principle consists in solving a new problem, for which the solution is unknown, to transfer solution of previously solved problem. Reasoning from cases is a current activity. If we watch the way people around us solve problems, we are likely to observe CBR in constant use [15]. A network manager, when he meets a new incident, will first use the "collective memory" of past incidents (e.g. in a logbook which records all incidents on the network) to construct and justify his decision. To retrieve and present "good" cases are the main aided functions that a CBR based system can offer.

Development of such a system requires a case base (i.e. a base of actual previous experiences), a mechanism to retrieve similar cases, an adaptation and justification mechanism to retrieve the solution in order to solve the current problem. Thus, at the highest level of generality, a CBR cycle may be described by the following four processes:

- *retrieve* the most similar case or cases;
- *reuse* the information and knowledge in that case to solve problem;
- *revise* the proposed solution;
- *retain* the parts of this experience likely to be useful for future problem- solving.

A new problem is solved by retrieving one or more previously experienced cases, reusing the case in one way or another, revising the solution based on reusing a previous case and retaining the new experience by incorporating it into the existing case base. Each of the four processes involves a number of more specific steps. In Figure 7, this cycle is illustrated [19].

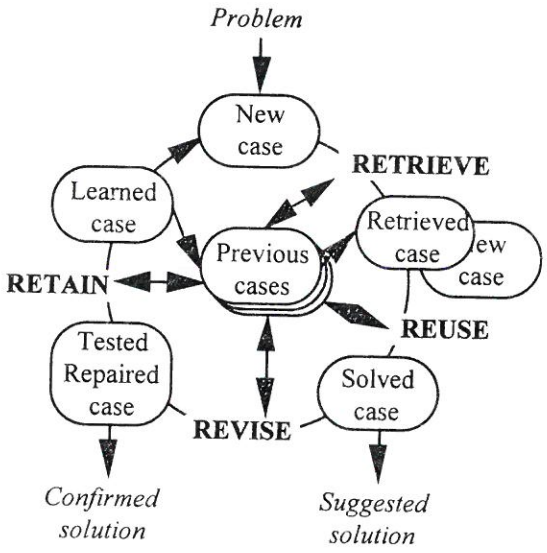

Figure 7. CBR Flowchart

An initial description of a *problem* to solve (top of Figure 7) defines a *new case*. This new case is used to *retrieve* a case from the collection of *previous cases*. The *retrieved case* is combined with the new case - through *reuse* - into a *solved case* (i.e. a proposed solution to the initial problem). Through the *revise* process this solution is tested for success (e.g. by being applied to the real world environment or evaluated by a teacher), and redressed if failed. During *retain*, useful experience is retained for future reuse, and the case base is updated by a new *learned case*, or by modification of some existing cases.

CBR can signify adapting old solutions to meet new demands, using old cases to explain new situations, using old cases to criticize new solutions, or reasoning from precedents to interpret a new situation (much like lawyers do) or create an equitable solution to a new problem (much like labour mediators do) [15].

These different processings are joined around the case memory which is the system's heart. In our case the memory contains descriptions of incident cases. An incident case is composed of data and knowledge about incident declarations (i.e. symptoms: place, alarms, threshold values, affected material), and incident solutions (i.e. explanations about the cause of symptoms, advised actions, consequences) and, possibly, a description of the process to converge on the solution (i.e. the strategy) and of the effects produced on the network by the solution.

5.3 Diagnosis Support System

Design of the CBR assistance system has been influenced by the requirements of the domain that were discussed during presentation of the first stage of our methodological approach. This paragraph presents an overview of this system with a detailed presentation of its major components. A functional description of the CBR assistance system is shown in Figure 8.

The CBR process of our system is common to many other CBR systems [20]. The processing starts with the *target incident case editor*, which parses the input data and forms problem

representations, each of which is called a *problem statement*. This problem statement is passed to the *indexer/matcher*, which is responsible for retrieving those stored cases from the *case-based memory* that are most relevant to the current problem statement. The *selector's* function is to choose the "most on point" successful case out of the set of cases retrieved by the indexer/matcher. The *modifier* (based on adaptation rules) must then apply the experience recorded in the selected case to the current situation, modifying it if necessary.

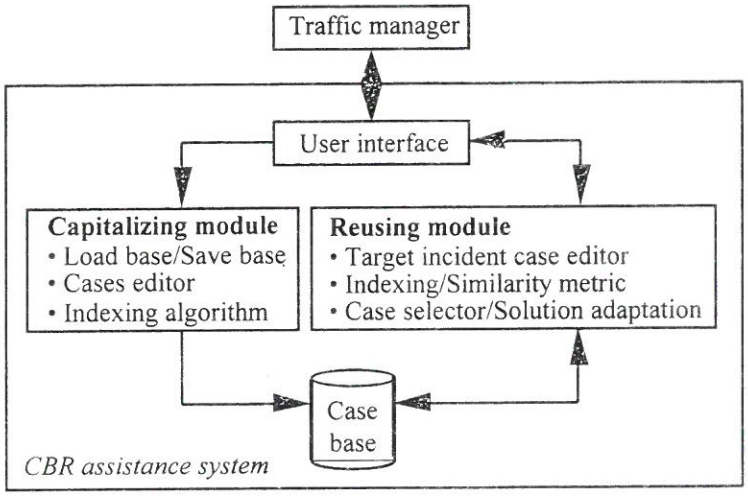

Figure 8. Functional Description of the CBR Support System

5.3.1 Incident Case Representation

The incident cases, of the case-based memory, are defined by means of an object oriented knowledge representation language . Thus, a case is represented within a class, as an object (i.e. an instance of this class). The different classes of the case base are categorized in an inheritance hierarchy [21]. A class may be shown as a data structure with its own properties. The object class *incident* has for sub-classes *switch incident* and *circuit group incident*. The most general classes appear at the top of the class hierarchy. For example, the class *circuit group incident* has the characteristics of its upper-class *incident* and its own attributes as *number of circuits in use*.

Main difficulty to design the case base is in the definition of the incident case. Domain level of the expertise model provides knowledge to answer this problem. The different attributes correspond to the situation when a problem was first detected in the network, to causes and consequences of that problem, to actions that should be taken to control the problem and some explanations about diagnosis and actions.

5.3.2 Case Base Categorization and Classification

The indexer/matcher is the part of the system that retrieves cases that are similar to the current problem statement. Given a situation and a starting point, the indexer/matcher produces a list of stored cases sorted out according to a preliminary hint of relevance. There is a number of characteristics of the domain which constrains the choice of indexing structure and technique. The domain supports a set of attributes which is always significant in discriminating among situations. We use a hierarchical discrimination tree; this discrimination net represents a conceptual map

of the situations which the system may face with [21]. According to the discrimination technique, the indexer/matcher searches for a good characterization of the current situation by a best first spreading activation process. During this process, "weight" is moved from one index (an index represents a characterization of a situation) to another. This weight gives confidence that the index characterizes the current situation. This search starts at one of a set of designated "root" indices. As candidate cases are retrieved by the match completely on the crucial "index" features in order to be retrieved by the indexer/matcher.

The task of the selector, then, is to perform a fine-grained match between the current problem statement and the candidate cases and to determine the most relevant of the candidate cases [21].

At present , domain knowledge modelling part is over. CommonKADS expertise model has been specified and validated by the managers. In the design stage, we have developed a library of C++ classes and functions which implement CBR stages and knowledge of the expertise model. Now, we develop the user interface of the CBR assistance system, with X-Designer, and we analyze results obtained, in laboratory, from a large base of real incident cases. In a next stage, we will compare these results with some re-using experiences realized, with the managers, in the control room. This validating phase allows to precise the case base indexing and matching, and the similarity measure used by the system to retrieve the most similar cases.

6. Conclusion

The application of artificial intelligence techniques to network traffic management has proved to be interesting as far as the monitoring function is concerned. The integration of these functions (monitoring and control) should provide the network managers with adequate tools in a more and more complex environment.

The methodological approach presented in this paper aims at offering to a user services of a collective memory in which he will find components necessary for his decision -making. Our framework has two main parts, which define the originality and complementarity of this "user-centred" approach:

- the expertise model developed during the first KA stage of the approach, including a process model of the CBR cycle and a task-method structure for CBR;

- development of the expertise model in the decision support system.

The two models are complementary and represent two views on CBR. The first is a dynamic model that identifies the main sub-processes of a CBR cycle, their interdependencies and products (see Figure 7). The second is a task-oriented view, where a task decomposition and related problem - solving methods are described.

Main originality resulting from this is the integration of a modelling process in the bottom-up approach of the example-based learning. These aspects place the approach in a rebuilding process of the reasoning from real knowledge of the managers. This approach allows to develop co-operative systems, which are more effective, more adapted to the user's needs and thus better accepted. Furthermore, the CBR approach allows to propose solutions to problems in a very short time. The fact of using a previously solved problem lets us not have to re-execute treatments and inferences to obtain a solution. Other (hybrid) approaches combine CBR with model-based reasoning (MBR) or/and rule-based reasoning (RBR), to solve diagnosis problems [22, 23].

The CBR support system design that has been described above specifically addresses problems of continuous control applications such as traffic management. The criteria imposed by a continuous control application are responsible for unique aspects of their design in the system. For example, as discussed in this paper, to limit the spread of a network problem and to provide a satisfactory response to meeting a deadline, the indexer/matcher performs a best-first spreading-activation search. The selector in this system is relatively sophisticated, devoting resources to increasing its chance of choosing the most relevant case at its first attempt. The design approach of this CBR support system, is then

motivated by the requirements of continuous control applications; the system thereby extends the applicability of CBR to a wide range of practical problems. As the system is implemented and evaluated, it will demonstrate the benefits of CBR paradigm, and test the generality of the approach for other related applications.

In conclusion, we can say that our methodological approach is intended to act as a support for the creation of a cognitive CBR support system usable in different ways:

- assistance to solve on line control problems;

- capitalize to exploit and preserve the collective memory of the control site (i.e. place in common and at managers disposal an expertise originally spread in multiple sources: managers, control system, files, papers, etc.);

- as a tutorial form intended for the new (or not) managers training.

We think that such a system can represent one key to solving a great challenge to a company: the management of its corporate knowledge and thus the improvement of its performances.

REFERENCES

1. CAULIER, P. and HOURIEZ, B., **Knowledge Acquisition and Case-Based Reasoning for a Case-Based Incident Diagnosis and Solving System**, Proc. IFAC Conference on Integrated Systems Engineering PERGAMON, Baden-Baden, Germany, September 27-29, 1994.

2. HAMMOND, K.J., **Case-Based Planning: An Integrated Theory of Planning, Learning and Memory**, Ph.D Thesis, Yale University, USA, 1986.

3. SLADE, S., **Case-Based Reasoning: A Research Paradigm**, A.I. MAGAZINE 12, 1, 1991.

4. SCHANK, R.C., **Dynamic Memory: A Theory of Learning in Computers and People**, Cambridge University Press, Cambridge, 1982.

5. WIELINGA, B., VAN DE VELDE, W., SCHREIBER, G. and AKKERMANS, H., **Towards A Unification of Knowledge Modelling Approaches**, KADS-II Report, KADS-II/T1.1b/UVA/RR/004/3.0, University of Amsterdam, The Netherlands, February 1992.

6. GUSTAFSSON, M. and MENEZES, W., **An Overview of CommonKADS**, KADS-II Report, KADS-II/P3/WP/CP/005/1.0, Cap Programmator, Sweden, March 15, 1994.

7. VISSER, W., **Acquisition de Connaissances : l'Approche Psychologie Cognitive Illustrée par le Recueil d'Expertise en Conception**, Actes des 2èmes Journées Acquisition de Connaissances du PRC-GDR-IA du CNRS, Lannion, France, 27 April 1990.

8. CAULIER, P. and HOURIEZ, B., **Knowledge Acquisition and Case-Based Reasoning: An Integrated Cognitive Approach**, Proc. 13th European Annual Conference on Human Decision Making and Manual Control, Espoo, Finland, June 13-14, 1994.

9. AAMODT, A., **A Knowledge-Intensive, Integrated Approach to Problem Solving and Sustained Learning**, Ph.D Thesis, University of Trondheim, Norway, May 1991.

10. WIELINGA, B. and SCHREIBER, G., **KADS: A Modelling Approach to Knowledge Engineering**, KNOWLEDGE ACQUISITION, 4, 1, 1992.

11. LAUBLET, P., **Objets et Intelligence Artificielle : des Langages aux Méthodes**, Actes de la Journée Méthodes Objets et Intelligence Artificielle, 14èmes Journées Internationales d'Avignon, IA'94, Paris, France, June 2, 1994.

12. DE HOOG, R., MARTIL, R., WIELINGA, B., TAYLOR, R., BRIGHT, C. and VAN DE VELDE, W., **The CommonKADS Model Set**, KADS-II Report,

KADSII/M1/DM1.1b/UvA/018/6.0/FINAL, University of Amsterdam, The Netherlands, June 6, 1994.

13. ALBERT, P., **KADS-Tool : un Atelier pour la Méthode CommonKADS**, GÉNIE LOGICIEL ET SYSTEMES EXPERTS 31, June 1993.

14. WIELINGA, B. and BREUKER, J., **Model of Expertise**, Proc. 7th European Conference on Artificial Intelligence , ELSEVIER SCIENCE, 1986.

15. KOLODNER, J.L., **Improving Human Decision Making Through Case-Based Decision Aiding**, A.I. MAGAZINE, 12, 2, 1991.

16. CAULIER, P. and HOURIEZ, B., **Apports de la Modélisation des Connaissances et du Raisonnement à Partir de Cas à la Capitalisation et la Réutilisation de Connaissances**, Actes des Journées Acquisition, Validation et Apprentissage du PRC-GDR-IA du CNRS, Grenoble, France, 5-7 April 1995.

17. CAULIER, P. and HOURIEZ, B., **Incident Case Reusing in Network Traffic Management Activities**, Proc. 5th European Conference on Cognitive Science Approaches to Process Control, Espoo, Finland, August 30-September 1, 1995.

18. CHOURAQUI, E., **Le Raisonnement Analogique: sa Problématique, ses Applications**, Actes des Journées Nationales sur l'Intelligence Artificielle, CEPADUES, PRC-GDR-IA du CNRS, Toulouse, France, 1986.

19. AAMODT, A. and PLAZA, E., **CBR: Foundational Issues, Methodological Variations and System Approaches**, A.I. COMMUNICATIONS, 7, 1, March 1994.

20. CAULIER, P. and HOURIEZ, B., **A Case-Based Reasoning Assistance System in Telecommunication Network Management**, Proc. 3rd German Workshop on Case-Based Reasoning, Kaiserslautern, Germany, February 28-March 1, 1995.

21. CAULIER, P. and HOURIEZ, B., **A Case-Based Reasoning Approach in Network Traffic Control**, Proc. IEEE International Conference on Systems, Man and Cybernetics, Vancouver, British Columbia, Canada, October 22-25, 1995.

22. KOTON, P.A., **Combining Causal-Models and Case-Based Reasoning**, Second Generation Expert Systems, SPRINGER-VERLAG, 1993.

23. MACCHION, D., **Un Systčme à Base de Connaissances Hybride pour l'Assistance et l'Apprentissage au Diagnostic Technique**, Actes du 2e Séminaire Français sur le Raisonnement à Partir de Cas, LAFORIA 93/42, Université de Paris VI, France, 1993.