Modeling and Using Context: Past, Present and Future

P. BRÉZILLON

LIP6, Case 169, University Paris 6, 8 rue du Capitaine Scott, 75015 Paris, France
E-mail: Patrick.Brezillon@lip6.fr

Abstract:

Over the ten years a community on context has emerged. Brézillon (1999) proposed a survey of the literature about context in artificial intelligence. There is a now series of conference on context, a web site and a mailing list. The number of web pages with the word "context" has been multiply by four in the last five years. Being among the instigators of the use of context in real-world applications, we present in this paper the evolution of our thoughts over the last years and the result that is obtained as a representation formalism based on contextual graphs and used in a real-world application called SART. We also point out some similarities between contextual graphs and some models developed in other domains. As a kind of compilation, this paper sums up and gives some comments on papers that we published along these last years.

1. INTRODUCTION

Context plays an important role since a long time in such domains where reasoning intervenes as in understanding, interpretation, diagnosis, etc. The reason is that these activities of reasoning rely heavily on a background or experience that is generally not make explicit and gives a contextual dimension to knowledge. Indeed, everybody uses context in his daily life as Mr. Jourdain used prose for speaking without being aware of it (Molière, 1670). However, there is a lack of a clear definition of this word.

In an expert system-like representation, knowledge is gathered as production rules. These rules are pieces of knowledge of the form "if *conditions* then *conclusions*." They are recorded in large rule bases difficult to update. The rules are structured pieces of knowledge, which are easily understood by the domain experts. However, the lack of structure of the rule-base impedes the comprehension (even for the experts of the domain) and the maintenance of the knowledge. Some works have been done on rule-bases structuring, namely by splitting of the rule bases into several rule packets, each containing a subset of the rules applied to solve a specific sub-problem (Brézillon, 1990). Clancey (1983, 1995) proposed to add screening clauses to the condition part of the rules so that they are activated only in some kind of context. This is burdensome because the designers must anticipate all the possible contexts to define the preconditions of the rules.

The decision tree approach (Raïffa, 1968) tries to represent the decision step by step. This is obtained by the presence of two types of nodes: the event nodes and the decision nodes. At an event node, paths are separated according to an event on which the decision maker has no influence. On a decision node, the person makes a choice. This approach might be a way to structure rule bases. For each new element analyzed in the conditions, a new event node is created. For each new value of an existing contextual element, a new branch is created, and so on. Rule after rule, a tree is constructed. The leaves give the conclusion rule. The main problem with this structure is the combinatorial explosion. The number of leaves increase exponentially with the deep of the tree. The addition of a contextual element may easily double the size of the tree.

Bayesian networks are composed of nodes, representing random variables, and links representing the causal relations between the different random variables (Jensen, 1996; Pearl, 1988). Each node is associated with a table giving the distribution of probabilities of the corresponding random variable according to the values of the random variables of which it depends on. The influence diagrams introduce decision nodes into Bayesian networks (Oliver and Smith, 1990; Neapolitan, 1990). These networks are possible solutions to limit the combinatorial and information explosions in decision trees with probabilities. Moreover, both approaches, Bayesian networks and influence diagrams, necessitate, to be handled, some information about the probabilistic dependence between the different random variables and are anyway limited to a reasonable number of variables.

Another network representation relies on simple Petri-net or colored Petri-net (CPN). CPN are very useful to represent the dynamics of a situation and simulate a process. Humphreys and Berkeley (1992) give such an example for simulating an organizational process.

Case-Based Reasoning (CBR) is a kind of analogy reasoning. To solve a current issue, one selects the most similar problem in a problem base and one adapts the solution to the problem at hand. Instead of adapting prior solutions, an interesting alternative consists of storing and reusing trace of how those solutions were derived (Leake, 1996). The main advantage is its great power of generalization and its maintenance. However, it fails to provide explanations on the obtained solution.

These computing approaches are well-known paradigms to introduce human-like reasoning in automatic systems or support systems. Psychology and ergonomics are also interested in this activity representation. In Artificial Intelligence, the lack of explicit representation of context is one of the reasons of the failures of many Knowledge-Based Systems (KBSs). For example, Vanwelkenhuysen and Mizoguchi (1995) show that if users are in different classes (i.e. different working context), the system should tailor its behavior to the different habits of the classes of users. Testers and engineers working on a same device (digital processor boards in a telecommunications production plant) solve the same problem differently (e.g., oscilloscope versus logic state analyzer), each way being effective for routine problems in their workplace but inadequate for the other's.

Studies of KBS use in real-world applications show four main failures (Brézillon and Pomerol, 1996a and 1996b):

(1) Exclusion of the user from the problem solving. KBSs were assimilated to oracles and users as novices. However, unexpected problems to solve are the norm rather than the exception. KBSs cannot solve such unexpected problems when users, with their practical experience, are not given the opportunity. What is required is a cooperation between the user and the system and the consideration of the context in which a problem has to be solved.

(2) KBSs do not use correctly their knowledge. Knowledge, which is acquired from human experts, has a high contextual component that is generally not acquired with the knowledge because knowledge engineers asked what experts' solution is, not how they reach it.

(3) KBSs cannot initially have all the needed knowledge. Any KBS has limited resources for problem solving and limited influence: One can never anticipate or "design away" all the misunderstandings and problems that might arise during the use of such systems (Fischer, 1990). This implies that knowledge must be acquired incrementally when needed, i.e., in a given context of use.

(4) KBSs cannot generate relevant explanations for users because they do not know the context in which the user asks a question. The unique way to generate a relevant explanation is by the explanation generation by the KBS and the user jointly (Karsenty and Brézillon, 1995).

This leads to include our thinking within the framework of Intelligent Assistant Systems, IASs, in which context plays a central role (Boy, 1991; Brézillon and Cases, 1995; Brézillon *et al.*, 2000): There is an asymmetry between the respective roles of the human being and the machine. The user makes the final decision and the system provides intelligent support by proposing satisfying alternatives and arguing over them, recalling points in the previous interaction, having a model of the user and a model of the process on which the user works, and within which he thinks. This asymmetry arises from the different responsibilities of the system and the user, and from the fact that it is only the latter who, besides responsibility, has intention and commitment (see, for example, Langley *et al.*, 1995 and the references therein). To emphasize the asymmetry between the leading role of the operator and the subsidiary role of the systems, we call such systems Intelligent Assistant Systems.

Hereafter, we present how context is considered in Artificial Intelligence (section 2) and decision-making area (section 3). Section 4 describes briefly the SART application for which the contextual graph approach has been developed. Section 5 gives our understanding of the context and section 6 how we have represented knowledge and reasoning in context. Section 7 compares contextual graphs, which have been developed in Artificial intelligence (AI), and models in other domains.

2. CONTEXT IN AI

2.a Introduction

AI moved from the view of KBSs replacing people towards two complementary approaches, namely the Situated Cognition approach (Clancey, 1991a, 1991b) and the Joint Cognitive systems (JCSs) approach (Woods *et al.*, 1990). Situated cognition insists on the need in problem solving to account for the environment (through our interaction with it) and the on-going context in organizing behavior. All processes of behaving, as problem solving, are generated on the spot, not by mechanical application of scripts or rules previously stored in the brain. JCS approach puts the emphasis on the competence complementarity of users and systems that leads to a symbiosis. It also emphasizes, more or less explicitly, the role of context for such a symbiosis.

Researchers in AI have used context since a long time as de Kleer (1987) in ATMS, McDermott (1982) in X1/RCON, Laird and col. (1987) in SOAR, Hendrix (1975) for the partition of semantic networks, Guha (1991) in the CYC project.

In this section, we give an overview on context in rule-based representation, and with incremental knowledge acquisition. We discuss after how context is considered in AI, the dynamic dimension of context, and conclude with some answers that emerged.

2.b Rule-based representation

In a rule-based representation, context may be expressed on the basis of either the knowledge structures (if explicitly represented) or the particularities of the chosen representation formalism. Knowledge structures are rule packets represented either at the level of the rules or at the level of the knowledge base. The former is managed by screening clauses, which are controlled by special rules (meta-rules) (Clancey, 1983). The latter organizes the knowledge base in a set of distinct small knowledge bases managed either directly by rules that call rule packets in the THEN part (Brézillon, 1990) or by interaction among rule packets for exchanging information in a multi-agent spirit.

For a representation at the rule level, a well-known example of screening clause is in the following rule in MYCIN (Clancey, 1983):

IF
1. The infection, which requires therapy, is meningitis,
2. Only circumstantial evidence is available for this case,
3. The type of meningitis is bacterial,
4. *The age of the patient is greater than 17 years old*, and
5. The patient is an alcoholic,

THEN
There is evidence that the organisms which might be causing the infection, are diplococcus-pneumoniae (.3) or e.coli (.2)

Such a rule is composed of different types of knowledge (strategic knowledge, causal knowledge, etc.). The clause 4 had been added to control the interaction with the user. The clause acts as a screening clause and implies that the use context of the rule is limited to adults. Such knowledge does not intervene directly in the problem solving, just constrains it.

Rule packets can also be represented explicitly in the knowledge base. The diagnostic expert system SEPT dealt with pieces of equipment as circuit breakers and protective relays (Brézillon, 1990). Checking the internal behavior of a circuit breaker implies an expertise (described in a rule packet) that is independent of the expertise on the internal behavior of a protective relay (described in another rule packet). Thus, the reasoning is local and needs not to tackle the overall expertise.

A rule of the SEPT expert system is:

```
! check_cb
"Checking a circuit breaker of the protection system $name"
> if failure freeze check_pw, check_teac
IF          equipment_piece (cos) := (cb) ,
        nature (cb)  := circuit_breaker .
THEN
        call the rule packet 'Circuit-Breaker_Diagnosis(cos, cb)' .
```

The rule entitled *check_cb* (written here in pseudo natural language) is used to trigger the diagnosis of a circuit breaker *cb* in a cut-off system *cos* ($name, *cb* and *cos* are variables). The rule belongs to a rule packet that checks all the equipment pieces in a cut-off system. The rule packet represents the diagnostic expertise at the level of the cut-off system, and local diagnostic expertise on pieces of equipment are in other rule packets as *Circuit-Breaker_Diagnosis* in the THEN part of the rule. Firing the rule *check_cb*, the inference engine will enter the rule packet *Circuit-Breaker_Diagnosis* with the instances fixed for the variables *cos* and *cb* in the IF part, when the rule packet may be applied to all the circuit breakers in the substation (around 20 circuit breakers in an EHV substation).

In such a rule, context is expressed at two levels:
(1) by a specialization of the circuit-breaker expertise for the given instances of the variables; and
(2) by a management of the expertise at the cut-off system level by the meta-knowledge *if failure, freeze choice-pw, choice_teac* that says that if a failure is found on the circuit breaker it is not necessary to check the equipment pieces *pw* and *teac* of the cut-off system.

2.c Incremental knowledge acquisition

An example of IAS design (Brézillon and Cases, 1995) convinced us of the prominent importance of two issues in cooperation, namely explanation generation and incremental knowledge acquisition. Explanation is an effective way of convincing others (Karsenty, 1996). Operators really do want to follow the system reasoning to feel comfortable with its recommendations. This is a matter of confidence. (It is not surprising that the question of confidence in system design is increasingly deserving attention, as stated by Moray, 1993.) The necessity of incremental knowledge acquisition is obvious to the extent that in real cases, the burden of data introduction is very heavy. Another less obvious reason is that automatic knowledge acquisition is the best way to acquire the context of use and avoid misuse of the system, as we will show.

For developing IAS, three aspects are particularly important:

(1) Explanations that must be considered as an intrinsic part of any cooperative problem solving (Karsenty and Brézillon, 1995). This implies that the system and the user must provide and understand explanations in order to cooperate, and cooperate to co-build explanations. Here, context is essential to the communication.

(2) Incremental knowledge acquisition by the system from users to relieve them after with the same type of problems. Thus knowledge is acquired during the problem solving with its context of use. In some way, this is a kind of explanation from the user to the system.

(3) Context of the user-system cooperation that must be made explicit to provide relevant explanation and acquire incrementally knowledge (Brézillon and Abu-Hakima, 1995).

Thus, making context explicit, acquiring knowledge incrementally and explaining in context must be considered as intrinsic aspects of any problem solving in which the user has a crucial role to play. An efficient cooperation between a human and a machine implies the consideration of the three related aspects: explanation, incremental knowledge acquisition and context.

Knowledge acquisition is a difficult and time-consuming task. In our opinion, the knowledge engineer associates a context to the couple (problem, solution) that may be different from those in

expert's mind. Encoding the knowledge relative to the task at hand leads to a proceduralization of the context because knowledge is (or should be) encoded into the system with a part of the contextual knowledge. Henninger (1992) noted that "you won't know what is really needed until you're in the design process."

Incremental knowledge acquisition plays an important role in two situations:

- When the knowledge is missing in the current context, the user adds new knowledge through explanations. Here, explanations enable the contextual knowledge to be proceduralized.
- A chunk of knowledge may be known by the system but incorrectly used, *i.e.* a link to the current context is missing. This link could be added in the form of a consequence of the current context. Here, incremental knowledge acquisition will focus on the refinement of the proceduralized context (as introduced hereafter), and explanation will support that refinement. Thus, gathering and using knowledge in the context of use greatly simplifies knowledge acquisition because the knowledge provided by experts is always in a specific context and is essentially a justification of the expert's judgment in that context.

Several approaches have been proposed to acquire knowledge in context. We give here two examples. Compton and Jansen (1988) attempt to capture the context by entering the expert's new rule only when necessary. More precisely, when a rule fails, the expert is requested to give some extra knowledge or rule. This new rule is directly related to the rule which failed and this latter is recalled to the expert because it gives the context for writing the new rule.

Gruber (1991) considers a justification-based knowledge acquisition. The machine provides the computational medium, including the knowledge representation and the context of use, such that everything that is acquired from the user can be assimilated into the computational model. The machine that can provide thus a frame to fill, allowing some kind of syntactic checking and ensures that all required parameters are given directs the dialogue.

The main idea of all these approaches is that experts provide their knowledge in a specific context and that knowledge can only be relied upon with in this context. Thus, knowledge cannot be generalized when it is acquired, and it is fundamental to record the context in which the knowledge is acquired and can be used. Nevertheless, acquiring knowledge in context is still a challenge.

2.d The two sides of context

The notion of context is dependent in its interpretation on a cognitive science versus an engineering (or system building) point of view, the practice viewpoint versus the theory one (Brézillon and Abu-Hakima, 1995). The *cognitive science view* is that context is used to model interactions and situations in a world of infinite breadth, and human dimension is the key for extracting a model. The *engineering view* is that context is useful in representing and reasoning about a restricted state space within which a problem can be solved. The identification of these two viewpoints permits to understand the contrasted views found in the literature (Brézillon and Abu-Hakima, 1995; Brézillon, 1994, 1997).

All works in knowledge representation consider a set of discrete contexts, and the effort relies on the crossing of contexts. Creating a context from existing contexts, as proposed by McCarthy, it is possible to establish a hierarchy of contexts where a formula relating two contexts involving contextual assumptions is itself in a context. The interest of a context hierarchy is that, working on an object in one context, something may be derived about that object in another context. The two contexts may use different vocabularies, and the treatment of the object may be easier in one context than another. Conversely, in Cognitive Science, researchers without reject the possibility of discrete contexts, consider that the context of interest is the context of the interaction because it is the unique context that may be perceived. The interaction context evolves continuously according to knowledge pieces introduced by an agent.

According to the engineering viewpoint, the context is static and considered at the level of the knowledge representation. As a consequence, there is a static view on contexts and the interest is on context management. Static contextual knowledge is attached to the domain knowledge, and thus may be described in knowledge bases. The static part of the context is what may be coded at the design time. Along its dynamical aspect, part of the problem is linked to the changing nature of context in time, by elaboration and shift (Clancey, 1991b). If it is (relatively) easy to represent

the static aspect of context, the dynamic aspects of context must be considered during its use, say a problem solving. Thus, one must account for both the *static aspect* (knowledge that remains constant throughout the interaction) and the *dynamic aspect* (knowledge that changes throughout interaction) of context.

Context is considered as a shared knowledge space that is explored and exploited by participants in the interaction. Contextual knowledge acts as a filter that defines, at a given time, what knowledge pieces must be taken into account (explicit knowledge) from those that are not necessary or already shared (implicit knowledge). A context is a structure, a frame of reference that permits to do not say all the things in a story. For example, "At his birthday's party, Paul blew up the candles." It is not said here there was a birthday cake because it is clear for everybody. Such an implicit piece of knowledge is supposed to be a part of our social inheritance.

Implicit knowledge is often assimilated to shared knowledge. For example, there is a French movie call 'Le Chat' (the cat) presenting the life a husband with his wife living together since 40 years. Knowing very well the other, they had highly limited their communication. For instance, with a light movement of the chin toward the cupboard, the husband means "Can you please darling give me the salt that is in the cupboard." With a computer system, however there is a compromise to find between the need to store a large number of information pieces and a tailored presentation of the answer to the user's question, i.e. to distinguish between contextual knowledge and the knowledge that stays external to the answer. We will see that often such a distinction can be made only a posteriori.

2.e The dynamical dimension of context

Beyond the two contrasted views on it, context possesses a time dimension that poses some problems in modeling. This temporal dimension of context arises from the interactions among agents, as opposed to context as a fixed concept relative to a particular problem or application domain (Maskery and Meads, 1992). That is, without interacting agents, there would be no context. In communication, the context is considered as the history of all that occurred over a period of time, the overall state of knowledge of the participating agents at a given moment, and the small set of things they are expecting at that particular moment. Context appears as a shared knowledge space. However, each entity involved in an interaction has its own context, which may or may not be consistent with some parts of the contexts of the others. One important point underlined by Mittal and Paris (1993) is that communication, including explanations, and context interacts with each other: the context of the situation triggers some actions, and this in turn modifies the context of the situation. These authors bring together different notions of context such as the elements of a global picture that might be taken into account by an explanation module, depending on the needs of the application.

The dynamics of the process are very important in diagnosis as well as control of complex systems (Hoc, 1996 ; Hoc, and Amalberti, 1995). We think that it is not only important to understand the dynamics of planning and action but also the dynamics of knowledge management. This is a twofold phenomenon that consists of focusing on some stimuli and, on moving contextual information from back-stage to front-stage. Interaction between agents appears to be a good way for moving a contextual knowledge into and out of the front-stage knowledge that we call proceduralized context hereafter.

At the interaction level, making context explicit enables knowledge to be shared with others. For example, in the example of "Sick traveler on a train" in the SART application (see section 4), the change from the step "Answer the alarm signal" to "Stop at the next station" was surprising to us (as knowledge engineers). The triggering of the alarm signal implied for many years an immediate stop of the train, even in a tunnel because an alarm signal needed immediate attention. To explain the skipping of the action “Answer the alarm signal,” the operator said that, based on company experience, they have decided to stop only at the next station for several reasons. Some reasons are easy to understand (*e.g.*, rescue is easier in a station than in a tunnel). Other reasons needed additional knowledge (*e.g.*, drivers follow procedures because a traveler may be sensible to claustrophobia in a tunnel). Once we were able to gather all these reasons into a coherent proceduralized context, we understood the change. Note that other reasons are left implicit (*e.g.*, if the stop will last a long time in a station, other travelers may leave the train to go by bus, reducing the number of travelers waiting on the platform).

2.f Some answers concerning context

We cannot speak of context out of its context. Context is something surroundings an item (e.g., the task at hand or the interaction) and giving meaning to this item. It cannot be considered out of its use. Giving meaning to an item, context acts then more on the relationships between items than on items themselves, modifies their extension and surface. Contextual knowledge concerns the future and consequences of actions, and intervenes in decision maker's look ahead reasoning. Context is considered as a shared knowledge space that is explored and exploited by participants in the interaction. Such shared knowledge includes elements from the domain (e.g., instantiated objects and constraints), the users (e.g., their goals), their environment (e.g., organizational knowledge), their interaction with a system (e.g., transaction history).

There are different types of context with respect to what we consider (knowledge, reasoning, interaction, and each agent in its socio-organizational environment) and in which domain we are. All these contexts are interdependent; e.g. the interaction context is constrained by the knowledge context through, and says the model chosen to represent knowledge.

There are different representations of the context depending if context is considered either as knowledge or as a process of contextualization (Edmondson and Meech, 1993). Context as knowledge implies that we must distinguish between the knowledge effectively used at a given step of a problem solving (the proceduralized context) and the contextual knowledge (the knowledge constraining the proceduralized context at that step). Considering context as a process--a viewpoint close from the previous one--implies a distinction between knowledge, information and data (Aamodt and Nygard, 1995). Data become information through the contextualization process on the basis of the available knowledge at the time of the observation.

The lack of context representation is responsible in AI of the failures of KBSs, knowledge acquisition, machine learning, explanation generation, etc. Making context explicit will permit to develop powerful systems in complex tasks where the user plays a crucial role, generally having to take the final decision. One already speaks of contextual explanations, context-sensitive machine learning and of incremental knowledge acquisition for intelligent assistant systems.

The context permits to guide the focus of attention, i.e., the subset of common ground that is pertinent to the current task. The focus of attention is defined as the immediate context, whilst the common ground (or common context) was seen as being the mutual context that already exists between the agents (Maskery and Meads, 1992). The relationships between context and focus of attention are at the center of the concerns of the community on "context-aware applications".

3. CONTEXT IN DECISION MAKING[1]

3.a Particularities of DSSs

Few multicriteria DSSs tackle decision problems with a large number of alternatives and/or with uncertainty on the alternatives. Indeed, most of DSSs deals with a small number of alternatives, known with certainty. A typical example, very often used in textbooks, is the choice of a car. The decision maker has to choose a car within a set of about ten known and well described cars. However, the choice is much more difficult in many real situations. Let us think about the choice of introducing or not a new product, the choice of the features of a privatization law (Lévine and Pomerol, 1989), the design of a good railway timetable (Pomerol *et al*., 1995). Gilboa and Schmeidler (1995) mentioned the choice of a baby-sitter or Clinton's decision about Bosnia. What characterizes such situations is first the uncertainty about competitors and about the possible reactions of many other agents (for example unions in the case of a privatization law or other countries in the Bosnia case). This uncertainty is generally modeled as uncertainty about the moves of nature. From these examples, it appears that the choice at hand in many real situations is the choice between scenarios. These scenarios constitute sequences of decision maker's and nature moves. Unfortunately the examination of all possible actions and events becomes quickly computationally unmanageable.

---

[1] The work presented in this section is the result of a fruitful collaboration with Jean-Charles Pomerol, as shown by the number of our joint publications.

Another interesting behavior of decision makers is the fact that they try to postpone the action after the nature moves. In our application in subway control (see Section 4 for a presentation), operators undertake only actions they are obliged to take (Brézillon *et al.*, 1997). Meanwhile, they try to get more information to transform chance nodes into certainty and carry out the actions at the leaves of the decision tree after most of nature moves are known. Another interpretation of this phenomenon is that operators try to diagnose carefully in which branch of the tree they are. They use to gather the maximum quantity of contextual information to match the beginning of an already recorded branch with the perceived situation. This is a typical case of diagnosis of the current situation before action. Each recognized state triggers an action according to a kind of recorded table resulting from operator's experience (Pomerol, 1997). Then, the question of the diagnosis quality of the current state arises and leads to the question of the number of possible states according to context. Here, operators use contextual knowledge to circumscribe a subset of possible states because the whole number of possible states is potentially very important.

The second issue is the difficult problem of context modeling. In a previous paper (Pomerol and Brézillon, 1997) we stressed that multicriteria DSSs should pay more attention to context. Actually, there are many relationships between context and multicriteria decision. A first one is, as in any system, the question of the modeling: What are the limits of the systems? What parts of the context are relevant or not? Has every relevant contextual trait been involved in the system during the knowledge acquisition and how could the knowledge acquirer establish it? Moreover, did the modelers address the very important question for on-line systems: Is the system able to recognize its validity domain, i.e. its context of use and is the system able to know that a given situation is outside its contextual designed range?

The multicriteria framework is another aspect of the contextual dimension of multicriteria DSSs. It has been observed for a long time that depending on the context, criteria may be changed into constraints, constraints into objectives and, more surprising, objectives into alternatives. This last point refers to the so-called "value thinking" of Keeney (1992). We pointed out that context affects more relationships between concepts (alternatives, criteria, constraints) than the core of the concepts themselves.

3.b DSS and scenario management

In the framework of classical decision theory, the decision maker chooses an alternative, the nature move then occurs, and the decision maker accordingly gets his result. Nowadays, the decision maker does not choose a simple alternative but a policy according to different events that might occur at different moments. We call this **a scenario** as represented in Figure 1.

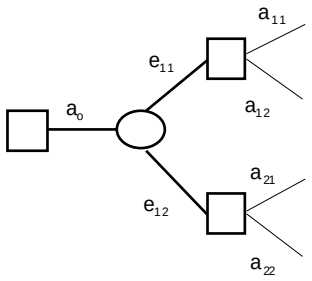

Figure 1: Scenario development.

In Figure 1, $\{a_0, a_{11}$ if e_{11}, a_{22} if e_{12}, etc.$\}$ is the beginning of a scenario when two events e_{11} and e_{12} are possible at the first chance node. Indeed, events and alternatives happen to be very numerous and the decision maker faces a combinatorial explosion. In order to manage this combinatorial explosion, the question is how do people practically reason? According to our examples, we suggest two ways of coping with the combinatorial explosion in scenario development.

The first way of thinking was observed in the system for evaluating timetable robustness (Pomerol *et al*., 1995) and in previous systems as for the choice of a privatization law (Lévine and Pomerol, 1989). The characteristics of these samples are a large number of events whereas the number of alternatives is relatively small (about five). The decision maker can theoretically define many sub-actions for each branch of an event node. However, he rather tries to define only one action that is

good enough for all the branches of a chance node. Following Roy (1997), we propose to call this notion of "good enough for every possible realization of the chance node" as robustness. A selected action following a chance node is robust as long as it gives "good enough" results against any event of the chance node. In this case, the decision maker can skip the chance node and the tree is flattened as shown in Figure 2 where the action a_1 is robust relative to the two realizations e_{11} et e_{12} of the chance node e_1.

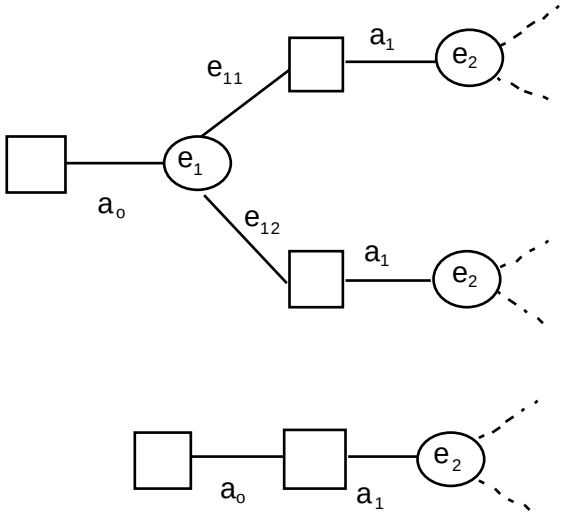

Figure 2: Transformation of a tree by using a robust action a_1.

The tree being flat, the decision maker can think in term of a sequence of actions. This is the reason we coined the term "fully expanded alternative" to describe this type of "alternative." We used the word "alternative" because after reduction the resulting scenario is made of a sequence of sub-actions. One may wonder how the decision maker reaches the conclusion that a sub-action is sufficiently robust. Again, from a theoretical point of view, the decision maker ought to use a folding back procedure and a pessimistic choice criterion such that max-min of minimax-regret. It now seems that a psychological process intervenes in many cases. During this process, the decision maker tries to persuade himself that event probabilities, which does not favor the "robust" alternative chosen, are very low. Montgomery (1983, 1987) pointed out this search for dominance, describing the different ways used to bolster attributes (events in our framework) that back an alternative while the bad attributes are more or less discarded.

In the case of timetable robustness, at each chance node the choice for a robust action was made by an expert system that, depending on various information pieces, gave what seems to be the better (the most robust) action. For the choice of a privatization law, as well as for the introduction of a new product, most of chance nodes actually model the behavior of other agents (unions, competitors, etc.). In systems previously mentioned, the other agents were modeled and simulated by knowledge-based systems. Then, according to the possible moves of other agents, an action was chosen, either for its robustness or because the expert system reaches the conclusion that the other agents should behave "almost certainly" in a given direction.

When an expert system carries out the reduction of the decision tree, its reasoning involves many contextual information pieces. This is the second way to reduce combinatorial complexity in decision trees. We think that the decision maker can replace subtrees by a simpler sequence of actions by using two types of reasoning: case-based reasoning and scripts. Case-based reasoning is now well-known in artificial intelligence (e.g. see Kolodner, 1993) but it has rarely been used for decision, probably due to a lack of theoretical framework (see Gilboa and Schmeidler, 1995; Tsatsoulis *et al.*, 1997; and Monnet *et al.*, 1998, for such formalization attempts).

A way to cope with the multiplicity of events is to try to put forward the action at the end of a sequence of chance nodes. Theoretically, this does not change the situation (see Figure 2). However, practically it does because people try to capture as much as information they can to transform chance nodes into certainty by determining which events occurred. The decision problem is thus transformed into a diagnosis problem. The question is now that of the diagnosis

quality of the current state and of the shrewdness of the representation relative to contextual information. This leads to a large number of possible current states and the diagnosis problem is not easier to handle than tree development. In many cases this also raises the question of context representation.

In many situations, people try to diagnose accurately the current state of the world beforehand to undertake an appropriate action. In terms of scenario generation and exploration, we propose to define this as **action postponing**. People try to delay the actions to get more information about the true state of nature. For example, a wise decision maker tries to have his competitors define their policy before he does. This usual view in management recall that often decision makers have the feeling that they can control risk (March and Shapira, 1987; Kahneman and Lovallo, 1993). This is a usual way of reasoning for the subway operators in incidental situation.

3.c DSS and partial planning

If scenarios bring some elements of solution, it is not always possible to make an entire planning beforehand because one has to face an environment that evolves dynamically. Thus, one often speaks of reactive planning, opportunistic planning, approximate planning, etc.

In reactive planning, no specific sequence of actions is planned in advance. The planner is given a set of initial conditions and a goal. However, instead of a plan with branches, it produces a set of condition-action rules: for example, universal plans (Schoppers 1987) or Situated Control Rules (SCRs) (Drummond 1989). In theory, a reactive planning system can handle exogenous events as well as uncertain effects and unknown initial conditions. It is thus possible to provide a reaction rule for every possible situation that may be encountered, whether or not the circumstances that would lead to it can be envisaged (Pryor and Collins, 1996).

Britanik and Marefat (1999) presents the notion of plan fragments that encapsulates actions with their context. They show that the mergeability of two or more actions cannot be determined based on the actions alone, but that the surrounding actions in the plan must also be considered, that is, the context in which the actions are used to play a role in determining their mergeability. This position is close from our observation of operators that try to collect first a maximum of contextual information and, as a consequence, postpone actions in sequences.

Ginsberg (1995) argues that planning systems, instead of being correct (every plan returned achieves the goal) and complete (all such plans are returned), should be approximately correct and complete, in that most plans returned achieve the goal and that most such plans are returned. The author also argues that the cached plans used by case-based planner are best thought of as approximate as opposed to exact. His also shows that one can use this approach to plan for sub-goals g1 and g2 separately and to combine the plans generated to produce a plan for the conjoined goal g1+g2. The computational benefits of working with sub-goals separately have long been recognized, but attempts to do so using correct and complete planners have failed.

For Hayes-Roth (1979), people's planning activity is largely opportunistic. That is, at each point in the process, the planner's current decisions and observations suggest various opportunities for plan development. Thus, planning appears as an incremental process. As a planner relates each new decision to some subset of his previous decisions, the plan grows by incremental accretion. Alternative sub-plans can develop independently either within or between levels of abstraction. The planner can incorporate these sub-plans into the final plan as she or he wishes. An important aspect of this approach is mental simulation that can answer a variety of questions for the subject. The subject can use this information to evaluate and revise prior planning and to constrain subsequent planning. The authors thus propose a so-called opportunistic approach to planning. This non-hierarchical planning assumes that a plan is executed with the help of some kind of mental blackboard where pieces of information, relevant cues and possible sub-goals are stored. They claimed and showed that planning happens asynchronously and is determined by the momentary aspects of the problem. No fixed order of operations exists; the plan execution and the steps to be taken grow out of the problem stage at hand. When planners solve a planning problem, they may start with the top-goal, but very soon they loose track of the goal structure and then they continue to fulfill the goals that are reachable within reasonable time. The hierarchy very soon vanishes and what remains is some sort of heterarchy. This is for this reason that the planning behavior is called opportunistic.

In the system Igor, Saint Amant and Cohen (1994) present a language based on techniques of opportunistic planning, which balances control and opportunism. By explicitly representing the goals of exploratory analysis, they can take advantage of strategic knowledge about data analysis to structure and reduce search. Explicit control structures monitor the search for 'interesting' results and select appropriate context-dependent sequences of operations during the analysis. The primitive operations provided by the representation are called actions. An action is a data transformation or decomposition of an input structure into an output structure. Each action has an associated goal form and may be triggered by the establishment of a matching goal. The authors propose to combine actions in scripts. A script is a sequence of sub-goals whose associated actions transform one structure into a more concise, better parametrized, more descriptive structure. Scripts, like actions, have associated goal forms, and thus may be combined hierarchically to satisfy the goals of other scripts. The script representation addresses the main tradeoff between control and opportunism and relies on an explicit representation of context.

Kott *et al.* (1999) describe an application of a dynamic replanning technique in a highly dynamic and complex domain: the military aeromedical evacuation of patients to medical treatment facilities. Doctrinally, patients requiring extended treatment must be evacuated by air to a suitable medical treatment facility. They developed a decision support system from planning and scheduling medical evacuation operations. The most challenging aspect of the problem had to do with the dynamics of a domain in which requirements and constraints continuously change over time. Continuous dynamic replanning is a key capability of their system. The key concern in reactive replanning is plan continuity--the new plan should not introduce unnecessary disruptions. In the reactive (dynamic or real time) problem, some of the activities are occurring at the same time that the planning problem is being solved.

3.d Role of the context in decision tree simplification

Conversely to Edmonds' idea (1997), context model is not only a way to specify which nodes and arcs are activated in a decision tree. Context model is a way to generate dynamically the right path of the actions (the fully expanded alternatives discussed previously), some contextual elements playing a role at different step of this process to determine the neglectable events or the robustness of an action. Indeed, we meet the positions of Hayes-Roth (1985) and Rogoff *et al.* (1987) for whom we cannot anticipate all aspects of our planning endeaviors, and then it is often both advantageous and efficient to plan opportunistically, developing and adjusting plans during the course of action. As a consequence, we have not to consider a complex decision tree with a high number of leaves. Instead, we may have to consider a set of elementary fragments of the decision tree that will be assembled according to different contextual elements. In that sense, a context may be considered as a viable unit of task strategy appropriate to some step of the task (Grant, 1992). Context thus permits a division of cognitive structure into modules accomplishing similar functions but in different contexts: each module has an analogous role in its own context.

According to this view, events and references specify the context of the decision, that is, the interactions and relationships among a set of elementary actions. Moreover, several branches in a decision tree differ by few items that are to be considered only in one branch and skipped in another branch. For example:

IF	Context is high activity on the line.
THEN	Establish a temporary service.
ELSE	Skip the scenario "establish a temporary service"

In other domains, diagnosis and problem solving are intertwined. The effect of an action may change the diagnosis and the context of the problem solving. This is the case of complex process where an important number of variables intervenes and interacts, often with delay between the action and the effect. Hoc (1996) gives such an example with a barge. A chemical plant is another example of the situation. Thus, the postponing of actions is not always possible, and it is preferable to look for pruning the decision tree.

In many practical situations, the decision maker tries to use contextual knowledge to reduce the uncertainty. This uncertainty often results from the other agent moves. It is therefore advantageous to anticipate their moves. In the example of the choice of a privatization law (Lévine and Pomerol, 1989), each agent is simulated by an expert system. According to the previous events and actions in the current scenario, and to the action context of this agent (formalized by rules), the expert systems reduce the possible reactions to one or two. In case of a unique reaction, the branch is reduced. When two possible reactions remain, the development of the two emerging branches is carried out by the system, the final choice between different policies being left to the Minister.

In the system for evaluating timetable robustness, there are very few possible actions, most of them being rather robust (Pomerol *et al.*, 1995). In this case, for each possible incident that may occur on the railway or on trains, an expert system makes the decision to stop a train, overcome or wait in a station for a while. This decision is made according to contextual information as the number of travelers, peak hours or not, weather, etc. What is noticeable in this system is that the action sequence thus generated is insensitive to events (fully expanded alternative). At each step, the expert system chooses a robust action that can resist to any occurring new events. Note that, like in SART (see next section), this is possible because there are actually few possible actions, none of them being dramatically bad. In some sense, an expert system is a primitive way for using contextual information to reduce the complexity of decision trees. However, it is not clear whether the simplification results from reducing the number of events (e.g. in the privatization law) or finding robust actions (e.g. the timetable robustness). In many systems, the two issues are tangled and the two ideas are very involved.

Again, making context explicit in the decision tree permits to simplify the decision tree representation by avoiding to duplicate paths in the tree. For instance, the four upper paths of Figure 8 (tree representation) may be represented as:

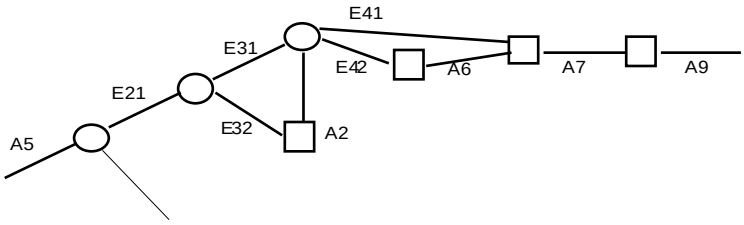

Figure 3: Simplification of the upper part of Figure 8

Another way to simplify the decision tree is to introduce robust actions. For instance, in Figure 3, the establishment of the temporary service (action A2) is normally realized only at peak hours (event E32). However, the operator may judge that the incident solving will last a too long time, and can decide to establish a temporary service even if it is off-peak hours. Thus, we can replace on Figure 3 the event (E31, E32) by the action A2 as:

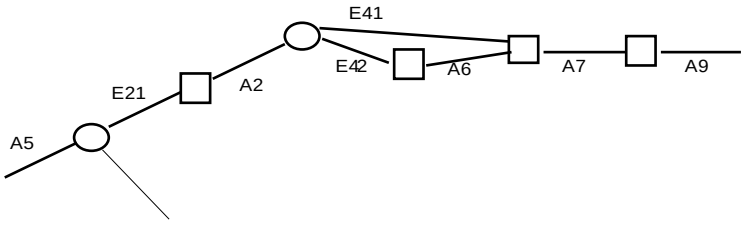

Figure 4: Simplification of the upper part of Figure 3.

Another way to prune a decision tree is to account for the context of the line itself. For example, in Figure 8 all the paths conclude to the same action: Evacuate the damaged train at the terminus, not, say, evacuate the damaged train on a secondary way. The reason is that, in Paris, the workrooms for repairing trains are in the terminus. The strategy for the metro in Rio de Janeiro is different because workrooms are in the middle of the line.

4. THE SART APPLICATION

4.a Introduction

The SART project (French acronym for support system for traffic control) aims at developing an intelligent decision support system to help the operators who control a subway line to react to incidents that occur on the line (Brézillon *et al.*, 1997; Pasquier, 2000; http://www.lip6.fr/SART). This project is based on the interaction between the operator and the system and will ease their mutual comprehension. We analyze the operational knowledge used by operators and store it in an adapted structure, which will be easily understood by operators and efficiently used by the computer. As an Intelligent Assistant System (IAS), SART has to accomplish several functions such as acquiring knowledge from operators; simulating train traffic on the line, possibly with incidents; changing the model of the line on operator's request for helping the operator to test alternative issues; proposing alternatives for an incident solving; training a new operator not familiar with a given line; etc. We use a multi-agent approach in SART development. To date, our group has developed three agents in the last two years, namely a line-configuration agent, a traffic-simulation agent and an incident-management agent.

Before describing an operating system designed according to the IAS philosophy, we should say a little about the implementation techniques. In the case described below, we choose a multi-agent approach to allow an evolutionary architecture in which other agents, *e.g.* a training agent, can be added after the first phase of development, in order to expand the functions of the system. Specifications are now completed and we are in the programming phase. An extended presentation can be found in (Brézillon *et al.*, 1997; Brézillon *et al*., 2000).

4.b The operational knowledge in the Parisian subway control

During the modeling phase, we tried to understand the troubleshooting strategies applied in some incidents, such as lack of motor power on a train. Our first representation used rules. This representation was not well accepted by operators and was difficult to maintain. The main reason was that the level of description of the operational knowledge was too low (too many details) and no global strategy was perceptible.

Then, we adopted a tree representation, made of two types of elements: the action and the contextual nodes. Contextual nodes lead to an incident solving according to a given context by selecting a path depending on the value of a contextual piece of knowledge. Figure 8 shows the decision tree representation of the official procedure for "train lack of power."

Decision trees inspire our tree representation, but they differ namely on two points. First, our trees have no decision node, only "chance" nodes where a contextual element is analyzed to select the corresponding path. Second, There are no probabilities. This representation shows important specificity having consequences on the size and structure of our tree:

- Operators have a single main goal: "to reestablish a normal traffic as fast as possible, respecting elementary security rules." This point places emphasis on the fact that in the same situation, several strategies permit to solve the incident.

- Operators use many contextual elements to perform their choice. This leads to a large number of practical strategies, even for the same incident. This point multiplies the number of branches and the tree grows rapidly.

- Operators prefer to gather a maximum of information before making their decision. This attitude postpones most of the actions to the end of the branches of the tree. This observation is very close to the observation of Watson and Perrera (1998) that consider a hierarchical case representation that holds more general contextual features at its top and specific building elements at its leaves.

- Operators choose actions that allow acceding to common intermediary situations. They can thus reuse common strategies to clear the incident. As a consequence, the terminal action sequences are often repeated from one branch to another.

- Several action sequences are done in the same order in different situations (paths). Other actions could be done in different order, but must be accomplished before a given other. For

example, before to link two trains, both have to be emptied, but the order in which they are emptied is not important (partial order on the actions).

The tree structure is heavy and does not permit to represent highly-contextual decision-making in complex applications. In a next section, we explain the modification we have introduced, based on the specificity discussed here, to obtain a manageable structure for representing operational knowledge on incident solving on subway lines.

4.c Decision making in a Control Room

Operators solve an incident by choosing a scenario, which is a sequence of actions conditional on possible events. The choice of a scenario greatly relies on contextual knowledge. One operator said us: "When an incident is announced, I first look at the context in which the incident occurs." The reason is that operators want to have a clear idea of future events; the purpose of this look-ahead reasoning (Pomerol, 1997) is to reduce, as far as possible, the uncertainty in the scenario. The problem for operators is that many scenarios are similar at the beginning and then diverge according to the context. Thus, a scenario is a sequence of actions intertwined with events that do not depend on the decision makers but that result in a limitation of their actions. For instance:

Focus of attention:	Removal of a damaged train from the line.
Contextual information:	Level of activity on the line.
Action:	Lead the damaged train to the:
	- terminal if the line activity is low
	- nearest secondary line if line activity is high.

As most of the contextual elements may intervene in several scenarios (*e.g.*, traffic activity, position of the next train), operators prefer to take them into account as soon as possible to get a general picture of the best path to choose. At this step, contextual knowledge is proceduralized and in the meantime operators postpone action. The main objective is to eliminate event nodes. By grouping together a set of actions in a macro-action, operators hope to make the following step easier.

4.d Looking first for contextual information

Previously, we said that the operators look first for information on the incident context. The reason is that they want to have a clear idea of what the future may be composed of (the look-ahead in Pomerol, 1997), and try to reduce, as far as possible, the uncertainty in the scenario. The problem for operators is that many scenarios are similar at the beginning and diverge according to context after. For instance:

Focus of attention:	Damaged-train elimination of the line.
Context:	Activity on the line.
Action:	Lead the damaged train:
	- at the terminal if the line activity is low
	- on a secondary line if line activity is high.

Landauer and Bellman (1993) find a similar conclusion by noting that the process of a problem study presents three major steps: find_context, pose_problem, and study_problem. The context includes conditions on the existence of certain information and partial mappings.

As most of the contextual elements intervene on several paths (e.g., traffic activity, position of the next train), operators prefer to take them into account as soon as possible to have a general picture

of the best path to choose. At this step, contextual knowledge will be contextualized and thus operators postpone the action during this contextualization step. The main objective is to eliminate chance nodes as said before. By grouping together a sequence of actions in a macro-action (a usable procedure), operators hope to plan beforehand the whole look-ahead. Briefly, such macro-actions are a way to make context explicit and introduce modularity in the diagnosis process by managing different modules accomplishing the same function (scenario) but in different ways according to the context.

4.e Procedures and practices

Since 1900, the company faces incidents and its agents solve them. These practices reflect the construction of operational knowledge, step by step, by the operators. Security sake and the willing of incident solving uniformization pushed the head of the company to compare the practices and to establish secure procedures for each encountered incident. In this sense, procedures are collections of safety action sequences permitting to solve a given incident in any case. These procedures are based on practices, but eliminate most of contextual information and particularities of each incident. Trying to promote sufficiently general procedures results often in sub-optimal solutions for incident solving. In this sense, procedures are useful guidelines for operators, but they have to be adapted for each new incident situation.

At RATP, most of the incidents have been well-known for a long time (object on the track, lack of power supply, suicide, etc.). Thus, the company has established procedures for incident solving on the basis of their experience. However, each operator develops his own practice to solve an incident, and one observes almost as many practices as operators for a given procedure because each operator tailors the procedure in order to take into account the current proceduralized context, which is particular and specific. In many working processes human beings can be observed to develop genuine procedures to reach the efficiency that decision makers intended when designing the task. Some parts of this practice are not coded (Hatchuel, 1992). Such know-how is generally built up case by case and is complemented by "makeshift repairs" (or non-written rules) that allow the operational agents to reach the required efficiency. This is a way of getting the result whatever the path followed. The validation of those unwritten rules is linked more to the result than to the procedure to reach it. De Terssac (1992) spoke of logic of efficiency.

In parallel, operators prefer to plan again their action in real time rather than to rely on these procedures based on company's experience, this is due to two main reasons. Firstly, the selected procedure is not always perfectly adapted to the situation at hand and can lead to improper actions or sub-optimal incident resolution strategies. Secondly, if the operator relies on a procedure, he can miss some important facts and notice them too late to adequately solve the incident. Operators prefer generally to plan again their action continuously according to the situation. Procedures are then used as frames to construct a genuine strategy tailored to the specificity of a given situation. Such practices are based on operational knowledge and are shared by operators. Leplat (1985) studied this well-known phenomenon that distinguished between the prescribed and the effective tasks. The former is the task conceived by the “method office” of the company, and the latter is the effective task that is executed by the employee. The effective task corresponds to the goals and conditions effectively taken into account during the activity.

The modeling of operators' reasoning is a difficult task because operators use a number of contextual elements, and by the fact that procedures for solving complex incidents have some degree of freedom. Their reasoning stems from some chunks of implicit knowledge, which are imposed on the driver because they correspond to mandatory procedures. Procedures are established from operator's experience during similar incidents and fixed by the company. As such, practices are what we call hereafter proceduralized contexts (see section 5.2). An implicit piece of knowledge is that travelers are safer in a station than in a tunnel. At a deeper level, the driver has to avoid stopping the train a long time in a tunnel. The reason is that some travelers may have behavioral troubles such as claustrophobia and could leave the train to wander about on the railway (and thus may generate another type of incident such as "Traveler on the railway"). These pieces of knowledge, which are not necessarily expressed, result in more or less proceduralized actions that are compiled as a proceduralized context. Very often many pieces of proceduralized context are structured together in comprehensive knowledge about actions.

Degani and Wiener (1997) distinguish procedures, practices and techniques. Procedures are specified beforehand by developers to save time during critical situations. Practices encompass what the users do with procedures. Ideally, procedures and practices should be the same, but the

users either conform to procedure or deviate from it, even if the procedure is mandatory. Techniques are defined as personal methods for carrying out specific tasks without violating procedural constraints. Users develop techniques over years of experience (Brézillon, 1996, 1999). Knowledge acquisition focuses on procedures and, eventually, practices, but rarely on techniques. Moreover, in most of real-world applications, a decision maker faces ill-defined situations where the form of the argumentation rather than the explicit decision proposal is crucial (Forslund, 1995). This implies that it would be better to store advantages and disadvantages rather than the complete decisions.

Medicine is also a domain where the distinction between procedure and practice in the one hand, and the notion of context in the other hand is very important. A practice can be considered as a kind of causal explanation for an incident solving in the spirit of the graphs in ABEL system described in Patil *et al.* (1981) called patient-specific model on the basis of the situation-specific models of Clancey (1992). In the ABEL system, a causal explanation is represented as a five-layered graph containing the particular findings, disorders, and their causal or subtype relations that are believed to be present in the particular patient being diagnosed. Such specific models are causal arguments having the structure of a proof (in our case, the effective solving of the incident). Bouaud *et al.* (1999) describe also in medicine this type of operationalization of the knowledge from procedures to practices when a physician can make different therapeutical choices for a diagnosis according to the instantiation of the clinical context built from his perception of his understanding of the patient. Strauss *et al.* (1985) give the example of the unravelling plan for an osteoarthritis patient that might state that an X-ray image of the hip is necessary. But when applying the plan to Mr. Jones, who doesn't have any problems with his hips, this part of the plan may be skipped – and other examinations, like a blood test, might be added to Mr. Jones' unravelling plan. Thus, for the last authors, a protocol in medicine is a standard operating procedure (Strauss *et al.*, 1985).

In high technical and heavily dynamical process regulation domains, operators who are responsible for the process control have to rapidly react. If an incident occurs, they have only few minutes to forge a representation of the issue, gather information on the situation, analyze the incident and undertake the correcting actions. To ease their job, many companies have established fixed procedures. Initially, general procedures have been designed to provide operators with a secure reference for incident solving. However, these general procedures forget the contextual dimension of the case at hand. Nowadays, companies are diversifying these procedures by introducing increasingly contextual considerations. This operation increases by specialization the available procedures for each type of incident type.

This discussion points out that if it is relatively easy to model procedures, the modeling of the corresponding practices is not an easy task because they are as many practices as contexts of occurrence. For example, when a train cannot move in a tunnel, there are procedures for evacuate travelers at the nearest station, for evacuate the damaged train by another train, etc. Some procedures are sequential, but others may be accomplished in any order compared to some ones. When a train must push a damaged train, both trains must be empty but the order in which travelers of the two trains are evacuated is not important and depends mainly on the context in which are trains. What is important is that the two actions must be realized before to intervene on the damaged train. For complex incident solving, it is not possible to establish a global procedure, but only a set of sub-procedures for solving parts of the complex incidents. Moreover, procedures cannot catch the high interaction between the solving of the incident itself and the number of related tasks that are generated by the complex incident. As a consequence, there are as many strategies for solving an incident as operators. Cases that are similar in one context may be totally dissimilar in others as already quoted by Tversky (1977).

4.f Planning is intertwined with the decision making

Nowadays, we observe the multiplication and the specialization of procedures in a number of domains. This is a general trend overtaking the railway control process. In aeronautics, operators always adapt the actual procedures to the current situation. De Brito and Boy (1999) explain that the aircraft operators prefer to plan again their actions instead of following non-adapted procedures, even if the new plan is inspired of these procedures. Britanik and Marefat (1999) also propose a plan merging methodology that merges partially-ordered plans based on the notion of plan fragments. Their notion of plan fragments encapsulates actions with their context. Hayes-Roth and Hayes-Roth (1979) proposed a so-called opportunistic approach to planning. This non-

hierarchical planning assumes that a plan is executed with the help of some mental blackboard where pieces of information, relevant cues and possible sub-goals are stored. They claimed and showed that planning happens asynchronously and is determined by the momentary aspects of the problem. No fixed order of operations exists; the plan execution and the steps to be taken grow out of the problem stage at hand.

Xiao *et al.* (1997) study anesthesiologists' work for which each patient can represent a drastically different “plant” and thus there are relatively few well-defined procedures stipulated either by inside professional communities or by outside regulatory agencies. They think that the planning process is necessarily fragmentary and non-exhaustive. When applying a plan to a concrete problem, the situated actions performed in the activity often mirror the plan, but are adjusted to the concrete details and conditions of the context.

The same conclusion is observed in other domains too (e.g., see Hoc, 1996; Debenham, 1997; Bainbridge, 1997). The main reason is the impossibility of making explicit the relationships between a system and its changing environment. Bardram (1997) speaks of situated planning and others of opportunistic planning, reactive planning, etc. (see Section 3.3) In an application in air control, Degani and Wiener (1997) shown that the descent phase is highly context dependent due to the uncertainty of the environment (e.g. air traffic control, weather), making it quite resistant to procedurization. Hollnagel (1993) proposes a contextual control model that distinguishes between a competence model (actions, heuristics, procedures, and plans) and a control model (mechanisms for constructing a sequence of actions in context), which are both heavily influenced by context (skills, knowledge, environmental cues, etc.).

Note that there are domains--as industrial projects-- in which one does not know the structure of the product in advance but the structure is specified by the design activities. This means that a large portion of the planning activities can only be executed after some design activities have been carried out. However, before we can execute any design activities, they have to be planned. There is thus a dynamic interplay between the design process that refines the goals of the project and the planning process that keeps the activities consistent with the goals. In other domains, as space exploration, no procedures are available when a space vessel is sent for the first time.

A solution to the impossibility of a comprehensive a priori planning is the acquisition of skills for anticipating. For Hoc (1996), the anticipative mode is the usual functioning mode of human beings: the human operator always checks, more or less explicitly, hypotheses instead of being in a situation of discovery or surprise. An anticipatory system would be a system that uses knowledge about future states to decide what action to make at the moment. An anticipatory system has a model of itself and of the relevant part of its environment and will use this model to anticipate the future (Ekdahl *et al.*, 1995). The system then uses the predictions to determine its behavior, i.e., it lets the future state to affect its present states. An intelligent system should be able to predict probably what will happen and pre-adapt itself for the occurrence of a crucial or time-critical event. This implies that an intelligent system must be equipped with a simulation component.

5. UNDERSTANDING AND MODELING CONTEXT

5.a Introduction

We have seen in Section 2 that the understanding of context is somewhat better than ten years ago. We know that context must be always considered in reference to its use, that there are different types of context according to what we consider, context representation is strongly dependent of knowledge and reasoning representation. However, in most of the studies on context, the dynamic dimension of context is not considered, and thus the use of context in real-world applications is very limited.

In this section, we present our view on context. First, we describe the three aspects of context. Second, we discuss the way in which contextual knowledge can be represented. Third, we give an example of context-based representation of the knowledge. Fourth, we give another example of movement from contextual knowledge to proceduralized context.

5.b Three parts in context

We consider three types of knowledge in a given decision making step. First, knowledge that is shared by those involved in the problem and is directly but tacitly used for the problem solving. Second, knowledge that is not explicitly used but influences the problem solving. Third, knowledge that has nothing to do with the current decision making step but is known by many of those involved. We call these three types of knowledge respectively: proceduralized context, contextual knowledge and external knowledge. We start by giving some insights about these three types of contextual knowledge in the framework of SART before stating more clear definitions in the next section.

We define context as the sum of all the knowledge possessed by the operators on the whole task. At a given step of a decision making process, we separate the part of the context that is relevant at this step of the decision making, and the part that is not relevant. The latter part is called **external knowledge**. The former part is called **contextual knowledge**, and obviously depends on the agent and on the decision at hand. A part of the contextual knowledge will, as explained below, be proceduralized. We call it the **proceduralized context**. Figure 5 illustrates the three types of context.

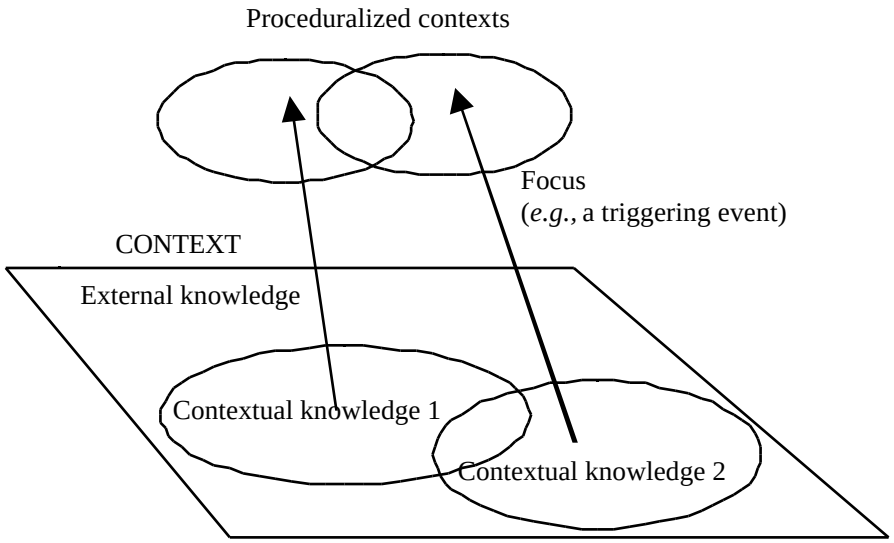

Figure 5: Different types of context

It is in fact difficult to define the concept of context without considering the people involved in a situation because, at first glance, context involves knowledge that is not explicit. This explicitness depends on the actors. Some common knowledge is implicit but well-known, for example the fact that it is easier to organize emergency operations in a station than in a tunnel. When the reasoning yields this type of knowledge, it is easily proceduralized and becomes an implicit part of the reasoning that can be elicited by knowledge engineers and finally included in the operation model.

The second fact is that each person involved uses a large amount of knowledge, different from one person to another, to picture the situation. We can define the contextual knowledge as all the knowledge that is relevant and can be mobilized to understand a given situated decision problem. By "situated" we mean in given, dated, and well-specified circumstances. Clancey (1991a, 1991b) introduced the word "situated" into artificial intelligence. In its artificial intelligence sense, "situated cognition" emphasizes the role of interaction and context in human behavior. This weak situated cognition hypothesis (Menzies, 1996), which links knowledge, interaction and context, provides a good background for our views.

Contextual knowledge is personal to an agent and it has no clear limit (McCarthy, 1993). Contextual knowledge is evoked by situations and events, and loosely tied to a task or a goal. However, when the task becomes more precise, a large part of this contextual knowledge can be proceduralized according to the current focus of the decision making. Although the contextual

knowledge exists in theory, it is actually implicit and latent, and is not usable unless a goal (or an intention) emerges. When an event occurs, the attention of the actor is focused and a part of the contextual knowledge will be proceduralized. In our definition, the contextual knowledge is dependent on the situation (date, location, and participants). It is a sub-part of the overall context (see Figure 6).

Proceduralized context is knowledge that is explicitly considered at a given step of a problem solving. Contextual knowledge intervenes implicitly by only constraining problem solving. For example, operators that ensure the monitoring of the distribution of water in Paris had noted that there was a peak in the water consumption late each evening. The peak was reproducible every day but not predictable because not exactly at the same time. After an inquiry, they discovered that persons use water for domestic needs (drink a glass of water, wash dishes, pour water on flowers, go to the toilets, etc.) during the advertisements introduced in the TV movie. The introduction of advertisements in the movie depends on the organization of the movie scenario. Such knowledge (the link between the peak of consumption and the advertisements at the TV) has a contextual nature for the water distribution. Contextual knowledge constrains a given step of the problem solving (water distribution at advertisements time in the example) without intervening in it explicitly.

The proceduralized context is a part of the contextual knowledge that is invoked, structured and situated according to a given focus and is common to the various people involved in decision making. The proceduralized context may be compiled but can generally be elicited with the usual techniques of knowledge acquisition; it is limited and should be a part of the operation model. For the model to be usable, people have to define a finite number of situations, and diagnosis consists mainly of trying to determine in which situation those involved find themselves.

5.c Representation of contextual knowledge

Contextual knowledge is more or less similar to what people generally have in mind about the term 'context'. It contains some general information about the situation and the environment of the problem. Contextual knowledge implicitly delimits the resolution space (this idea is also evoked in Bainbridge, 1997). It is always evoked by a task or, in our case, an event. Contextual knowledge does not focus on a task or on the achievement of a goal but is mobilized, even though it has not yet been proceduralized for use.

For instance, in a normal situation, the control operator faces the following concern:

F0: the *normal* focus of attention is to see that schedules and intervals between trains are respected.

This task can be regarded as routine and does not require special attention. Nevertheless, contextual knowledge about control is involved. In this task, the word *normal* has different meanings according to the context. Let us look at some possible contexts.

C0: the normal context associated with F0 involves:
- k1: type of day (e.g., working day, Saturday, Sunday, Holidays),
- k2: period of the day (morning, afternoon, evening),
- k3: traffic state (rush hours, off-peak hours),
- k4: the section load (very busy, few people),

All these pieces of knowledge are some of the elements defining the contextual knowledge describing the environment of the problem with which the following pieces of knowledge are associated:

k5: the interval between trains according to the situation,
k6: the stopping time in stations,
etc.

Contextual knowledge is therefore quite large and not focused. Many "normal" contexts are contained in this contextual knowledge. Assume now that an incident occurs on the subway line; the pieces of knowledge k1 to k4 are (or should be) immediately invoked. This results in k5 and k6 being invoked. They become a part of the proceduralized context in which the incident is resolved.

Contextual knowledge appears back-stage, whereas the proceduralized context is front-stage in the spotlights. It is noteworthy that, as far as engineering is concerned, only the proceduralized context matters, but contextual knowledge is necessary because this is the raw material from which proceduralized context is made. One can say that contextual knowledge is proceduralized, not necessarily explicitly, to become the proceduralized context. In a sense, the proceduralized context is the contextual knowledge activated and structured to make diagnoses, decisions and actions.

In our work, "contextual" is not opposed to "linear" reasoning as in Bainbridge (1997); see also Hoc and Amalberti (1995) for a criticism of "linear" models of diagnosis. As Pomerol (1997) stated, the analysis of decision systems which separate diagnosis from anticipations is just an engineer's simplification and not a cognitive model, however many observations about what is called "decision bias" in decision theory may be due to the wrong intertwining of diagnosis, anticipation and preferences. It is also noteworthy that during the transformation of contextual knowledge into proceduralized context, some interpretations and rationalizations occur; this is reminiscent of the transformation of data into facts in the sense of Hoc and Amalberti (1995). Note also that our view about context is not inspired by a holistic point of view as opposed to a decomposition of the actions and reasoning (see Hollnagel, 1993 for his holistic argumentation).

5.d Context-based representation and the onion metaphor

In (Brézillon *et al.*, 1997; Brézillon *et al.*, 2000), we present the SART application for the subway in Paris. The goal is to support the operator who is responsible of a subway line when an incident occurs. The Figure 6 gives a very partial view of the solving of the incident "Ill traveler in a train." Ovals represent incidents and rectangular boxes represent steps of the incident solving (e.g., "Alarm signal").

Consider the step "Stop at the next station." This step--proceduralized context--is imposed on the driver because, for example, this corresponds to procedures. Procedures arise from operators' experience with similar incidents. For example, travelers' security is better ensured in station than in a tunnel. At a deeper level, the driver has to avoid stopping the train a long time in a tunnel because some travelers may have behavior's trouble as claustrophobia and leave the train to go on the railway (and thus generate another type of incident).

All the pieces of contextual knowledge are not at the same distance from the proceduralized context "Stop at the next station." For instance, "Procedures" is a contextual knowledge that is close to the incident-solving step, when "Avoid stopping in a tunnel" is another piece of contextual knowledge that is far from the same step. However, both of them make this step necessary. Such a kind of distance permits us to order pieces of contextual knowledge in layers around the step like skins of an onion. We call this the onion metaphor. Stippled circles in Figure 8 represent layers of contextual knowledge. Agabra *et al.* (1997) obtain a quite similar result in Enology. Tichener (cited in Jansen, 1995) also considered the notion of a situation surrounding the organism as one of the roots of context. Tichener's definition implies that context (i.e. our contextual knowledge) is not part of the actual chunk of knowledge (i.e. our proceduralized context) but forms a layer, or a set of layers, around the knowledge.

Contextual knowledge also ensures a link between the different steps of a given incident solving and across different incident solvings. Thus, if contexts at the level of problem-solving steps

constitute a discrete set of contexts, there is a unique context at the level of the problem solving itself that evolves continuously along the solving.

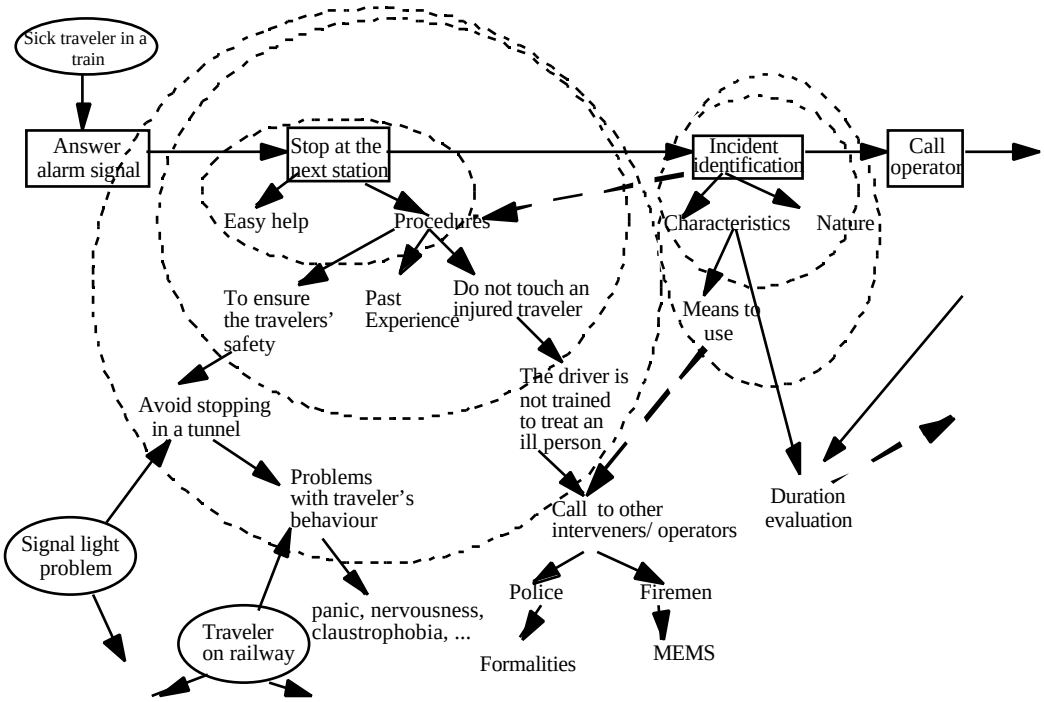

Figure 6: Context-based representation of the incident "Ill traveler in a subway"

Our context modeling along the onion metaphor reveals several interesting results:

(1) A step takes a meaning in a given context. Contextual knowledge does not intervene directly at this step but constrains it. For the step "Stop at the next station," the contextual knowledge "Easy help" is not the main reason for stopping the train at the station. However, it intervenes in its realization.

(2) Pieces of contextual knowledge may be partially ordered. If we consider the step "Stop at the next station," we observe that some knowledge pieces of its context (e.g., "Easy help") are closer to the step than other (e.g., "Do not touch an injured traveler") because the constraints applied on the step are more direct.

(3) Each piece of contextual knowledge takes a meaning in a context. The piece of contextual knowledge "Procedures" of the step "Stop at the next station" has its own context with elements as "Past experience" and "Do not touch an injured traveler." Recursively, one can link knowledge pieces together by layers. This result is a concrete example of McCarthy's claims (1993) about the definition of a context in an outer context and the infinite dimension of context.

- Pieces of contextual knowledge relate incidents together. With an association between a number of contextual-knowledge pieces, it appears that there is a relationship between a given incident and others. Figure 8 shows how such a relationship is established between "Sick traveler in a train" and other incidents as "Signal light problem" through contextual knowledge. Establishing such an incident net accounts for the occurrence in real-life conditions of combinations of incidents that seem apparently not related (e.g., "Ill traveler in a train" and "Signal light problem"). Common contextual components (or pieces of knowledge) of two contexts are highlighted by dotted, bold arrows in Figure 6.

5.e Movement between contextual knowledge and proceduralized context

Let us now present how layers of contextual knowledge are used in a diagnosis process. Proceduralized context defines what the focus of attention is, and contextual knowledge defines the context of the focus of attention.

For example, in a normal situation, the operator in the control room faces:

F0: normal focus of attention with pieces of the proceduralized context such as the
traffic evolution on the line.
C0: normal context associated with F0 with elements as:
k1: type of the day (e.g. work day, Saturday, Sunday, Holidays)
k2: part of the day (morning, afternoon, evening)
k3: traffic state (peak hours, off-peak hours)
k4: charge of the section (high charge, low charge)
k5: incident-elimination know-how.

At the announcement of an incident on the subway line, the piece k5 of contextual knowledge enters the focus of attention and becomes a piece of the proceduralized context. Some pieces of external knowledge as the position of the incident on the line also enter the focus of attention. The context also evolves to integrate some external-knowledge pieces as maintenance activity on the line, the number of trains on the line and available helps. Thus, the context of the diagnosis evolves continuously along the diagnosis process.

The operators' reasoning during a problem solving is the following. When an incident is announced, they first look at the context in which that incident occurs. This means that they consider, at a first step, the process under their control (e.g., the line control) as a set of contextual elements. They extract a subset of contextual knowledge pieces that become proceduralized context (entering the focus of attention). On the basis of this subset of proceduralized context pieces, they try to have a clear idea of all the alternatives, and make their decision that is concretized by a sequence of actions.

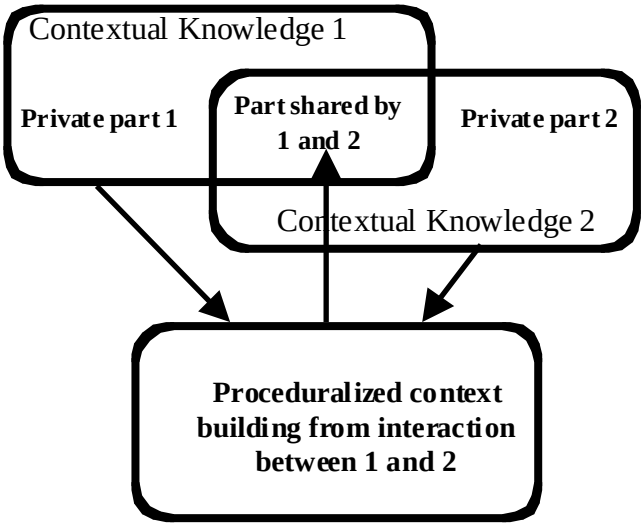

Figure 7: A representation of the interaction context

From an analysis of the possible action sequences, operators may face three alternatives:
(1) The action sequence leads to a successful incident solving,
(2) The action sequence must be refined, and
(3) The action sequence must be changed.
In the first alternative, the set of proceduralized context pieces leads to the incident solving. In the second alternative, the set of proceduralized context pieces must be extended, new contextual knowledge pieces are introduced as proceduralized context pieces, the decision making process is reiterated and leads to a modified action sequence. In the third alternative, a deep change of the

proceduralized context pieces (i.e. the new set of contextual knowledge pieces that is chosen is different of the previous one) leads to a different decision making process and a different sequence of actions. In the two last alternatives, the process of decision making is reiterated until a successful incident solving is reached.

The third alternative--change of the action sequence--corresponds to a new process of decision making by adjustment of scenarios. A change in this process is due to a lack of contextual information that is needed to choose among alternatives in the decision making. This is because a scenario is a sequence of actions intertwined with events that are external to the decision making but act as constraints on it.

Figure 7 represents how the proceduralized context is built from contextual knowledge during an interaction between two agents. The interaction context contains proceduralized context pieces in the focus of attention of the two agents. These pieces of knowledge are extracted from the contextual knowledge of each agent; they are organized and structured jointly by both agents and result in a shared knowledge. Generally, the first utterance of an agent gives a rule such as “Stop at the next station” if the alarm signal is triggered. Then on the request of the second agent, the first agent may add some pieces of knowledge related to his first utterance. If this knowledge chunk belongs to the common part of the contextual knowledge of the two agents, the pieces are integrated into a mutually acceptable knowledge structure, and the knowledge structure may then be moved to the shared part of the contextual knowledge of each agent. Thus, the proceduralized context contains all the pieces of knowledge that have been discussed and accepted (at least made compatible as quoted in Karsenty and Brézillon, 1995) by all the agents. These pieces of proceduralized context then become part of the shared contextual knowledge of each agent, even if they do not remain within the focus of the proceduralized context.

Decision making is a dynamic process. From one step to the next one, a piece of contextual knowledge either enters the proceduralized context or become external knowledge. Conversely, a piece of proceduralized context may become either contextual knowledge or external knowledge. Thus, the content of the context evolves continuously all through the decision making process. The proceduralized context may also evolve to integrate some knowledge that, up to now, has neither been proceduralized nor is contextual (*i.e.* external knowledge). Such a continuous building of the proceduralized context has the consequence that we face a context-based approach: context-based reasoning, context-based planning, etc.

5.f Related works on context-based approaches

Saint Amant and Cohen (1994) address in their system Igor how scripts should be triggered, in which order they should run, and how results may be shared between different threads of exploration. An explicit representation of context, and two mechanisms that depend on context, namely monitoring and focusing obtain this. A script matches a goal not in isolation but in an environment containing the goals and scripts that have been expanded beforehand. The environment is made explicit in a context structure associated with each plan. A context structure is simply the sequence of intermediate and end results produced by execution of a script. Because other scripts may satisfy script sub-goals, a hierarchy of contexts is built up during exploration. This hierarchy provides contextual information to monitoring and focusing mechanisms. As intermediate and final results are generated, they become available in the context of the script that produced them. When a script matches a goal, posts its sub-goals, and eventually runs to completion, its results propagate to the context of the higher level script that posted its goal. Results propagate from low levels up to the more abstract levels, where they can be evaluated for their relevance to other areas of exploration.

Gonzalez and Ahlers (1993, 1995, 1997) describe the development of a knowledge representation paradigm that used to model the intelligent behavior of simulated agents in a simulator-based tactical trainer. Their hypothesis is that tactical knowledge is highly dependent upon the context (i.e., the situation being faced). Thus, a combination of script-like structures and pattern-matching rules in an object-oriented environment could serve as a concise means of representing the knowledge involved, as well as an efficient means of reasoning with that knowledge. This hypothesis was tested through the development of a prototype system that implemented the knowledge of a submarine tactical officer on a patrol mission. The results of the prototype show that the combination of scripts and rules in object-oriented environment meets the requirements described above.

Their concise means of representing the knowledge involved, as well as an efficient means of reasoning with that knowledge, is called context-based reasoning (CxBR). CxBR encapsulates knowledge about appropriate actions and/or procedures, as well as possible new situations, into contexts. Applying context-based reasoning presents a highly effective and efficient methodology for imparting sufficient intelligence to agents so as to achieve their objective in a training simulator. The context would also contain a set of rules together with its own “mini” inference engine consisting of a pattern matcher, a Rete net as an agenda, as well as the capability to assert and retract facts from the local fact base, to call procedures, and to change contexts.

The representation paradigm proposed is based on the idea that the applicable tactical knowledge is highly dependent upon the situation being faced by the decision maker (i.e., the context). A combination of script-like structures and pattern-matching rules in an object-oriented environment could serve to hold all knowledge pertinent to the context present at a specific time. This paradigm has been preliminary tested in a prototype system that incorporates the knowledge of a submarine tactical officer on a patrol mission. Evaluation of the prototype shows that the context-based paradigm promises to meet the desired levels of conciseness and effectiveness required for the task.

Tactical knowledge is required in order to endow autonomous intelligent agents with the ability to act, not only intelligently, but also realistically, in light of a trainee's action. In general, tactical knowledge can be said to address rime-stressed tasks, which require (1) assessment of the situation at hand, (2) selection of a plan to most properly address the present situation, and (3) execution of that plan. The work described by Gonzalez and Ahlers is based on the idea that by associating the possible situations and corresponding actions to specific contexts, the identification of a situation is simplified because only a subset of all possible situations are applicable under the active context.

In CxBR, contexts are the most important representational item. Much knowledge about how an agent should behave, as well as what other contexts it can transition is stored in the context objects themselves. There are three levels of contexts that can be represented, and they are ordered hierarchically: (1) the mission context, the major contexts, and the sub-contexts. Active contexts change in response to external events but also as a result of actions taken by the decision maker. A context can be likened to a situation that has bee recognized, and which has a prescribed set of procedures that must be carried out, sequentially, methodically, or arbitrarily.

Turner (1993, 1997, 1998) has developed a system--an adaptive reasoner--to make context explicit for autonomous underwater vehicles to tackle unanticipated events in complex environments. Contextual information helps the agent to focus its attention on appropriate goals to achieve in the current situation. Thus, context intervenes in at least five different ways: (1) make predictions about the situation; (2) modulate agent's behavior; (3) focus agent's attention; (4) influence an agent's choice of actions; (5) determine how an agent should handle unanticipated events. An agent should be able to recognize its current context as an instance of a class of contexts it knows about. It should be able to reason about its context, bringing to bear knowledge that is explicitly known to be contextual in nature.

Contextual knowledge is represented as a set of contextual schemas (c-schemas), then retrieving the most appropriate of those and using them to help the reasoner behave appropriately for its current context. Thus, c-schemas contain information not only describing the context they represent, but also information prescribing how to behave in situations that are instances of that context. Schemas provide a natural way to represent contexts, which should facilitate knowledge acquisition and potentially provide a tie to established machine learning approaches such as case-based reasoning.

An agent's context manager retrieves the best c-schemas from its memory based on features of its current situation, then merges them to form a view of the current context, the current c-schema. Thus, relatively few contexts are represented as c-schemas, but they are combining as needed to adequately represent a particular situation. The major difference with case-based reasoning is how c-schemas are used: generalized cases are usually used as indexing structures, while c-schemas are problem-solving structures in addition to their role in memory organization. Context is mainly considered as a way to cluster knowledge for search efficiency, for representing counter-factual or hypothetical situations, for circumscribing the effects of particular actions to particular situations, and for directing an agent's focus of attention to salient features of a situation.

In this framework, Turner describes context-mediated behavior (CMB), an approach to context-sensitive behavior developed over the past few years for intelligent autonomous agents. Context-mediated behavior (CMB) is based on the idea that an agent have explicit knowledge about contexts in which it may find itself, then use that knowledge when in those contexts. CMB is

automatic once a context is recognized. In CMB, contexts are represented explicitly as contextual schemas (c-schemas). An agent recognizes its context by finding the c-schemas that match it, then it merges these to form a coherent representation of the current context. This includes not only a description of the context, but also information about how to behave in it. From that point until the next context change, knowledge for context-sensitive behavior is available with no additional effort. This is used to influence perception, make predictions about the world, handle unanticipated events, determine the context-dependent meaning of concepts, focus attention, and select actions. CMB is being implemented in the Orca program, an intelligent controller for autonomous underwater vehicles.

In case-based reasoning, it is assumed that similar problems have similar solutions. Retrieving relevant cases is a crucial component of case-based reasoning systems. A main problem is that what is considered similar in one situation may not be similar in another one. Jurisica (1994) proposes a context-based similarity as a basis for flexible retrieval. Case similarity is assessed with respect to a given context that defines constraints for matching. Context allows to specify what parts of information representation to compare and what kind of matching criteria to use. Context can thus be used as a basis of relevance measure, i.e. items are considered relevant if they are similar with respect to the current context. This allows, for instance, for excluding similar but irrelevant items.

Jurisica (1994) defines context-based similarity, where context is a set of attributes with associated constraints on the attribute values. The tasks that can be addressed by a context-based similarity are comparing items, retrieving items, finding a context, and knowledge mining. Similarity is thus considered as a relation with three parameters: a set of relevant items, a context and an information base. Context-based similarity has two levels. The first level is an equivalence of items and is called a surface similarity. The second level deals with similarity between contexts and is called deep similarity. Context specifies how close retrieved items are, i.e. it can be perceived as a measure of usefulness. Thus, context in context-based similarity is useful during the process of judging how relevant the returned answer is to the current goal.

Jurisica and Glascow (1997, 1998) proposed an algorithm that is based on a notion of relevance assessment and on a modified nearest-neighbor matching. Its modifications include: (1) Grouping attribute into categories of different priorities so that different preferences and constraints can be used for individual categories during query relaxation; (2) Using an explicit context, a form a bias represented as a set of constraints, during similarity assessment; and (3) Using an efficient query relaxation algorithm based on incremental context modifications. The goal is to retrieve not only exact matches, but partial matches (similar cases) as well. In short, context is a parameter of a relevance relation, which maps a case base onto a set of relevant (in terms of context) cases. There are several factors affecting performance of the algorithm (Jurisica and Glascow, 1997): the size of a case base (measured in terms of the number of cases in the case base); the size of a case (measured in terms of an average number of attributes used to describe cases); and context (measured in terms of the number of attributes defined and constraints specified); the query complexity (measured in terms of the size of a query and the complexity of the operations required); and the relaxation/restriction strategy used.

6. REPRESENTATION BY CONTEXTUAL GRAPHS[2]

6.a Introduction

Initially, we have use a rule-based representation to describe incident solving (Pasquier, 2000). This representation was not adapted to operators, and we have then chosen a tree representation as presented in Figure 8. This part of the work has been made with L. Pasquier (Pasquier, 2000; Pasquier *et al.*, 1999, 2000; Brézillon *et al.*, 2000a, 2000b) in the framework of the SART application.

Decision trees inspire our tree representation, but they differ namely on two points. First, our trees have no decision node, only “chance” nodes where a contextual element is analyzed to select the corresponding path. Second, There are no probabilities.

[2] The work presented in this section has been developed with Laurent Pasquier in the framework of his Ph.D. Thesis.

Hereafter, we present the way in which we have transformed our tree representation into a graph representation, with: (1) the introduction of macro-actions, (2) the transformation in graph representation, and (3) the development in contextual graph.

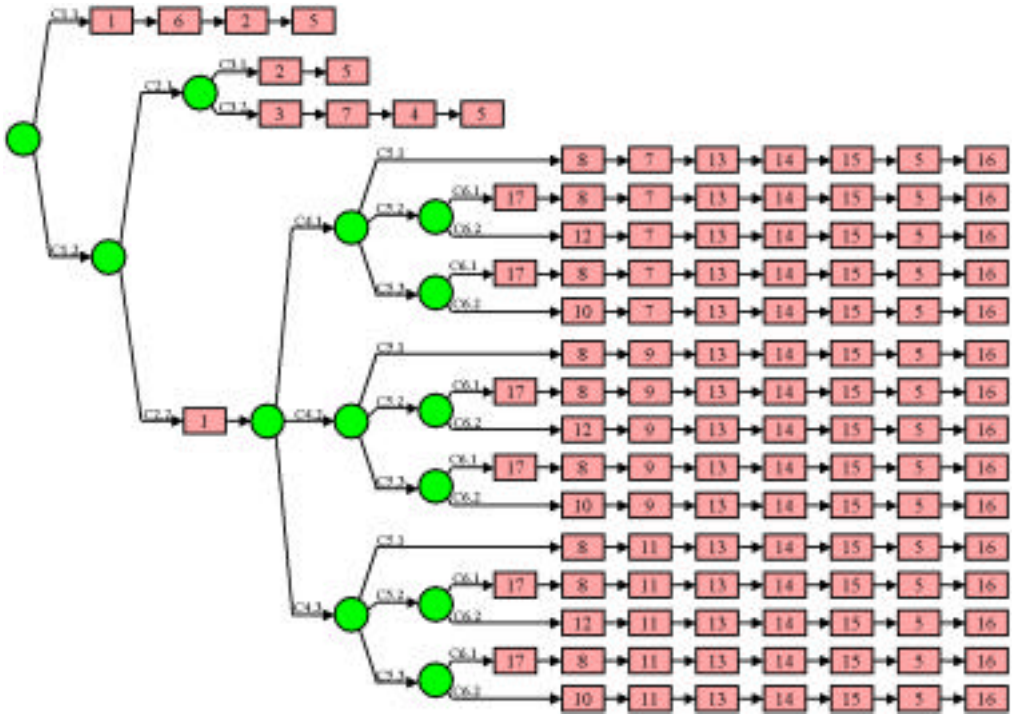

Figure 8: A tree in which many actions are postponed to the end of the branches

Some Actions (A) and Events (E) in case of incident on a metro line are given in the following table.

Actions	
1	Residual traffic regulation
2	Damaged train continues with travelers
3	Damaged train continues with travelers until a steep incline
4	Damaged train restarts without travelers
5	Stable damaged train at end station
6	Repair damage
7	Exit of the travelers out of the damaged train
8	Exit of the travelers out of next train
9	Exit of the travelers out of damaged train via available cars
10	Exit of the travelers out of next train via available cars
11	Exit of the travelers out of damaged train via track
12	Exit of the travelers out of next train via track
13	Next train joins damaged train
14	Link both trains
15	Convoy return to end station
16	Disassemble convoy
17	Next train goes to next station

Events	
C11	Immediate repair possible
C12	Immediate repair impossible
C21	Enough motor coaches available
C22	Not enough motor coaches available
C31	No steep incline between damaged train and end station
C32	Presence of steep incline until end station
C41	Damaged train at station
C42	Damaged train under tunnel
C43	Damaged train partially at station
C51	Next train at station
C52	Next train under tunnel
C53	Next train partially at station
C61	Presence of a station between damaged train and next train
C62	No station between both trains

6.b Particularities of our tree representation

Without entering details (presented in a following section), macro-actions are a way of proceduralizing the contextual knowledge and introducing modularity into the diagnosis process by managing different modules that accomplish the same function in different ways according to the context. However, action postponement is not always possible, and it is preferable to try to prune the decision tree in some situations (Brézillon, Pomerol, and Saker, 1998). In figure 4, several macro-actions existing on different branches can be identified (e.g., A6-A7-A9). Such macro-actions are a kind of compilation, originating from experience, of several actions. In this compilation, a part of the knowledge on each action becomes implicit in the proceduralized context. This is close to Edmondson and Meech's view (1993) on context as a process of contextualization. However, to explain a macro-action (and thus the whole reasoning involved), an operator needs to decompile the macro-action to retrieve the rationales. Such an operation is not always easy, especially when experience comes from a previous generation of operators.

As we have seen, it appears that, for many decision makers, the most urgent need is, on the one hand, to reduce the number of events to envisage and, on the other hand, to undertake robust actions and, if possible, macro-actions. At that level, contextual knowledge enables the uncertainty in the decision making to be reduced. However, some contextual knowledge may be very far removed from the incident solution. For example, all the paths in Figure 8 conclude with the same action (except one): evacuate the damaged train to the terminus, and not evacuate the damaged train on a secondary line. The reason is that the workshops for repairing trains in Paris are in the terminus. (The strategy for the metro in Rio de Janeiro is different because workshops are in the middle of the lines.)

We arrive now to a main point of our representation. Each event at an event node carries on a part of the uncertainty of the situation. For example, C1, in previous figures, means that either an immediate repair would be possible or not. There are two ways to manage this uncertainty either assess some probabilities for each events, which is the usual view in decision theory or consider that, anyway, the two events are possible depending on the circumstances. Our graph must provide an answer for any possible circumstances whatever the probability is. The problem of an operator is to get an adapted answer as soon as he knows what the circumstances are. Thus, the main problem is to diagnose the exact situations according to the contextual information reaching the operators. For this reason we considered that each branch, actually describes a contextual knowledge, which becomes more and more accurate as long as the branch is followed. We are no longer interested by the probability of a branch but by the possibility to determine, as soon as possible, on which path the operator is, to determine what is the next action to undertake. In other words, we come back to the original idea of Savage (1954), namely that each nature state (a sequence of events is a nature state) describes a state of the world, or using our words, a context for action.

At this step there is no probabilities on the events, the main purpose is to describe, with the maximum of parsimony, all the possible contexts in which the decision has to be made. For example, the branch with {C12, C21, C32} describes a context in which there is no immediate repair possible, but enough motor coach power is still available and a steep incline. For this reason we will talk hereafter of context nodes instead of event nodes and of context instead of nature's state. The action postponement observed in Figure 8 amounts to relating each decision to a state of nature, here the context described in the branch. Thus, our representation tends to stick to operators' behavior. In some cases, especially at the beginning of the tree, making decision under uncertainty is probably interesting, but by trying to gather relevant information the operators endeavor to make decision under certainty. This means that when undertaking an action in the tree they consider that, due to the contextual information they got, the state of nature between the root and the action undertaken is the true state of nature.

Under this interpretation and with the modification implied by our action-driven view, decision trees are changed into contextual graphs.

6.c From tree to graph representation

First, we reduced the number of objects in the structure by replacing repeated sub-sequences of actions by a single object called **macro-action**. The choice made for defining the different macro-

actions is based on common sub-procedures known by the operators, such as “linking trains,” “return to end-station without travelers”… The principle of replacing is the following:

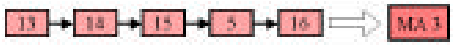

Figure 9: From a sequence of actions to a macro-action

By using macro-actions, the tree representation in Figure 8 is simplified as represented on Figure 10.

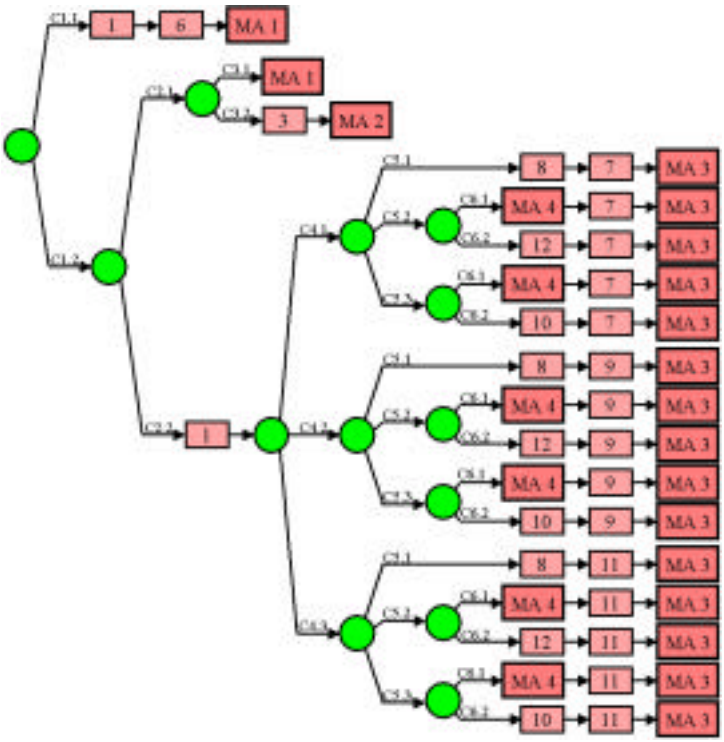

Figure 10: A tree representation with macro-actions

If this replacement simplifies the lecture of the tree, it does not reduce the structural issue of the tree. Secondly, there is a **merging of the branches** of the tree as soon as the sequences leading to the end of the incident are similar, as shown in the figure 11.

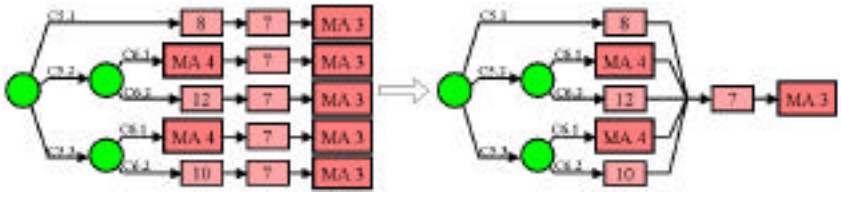

Figure 11: From tree to graph representation

Macro-actions		Actions lists
MA 1	Damaged train continues service	Actions 2 and 5
MA 2	Damaged train stops service	Actions 7, 4 and 5
MA 3	Make a convoy with damaged train and next train	Actions 13, 14, 15, 5 and 16
MA 4	Empty next train at a station	Actions 17 and 8

Cognitively speaking, this amounts to use a scarcity principle that leads the operators to try to reuse well-known procedures as soon as possible. This operation has several main consequences on the structure of the representation and on the meaning of the model.

1. We no longer face a tree but a graph. This graph is oriented without circuits, with exactly one root and one goal (because operators have only one goal and branches express only different strategies, depending on the context, to achieve this goal). The graph structure moreover allows extending of the representation.
2. The size of the structure is now under control and the consideration of a new contextual element will add some elements in the graph, but not increase drastically its size.
3. The change of the structure introduces a dynamics comparable to the dynamics of the change between the proceduralized context and contextual knowledge. Indeed, when two branches are merged, it means that the undertaken actions led to a common situation from different contexts. The contextual elements attached to the different branches are proceduralized at the diverging node. They stay in this state for the different action sequences, because they intervene in the branch decisions. Finally, they are deproceduralized when the branches are merged. By this way we explicitly express the life duration of the contextual elements (see Figure 12).

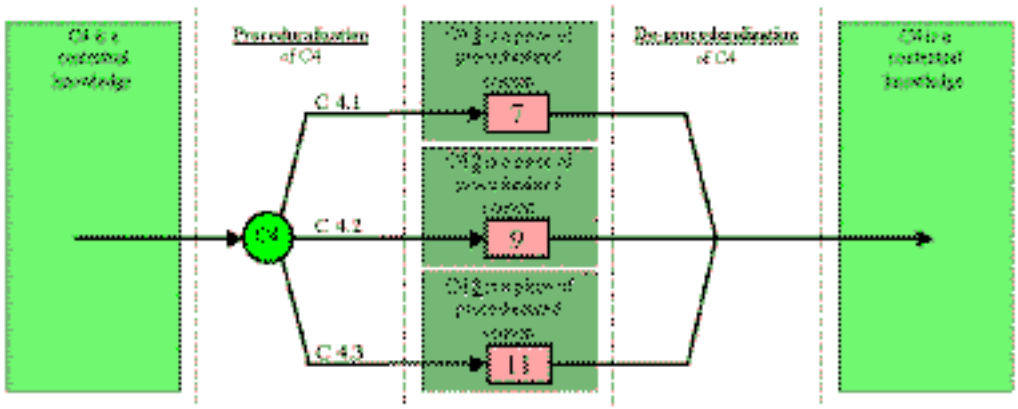

Figure 12: Proceduralization and de-proceduralization

4. Contextual graphs give a unified representation and practices as discussed in previous sections. Moreover, they permit naturally a learning mechanism for integrating new situations by assimilation and accommodation. Here the learning mechanism incorporates in the contextual graph a new incident solving based on its context. It looks for the most similar incident solving known and recombines the known elements to express the new incident solving.
5. The sub-graphs, associated with a name (operators know the corresponding procedures and can name them) and a goal, are similar to schemes identified by the cognitive ergonomists as discussed later. These structures may evolve respecting two rules. First, one structure can be duplicated and adapted to another action. For example in the Figure 12 the scheme "*Helping train emptying*" is derived from the scheme "*Damaged train emptying*" and adapted by the introduction of the fact that an available train may run to the next station, if there is not another train at this station, to evacuate its travelers if necessary. The second possible evolution of the structures is the update of the acting rules by combination with a new action sequence achieving this goal. We thus obtain a set of contextual graphs that interact one each other according to a given context for an incident solving. We worked on the possible algorithms for this combination (Niang, 1999). These algorithms will be tested as soon as the whole structure will be coded.

Moreover, even in a part of a subgraph it may happens that the actions are only partially ordered (see the example above of two trains on the same line which have to be cleared of the travelers whatever the order of the operations). For representing this issue, we introduced the **temporal branching** symbols to represent action sequences that can be done in different order. This symbol is made of two parts: a divergent branching and a convergent branching; the parallel branches

wear the temporally independent decision blocs. Finally we obtain the following structure (Figure 13) that we call **contextual graph**.

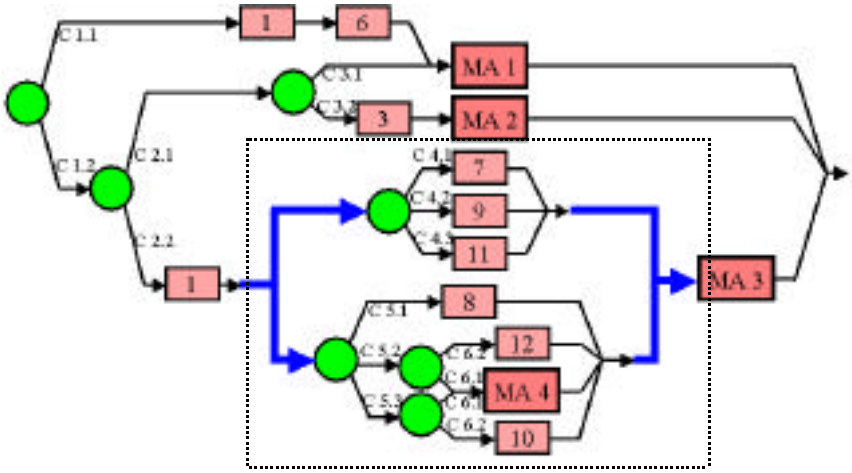

Figure 13: Contextual graph representing the official procedure for "lack of trains power" incident on Figure 8

This structure is called contextual graph to recall that it makes explicit the context and its dynamics for decision-making. This representation is more compact than trees and seems to be well accepted by the operators. As our initial trees were not decision trees, these directed acyclic graphs are not influence diagrams. They simply represent the succession of actions to do to solve an incident; the different possible paths express the possible strategies according to the situation. We must also mention that our representation is much simpler than colored Petri-net (Humphreys and Berkeley, 1992) but it has not the same expressiveness as regards the dynamics because the process of proceduralization de-proceduralization must "linearly" follow the left to right reading of the graph.

We already have studied the solving of different incidents. As a result, macro-actions give a simple picture of decision graphs and permit to point out similarities and possible shared parts between the solving of incidents quite different. Operators and managers at RATP have well accepted our modeling, mainly because of the clarity and understandability of the representation: it is easier to understand a set of decision graphs than the corresponding decision tree, when both of them have the same mode of investigation. Different objects are coded to permit the handling of macro-actions and the first results are very interesting (we code in C++). The methods actually implemented concern the basic operations (creation, change, visualization, and file management) and establishment of links among objects. Several procedures are also coded. Operators will have the possibility to modify the structure of a macro-action (i.e., a procedure) to model their own strategy for solving an incident. However, each procedure and the associated strategies will have the same structure, and there will have as many items as strategies used by operators. Concretely, a strategy is a copy of a procedure applied for an incident in a given context.

6.d Contextual graph representation of the reasoning

A contextual graph is a directed acyclic graph that represents the actions to undertake according to the context. The action nodes represent actions to undertake to achieve a goal while the event nodes become as explained above, contextual nodes describing the possible contextual issues of a given. The contextual graph is intended to represent the part of the context that we have denoted proceduralized context, i.e. context chunks ready to be used for action.

The proceduralization of the contextual knowledge is a process that makes explicit the links, especially the causal and consequential links, between contextual knowledge chunks and as such the links become a part of the proceduralized context. Thus, the proceduralized context appears as a kind of compiled knowledge that the system will have to decompile to explain its reasoning. Consequently, one can regard the above contextual graph (Figure 13) as representing the proceduralized context, because for each context represented by a sequence of contextual nodes the implicit reasoning about causes and consequences implies that the action to undertake is defined without ambiguity. In other words, each sequence of contextual nodes along a branch triggers some actions due to rationales which are not represented on the graph but are generally

known and compiled in the operators' mind (proceduralized knowledge). Thus, the contextual graph explicitly represents the reasoning involved in the proceduralized context.

The evolution from a decision tree to a contextual graph is not only a graphical simplification. The purpose of contextual graphs is not to make a decision under uncertainty but to represent along each branch a state of nature that the operators try to identify. The operators endeavor to work with the certainty of having diagnosed the right branch. The chaining of the different contexts along a path figures the evolving proceduralized context. Thus, in a contextual graph, the proceduralized knowledge evolves continuously. In such a dynamical domain (subway line control), it is fundamental to represent accurately the dynamics of operators' reasoning.

Another problem of modeling by trees concerns parallel sequences of actions. In our application, the order in which some actions are executed may be indifferent. For example, when a train must push a damaged train, both trains must be empty but the order in which the trains are emptied is not important and mainly depends on the context in which each train is. As the contextual chunks of knowledge are too numerous to be exhaustively considered, we have introduced a new type of branching in contextual graphs, called temporal branching. A temporal branching figures several branches that may be followed independently and in any order. Without such a representation we would be obliged to multiply the number of paths, one for each possible ordering.

The temporal-branching structure is shown in the slashed square in Figure 13. One can see the opening and closing square bracket in dashed lines between action 1 and macro-action MA3. Both branches are composed of a contextual sub-graph. The upper one tells how to empty the damaged train, the lower one shows how to empty the helping train. The actions of both sequences are locally and independently carried out. To detect such temporal branching in real applications, one has to detect sequences of actions that may actually be indifferently carried out in an order or another. For example, if you see that sometimes the operators decide to do a sequence "A-B" and sometimes the sequence "B-A," one can suppose that actions "A" and "B" are independent, but must be accomplished before a given step. Such situations are easy to detect automatically from records.

In (Pasquier, 2000, Pasquier *et al.*, 2000), we show how a series of changes leads from a decision tree as represented in Figure 8 to a contextual graph (Figure 13). These changes are:

1. We identify sequences of actions that appear on several branches and replace each of them by a macro-action (e.g. "assembling trains" is a macro-action),
2. Then, we merge the branches of the tree as soon as the sequence leading to the end of the incident is identical,
3. We introduce the notion of temporal branching for representing action sequences that can be executed in any order; and finally,
4. We identify sub-graphs that appear in the contextual-graphs representation of different incident solvings.

Figure 13 is the contextual-graph representation of the tree representation given in Figure 8. Such a representation by contextual graphs/sub-graphs is very similar to the generic tasks proposed by Chandrasekaran some years ago (Chandrasekaran *et al.*, 1992).

6.e Sub-graphs and contextual graph

In Figure 13, the structure in the slashed square has its own signification. It can be thought as an independent plan named "assistance to a damaged train." This plan has a goal (to push a damaged train with another train up to the end-station) and an explanation about the way to carry it out. This explanation results in the proceduralization of the context contained in the contextual sub-graph shown in Figure 13. This sub-graph could be found again in other contextual graphs. Each sub-graph can be considered for its own, by operators as well as by designers. This is a chunk of knowledge that can be considered as an elementary piece of knowledge. We have elsewhere argued that, on the one hand, such an elementary sub-graph can be considered as an atomic task or a scheme of action (Pasquier *et al.*, 2000). On the other hand, our representation is reminiscent to Sowa's conceptual graphs (Sowa, 1984, 2000) with their mechanisms of aggregation and expansion. Macro-actions are then considered as a particular case of action structure in which the contextual graphs are reduced to sequences of actions not intertwined with context nodes. However, one must notice that our representation is simpler than Sowa's one because our graph is oriented and the path gives the reasoning followed by the designer according to the different contexts.

The block “train aid” in Figure 14 is found in the solving of different incidents. Such a block corresponds to the right level of operator’s interpretation of events and the type of utterance between the operator and drivers of the concerned trains because all of them use the same language (i.e. they are all able to decompile the action “train aid” in elementary actions).

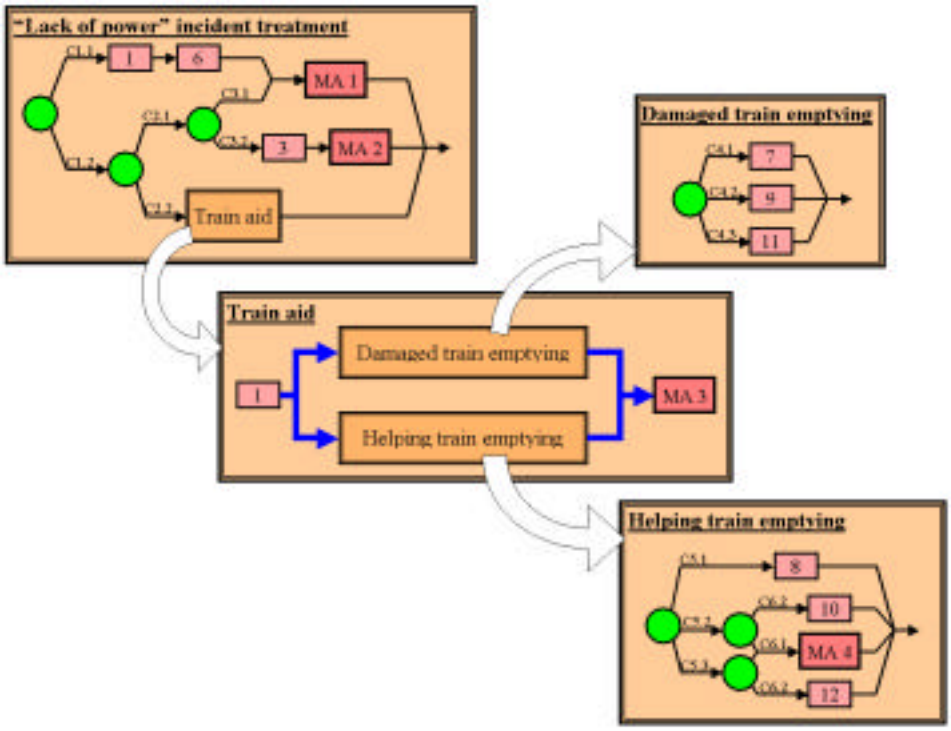

Figure 14: Set of contextual graphs used while "lack of train power" incident resolution

This contextual graph recalls that the structure makes explicit the context and its dynamics for decision-making. This representation is more compact than trees and seems to be well accepted by the operators. As our initial trees were not decision trees, these directed acyclic graphs are not influence diagrams. They simply represent the succession of actions to do for solving an incident; the different possible paths express the possible strategies according to the situation. We must also mention that our representation is much simpler than colored Petri-net (Humphreys and Berkeley, 1992) but it has not the same expressiveness as regards the dynamics because the process of proceduralization de-proceduralization must «linearly» follow the left to right reading of the graph.

6.f Learning in contextual graphs

Consider the following sequence of actions from the operator that must be compared to the contextual graph in Figure 13:

The system begins by identifying the part of the contextual graph that is concern by the action sequence (see Figures 15), The system thus associates the actions 1, 17 and 18. However, the identification stops here because the system knows that the operator is at the step to make empty the trains (i.e. the damaged train and the aiding train). In the given action sequence, it appears however that the operator has skipped this step. Asking the operator for his reason, the system learns that the train that is concerned by the action sequence to identify is already empty because it just leaves the terminal. For integrating the new alternative (branch) in the contextual graph, the system introduces a new contextual node to check the state of the train and skip the actions to make the train empty. The operator explaining why, the system learns by doing. There is thus a learning that is an incremental knowledge acquisition at the level of the contextual graph.

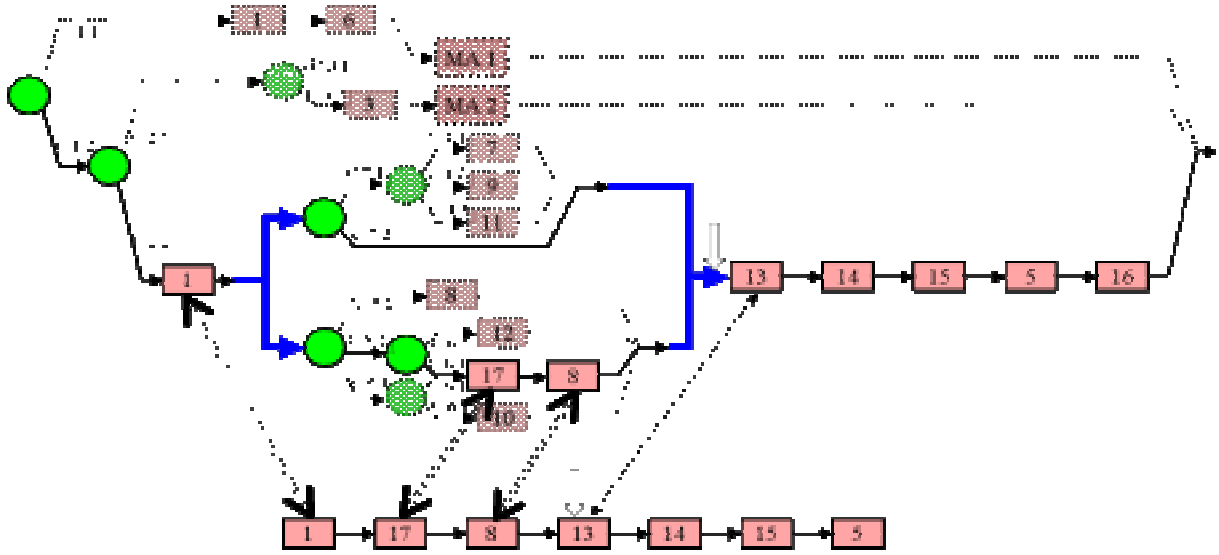

Figure 15: Evolution of a contextual graph by learning

6.g Explanation generation and contextual graphs

Most of researchers focused on explanations considering them as a transfer of knowledge **from** the system **to** the user. Feedbacks are used by the system to tailor its explanation to user's needs. Thus, users rarely might intervene in the generation of explanations. Conversely, the approach taken in SEPT (Brézillon, 1991) lets the user build alone his explanation. This was not a better solution than the previous one: users must tackle complex commands that are not always compatible with their work and temporal constraints. The lesson learned is that the user and the system must cooperate to solve jointly the problem and to co-construct an explanation for the solution, explanation generation being an intrinsic part of the task at hand.

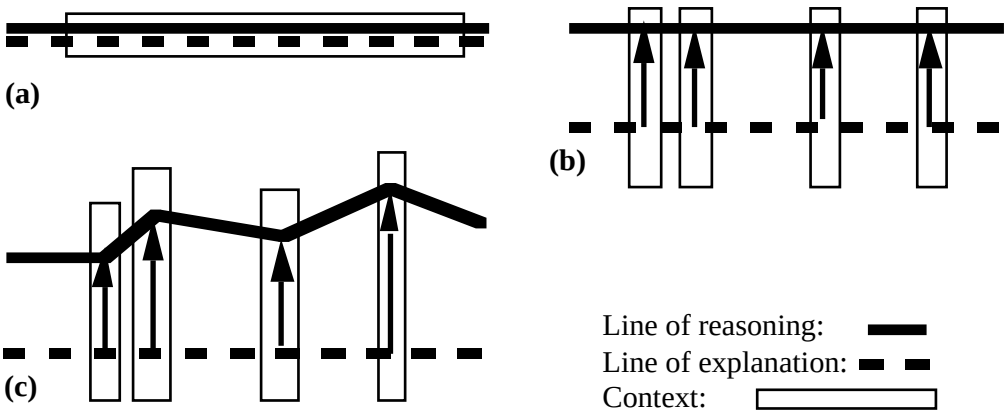

Figure 16: Line of reasoning versus line of explanation

As represented in Figure 16-a, MYCIN, and all earlier expert systems, align their explanation facility directly with the reasoning paths that define movement across contexts of the diagnostic system. Thus, all traces of reasoning that represent the traversed contexts are kept and their contents provided to the user for explanation. Here, the definition of context is restricted to knowledge and reasoning of the problem solving. MYCIN is not selective in its construction of explanations.

In other systems, such as that described by Wick (Paris *et al*. 88; Wick and Thompson, 1992), an explanation facility is aligned only periodically with the reasoning of the system (see Figure 16-

b). In Wick's system, only some of the contexts that the system reasons with are explained to the user. In this approach, additional explanatory knowledge (knowledge on the domain and the expertise that are not directly necessary for the task at hand) may be used to generate enhanced explanations. This implies that the explanation path separates from the path of reasoning to produce effective explanations. Context is here an extended version of the previous one because it also contains domain and task knowledge not directly considered in the reasoning of the problem solving, and eventually some information on users through a model. One problem with such an approach is that it may be unsuitable for critical applications whose results may affect the safety of processes and people.

Another approach for explanation is to accept that the reasoning of the system is often different from that of the user as in the case of SEPT. Thus, the user and the system may have different interpretations on the current state of the problem solving. The differing interpretations will be compatible if the user and the system make proposals, explain their viewpoints and spontaneously produce information (Karsenty and Brézillon 1995). In order to align the system's reasoning with that of the user and vice versa, the user and the system must co-construct the explanation in the current context of the problem solving. People who are trying to understand something often may offer an explanation that embodies their current understanding, expecting to have it corrected (Mark, 1988). Thus, explanations become an intrinsic part of the problem solving and the line of reasoning of the system may be modified by explanation (see Figure 16-c). This leads to cooperative problem solving. Again, context here is an extended version of the context in the previous approach because it also integrates direct information from users, mainly on the basis of their actions on the system and on the real-world process.

Mackie (1965) already stressed the context-dependence of explanation as a process of making a distinction between some current situation and another class of situations. Thus, context--involving both explainer beliefs and goals-- is crucial in deciding an explanation's goodness, and a theory of contextual influences can be used to determine which explanations are appropriate. Deciding explanations' goodness makes sense only in context of what triggered explanation and how the resultant explanation will be used. This leads to view explanation as primarily a process of hypothesizing causes of events and expectation failures, rather than deriving them from known factors.

Many theories of explanation evaluation were based on context-independent criteria. Such theories either restrict consideration of explanation towards a fixed goal, or assume that all valid explanations are equivalent. However, explanation can serve a range of purposes that place widely divergent requirements on the information an explanation must provide. Leake (1991) argues that understanding what determines explanations' goodness requires a dynamic theory of evaluation. His claim is based on analysis of the information needed to satisfy the many goals that can prompt explanation. A way to address the problem is to generate explanation from a given viewpoint. Making context explicit permits to highlight a few important factors from the many that are causally involved: those that give the explainer the information it needs, that are good explanations. In order to judge the goodness of an explanation, we need to know how it will be used, and what information that use requires.

Leake (1992) considers the relationships between explanation and context in the framework of case-based reasoning. An explanation is required when there is a conflict between an event and a model that we have of the place where occurs the event. Leake argues that such a conflict is a property of the interaction between events and context: Any particular fact can be anomalous or non anomalous, depending on the situation and on the processing we are doing. To be relevant to an anomaly, explanations must resolve the belief conflict underlying the anomaly. To resolve an anomaly, the information in an explanation must account for why prior reasoning led to false expectations or beliefs. Any anomaly vocabulary would allow retrieval of explanation for identical anomalies, provided that the same anomaly was always described the same way and that distinct anomalies always received distinct characterization.

Leake implemented his ideas in the system ACCEPTER. ACCEPTER characterizes anomalies to index into explanatory information stored in memory. The facts that ACCEPTER retrieves during routine understanding, and the expectations based on those facts, become the context in which new inputs are understood. ACCEPTER focuses its explanation effort on filling current gaps in its knowledge. It maintains conflicting families of beliefs by placing them in hierarchical understanding contexts, which are used to maintain a tree of alternative world models; by

evaluating competing explanations in different contexts, ACCEPTER avoids interactions between their beliefs.

Turney (1996) distinguishes three types of features: primary, contextual and irrelevant. Primary features are useful for classification even when they are considered in isolation , without regard for the other features. Contextual features are useful for classification only when they are considered in combination with other features. Irrelevant features are not useful. For example, when classifying spoken vowels, the primary features are based on the sound spectrum. The accent of the speaker is a contextual feature. The color of the speaker's hair is irrelevant. The three types of context of Turney are close to our distinction between external knowledge, contextual knowledge and proceduralized context.

Moore (1995) presents a view similar to that of Leake in Natural Language. She states the explanation problem as follows: given a communicative goal, find information from the expert system's knowledge sources that is relevant to achieving this goal, and organize this knowledge into a coherent multi-sentential text. In order to allow the explanation to be responsive to the user's feedback and sensitive to previous explanations, the system must be able to reason about its own previous responses, for example, to interpret the user's follow-up questions in the context of the ongoing conversation and to determine how to elaborate or clarify a response when necessary. New information must be related to what the information seeker already knows to provide missing information in a way that facilitates understanding and learning.

Moore's approach employs rhetorical strategies as compiled knowledge about what actions may be used to satisfy certain intentions. Explanation strategies thus enable systems to generate a range of different explanations from the same knowledge representation (Carenini and Moore, 1993). Information about focus is inferred from the dialogue history, using two pointers into its structure: *global-context* and *local-context*. *Global-context* indicates the current topic under discussion. It points to the place in the dialogue history where this topic was begun. *Local-context* points to the most recent thing said. It always points to the node in the text plan representing the last clause uttered by the system.

Always in the area of Natural Language Generation, Pereira and Pollack (1991) present a system, called Candide, to incrementally interpret natural-language utterances in context. Context-independent and context-dependent aspects of an interpretation are separated. The context-independent part of the information that is invariant with respect to further incremental interpretation is separated from the context-dependent part that may vary. Candide's discourse model has three components. The most readily accessible information concerns entities referred to in the “current” utterance, the one that is undergoing interpretation at any point in time. Detailed information about those entities is encoded in the first component of the discourse model, which is called the immediate context. The immediate context is used primarily for resolving intra-sentential anaphora and for making modification decisions that depend on sortal information. The second component is the local context that contained detailed information about slightly less-accessible entities, generally those referred to in the immediately preceding utterance. The local context is primarily used for pronoun resolution. The third component of the discourse model is the global context, which contains somewhat less detailed information about entities referred to throughout longer stretches of the discourse. The global context is employed primarily for the resolution of definite anaphora, and is structured as a stack to make use of the theory of focusing. Global, local and immediate contexts are close to our three types of context.

Karsenty (1994) argues that communication implies the sharing of a linguistic code and a context. The context is a set of information pieces that are accessed or built to give a meaning at a message. Thus, explanation is a means to share the context that is needed for the actor understanding. Its aim is to differentiate between an initial context, in which an information is not understood or misunderstood, and a target-context, in which the information becomes comprehensible. It is a way to make explicit the implicit knowledge in a procedure and interpret a new information. Explanation generation acts as a contextualization process (Edmonson and Meech, 1993; Karsenty and Brezillon, 1995), and manages the interaction context. Explanations are a type of validation of the context. Conversely, the explicit use of context permits explanations to be tailored to a specific request. It is the context that supplies any explanation needed to validate suggestions (Karsenty and Falzon, 1992).

Explanation and context are strongly intertwined. Making context explicit permits the tailoring of explanations to a precise need, to decrease the amount of knowledge required for the exchange between the user and the KBS, to show the coherence of an explanation and rapidly reach an agreement between the user and the system. Explanations permit the context to be explicit in order to understand a step of the reasoning and shared knowledge and experience. They are a means to point out the links between the problem at hand and shared knowledge on its current state.

The way in which an explanation must be chosen and produced depends essentially of the context in which are the two actors in the explanation generation. The sensitivity of “knowledge” to context becomes evident when one attempts to implement an automated explainer that chooses between alternate explanations (Suthers, 1993). An explanation always takes place relative to a space of alternatives, which require different explanations according to the current context. Comparing two explanations leads to see how their contrast spaces differ. This gives us a measure of dislocation between two explanations and a basic presupposition of the explanation context. Thus, an additional piece of structure--the explanation context--is necessary to explain how explanations function (Garfinkel, 1981; Lester and Porter, 1991).

Context appears to guide the interactions among actors and search for suitable explanations. Interaction is seen as navigation in context space. For instance, Huuskonen and Korteniemi (1992) propose to follow different steps for producing an explanation accounting for the context:

- Find context (context can be inferred from previous explanations, assuming that the user likes to ask more on the same subject);
- Find question: questions matching with the context are displayed, and the user chooses one among them;
- Refine context if necessary;
- Find answer and show it.

With such a method, it seems that the user must accept easily the idea of conversation through context refinement. According to this view, producing acceptable explanations means identifying mutual knowledge of specific events, objects and contexts and relating explanations to the explainee's personal characteristics and the interaction history (Johnson and Johnson, 1992). Using context permits a kind of “cognitive coupling” between the user and the system.

If it seems acceptable that explanations intervene in the evolving context of interaction, it is difficult to say more about this for two reasons. Firstly, the co-building of explanations is an accepted idea but rather very few studies consider it. Secondly, context being not a mature domain of research, its dependency upon explanations is not really considered. For example, Lester and Porter (1991) propose a model of explanation generation that includes simple methods for representing and updating context. However, their model makes assumptions about the representation of the context, not about how it is inferred.

Contextual graphs are a type of unified framework in which it is possible to associate explanation, learning and incremental knowledge acquisition that considered as the minimal requirement for any cooperative system.

7. CONTEXT AND RELATED NOTIONS

7.a Context and schemata

Scripts are a form of schemata structures (Bobrow , 1975; Schank and Abelson, 1977; Rumelhart 1980). They represent the elements shared by similar events or experiences (such as been in certain kinds of rooms, or going to children's birthday parties, or eating in restaurants). In scripts, specific details of events are dropped out and the common features between similar experiences are retained. Thus, scripts contain organized sequences of stereotypical actions. Scripts are one of the main structures used to explain the organization of episodic memory, jointly with other knowledge structures that have been proposed (scenes, MOPs, MetaMOPs, and TOPs). Scripts can eventually be organized hierarchically in subscripts.

An episode, also called an experience, is like a specific visit to a dentist, or a particular occasion going to a restaurant. Elements of episodes, which are identical, are treated as a unit, a script, like [Doctor Jones' waiting room] script, or a standardized script [dentist's waiting room]. Scripts are collections of specific situations, organized around common parts of similar episodes, i.e., a script is formed with common situations among similar episodes, built over time, by repeated encounters with those situations.

Bower *et al*. (1979) discussed the segmentation of scripts into low level action sequences called scenes. Scenes organize a number of scripts, or parts of them. Abstracting and generalizing from multiple experiences creates them. A scene is generally organized by a context, the context of the relevant setting. Scenes capture generalities when scripts provide the specifics. For example, [waiting room] is a typical case of a scene formed by scripts embedded as standardization of various general scripts. In this case, scripts about visiting different professional consultants (doctors, lawyers, financial advisors, etc.), or any other activity that have in common a similar procedure for [waiting room].

Memory Organization Packets (MOPs) are used as organizers of scenes. They organize events by controlling the sequence in which the scenes occur in the events. A MOP consists of a group of scenes directed toward the achievement of a goal. MOPs provide information about how a number of scenes are related to one another. Thus, a generalized scene is a decontextualized description of a setting and the activities in pursuit of a goal relevant to that setting.

In artificial intelligence, Minsky (1975, 1981) proposed another schemata-like structures, called frames. Minsky (1981) presents the essence of the theory as follows: ``When one encounters a new situation (or makes a substantial change to one's view of the present problem), one selects from memory a structure called a frame. This is a remembered framework to be adapted to fit reality by changing details as necessary. A frame is a data structure for representing a stereotyped situation, like been in a certain kind of room, or going to a child's birthday party.'' Attached to each frame are several kinds of information. Some of this information is about how to use the frame. Some is about what can one expect to happen next. Some is about what to do if these expectations are not confirmed. We can think of a frame as a network of nodes and relations. The top level frames are fixed and represent things that are always true about the supposed situation.'' Frames are intended mainly for the representation of concepts, by grouping together sets of attributes. They are an extension of the idea of representing concepts through semantic networks (Collins and Loftus, 1975). It is then possible to regroup sets of frames in arbitrarily complex forms to model or express the facts concerned with daily aspects of the world.

Frames represent declarative and procedural information in terms of knowledge represented and the expectations related to it more in a functional way rather than a structural way. They organize the knowledge that they represent according to the function of that knowledge. A frame consist of a collection of knowledge slots, which contain the attributes associated with the represented concept or event (see Bayle, 1988). Slots can have procedures (self-contained pieces of code) attached to them for the purpose of performing operations on the slot values when a particular context is identified. Some of the information in a frame is also about how to use the frame, what to do if the expectations are realized (or not realized).

Scripts, frames and relatives provide a static description of the world (Ramirez, 1997). In such representations, the dynamics is given by at the assembling stage, i.e. before the use of these items. In the same way, these representations do not deal correctly with context, only through an implicit coding. By abstracting specific details and representing only stereotypical items, scripts are very close of the representation of the procedures established by companies, procedures in which contextual considerations are not retained. Conversely, contextual graphs permit to represent procedures, but also all the variants of the procedures that deal with contextual cues (our practices). As a limit case, an elementary contextual graph can be compared to a script. A scene could be compared to a more complex contextual graph (i.e. composed of elementary sub-graph), except that a scene gives a static organization of sub-graphs in a contextual graph, but does not represent a dynamic organization of the sub-graphs. Moreover, there is no possibility to add new paths in scenes as in contextual graphs.

7.b Context and patterns

The notion of pattern has been introduced first by the architect in building Christopher Alexander in 1977 (Alexander *et al.*, 1977). He observed that each model describes a problem that recur in

the environment and thus describes the heart of the solution to this problem in a way that permits a reuse of the solution without to do it twice exactly in the same way. Now, patterns have roots in many disciplines, and most notably in works on urban planning and building architecture. Appleton (2000) presents different definitions of a "pattern":

- An abstraction from a concrete form which keeps recurring in specific, non-arbitrary contexts.
- A recurring solution to a common problem in a given context and system of forces.
- A named "nugget" of instructive insight, conveying the essence of a proven solution to a recurring problem in a given context amidst competing concerns.
- A successfully recurring "best practice" that has proven itself in the "trenches".
- A literary format for capturing the wisdom and experience of expert designers, and communicating it to novices

The intend behind the construction of patterns is that they will serve both as a means of documenting and describing common interactions, and as a vehicle for communicating the results of a specific analysis to designers. They describe the high level concepts that are used in a chosen domain (Wallace, 2000). This is especially true in ethnography studies where the interest is in the possibilities surrounding the use of patterns as a means of organizing, presenting and representing this growing corpus of ethnographic material (Martin *et al.*, 2001). Patterns are powerful because they avoid being too precise and too formal all the time (Chalmers, 2000). Their descriptions are often independent of any programming language or implementation details. Thus, patterns are an informal representation based on the features that are significant to the programmer's everyday job. They express how a software component is used and how it acts and fits with other components' uses and actions when designing a larger structure.

More specifically in design, patterns contain the result of years of experience, collaboration and refinement (Schmidt *et al.*, 1996), not just abstract principles or strategies. A design pattern is a description of a problem and its solution. A pattern should document the problem, its solution and the consequences of using it. This permits to compare alternative solutions with full awareness of the consequences of each alternative. Design patterns capture the static and dynamic structures and collaborations of successful solutions to problems that arise when building applications in a particular domain. Design patterns are similar, at least in the spirit, with our notion of context when we consider a 3-uple (problem, context, solution) as represented by contextual graphs.

In Artificial Intelligence, the lack of consideration of patterns could be a reason of the failure of the first expert systems (Brézillon and Pomerol, 1996a). An expert possesses specialized situation patterns for his domain of expertise. These patterns have been built with experience, and generally expressed as rules. Generalization consists of building disjunctive chunks, while specialization consists of building conjunctive chunks. In machine learning, a computer system learns by connecting new information to patterns that it already understands. Among the main failures pointed out, there are the fixed patterns defined out of their context of use, and the impossibility to evolve dynamically.

The terms patterns and pattern languages are an attempt to describe successful solutions to common software problems from a decade of field studies (Schmidt, 1995). By definition, patterns capture experience, and parts of this experience appear in different problems as building blocks in artificial intelligence some years ago (Chandrasekaran, 1992). A collection of interrelated patterns provide a vocabulary for talking about a particular problem. A pattern language defines a collection of patterns and the rules to combine them into an architectural style. Pattern languages describe software frameworks or families of related systems. The term pattern language does not correspond to a programming language. The pattern language is like a document that guides and informs the designer in the use of several patterns. The pattern languages are a set of patterns, classified by categories, intimately linked together by identical contexts (Lea, 1997).

Some of the most useful patterns describe, in an abstract way, frameworks and, thus, facilitate widespread reuse of software architecture. Similarly, frameworks can be viewed as concrete realizations of patterns that facilitate direct reuse of design and code. One difference between patterns and frameworks is that patterns are described in language-independent manner, whereas frameworks are generally implemented in a particular language. They can be viewed as a concrete reification of families of design patterns that are targeted for a particular application-domain.

A number of patterns are described in the literature. Roth (2001) presents mobility patterns that are derived from successful mobile applications. Martin *et al.* (2001) present patterns of cooperative interaction that support the development of general design principles drawn from a range of work settings. In all this, the long term goal is to develop handbooks for software engineers.

Thus, patterns help people to reuse successful practices (a successful solution is considered as a recurring solution to a standard problem, the meaning of practice being different of those of practice when contrasted with procedure previously): Patterns are a step toward handbooks for software engineers. People are concentrating on documenting the key patterns that successful developers use, but that relatively few developers thoroughly understand and consistently apply in their daily work. Degler and Battle (2000) propose a synthetic view of this as presented in Figure 17.

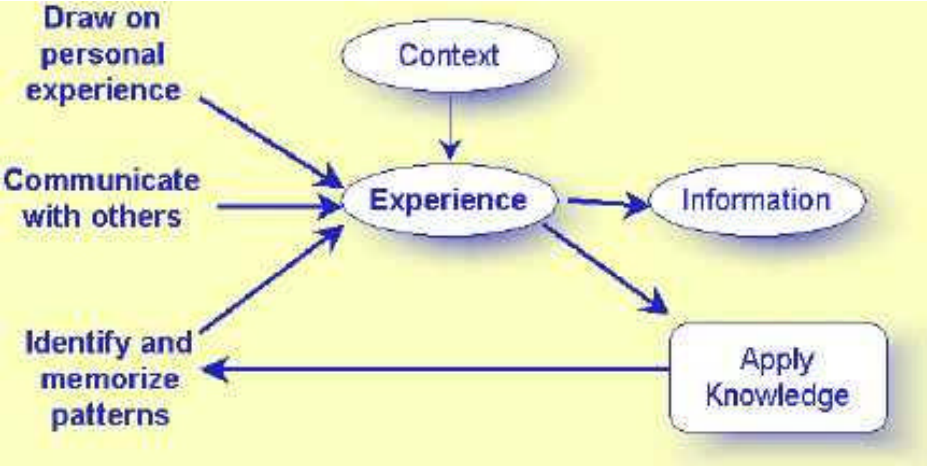

Figure 17: From (Degler and Battle, 2000)

A pattern is a proven solution to a problem in a context. Here, context refers to a set of recurring situations in which patterns are applied. Alexander *et al.* (1977) quoted that every pattern is formulated as a rule which establishes a relationship between a context, a system of forces which arises in that context, and a configuration that allows these forces to resolve themselves in that context. Thus, patterns are a way to contextualize knowledge.

A strong parallel can be lead between contextual graphs and patterns, although that what is made explicit in contextual graphs seems to be rather implicit or informal in the pattern paradigm. First, both of them capture experience. This point is particularly important when we refer to our discussion about procedures and practices. There is a clear need to express the different ways to reach a given solution. Second, both of them try to make explicit static aspects as well as dynamic aspects of a problem solving. However, in contextual graphs we try to use the dynamic aspects when patterns only give a representation of them with weak means to use it. Third, both of them possess an organization in items/sub-items that appears as a kind of language. In contextual graphs the organization present the same organization that permits a reuse of some sub-items, but contextual graphs permit also the generation of explanation at different levels of detail. Both of them give a description above programming language. With respect to the notion of context, contextual graphs use it explicitly when patterns and frameworks introduce it implicitly. Globally, patterns help designer to develop a computer system, when the contextual-graph formalism is to permit the computer system itself to reuse the experience.

7.c Context and the Unified Modeling Language

The manual V1.3 of UML (2000, Eriksson and Penker, 1998) presents a sub-activity state as an activity graph. When a sub-activity state is entered, the activity graph nested in it is executed as any activity graph. The sub-activity state is not exited until the final state of the nested graph is reached, or when trigger events occur on transitions coming out of the sub-activity state. Since states in activity graphs do not normally have trigger events, sub-activity states are normally exited

when their nested graph is finished. A single activity graph may be invoked by many sub-activity states, and may be executed more than once concurrently. In the latter situation, the number of concurrent invocations is determined at runtime by a concurrency expression, which evaluates to a set of argument lists, one argument list for each invocation. The different aspects of the UML are generally discussed in the literature on the example presented in Figure 18 (UML 2000).

UML let the possibility to represent actions to execute in parallel in any order before a given action. Thick bars that act to synchronize transitions represent this. A fork starts 2 or more threads of control. A join will allow the threads to synchronize. In Figure 18 the actions "put coffee in filter" and "add water to reservoir" are independent actions that must be all accomplished before the action "Turn on machine"? This situation is quite similar to the temporal branching introduced in our contextual-graph formalism as shown in Figure 13.

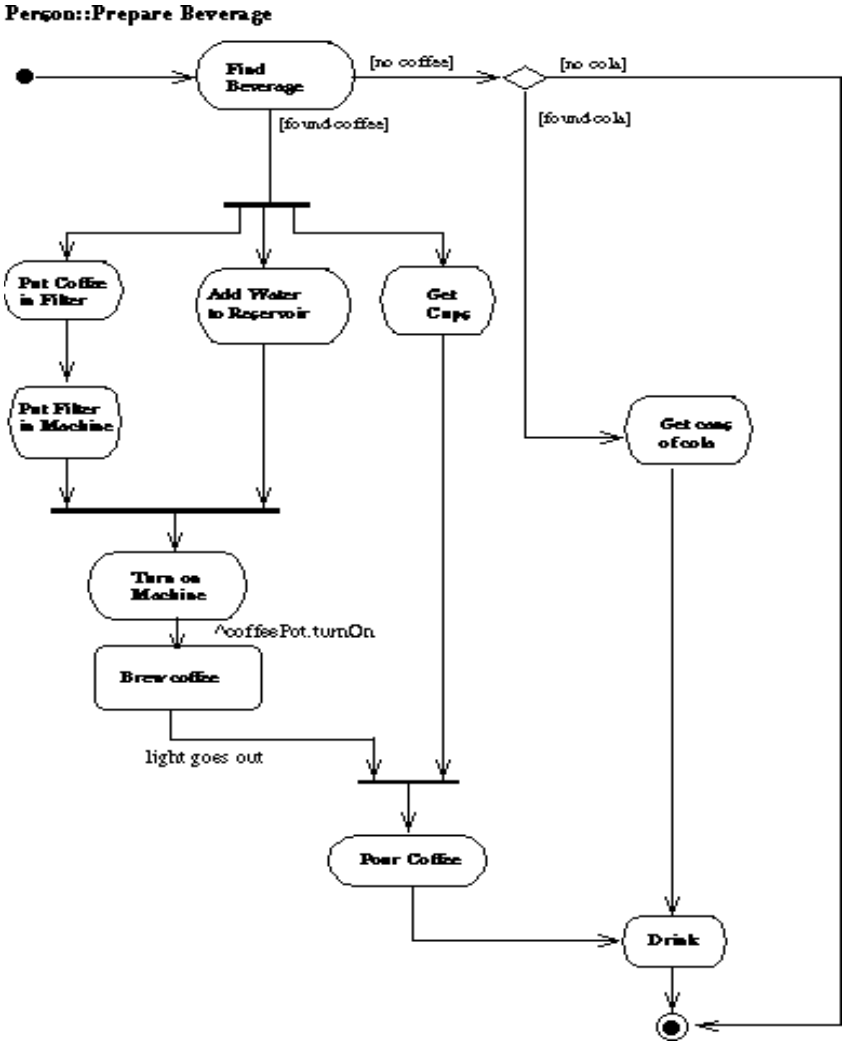

Figure 18: The example discussed in papers on UML (from URL8)

A use case describes a way in which a real world actor - a person, organization, or external system - interacts with an organization. Each use case describes the sequence of events of an actor (human or system) using the system to complete a process. This description views the system as a black box. That is, the internal design and functionality of the software are not captured.

Typically, use cases represent business processes. They capture key uses of the system and hence, are an excellent means of communicating requirements. It provides a natural high-level view of the intended external functionality of the system that is understandable by engineers and non-engineers alike. A use case is a generic description of an entire transaction involving several objects. It can also describe the behavior of a set of objects, such as an organization. A use case

model thus presents a collection of use cases and is typically used to specify or characterize the behavior of a whole application system or a part of an application system together with one or more external actors that interact with that system. An individual use case may have a name (although it is typically not a simple name). Its meaning is often written as an informal text description of the external actors and the sequences of events between objects that make up the transaction. Instances of this behavior may be formally specified using scenarios, but iteration and conditionally within scenarios is usually best expressed as informal text.

The goal of use case analysis is to describe the system's behavior according to how the user will use the system, but the use cases alone won't do. To analyze system behavior, we need to add scenarios. While the use case is a general description of an action, the scenario is specific (in the spirit of our distinction between procedures and practices). A use case says "Dispatcher dispatches driver" but the scenario indicates all of the steps the driver took in delivering the package, as well as who the driver is, what kind of truck the driver drove, and who the recipient is. Thus, a potential scenario is: “The dispatcher instructs the driver to get a packages at 10:00 and deliver it to the recipient at 2:00. The package is C.O.D. and the person who signs for the package is not the person it is addressed to.” There are a number of scenarios for a use case, so one must choose to analyze the scenarios that will best elucidate the operation of the system. There is a parallel to lead with our previous discussion about procedure (similar to use case) and practices (similar to scenarios).

The Unified Modeling Language (UML) possesses some notions, as activity graph, use case and scenario that are quite similar with our approach based on contextual graphs. Notions as activity graph, use case and scenario that are quite similar with our approach based on contextual graphs. A first similarity relies on a structured description of a problem solving. Activities are organized in sub-activities when possible, and in contextual graphs, some sub-graphs are found in different tasks of higher levels. An advantage of this approach in contextual graphs is to provide naturally learning and explanation capabilities in the system. Both of them, also authorize a representation of temporal branching. Beyond these shared properties, contextual graphs also have learning capabilities and possibilities of reorganization of the memory that are not considered in UML in which the description of the problem stays static. Again, the notion of context that is made explicit in contextual graphs stays implicit in UML.

7.d Context and scenarios

Design also uses a notion of scenario but in a slightly different sense than in software engineering. The defining property of a scenario is that it projects a concrete description of activity that the user engages in when performing a specific task, a description sufficiently detailed so that design implications can be inferred and reasoned about (Carroll, 1995). Central to most scenario based design is a textual description or narrative of a use episode. Stories are often about atypical situations; they are about events that are exceptional in some way (Erickson, 1995).

A scenario is a narrative that describes someone trying to do something in some environment (Karat, 1995). The use of scenario is a natural and efficient way to capture end users' needs in their context. By using a narrative it is possible to capture more information about the user's goals, and the context the user is operating in. The context might include details about the work place or social situation, and information about resource constraints. This provides some more help in understanding why users do what they do. In much current design work the users goals and context are often assumed implicitly, or may not be taken into account. As such, it is a description of a context, which contains information about users, tasks, and environment. The scenario is described from the user point of view and may include social background, resource (e.g. disk space, time) constraints and background information. Scenarios seek to be concrete; they focus on describing particular instances of use, and on a user's view of what happens, how it happens, and why (Carroll, 1995). When stories are concrete accounts of particular people and events, in particular situations, scenarios are often more abstract -- they are scripts of events that may leave out detail of history, motivation, and personality (Erickson, 1995). Each scenario can be expanded into a set of causal relations between elements of the design and specific consequences for the user's activity and experience (Carroll, 1995).

By using a narrative it is possible to capture more information about the user's goals, and the context the user is operating in. Context provides some more help in understanding why users do what they do. In much current design work the user goals and context are often assumed

implicitly, or may not be taken into account. The scenario helps to expose the complexity of the working context that is being designed, and revealed some of the organizational tradeoffs implicit in developing new automation (Dearden *et al*., 2000).

The scenario becomes a design object or artifact and may be augmented and rearranged as the design evolves. It can become a hypothetical interaction scenario for a new design and allow better understanding of the new design. It is also desirable to maintain a history of past scenarios as a way of capturing past design rationale. In one sense, scenarios are not really new in design activity. It's extremely common in design to imagine "what if" situations, or to walk through a design in ones mind or in a group.

Dearden *et al.* (2000) use scenarios to ensure that allocation decision are sensitive to the context in which the system will be used and by insights from economic utility theory. Thus, the attention of decision makers focuses on the relative costs and benefits of developing automated support for the activities of the scenario, the relative impact of functions on the performance of the operator's primary role on the relative demands placed on an operator within the scenario.

7.e Context and affordance

Affordances are what objects or things offer people to do with them. The theory of affordances states that all the information necessary for correct perception is already present in the environment. Gibson first introduced this theory in 1966 (Gibson, 1977, 1979). His definition states that: "the affordance of anything is a specific combination of the properties of its substances and its surfaces taken with reference to an animal." In other words, by looking at objects, people perceive their affordances and not their physical qualities (e.g., size, color). Gibson's affordances are relative to animals (and can only be measured in ecology), invariant, holistic, jointly determined by the environment and the organism. They are the allowable actions specified by the environment coupled with the properties of the organism.

In a given environment, two aspects of affordances are considered: (1) affordances are jointly determined by the environment and the organism, and, (2) affordances are relationships or properties of relationships, actions, perceived properties, and mental constructs.

People give meaning to places based on their interactions with them (Jordan *et al.*, 1998). Places provide a context for everyday action and a means for identification with the surrounding environment. The concept of place integrates both its location and its meaning in the context of human action. Jordan *et al.* (1998) suggest the following aspects for an affordance-based model of place:

- Physical features: Places consist of collections of objects.
- Actions: People perform actions in places, actions being one of the most important aspects that give meaning to a place.
- Narrative: Stories help to characterize the uniqueness of a place by revealing the past actions of others.
- Symbolic representations/Names: Certain places are referenced by symbols having symbolic and/or mythical meanings.
- Socioeconomic and cultural factors: For example, black is the color of mourning in the west, whereas in China it is white.
- Typologies: Categorization represents an important mental strategy for dealing with complexity and new situations.

Moreover, affordances depend on perceiver’s intention. Gibson (1979) described affordances as the process of perception as the extraction of invariants from the stimulus flux. Here, the role of the perceiver is to pick up the available information rather than to construct a percept from cues (Allard, 2000). Experimentally, subjects of different heights perceived stairs as climbable or seat depending on their own leg length, as opposed to some extrinsically quantified value (e.g., 18 inches, 2 feet). In the act of sitting down in a chair, there are at least four separate affordance-related concepts involved (Zhang, 2000):

- the affordances proper: the seat of the chair is horizontal, flat, extended, rigid, and approximately knee-high off the ground, all relative to your own proportions and position,
- your perception of these properties, the surfaces, distances, areas, textures, relationships between parts, and so forth,

- the mental interpretation you derive from the perceptions,
- the act of sitting itself.

Note that the mental interpretation is derived in the framework of the perceiver's intention: Instead of sitting on the chair, the perceiver can look at the chair for putting a foot to tie up the shoe.

Affordances for planning environments apply equally well to user interface environments (Saint Amant, 1999). For example, a number of different affordances in user interface can be proposed:

- Icon size (St Amant, 1999): Fitts' law tells us that increasing the width of an icon will reduce the time to reach it, thus reducing the cost of establishing the pointer-over predicate.
- Sticky icons (Worden *et al.*, 1997): When the pointer is over an icon that is sticky, the gain of the mouse is reduced, requiring an increased effort for the user to move the pointer off the icon.
- Haptic feedback (Munch and Dillman, 1997): On predicting the target of a user's movement, one implementation of a haptic mouse activates an electromagnetic field to stop the mouse on the target (and thus the pointer over the icon), eliminating the possibility of overshooting and the need for detailed adjustment.
- Capture effects: If a mouse down is detected in the neighborhood of a selectable icon, a system can interpose an additional movement that puts the pointer over the icon, effectively defining an area of "magnetism" around it. Selection of narrow or small objects such as line segments in graphical drawing interfaces often rely on such a mechanism.

Kuhn (1996) applied the theory of affordances to spatialized user interfaces. Affordances of physical space are being mapped onto abstract computational domains through spatial metaphors in order to bring human-computer interaction closer to people's experiences with real-world objects. Jordan *et al.* (1998) present a methodology to model places in GIS area with affordances. Their conclusion is that the integration of affordance-based models of places into future GIS will lead to a better communication between users and systems." It is our task to build affordances into the environment that facilitate the activities intended by our design" (Saint Amant, 1999).

Another important point is that affordances are about actions, and thus exclusively concerned with plan execution and, to some extend, to the decision making process. A real affordance describes the relationship between action and the physical environment, but the decision maker adjusts external data to fit internal knowledge and representations that are a kind of schemes (Tijus, 2001). Thus, the action can be related directly by perception, e.g. as one turns the head when one hears a violent noise.

In the relationships between a person and a system, the notion of affordance covers a large spectrum of items, in the one hand, concerning the place as the environment, the agent, the task, their relationships, and, in the other hand, actions, mental constructs and perceived properties. Making context explicit here permits to identify clearly the right affordance in a given situation.

Norman (1988) proposed the notion of perceived affordance as "tells the user what actions can be performed on the object and, to some extent, how to do them." When a user sees a printer icon, s/he immediately thinks of the function of a printer - to print out a hard copy of whatever the user is intending. The icon becomes a perceived affordance by associating the function of the printer with the printer icon. The concept of perceived affordance encompasses not only physical objects, but certain types of text that "offer" or "afford" clicking to jump from one place to another. For example, the convention used is that links are underlined and in blue, visited links are underlined and purple, and active links are underlined and in red. This convention is very well known in the Internet culture and a designer who deviates from this concept risks users being frustrated and annoyed.

The perceived affordance is, in our interpretation, the contextual knowledge and the proceduralized context considered jointly. When we want to print a document, the printer icon is similar to the proceduralized context, and its associated function the contextual knowledge.

7.f Context and scheme

The notion of scheme was proposed first by Kant around 1800, with an emphasis on its temporal dimension (Eco, 1997). Schemes are a kind of collection of tape-recorded thoughts and actions which human beings use (or replay) to interact with the world and to solve problems. Piaget

(1936), working on learning during the childhood, defined the notion of scheme as organized patterns of actions or thoughts used to represent the world. In Piaget (1973), schemes as what is transposable, generalizable or differentiable from one situation to another one. In other words, they are what is common in the different repetitions or applications of a given action. Schemes are instruments of adaptation capable of differentiation, accommodation to new facts, and of self-coordination. Schemes do not constitute a complete library of actions ready to use. Rather, they are a "frame" from which can be applied assimilation and accommodation, the latter intervening when a situation totally new necessitate an original action. The difference between assimilation and accommodation is illustrated on Figure 19.

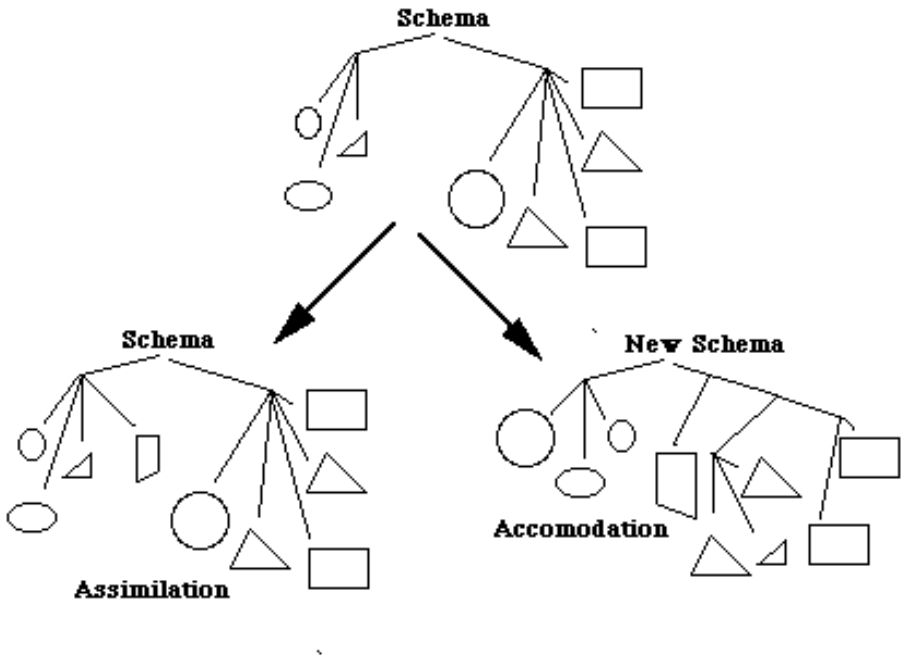

Figure 19: Example drawn from Danmac[3]

Moreover, it seems to have a close analogy between the notions of Piaget's scheme and Bourdieu's habitus (1972, cited by Perrenoud, 1976). The habitus is understood as a system of dispositions durables and transposables integrating all the past experience. The notion of habitus can be used at each moment as a matrix of perceptions, appreciations and actions, and thus make possible the accomplishment of infinitely differenciated tasks.

The notion of scheme plays an important role also for structuring operators' activity: Béguin (1994) for drawers, Galinier (1996) for truck drivers and Duvenci-Langa (1997) for workers on tool machine, have all identify schemes of activity. A scheme is composed of a structure of actions, but also other things as the means used to accomplish the actions. Eventually, an action structure may be composed of several elementary schemes. Vergnaud (1985) describes a scheme as a dynamical organized totality containing four categories of elements:
- operational invariant items (objects, properties, relationships and processes),
- acting rules for guiding and create the action,
- inference or calculi (kind of reasoning),
- predictions, which are goals or intermediary steps.

Conversely to a schema, a frame or a script, a scheme possesses a dynamical dimension: a scheme is triggered by a change in the current context when a new event occurs. A context change implies a change of the scheme, and the change of the scheme of a given task may concern the task itself or the tool that is used to solved the task. Again, this point shows some similarities with our previous discussion about procedures and practices.

Schemes evolve respecting different rules:

---

3 http://www.phy.nau.edu/~danmac/kuhnpiaget/KP1.html

- A scheme may be updated as a set of procedures that are relevant according to the singularities of the given situation.
- A scheme may be enriched by addition of new strategies. Following Piaget (1936), Rabardel (1995) discusses the addition by assimilation and accommodation. Assimilation involves putting information into an existing scheme without changing the scheme. Accommodation is the process of changing our existing schemes in order to create new ones suitable to the new information or situation.
- Schemes may be build by adaptation of existing schemes to a new one. For example, you use to work with a tool (you have constructed a scheme for using this tool) and you discover a new tool based on the same idea. You try to use the new tool like the old one (this does not work exactly the same way) and thus you create a new scheme derived from the old one integrating the differences between both tools.

This notion of scheme is different from the notions of scenario or script proposed by Schank and Abelson (1977). For example, each event in a script follows each other in a fixed manner. There is also a difference with the notion of schema or Memory Organization Packets, MOPs (Schank, 1982). Again, all the information on the way an entity (scenario, script, MOP) behaves is coded initially in the entity in a static way. Thus, these entities are invariant structures of activities and actions. They do not address the problem of the representation of an activity in a context evolving dynamically. Conversely, a scheme is not applied directly, but instantiated depending on the specific context of the current situation.

Contextual graphs seem to us a computer expression of schemes, some for solving problems, others for completing sub-goals. Each scheme has a name, a goal and a contextual graph representing the decision-making that permits to achieve its goal depending on the context. Each scheme organizes the activity around an object and can call other schemes to complete specific sub-goals. A contextual graph, as a scheme, permits:

- to represent and organize clearly operators' activity and all its variant (procedures and practices),
- to include automatic learning and adaptation capabilities in a system, and
- to make context explicit in the representation of operators' reasoning.

The notions of scheme (mainly scheme of action) and our contextual graphs are quite similar. The SART decision support system uses the contextual-graph representation in association with case-based reasoning, respecting three main modes. The first mode updates the databases used by SART according to a new incident declaration and description. The two other modes are mainly databases interrogations, but differ in their principle. One of these two last modes helps the operator when a new incident occurs. This mode must gather a maximum of information automatically and propose well-adapted solutions. The third mode is a support system for experiencing and training. The two later modes are based on the following reasoning. Given a scheme base, an incident and its context description, the system proposes several possible solutions to this problem. First it selects the scheme corresponding to the resolution of this type of incident. Then, for each contextual element encountered in the associated contextual graph, it determines if this element is known or not for the current incident. If so, it selects only the corresponding branch. Otherwise several policies are acceptable: either it selects all the branches (this presents to the operator all possible strategies in this situation), or it selects the most often followed path, or it follows the path closer to the official procedure. It continues the path selection up to the end of the contextual graph and return the path(s) found. It presents the possible sequences of action, representing the integrated schemes as expandable actions. If the operator wants precision on one of the integrated schemes, he asks the system to zoom on it. This mechanism looks like aggregation and expansion of Sowa's conceptual graph (1984, 1991, 2000).

8. CONTEXT AWARENESS

8.a Introduction

In parallel to AI community, mainly interested by computational aspects of context, another community emerges since few years. This community is more oriented towards the practical use of context through data (location and time). One goal of this community is to support person mobility and the different ways to support it. Such applications are called context-aware applications (CAS, and, to some extend, mobile, ubiquitous and pervasive computing, and smart devices too) and are developed in domains, say, as different as tourism and e-maintenance. In the

tourism example, the idea is to provide tourists with real-time information, context being here mainly tourist's location and request time. Means for interaction between tourists and the system can be mobile phone, PDA, fixed, information dispenser or FM radio. Weaknesses of these applications are threefold: (1) context is mainly limited to location and time: information on the tourists (e.g. from a model of the tourist) is generally ignored, (2) there is no consideration for the dynamic dimension of the context, and (3) works in this area are only equipment-based (e.g., the video-tie).

For example, in HCI, a definition is that a context feature is any information that can be used to characterize and interpret the situation in which a user interacts with an application at a certain time. In context-aware applications area, Dey and Abowd (1998) define context as any information that characterizes a situation related to the interaction between humans, applications and the surrounding environment.

From an *informational* perspective, context aims to provide an information space that can be "adapted according to the user's context (modulo group context)", and complemented by tools/processes that promote the socially situated construction and sharing of context (Fitzpatrick and Bruza, 2000). A fact is not meaningful in itself, but acquires meaning in the context of someone using it, in an account of what happened today. From a *device* (albeit in-use) perspective, context concerns how devices can be made to "blend naturally into the normal interactions and physical space of human work practice as they gather, integrate and display information that assists work." From an *infrastructural* view, context provides a computing device with information about its environment as provided by other system components. As a consequence, different 'types' or 'dimensions' of context emerge, e.g., physical context (ambience), computational context, informational context modified by notions of individual and group (information ecology), situated context. In the example of tourism, the objective is to provide tourists with real-time information. Information concerns the environment in which tourists are (from inside a building to a whole geographical region), the moment at which they send their request, external factors (e.g. weather and events in progress), and above all tourists' interest. Such information is called contextual information because it does not intervene directly in the tourist's request but constrains the answer that will be made.

Such context-aware systems have an efficiency rely heavily on contextual information (e.g. tourist's preferences and weather) they take into account explicitly. For example, a context-aware system can propose the visit of a castle at 10 mn by foot from where is the user. The user indicates to the CAS that he is tired or the CAS checks the weather and identifies that is just beginning to rain. The CAS then proposes an alternative by bus, with where and how to have a ticket, where is the station, at which stop to leave the bus, etc. This points out the dynamic dimension of the context and a need for an active wake state of the CAS during the accomplishment of the task by the user.

A context-aware intelligent environment is a space in which a ubiquitous computing system has contextual awareness of its users and the ability to maintain consistent, coherent interaction across a number of heterogeneous smart devices. Specifically, the CAS community recognize four primary methods of utilizing context information: Resolving references, Tailoring lists of options, Triggering automatic behaviors, and Tagging information for later retrieval.

Moreover, context-awareness implies two attributes of a system: the ability to obtain context, and the ability to utilize contextual information. Obtaining a context-aware system supposes to tackle seriously several key elements that we present in the next section.

8.b Context-aware or context-based?

In 1969, Frederik Pohl wrote a science fiction novel called "The Era of the Satisfactor". In the story, each person possess a mobile device--called Satisfactor--by which the person can order and buy a product, pay something in a store, have news, communicate with another person, command the TV channel, control what the children are doing, and even can receive (or send on someone) some chemical drugs (e.g. a relaxing drug). As a corollary, if a person loses his satisfactor, this person is anymore a person recognized by the society, cannot buy food, etc. This is a novel of science fiction. However, some research is in the realm of this story: identification of a person

entering a room, location of a person from his cellular phone, paiement through internet, etc. The focus is now on mobile devices, and the hardware aspect of such devices would be soon under control.

More than for devices, the acute identification of the context is crucial for mobile device. In the later case, most of the parameters stay constant, when in the latter case parameters can be different, for a same mobile device, from one place to another one, and at the same place from one instant to another. For a given mobile device in a given environment, some external events (e.g. the rain) can modify the current context and makes non-optimal the current plan, and a need for re-planning the actions.

Now, it is possible to specify a fixed context for a mobile device, and, eventually, to take into account some changes in the environment. This concerns data and partially information from the environment. Accounting continuously for knowledge coming from the environment or the user is the next step along which we present some challenges.

Some challenges to explore are:
- To provide users with a first approximation of environmental trends and events;
- To point out useful information implicit in large volumes of data to alert users of sudden changes;
- To develop multiple scenarios and perspectives on a given line of actions;
- To attract users' attention upon existing and emerging strategic issues;
- To support users in sharing and communicating their views and perspectives;
- To guide users' attention to specific data and the interpretation of the accessed data in relation to particular issues.

Hereafter, we discuss in more details three topics in which context plays a leading role.

Dealing with individual contexts and interaction between users

We have shown in previous sections that making context explicit allows data, information and knowledge to be structured in order to facilitate the user's navigation (e.g., at each time, present the minimum number of relevant items). However, a computer system can do more by being a real intelligent assistant system: the system can "learn by interacting". Thus, the system becomes increasingly familiar with the user, and his way to accomplish his activity.

From an observation of operator's actions to accomplish a task (e.g. using shortcuts or not), the system can accompany the user (attentive wake state) in the accomplishment of his task by using anticipation means (e.g. forecast). For example, when a tourist is in a street, the system can warn and propose alternatives, by recall of memo with respect to the location (buy bread before to go home), in a foreign country by automatic changes for currency or translation. Moreover, having support the operator in the accomplishment of a task may help the system to support the user in the accomplishment of other tasks.

Here a system will have to make compatible the context of the task (as context-aware applications) and the context of the user (as in artificial intelligence). A goal could be to present an ordered list of alternatives with explanations to the user. This is not science fiction: Schmidt *et al.* (2000) present a study in which a system assists a user wishing to give a phone call to a friend. The system presents the list of his friends with their availability before the call itself (e.g. Tom does not want to be disturb, Mary is already in communication and John has put his voice recorder). Then, the user makes himself the compromise between his wish to contact the person and the availability of the person. Dourish (2001) claims that context-aware computing must integrate a large set of different perspectives like this one.

Beyond the support that the system can bring to a user by managing his personal context in relationship to the working context, a system can also reuse its experience with a user when it interacts with other users taken individually or in a collaborative work. The former case is similar to the case considered by Maes' team for which an agent interacting with a user can require a support from other agents with other users to help it to solve a particular problem. The latter case is more general and concerns different aspects of cooperation as negotiation. However, it seems to us that an important point by making context explicit is the adjustment of all users' contexts to make compatible their interpretations and understandings on a given problem (Karsenty and Brezillon, 1995), even with radically different viewpoints as specialists in the building of a spacecraft or a family of tourists in a city with different objectives.

Granularity of the context

In context-aware applications, a particular credit is given to sensors in the capture of the context to understand events in the world. (Here, context is mainly defined by data--location and time.) Context-aware computer systems have not to capture only the local environment (immediate context), but also the far context (e.g. forecasting for the weather for example). There is then a problem of context granularity.

The context granularity is a kind of density measure that can be used as a function of the distance to the focus of attention. Such a view allows a person to address local questions like, "Where is the closest mail box?", but also more global questions like "How many planets are between Sun and Earth?" Fisheye views (Furnas, 1986; Sarkar and Brown, 1994; Pook *et al.*, 2000) are one way of integrating the context and focus into a single view. Some of the information surrounding the focus is shown following the rule: the greater the distance of the information from the focus, the more interesting it must be for it to be shown. Thus, it is possible to view local details and global context simultaneously. Note that a general transformation would allow global information about the graph to affect a view. However, we are yet far from such a granularity representation that varies continuously with the distance to the focus of attention.

Now, one finds the management of a local context in reference to a global context. The literature on context-aware systems distinguishes two types of context: (1) the "local" context that is close of the focus of attention and highly detailed, and (2) the "distant" context that is general (with less details). In the spirit of Sowa's conceptual graph (1984, 1991, 2000), one can say that all the concepts are expanded in the local context, and aggregated in the distant context. This is a fisheye strategy providing a balance between local details and global context. Local detail is needed for the local interactions with a structure. The global context is needed to tell the user what other parts of the structure exist and where they are. Global information may also be important even in the mere interpretation of local detail.

This approach is also found in other domains. For example, van Dijk (1998) presents a similar position on political discourses with:

- À local or micro context (often also called situation), defined by a specific setting and specific participants, and
- A global or macro context, informally defined in terms of higher level societal structures, involving, e.g., groups, group relations (such as power and inequality), organizations, institutions or even whole states and nations.

The notion of granularity can be extended to user's context, especially in situation of collaborative work. It is a way to manage different users' context (far or global context) and the collaborative work context (the local context). There is a link too with our context-based representation of the knowledge based on the onion metaphor where contextual-knowledge pieces are ordered in layers around the current focus of attention.

Context as a managing tool for the informational mass

Methods of introducing deliberate distortion in order to show a large amount of contextual information in a given amount of screen area are collectively known as Focus+Context views. A basic idea with focus+context visualizations (very close indeed from the fisheye view) is to enable users to have the object of primary interest presented in detail while having an overview or a context available at the same time (Bjork and Redstrom, 2000). A description of these techniques is given as "focus+context start from three premisses: First, the user needs both overview (context) and detail information (focus) simultaneously. Second, information needed in the overview may be different from that needed in detail. Third, these two types of information can be combined within a single (dynamic) display, much as in human vision" (Card *et al.*, 1999).

For various reasons, we are now increasingly concerned by the management of data, information and knowledge in large and heterogeneous bases. The keystone here is clearly, first a distinction between data, information and knowledge, and, second, a dynamic organization of the data, information and knowledge in bases for a given focus of attention at a given moment. Making context explicit would help for such a dynamic organization of the knowledge in the memory (a kind of dynamic conceptual schema for a database) for (1) addressing a request in a given context, (2) include a new piece of information/knowledge by learning (especially useful for representing procedure and its practices), and (3) Retrieval/storage of items in the bases.

The design and development of a context-aware system supposes:

- Access to large databases updated in real-time (e.g. social events, weather);
- An efficient context-based representation of a user model, data, information and knowledge;
- Communication through mobile phone, PDA, fixed information dispensers or FM radio; and
- An efficient association of software and hardware constraints.

Clearly, this can only be the result of a cooperative work with abilities in different technical domains.

Context representation supposes a dynamic organization of the knowledge in the memory. This is necessary for giving learning capability to a system, and for making memory organization dependent of external conditions, dependent of its context of use.

Each approach considers the context from a different viewpoint when we compare them with the distinction data/information/knowledge: The Context-Aware Systems (CASs) deal essentially with data, when Context-based Intelligent Assistant Systems (CIASs as in the SART application) focus mainly with knowledge. Figure 20 gives a picture of this position.

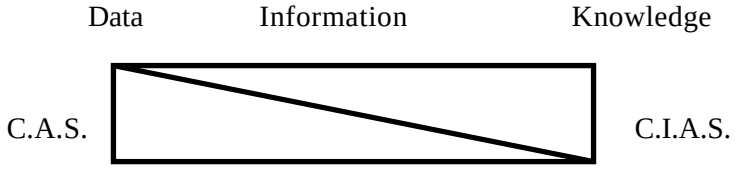

Figure 20: From CAS to CIAS through data-information-knowledge

The innovative aspect of our proposal concerns different consideration on context: (1) the modeling of the context (including a maximum of contextual elements), (2) the representation and storage of data, information and knowledge related to context, (3) the retrieval of the relevant data, information and knowledge in the context of the user's query, and (4) the presentation of the data, information and knowledge in the user of the user. Globally the proposed approach is a user centered approach accounting for preferences, language, financial possibility, the number of persons in the group, his/her interest for the History or stories, etc. In that sense, a context-aware approach will aim at developing social sensitive applications. We thus share Dourish's position (2001) with (1) embodiment about establishing meaning, and (2) Meaning arising in the course of actions.

9. CONCLUSION

The two questions of contextual knowledge and knowledge sharing have received wide consideration in recent years. In this paper we have addressed these two topics through Intelligent Assistant Systems and, more specifically, by using our experience in the development of IASs for process control. As a result of our analyses, we stress the dynamic aspect of the contextual knowledge. We propose to define contextual knowledge as a possibly unlimited, personal and situated set of relevant knowledge involved in problem solving. Part of this contextual knowledge is proceduralized to enable cooperation and this results in a shared, limited proceduralized context that can be elicited, and therefore made explicit by the usual methods of knowledge engineering. There are a finite number of pieces of knowledge in the proceduralized context, each of them being related to a situation, date, locations, participants, problem, etc. One difficult question, which can be the main question in diagnosis, is to identify the relevant situation. The second question that we can address through case-based reasoning (Brézillon and Pomerol, 1998) is to use proceduralized context for situations, which are close or similar to a reference situation (*i.e.*, a kind of context-based reasoning). Using our definition we show how contextual knowledge is partially proceduralized in cooperative settings. The proceduralization process being itself a cooperative process we show how the proceduralized context is incrementally enriched and we examine the role of explanation in this process.

Our opinion is that contextual issues cannot be addressed in a static framework only and that eliciting and sharing contextual knowledge in a dynamic way is a key process in addressing and understanding context problems. The parallel to the AI community reinforces this; there is now another community interested by context in a more hardware way than in IA. Each community tackle ambitious challenges. However, it is clear that none of the two communities is totally right. It is necessary to find a generalization based on the advantages of both communities, but avoiding their respective weaknesses. This supposes the ability to manage: Information that evolves with time (e.g. time schedule for visiting a castle); Information coming from heterogeneous sources (e.g. weather, hotels, press, etc.); Knowledge about users (including their preferences and profiles as perceived by the system through users' actions); Software with a centralized part and mobile parts attached to a user or group of users; and Hardware and interaction among pieces of hardware.

10. REFERENCES

Aamodt, A, 1993. "A case-based answer to some problems of knowledge-based systems" *Scandinavian Conference on Artificial Intelligence*. Sandewall E. and Jansson C.G. (Eds.). IOS Press. pp 168-182.

Agabra, J, Alvarez, I, and Brézillon, P, 1997. "Contextual Knowledge Based System: A study and design in enology" *Proceedings of the First International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)* Federal University of Rio de Janeiro Ed., pp 351-362.

Alexander, C, Ishikawa, S, Silverstein, M, Jacobson, M I, Fiskdahl-King, S and Angel., S, 1996. *A Pattern Language*, New York: Oxford University Press.

Allard, F, 1999. "Affordances in Human-Computer Interactions: Do Affordances for Websites Really Exist?" *Course on Information Processing in Human Perceptual Motor Performance*. http://ahs.uwaterloo.ca/~bghlee/affordances/frames.htm

Allard, F, 2000. *Kinesiology 356: Course Notes,* WaterlooGraphics Visual Solutions.

Appleton, B, 2000. Patterns in a Nutshell: The "bare essentials" of Software Patterns. http://www.enteract.com/~bradapp/

Bainbridge, L, 1997. The change in concepts needed to account for human behavior in complex dynamic tasks_*IEEE transactions on Systems, Man and Cybernetics*, **27**, 351-359.

Bardram, J E, 1997. "Plans as situated action: An Activity Theory approach to workflow systems" *Proceedings of ECSCW'97 Conference*, Lancaster UK, September 1997. http://www.daimi.aau.dk/~bardram/ECSCW97.html

Barewise, J and Perry, J, 1983. *Situations and Attitudes*, Cambridge: MIT Press,

Bayle, A, 1988. *An epistemological analysis of frames*. Simposium Internacional de Intelligencia Artificial: Memorias, Monterrey, Mexico: McGraw Hill.

Béguin, P, 1994. Travailler avec la CAO en ingénierie industrielle : de l'individuel au collectif dans les activités avec instruments. *Thèse de doctorat en ergonomie*. Paris, CNAM.

Bellinger, G, 1999. Knowledge Management—Emerging Perspectives. OutSights. http://www.outsights.com/systems/kmgmt/kmgmt.htm

http://www.outsights.com/systems/kmgmt/kmgmt.htm

Bjork, S and Redstrom, J, 2000. Redefining the focus and context of focus+context visualizations. *Proceedings of InfoVis2000.* http://www.infovis.org/infovis2000

Bobrow, D G, 1975. *Some principles of memory schemata*, In D. Bobrow and A. Collins (Eds.), Representation and Understanding: Studies in Cognitive Science. London: Academic Press.

Bobrow, D G and Winograd, T, 1977. An overview of KRL, a Knowledge Representation Language. *Cognitive Science*, **1**(1): 3-46.

Bouaud, J, Séroussi, B and Antoine, E-Ch, 1999. OncoDoc: modélisation et "opérationalisation" d'une expertise thérapeutique au niveau des connaissances. *Actes de Ingénierie des Connaissances (IC'99)*, Palaiseau, pp 61-69.

Bourdieu, P, 1972. *Esquisse d'une théorie de la pratique*, Genève: Droz.

Bower, G H, Black, J B, and Turner, T T, 1979. Scripts in memory for text. *Cognitive Psychology*„ **11**, pp 177-220.

Boy, G, 1991. *Intelligent Assistant Systems* Knowledge-Based Systems, 6. London: Academic Press.

Brézillon, P, 1990. Interpretation and rule packet in expert systems. Application to the SEPT expert system. In S. Ramani, R. Chandrasekar, & K.S.R. Anjaneyulu (Eds.), *Knowledge Based Computer Systems* pp 78-87. Heidelberg: Springer-Verlag.

Brézillon, P, 1994. Context needs in cooperative building of explanations. *First European Conference on Cognitive Science in Industry*. Luxembourg,

Brézillon, P, 1996. Context in problem solving. Technical Report 96/29, LAFORIA, University Paris 6, October, 37 pages (available at ftp://ftp.ibp.fr/ ibp/reports/ laforia.96/laforia.96.29.ps).

Brézillon, P, 1997. A first step towards the modeling of contexts. *Proceedings of the 2nd European Conference on Cognitive Science (ECCS'97)*, pp 195-198.

Brézillon, P., 1999. Context in human-machine problem solving: A survey. *Knowledge Engineering Review*, **14**,1-34.

Brézillon, P and Abu-Hakima, S, 1995. Using knowledge in its context: Report on the IJCAI-93 Workshop. *The AI Magazine*, **16**(1) pp 87-91.

Brézillon, P and Cases, E, 1995. Cooperating for assisting intelligently operators. *Proceedings of the International Workshop on the Design of Cooperative Systems*, INRIA Ed., pp 370-384.

Brézillon, P and Pomerol, J-Ch, 1996a. Misuse and Nonuse of Knowledge-Bases Systems: the past experiences revisited. In *Implementing Systems for Supporting Management Decisions*, P. Humphreys, L. Bannon, A. McCosh, P. Migliarese, J.-Ch. Pomerol, Eds., London: Chapman and Hall, pp 44-60.

Brézillon, P, and Pomerol, J-Ch, 1996b, "User Acceptance of Interactive Systems: Lessons from Knowledge-Based and Decision Support Systems" *International Journal on Failures and Lessons Learned in Information Technology Management*, **1**(1) pp 67-75.

Brézillon, P and Pomerol, J-Ch, 1997a. "Lessons learned on successes and failures of KBSs" *Special Issue on Successes and Pitfalls of Knowledge-Based Systems in Real-World Applications, Int. Journal of Failures & Lessons Learned in Information Technology Management* **1**(2) pp 89-98.

Brézillon, P and Pomerol, J-Ch, 1997b. *Contextual issues in the framework of multicriteria decision making* First International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97). Federal University of Rio de Janeiro (Ed.) pp 256-268. 1997 (also available at http://www-poleia.lip6.fr/~brezil/Pages2/CONTEXT-97/index.html).

Brézillon, P and Pomerol, J-Ch, 1998. Using contextual information in decision making. In G. Widmeyer, D. Berkeley, P. Brézillon & V. Rajkovic (Eds.) *Context-Sensitive Decision Support Systems* (pp 158-173). London, UK: Chapman & Hall.

Brézillon, P, and Pomerol, J-Ch, 1999. "Contextual knowledge sharing and cooperation in intelligent assistant systems" *Le Travail Humain*, **62**(3), Paris: PUF, pp 223-246.

Brézillon, P, Pasquier, L and Pomerol, J-Ch, 2000a. "Reasoning with contextual graphs" *European Journal of Operational Research* **136**(2) pp 290-298.

Brézillon, P, Pasquier, L and Pomerol, J-Ch, 2000b. Representing operational knowledge by contextual graphs. *International Joint Conference IBERAMIA'2000-SBIA'2000*, Sao Paulo, Brazil, November 2000.

Brézillon, P, Pomerol, J-Ch and Saker, I 1998. "Contextual and contextualized knowledge: An application in subway control" *International Journal of Human-Computer Studies* **98** pp 357-373.

Brézillon, P, Pasquier, L and Saker, I, 1999. Context-based reasoning and decision graphs: Application in incident management on a subway line. *Seventh European Conference on Cognitive Science Approaches to Process Control, CSAPC'99*, September 20-24, Presses Universitaires de Valenciennes, pp 189-194.

Brézillon, P, Gentile, C, Saker, I and Secron, M, 1997. SART: A system for supporting operators with contextual knowledge. *Proceedings of the First International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97).* Federal University of Rio de Janeiro Ed. pp 209-222. (also available at http://www-poleia.lip6.fr/~brezil/Pages2/CONTEXT-97/index.html).

Brézillon, P, Cavalcanti, M, Naveiro, R and Pomerol, J-Ch, 2000. "SART: An intelligent assistant for subway control" *Pesquisa Operacional, Brazilian Operations Research Society* **20**(2) pp 247-268.

Britanik, J M and Marefat, M M, 1999. "Hierarchically merging plans in decomposable domains" *IEEE Trans. on Systems, Man, and Cybernetics* **29**(1) pp 27-39.

Card, S K, Mackinlay, J D and Shneiderman, B, eds. 1999. *Readings in Information Visualization: Using Vision to Think* San Francisco, California: Morgan Kaufmann Publishers pp 1-34.

Carenini, G and Moore, J D 1993. Generating explanations in context *International Workshop on Intelligent User Interfaces*, Orlando, Florida.

Carroll, J M, 1995. Introduction: The scenario perspective on system development. In: *Scenario-Based Design: Envisioning Work and Technology in System Development*. J.M. Carroll Ed. J. Wiley & Sons, pp 1-17.

Chalmers, M, 2000. Affordances and Interpretation. *Lecture on Human-Computer Interaction* 4. http://www.dcs.gla.ac.uk/~matthew/lectures/HCI4/HCI4Supp3.pdf

Chandrasekaran, B, Johnson, T R and Smith, J W, 1992. "Task-structure analysis for knowledge modeling" *Communications of the ACM*, **35**(9) pp 124-137.

Clancey, .J, 1979. "Tutoring rules for guiding a case method dialogue" *International Journal of Man-Machine Studies* 11(1), pp 25-49.

Clancey, W J, 1983. "The epistemology of a rule-based expert system: A framework for explanation" *Artificial Intelligence Journal* **20**(3) pp 197-204.

Clancey, W J, 1989. "The knowledge level reinterpreted: modeling how systems interact" *Machine Learning* **4** pp 283-291.

Clancey, W J 1991a. "Israel Rosenfield, The Invention of Memory: A New View of the Brain (Book Review)" *Artificial Intelligence Journal* **50** pp 241-284.

Clancey, W J, 1991b. "Situated cognition: stepping out of representational flatland" *AI Communications* **4** 2/3 pp 109-112.

Clancey, W J, 1992. The knowledge level reinterpreted: modeling socio-technical systems, *Proceedings of the AAAI'92 Worshop on Cognitive Aspects of Knowledge Acquisition*, Stanford, CA, March, pp 47-56.

Clancey, W J, 1995. The conceptual, non-descriptive nature of knowledge, situations, and activity. *Proceedings of the IJCAI-95 Workshop on Modelling Context in Knowledge Representation and Reasoning* Research Report 95/11 of LAFORIA, University Paris 6, France.

Collins, A M and Loftus, E F, 1975. "A spreading activation theory of semantic processing" *Psychological Review* **82** pp 407-428.

Compton, P and Jansen, B, 1988. Knowledge in context: A strategy for expert system maintenance. In: J. Siekmann ed. *Lecture Notes in Artificial Intelligence* Heidelberg: Springer-Verlag pp 37-49.

de Brito, G and Boy, G, 1999. Situation awareness and procedure following. *Proceedings of CSAPC'99*, Villeneuve d'Ascq, Presses Universitaires de Valenciennes pp 9-14.

de Kleer, J, 1987. An assumption based truth maintenance system. In: M Ginsberg ed., *Readings In Nonmonotonic Reasoning* Los Altos, CA: Morgan Kaufmann.

Dearden, A, Harrisson, M and Wright, P, 2000. "Allocation of function: scenarios, context, and the economics of efforts International" *Journal of Human Computer Studies* **52**(2) pp 289-318.

Debenham, J, 1997. Strategic workflow management: An experiment. *Proceedings of the International Workshop "Distributed Artificial Intelligence and Multi-Agent Systems", DAIMAS'97*.

Degani, A and Wiener, E L, 1997. "Procedures in complex systems: The airline cockpit" *IEEE Trans. on Systems, Man, and Cybernetics-Part A: Systems and Humans* **27**(3) pp 302-312.

Degler, D and Battle, L, 2000. "The challenge of context" *Performance Improvement* **39**(6) pp 25-31.

Dey, A K and Abowd, G D, 1998. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. http://www.cc.gatech. edu/fce/contexttoolkit

Douglas, C, Schmidt, D C, Johnson, R E and Fayad, M, 1996. "Software Patterns" Guest editorial for *The Communications of the ACM, Special Issue on Patterns and Pattern Languages* **39**(10),

Dourish, P, 2001. "Seeking a Foundation for context-aware computing" *Human-Computer Interaction* **16** (2-4). http://hci-journal.com/editorial/vol-16.html

Drummond, M, 1989. Situated control rules. *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning* Toronto: Morgan Kaufmann pp 103-113.

Duvenci-Lenga, S, 1997. Evolution de l'activité et des compétences en situation d'automatisation: le cas des machines-outils. *Thèse de doctorat d'ergonomie*. Paris: CNAM.

Edmonds, B, 1997. A simple-minded network model with context-like objects. *Proceedings of the 2nd European Conference on Cognitive Science*, Manchester, UK, April, 181-184.

Eco, U. (1997). *Kant et l'Ornithorynque*, Paris: Grasset.

Edmondson, W H and Meech, J F, 1993. A model of context for human-computer interaction. *Proceeding of the IJCAI-93 Workshop on Using Knowledge in its Context*. Technical Report 93/13, LAFORIA, University Paris 6 pp 31-38.

Ekdahl, B, Astor, E and Davidsson, P, 1995. Toward anticipatory agents. In: *Intelligent Agents* M. Wooldridge & Jennings N. eds. Lecture Notes in AI, 890, Berlin: Springer Verlag pp 191-202.

Erickson, T, 1995. Note on design practice: Stories and prototypes as catalysts for communication. In: *Scenario-Based Design: En visioning Work and Technology in System Development* J.M. Carroll ed. J. Wiley & Sons pp 37-58.

Eriksson, H-E and Penker, M, 1998. *UML Toolkit.* New York: Wiley Computer Publishing.

Fayad, M and Schmidt, D C, 1997. "Object-Oriented Application Frameworks" Guest editorial for *The Communications of the ACM, Special Issue on Object-Oriented Application Frameworks*, **40**(10).

Fischer, G, 1990. "Communication requirements for cooperative problem solving systems" *Information Systems* **15**(1) pp 21-36.

Fitzpatrick, G and Bruza, P, 2000. Views on Context in the Enterprise. *CHI 2000 Workshop, ACM Conference on Human Factors in Computing Systems*, pp 22-24. http://www.daimi.au.dk/~mbl/chi2000-sitcomp/pospapers.pdf

Forslund, G, 1995. Toward cooperative advice-giving systems. The expert systems experience, *Ph. D. Thesis 518, Linkoping University, Sweden.*

Furnas, G W 1986. Generalized fisheye views. *Proceedings of Human Factors in Computing Systems CHI'86 Conference*, pp 16-23.

Galinier, V, 1996. Apports de l'ergonomie à la conception d'instruments: concevoir autour des schèmes d'utilisation. Un exemple dans le domaine du transport routier. *Thèse de doctorat en ergonomie. Paris: CNAM.*

Garfinkel, A, 1981. *Forms of Explanation : Rethinking the Questions in Social Theory* New Haven and London: Yale University Press.

Gibson, J .J, 1977. The theory of affordances. In R. Shaw & J. Bransford eds. *Perceiving, Acting and Knowing*. Hillsdale, NJ: Erlbaum.

Gibson, J J, 1979. *The Ecological Approach to Visual Perception* Boston: Houghton Mifflin.

Gilboa, I and Schmeidler, D, 1995. "Case-based Decision Theory" *Quaterly Journal of Economics*, **110** pp 605-639.

Ginsberg, M L, 1995. "Approximate planning" *Artificial Intelligence Journal* **76**(1-2) pp 89-123.

Gonzalez, A J and Ahlers, R H, 1993. A context-based representation of tactical knowledge for use in simulation-based autonomous intelligent platforms. *Proceedings of the Interservice/Industry Training Systems Conference*, Orlando, FL, USA.

Gonzalez, A J and Ahlers, R, 1995. Context-based representation of intelligent behavior in simulated opponents. *Proceedings of the Computer Generated Forces & Behavioral Representation Conference*, Orlando.

Gonzalez, A J and Ahlers, R, 1997. "Context-based representation of intelligent behavior in training simulations" *Transactions* **15**(4) pp 153-166.

Grant, A S, 1992. Mental models and everyday activities. *Proceedings of the 2nd Interdisciplinary Workshop on Mental Models*, Cambridge, UK, pp 94-102.

Gruber, T, 1991. Justification-based knowledge acquisition. In H. Motoda, R. Mizoguchi, J. Boose, & B. Gaines (Eds.), *Knowledge Acquisition for Knowledge-Based Systems* Amsterdam, The Netherlands: IOS Press pp 81-97.

Guha, R V, 1991. Contexts: a formalization and some applications. MCC Technical Report ACT-CYC-423-91 December.

Hatchuel, A. and Weil, B, 1992. *L'expert et le système.* Paris: Economica.

Hayes-Roth, B, 1985. A blackboard architecture for control. *Artificial Intelligence Journal* **26** pp 251-322.

Hayes-Roth, B and Hayes-Roth, F, 1979. "A cognitive model of planning" *Cognitive Science* **3** pp 275-310.

Hendrix, G, 1975. Expanding the utility of semantic networks through partitioning. *Proceedings of the Fourth IJCAI,* pp 115-121.

Henninger, S, 1992. *The knowledge acquisition trap. Proceedings of the IEEE Workshop on Applying Artificial Intelligence to Software Problems: Assessing Promises and Pitfalls (CAIA-92)*. Monterey, Canada, March.

Hoc, J-M, 1996. *Supervision et Contrôle de Processus: La cognition en Situation Dynamique*, Grenoble: Presses Universitaires de Grenoble.

Hoc, J-M and Amalberti, R, 1994. "Diagnostic et prise de décision dans les situations dynamiques" *Psychologie Française* **39**(2) pp 177-192.

Hoc, J-M and Amalberti, R, 1995. "Diagnosis : Some theoretical questions raised by applied research" *Current Psychology of Cognition*, **14** pp 73-101.

Hollnagel, E, ed. 1993. *Human Reliability Analysis, Context and Control*. London: Academic Press.

Humphreys, P and Berkeley, D, 1992. Support for the synthesis and analysis of organisational systems in deciding on change *Decision Support Systems: Experiences and Expectations,* T .Jelassi, M R Klein and W M Mayon-White, eds. Amsterdam, North-Holland: Elsevier Science Publishers pp 29-50.

Huuskonen, P and Korteniemi, A, 1992. Explanation based on contexts *Procedings of the 8th Conference on Artificial Intelligence for Applications,* Monterey, CA, March, pp 179-185.

Jansen, B, 1995. "Context in context" http://mac145.syd.dit.csiro.au/Context/context.html (Working Draft V4).

Jensen, F, 1996. *An introduction to Bayesian networks* London: UCL Press.

Johnson, H and Johnson, P, 1992. Acceptable explanations: theory before implementation *Proceedings of the AAAI Spring Symposium on Producing Cooperative Explanation*, Stanford, CA, March pp 95-102.

Jordan, T, Raubal, M, Gartrell, B and Egenhofer, M .J, 1998. An Affordance-Based Model of Place in GIS. n: T. Poiker and N. Chrisman, eds. *Proceedings of the 8th International Symposium on Spatial Data Handling, SDH'98,* Vancouver, Canada, pp 98-109.

Jurisica, I, 1994. Context-based similarity applied to retrieval of relevant cases *Proceedings of AAAI Fall Symposium Series on Relevance. New Orleans,* Louisiana.

Jurisica, I and Glascow, J, 1997. "Improving performance of case-based classification using context-based relevance" *International Journal of Artificial Intelligence Tools* **6** pp 3-4.

Jurisica, I and Glasgow, J, 1998. An efficient approach to iterative browsing and retrieval for case-based reasoning Proceedings of the 11th IEA-98-AIE, Tasks and Methods in Applied Artificial Intelligence. del Pobil A P, Mira J and Ali M, eds. Berlin: Springer Verlag, Vol. II, pp 535-546.

Kahneman, D and Lovallo, D, 1993. "Timid choices and bold forecasts: A cognitive perspective on risk taking" *Management Science* **39** pp 17-31.

Karat, J, 1995. Scenario use in the design of a pseech recognition system. In: *Scenario-Based Design: En visioning Work and Technology in System Development.* J.M. Carroll, ed. J. Wiley & Sons pp 109-133.

Karsenty, L, 1994. *L'explication d'une solution dans les dialogues de conception* Ph.D. Thesis, University Paris 8, France.

Karsenty, L, 1996. "Une définition psychologique de l'explication" *Intellectica,* **2** pp 299-317.

Karsenty, L and Brézillon, P, 1995. "Cooperative problem solving and explanation" *International Journal of Expert Systems With Applications* **4** pp 445-462.

Karsenty, L and Falzon, P, 1992. Spontaneous explanations in cooperative dialogues *Proceedings of the ECAI'92 Workshop on Improving the Use of KBS with Explanation,* Technical Report 92/21, LAFORIA, University Paris 6, France, June, pp 115-124.

Keeney, R L, 1992. *Value-Focused thinking* Harvard University Press.

Kolodner, J, 1993. *Case-based Reasoning* San Francisco, USA: Morgan Kaufmann.

Kott, A, Saks, V and Mercer, A, 1999. "A new technique enables dynamic replanning and rescheduling of aeromedical evacuation" *AI Magazine* Spring, pp 43-53.

Kuhn, W, 1996. Handling Data Spatially: Spatializing User Interfaces. in: Kraak M. and Molenaar M., eds.

Laird, J E, Newell, A and Rosenbloom, P S, 1987. "Soar: An architecture for general intelligence" *Artificial Intelligence Journal* **33** pp 1-64.

Landauer, C and Bellman, K, 1993. The role of self-referential logics in a software architecture using wrappings *Proceedings of ISS'93: 3rd Irvine Software Symposium, Irvine*, pp 1-12.

Langley, A, Mintzberg, H, Pitcher, A, Posada, E. and Saint-Macary, J, 1995. "Opening up decision making : the view from the black stool" *Organization Science*, **63** pp 260-279.

Lea,, D, 1997. Patterns Discussion FAQ, http://g.oswego.edu/dl/pd-FAQ/pd-FAQ.html December.

Leake, D B, 1991. "Goal-based explanation evaluation" *Cognitive Science* **15**(4).

Leake, D B, 1992. *Evaluating explanations* Lawrence Erlbaum Associates Inc.

Leake, D B, 1996. "Case-based reasoning: Experiences, lessons, and future directions. Chapter I" *CBR in context: The present and future* Menlo Park: AAAI Press/MIT Press.

Leplat, J, 1985. *Erreur humaine, fiabilité humaine dans le travail* In: A. Colin, ed. Collection Universitaire, pp 100-120.

Lester, J-C and Porter, B W, 1991. Generating context-sensitive explanations in interactive knowledge-based systems *Proceedings of the AAAI'91 Workshop on Comparative Analysis of Explanation Planning Architectures* pp 27-41.

Lévine, P and Pomerol, J-Ch, 1989. *Systèmes interactifs d'aide à la décision et systèmes experts* Paris, France: Hermès.

McCarthy, J, 1993. Notes on formalizing context *Proceedings of the 13th IJCAI* Vol.1, pp 555-560.

McDermott, J, 1982. "R1: A rule based configurer of computer systems" *Artificial Intelligence Journal* **19** pp 39-88.

Mackie, J, 1965. "Causes and conditions" *American Philosophical Quaterly* **2**(4) pp 245-264.

March, J G and Shapira, Z, 1987. "Managerial perspectives on risk and risk taking" *Management Science* **33** pp 1404-1418.

Mark, B, 1988. Explanation and interactive knowledge acquisition *Proceedings of AAAI'88, Workshop on Explanation*.

Maskery, H and Meads, J, 1992. "Context: In the eyes of users and in computer systems" *SIGCHI Bulletin* **24** pp 12-21.

Martin, D, Rodden, T, Rouncefield, M, Sommerville, S and Viller, S, 2001. Finding patterns in the fieldwork *Proceedings of the Seventh European Conference on Computer-Supported Cooperative Work* Kluwer Academic Publishers, W.Prinz, M.Jarke, Y.Rogers, K.Schmidt, and V.Wulf, eds pp 39-58.

Menzies, T, 1996. Assessing responses to situated cognition *Proceedings of the Banff Knowledge Acquisition Workshop* (http://www.cse.edu.au/~timm/pub/docs/papersonly.html).

Minsky, M, 1975. A framework for representing knowledge. In P Winston ed *The psychology of computer vision* New York: McGraw-Hill.

Minsky, M, 1981. A Framework for Representing Knowledge. In J Haugeland ed *Mind Design-Philosophy, Psychology, Artificial Intelligence* Cambridge, MA: The MIT Press.

Mittal, V O and Paris, C L, 1993. Context : identifying its elements from the communication point of view *Proceedings of the International Joint Conference on Artificial Intelligence Workshop on Using Knowledge in its Context*. Paris: France.

Moliere, 1670. Le Bourgeois Gentilhomme, Acte 2, Scene 4.

Monnet, J-M, Lagrange, J-Ph, Pomerol, J-Ch and Teulier, R, 1998. A formal framework for decision-making in case-based reasoning. IPMU.

Montgomery, H, 1983. Decision rules and the search for a dominance structure: towards a process model of decision making In: P.C. Humphreys, O. Svenson and A. Vari eds *Analysing and Aiding Decision Processes* Amsterdam: North Holland.

Montgomery, G, 1987. "Image theory and dominance search theory: how is decision making actually done?" *Acta Psychologica* **66** pp 221-224.

Moore, J D, 1995. *Participating in Explanatory Dialogues. Interpreting and Responding to Questions in Context* Series Natural Language Processing Cambridge, MA, USA: A Bradford Book, The MIT Press,

Moray, N, 1993. Designing for attention. In A Baddeley & L Weiskrantz eds *Attention: selection, awareness, and control: a tribute to Donald Broadbent* Oxford, UK: Oxford University Press pp 111-134.

Munch, S and Dillman, R, 1997. Haptic output in multimodal interfaces *Proceedings of IUI'97* Orlando, FL: ACM Press pp 105–112.

Neapolitan, R, 1990. *Probabilistic reasoning in expert systems* New York: John Wiley & Sons.

Newell, A. and Simon, H A, 1972. *Human Problem Solving* Englewoods Cliffs, New Jersey: Prentice-Hall Inc.

Niang, D, 1999. Méthodes et techniques d'Intelligence Artificielle pour prévenir l'explosion combinatoire - Application au Gestionnaire d'Incidents de SART *Rapport de DEA IRO*, University Paris 6, France.

Norman, D A, 1988. *The psychology of everyday things* New York: Basic Books.

Norman, D A, 1990. *The design of everyday things* New York: Doubleday (see also at http://www.jnd.org/dn.pubs.html)

Oliver, R and Smith, J, 1990. Influence Diagrams, Belief Nets and Decision analysis *Wiley Series in Probability and Mathematical Statistics* New York: John Wiley and Sons.

Ozturk, P and Aamodt, A, 1998. "A context model for knowledge-intensive case-based reasoning" Special Issue on Using Context in Applications *International Journal on Human-Computer Studies* **48**(3) pp 331-355.

Paris, C L, Wick, M R and Thompson, W B, 1988. The line of reasoning versus the line of explanation *Proceedings of AAAI Workshop on Explanation* pp. 4-7.

Pasquier, L, 2000. Raisonnements basés sur le contexte: Contextes procéduralisés, graphes contextuels et schèmes d'action Research Report LIP6 N.2000-010, University Paris 6, France.

Pasquier, L, Brézillon, P and Pomerol, J-Ch, 1999. Context and decision graphs in incident management on a subway line *Modeling and Using Context (CONTEXT-99)* Lecture Notes in Artificial Intelligence, N° 1688, Berlin: Springer Verlag pp 499-502.

Pasquier, L, Brézillon, P and Pomerol, J-Ch, 2000. From representation of operational knowledge to practical decision making in operations. *Decision Support throught Knowledge Management* S. Carlson, P. Brézillon, P. Humphreys, B. Lundberg, A. McCosh and V. Rajkovic eds pp 301-320.

Patil, R S, Szolovits, P and Schwartz, W B, 1981. Causal understanding of patient illness in medical diagnosis *Proceedings of the 7th International Joint Conference on Artificial Intelligence* pp 893-899.

Pearl, J, 1988. *Probabilistic reasoning in intelligent systems* San Mateo, California: Morgan Kaufmann Publishers.

Pereira, F C N and Pollack, M E, 1991. "Incremental interpretation" *Artificial Intelligence Journal* **50**(1) pp 37-82.

Perrenoud, Ph, 1976. "De quelques apports piagétiens a une sociologie de la pratique" *Revue européenne des sciences sociales* n° 38-39, pp 45l-470.

Piaget, J, 1936. *La naissance de l'intelligence chez l'enfant* Paris, Lausanne.

Piaget J, 1973. *Biologie et connaissance* Collection Idées Paris: Gallimard.

Pohl, F 1969. *L'Ère du Satisfacteur* Paris: Librairie des Champs Elysées, Série Science Fiction.

Pomerol, J-Ch, 1997. "Artificial Intelligence and Human Decision Making" *European Journal of Operational Research* **99** pp 3-25.

Pomerol, J-Ch, 1998. Scenario development and practical decision making under uncertainty: robustness, case-based reasoning and risk control *Proceedings of the IEEE Conference Engineering in Systems Application*. Hammamet: Tunisie.

Pomerol, J-Ch and Brézillon, P, 1997. Organizational Experiences with multicriteria support systems : Problems and Issues *Proceedings of the 30th Hawaii International Conference on System Sciences* J. F. Nunamaker and R.H. Sprague eds p 3-10.

Pomerol, J-Ch and Brézillon, P, 1999. Dynamics between contextual knowledge and proceduralized context *Modeling and Using Context (CONTEXT-99)* In: Lecture Notes in Artificial Intelligence, N° 1688, Berlin: Springer Verlag pp 284-295.

Pomerol, J-Ch, Roy, B, Rosenthal-Sabroux, C and Saad, A, 1995. "An "intelligent" DSS for the Multicriteria Evaluation of Railway Timetables" *Foundations of Computing and Decision Sciences* **20**(3) pp 219-238.

Pook, S, Lecolinet, E, Vaysseix, G and Barillot, E, 2000. Context and interaction in Zoomable User Interfaces *AVI 2000 Conference Proceedings (ACM Press)* pp 227-231.

Pryor, L and Collins, G, 1996. "Planning for contingencies: A decision-based approach" *Journal of Artificial Intelligence Research* **4** pp 287-339.

Rabardel, P, 1995. Les Hommes et les technologies. *Approche cognitive des instruments contemporains* Paris: A. Colin.

Raïffa, G, 1968. *Decision Analysis* New York: Mac Graw Hill.

Ramirez, C, 1997. Schemata, Frames, and Dynamic Memory Structures Technical Report 7-97, University of Kent at Canterbury, Computing Laboratory, Canterbury, Kent CT2 8DS.

Ramirez, C and Cooley, R, 1997. A Theory of the Acquisition of Episodic Memory. D Wettschereck and D W Aha eds *Proceeding of the European Conference in Machine Learning* Prague: Springer-Verlag.

Rasmussen, J, 1986. *Information processing and human machine interaction: An approach to cognitive engineering* New York, NY: North-Holland.

Rogoff, B, Gauvain, M and Gardner, W, 1987. The development of children's skills in adjusting plans to circumstances In: Friedman S, Scholnick S & Cocking R eds *Blueprints for Thinking: The Rule of Planning in Psychological Development*. New York: Cambridge University Press pp 303-320.

Roth, J, 2001. Patterns of mobile interaction *Proceedings of Mobile HCI 2001: Third International Workshop on Human Computer Interaction with Mobile Devices* M D Dunlop and S A Brewster eds.

Roy, B, 1997. "Un chaînon manquant en RO-AD, les conclusions robustes" *Cahier du LAMSADE*, n°114.

Rumelhart, D E 1980. Schemata: The basic building blocks of cognition In R. Spiro, B. Bruce, and B. Brewer eds *Theoretical Issues in Reading Comprehension* Hillsdale, NJ: Lawrence Earlbaum.

Saint Amant, R, 1999. Planning and user interface affordances *Proceedings of the 5th International Conference on Intelligent User Interfaces*.

Saint Amant, R and Cohen, P R, 1994. A planning representation for automated exploatory data analysis. *Proceedings of SPIE, Knowledge-based Artificial Intelligent Systems in Aerospace and Industry* **2244** pp 44-52. Computer Science Technical Report 94-18.

Sarkar, M and Brown, M H, 1994. "Graphical fisheye views" *Communications of the ACM* **37**(12) pp 73-84.

Savage, L .J, 1954. *The foundations of statistics* John Wiley and Sons.

Schank, R C, 1972. "Conceptual dependency: A theory of natural language understanding" *Cognitive Psychology* **3** pp 552-631.

Schank, R C, 1975. The Structure of Episodes in Memory In D. Bobrow and A. Collins eds *Representation and Understanding: Studies in Cognitive Science* New York: Academic Press.

Schank, R C, 1982. *Dynamic memory, a theory of learning in computers and people* Cambridge University Press.

Schank, R C and Alberson, D J, 1975. *Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures* Hillsdale, NJ: L. Erlbaum.

Schmidt D C, ed. 1995. *Pattern Languages of Program Design* Addison-Wesley.

Schmidt, D C, Johnson, R E and Fayad, M, 1996. "Software Patterns" *Communications of the ACM* Special Issue on Patterns and Pattern Languages **39**(10). http://www.cs.wustl.edu/~schmidt/CACM-editorial.html

Schmidt, A, Takaluoma, A and Mantyjarvi, B, 2000. "Context-aware telephony over wap" *Personal Technologies* **4**(4) pp 225-229.

Schoppers, M J, 1987. Universal plans for reactive robots in unpredictable environments *Proceedings of the Tenth International Joint Conference on Artificial Intelligence* pp 1039-1046.

Sowa, J F, 1984. *Conceptual structures information processing in mind and machine* Addison Wesley Publishing Company.

Sowa, J F, 1991. Toward the expressive power of natural language In: *Principles of Semantic Networks - Exploration in the representation of Knowledge* San Mateo, CA: Morgan Kaufmann pp 157-189.

Sowa, J F, 2000. *Knowledge Representation: Logical, Philosophical, and Computational Foundations* Pacific Grove, CA: Brooks Cole Publishing Co.

Strauss, A, Fagerhaugh, S, Suczek, B and Wiener, C, 1985. *Social Organization of Medical Work* Chicago and London: University of Chicago Press.

Suthers, D, 1993. Influences of the epistemic context on explanation *Proceedings of the IJCAI-93 Workshop on Using Knowledge In Its Context* Research Report 93/13, LAFORIA, Box 169, University Paris 6, France.

de Terssac, G, 1992. *Autonomie dans le travail* Série Sociologie d'Aujourd'hui. Paris: Presses Universitaires de France.

de Terssac, G and Chabot, C, 1990. Réferentiel opératif commun et fiabilité In J Leplat & G de Terssac eds *Les facteurs humains de la fiabilité* Toulouse: Octarès pp 110-139.

Tiberghien, G, 1986. "Context and cognition: Introduction" *Cahier de Psychologie Cognitive* **6**(2) pp 105-119.

Tijus, Ch, 2001. *Introduction à la Psychologie Cognitive* Paris: Nathan Université.

Tsatsoulis, C, Cheng Q and Weil H-Y, 1997. "Integrating Case-based Reasoning and Decision Theory" *IEEE Expert*, July/August, **12** pp 46-55.

Turner, R M, 1993. Context-sensitive reasoning for autonomous agents and cooperative distributed problem solving *Proceedings of the IJCAI-93 Workshop on Using Knowledge in its Context* Rapport de Recherche 93/13, LAFORIA, Université Paris 6, Paris, France.

Turner, R M, 1997. Determining the context-dependent meaning of fuzzy subsets *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)* University of Rio de Janeiro, Brazil, February.

Turner, R M, 1998. "Context-mediated behavior for intelligent agents" *International Journal of Human-Computer Studies* Special Issue on Using Context in Applications **48**(3) pp 307--330.

Turney, P, 1996. The identification of context-sensitive features: A formal definition of context for concept learning *Proceedings of the ICML'96 Workshop on Learning in Context-Sensitive Domains* pp 53-59.

Tversky, A, 1977. "Features of similarity" *Psychological Review* **84**(4) pp 327--352.

UML 2000 http://www.dalmatian.com/unified_modeling_language.htm

Van der Sandt, R, 1992. "Presupposition Projection as Anaphora Resolution" *Journal of Semantics*

Van Dijk, T A, 1998. Cognitive Context Models and Discourse. In Maxim Stamenov ed *Language Structure, Discourse and the Access to Consciousness* Amsterdam: Benjamins pp 189-226.

Vanwelkenhuysen, J and Mizoguchi, R, 1995. Adaptation of reusable knowledge for workplace integration *Proceedings of the IJCAI-95 Workshop on Modelling Context in Knowledge Representation and Reasoning* Technical Repport 95/11, LAFORIA, University Paris 6, France pp 167-177.

Vergnaud, G, 1985. "Concepts et schèmes dans la théorie opératoire de la représentation" *Les Représentation, Psychologie Française* **30** (3 et 4) pp 245-252.

Wallace, N, 2000). Design Patterns in Web Programming http://www.e-gineer. com/articles/design-patterns-in-web-programming.phtml

Warren, W H, 1995. Constructing an Econiche In Flack J, Hancock P, Caird J and Vicente K eds *Global Perspectives on the Ecology of Human-Machine Systems (volume 1)* Hillsdale, New Jersey: Lawrence Erlbaum Associates pp 121-156.

Watson, I and Perera, R S, 1998. "A hierarchical case representation using context guided retrieval" *Knowledge Based Systems Journal* **11**(5-6) pp 285-292.

Wick, M R and Thompson W B, 1992. "Reconstructive expert system explanation" *Artificial Intelligence Journal* **54** pp 33-70.

Woods, D D, Roth, E M and Benett, K, 1990. Explorations in joint human-machine cognitive systems In Robertson S, Zachary W and Black J B eds *Cognition, Computing and Cooperation* Ablex pp 123-158.

Worden, A, Walker, N, Bharat, K and Hudson, S, 1997. Making computers easier for older adults to use: Area cursors and sticky icons *Proceedings of the CHI'97 Conference* pp 266–271.

Xiao, Y, Milgram, P and Doyle, D J, 1997. "Planning behavior and its functional role in interactions with complex systems" IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans **27**(3) pp 313-324.

Zhang, J, 2000. Categorization of Affordances http://acad88.sahs.uth.tmc.edu/courses/hi6301/affordance.html