The Knowledge Engineering Review, Vol. 18:2, 147–174. © 2003, Cambridge University Press
DOI: 10.1017/S0269888903000675 Printed in the United Kingdom

Representation of procedures and practices in contextual graphs

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 last ten years a community that is interested in context has emerged. Brézillon (1999) gave a survey of the literature on context in artificial intelligence. There is now a series of conferences on context, a website and a mailing list. The number of web pages with the word "context" has increased tenfold in the last five years. Being among the instigators of the use of context in real-world applications, I present in this paper the evolution of my thoughts over the last years and the results that have been obtained, including a representation formalism based on contextual graphs and the use of this formalism in a real-world application called SART. I present how procedures, practices and context are intertwined, as identified in the SART application and in different domains. I root my view of context in the artificial intelligence area and give a general presentation of my view of context under the three aspects – external knowledge, contextual knowledge and proceduralised context – with the implementation of this view in contextual graphs. I discuss how reasoning is carried out, based on procedure and practices, in the formalism of contextual graphs and show how incremental acquisition of practices is integrated in this formalism.

1 Introduction

For a long time, context has played an important role in domains in which reasoning is important. These domains include understanding, interpretation, diagnosis and so on. The reason context is important is that the types of reasoning undertaken in these domains rely heavily on a background or experience that is generally not made explicit but gives a contextual dimension to knowledge. Indeed, everybody uses context in their daily life, just as M. Jourdain used prose for speaking without being aware of it (Molière, 1670, Act 2, Scene 4). However, there is a lack of a clear definition of the word "context". In this paper, we present the evolution of thoughts on context over the last few years, thoughts that have been motivated by work on the SART application.

The SART project (French acronym for "support system for traffic control") aims to develop 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`). The operational knowledge used by operators is stored in an adapted structure, which will be easily understood by operators and efficiently used by the computer. As an intelligent assistant system, 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 when an operator requests it (to help the operator to test alternative issues, for example); proposing alternatives that might deal with an incident; training a new operator not familiar with a given line; and so on. A presentation of SART can be found in Pasquier (2002) and Brézillon *et al*. (2003).

Operators deal with 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 told 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 does not depend on the decision-makers but results in a limitation of their actions.

The contextual elements may intervene in several scenarios (e.g. traffic activity, position of the next train), and 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 proceduralised and in the meantime operators postpone actions as sequences. By grouping together a set of actions in a sequence, operators hope to make the following step easier.

Hereafter, I present in Section 2 how procedures, practices and context are intertwined, as identified in the SART application and in different domains. I root my view of context in the artificial intelligence area in Section 3. Section 4 gives a general presentation of my view of context under the three aspects of external knowledge, contextual knowledge and proceduralised context, and the implementation of this view in a context-based formalism called contextual graphs. Section 5 focuses on reasoning based on procedure and practices in the formalism of contextual graphs, and shows how incremental acquisition of practices is integrated in this formalism.

2 Context, procedures and practices

The subway company in Paris (RATP) has been in operation since 1900, and as day to day incidents occur the agents of RATP deal with them. These practices reflect the construction of operational knowledge, step by step, by the operators. The need for security and the desire to have a uniform procedure for dealing with incidents led the head of the company to compare operational practice across the company and to establish secure procedures for each incident that is encountered. In this sense, procedures are collections of safety action sequences allowing a given incident to be dealt with in any case. These procedures are based on practices, but eliminate most of the 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 kinds of incident have been well known for a long time (object on the track, lack of power supply, suicide and so on). Thus the company has established procedures for incident-handling on the basis of their experience. However, each operator develops his own practice to handle 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 proceduralised 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 & Weil, 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 the "logic of efficiency" when referring to such unwritten rules.

In parallel, operators prefer to plan their actions themselves, in real time, rather than rely on these procedures based on previous experience on the part of the company. This happens for two main reasons. First, 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. Second, if the operator relies on a procedure, he can miss some important facts and notice them too late to adequately handle the incident. Operators generally prefer to re-plan their actions continuously according to the situation. Procedures are then used as frames to construct a genuine strategy tailored to the specific detail 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.

Modelling the reasoning carried out by operators is a difficult task because operators use a number of contextual elements and because procedures for solving complex incidents have some degree of freedom. The reasoning carried out by operators stems from some chunks of implicit knowledge, which are imposed on the driver because they correspond to mandatory procedures. Procedures are established from the operator's prior experience during similar incidents and then fixed by the company. As such, practices are what I call hereafter "proceduralised contexts." In Figure 2 an implicit piece of knowledge is that travellers are safer in a station than in a tunnel. At a deeper level, the driver has to avoid stopping the train in a tunnel for a long time. The reason is that some travellers may have behavioural issues such as claustrophobia and could leave the train to wander about on the railway (and thus may generate another type of incident, such as "Traveller on the railway"). These pieces of knowledge, which are not necessarily expressed, result in more or less proceduralised actions that are compiled as a proceduralised context 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 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 according to the current context of use of practices rather than store the complete decisions.

Medicine is also a domain where the distinction between procedure and practice on the one hand, and the notion of context on the other hand, is very important. A practice can be considered as a kind of causal explanation for incident-handling in the spirit of the graphs in the ABEL system described in Patil *et al*. (1981). These graphs are called patient-specific models. In the ABEL system a causal explanation is represented as a five-layered graph containing the particular findings and 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 the medical domain, this type of operationalisation of 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 does not have any problems with his hips, this part of the plan may be skipped; other examinations, like a blood test, might be added to Mr Jones's unfolding plan. Thus, for the last authors, a protocol in medicine is a standard operating procedure (Strauss *et al*., 1985). Here, practices appear as a contextualised expression of procedures.

In high technical and heavily dynamic process-regulation domains, operators who are responsible for the process control have to react rapidly. If an incident occurs, they have only a few minutes to forge a representation of the issue, gather information on the situation, analyse the incident and undertake the correcting actions. To ease their job, many companies have established fixed procedures. Initially, general procedures were designed to provide operators with a secure reference for incident-handling. 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 specialisation the available procedures for each type of incident.

This discussion points out that if it is relatively easy to model procedures, the modelling of the corresponding practices is not an easy task because there are as many practices as contexts of

occurrence. For complex incident-solving, it is not possible to establish a global procedure, but only a set of subprocedures for solving parts of 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 there are operators. Cases that are similar in one context may be totally dissimilar in others, as already noted by Tversky (1977).

3 Context in AI

3.1 Introduction

AI has moved from the view of Knowledge-Based Systems (KBSs) replacing people towards two complementary approaches, namely the situated cognition approach (Clancey, 1991) and the Joint Cognitive System (JCS) approach (Woods *et al*., 1990). Situated cognition insists on the need in problem-solving to account for environment (through our interaction with it) and ongoing context in organising 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. The JCS approach puts the emphasis on the competence and complementarity of users and systems, and the symbiosis between them. It also emphasises, more or less explicitly, the role of context for such a symbiosis.

In artificial intelligence, researchers have implicitly made use of context for a long time. Examples include de Kleer (1987) with the ATMS, McDermott (1982) in X1/RCON, Laird *et al*. (1987) in SOAR, Hendrix (1975) for the partition of semantic networks, and Guha (1991) in the CYC project. However, the lack of an explicit representation of context is one of the reasons for the failure of many KBSs. Brézillon and Pomerol (1996) note the following problems:

1. Exclusion of the user from the problem-solving process. KBSs were assimilated to oracles and users were treated as novices. However, being faced with unexpected problems to solve is the norm rather than the exception. KBSs cannot solve such unexpected problems when users, with their practical experience, are not given the opportunity to help. What is required is 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 their knowledge correctly. Knowledge, which is acquired from human experts, has a high contextual component that is generally not acquired with the knowledge because knowledge engineers ask what the experts' solution is, not how the experts reach it.
3. KBSs cannot initially have all of the necessary 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. with its 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 to let the KBS generate the explanation jointly with the user (Karsenty & Brézillon, 1995).

In this section, I give an overview of context in rule-based representation, with incremental knowledge acquisition and explanation generation. I then discuss how context is considered in AI and the dynamic dimension of context, and conclude with some answers that have emerged.

3.2 Rule-based representation

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 that are 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 comprehension (even for the experts of the domain) and the maintenance of the knowledge.

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 organises 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 a 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.). Clause 4 was added to control the interaction with the user. The clause acts as a screening clause and implies that the context of use of the rule is limited to adults only. Such knowledge does not intervene directly in 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 such as circuit breakers and protective relays (Brézillon, 1990). Checking the internal behaviour of a circuit breaker implies an expertise (described in a rule packet) that is independent of the expertise on the internal behaviour 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 an isolated 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 isolated 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 specialisation 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*, which says that if a failure is found on the circuit breaker it is not necessary to check the equipment *pw* and *teac* of the cut-off system.

3.3 Context and incremental acquisition

An example of AI system design (Brézillon & Cases, 1995) convinced us of the prominent importance of two issues in cooperation, namely explanation generation and incremental knowledge acquisition. 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 I will show. In early AI system design, the knowledge engineer acquired expert knowledge as a pair ⟨problem, solution⟩, when the expert restricted the use of a solution to the given problem for a given context and thus provided the triple ⟨problem, context, solution⟩.

Thus, for developing AI systems, three aspects are particularly important:

1. Explanations that must be considered as an intrinsic part of any cooperative problem-solving (Karsenty & 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 about cases that have just been dealt with. Thus knowledge is acquired during problem-solving with its context of use. In some way, this is a kind of explanation from the user to the system.
3. The context of user-system cooperation must be made explicit to provide relevant explanation and acquire knowledge incrementally (Brézillon & 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. Efficient cooperation between a human and a machine implies consideration of the three related aspects: explanation, incremental knowledge acquisition and context.

There are two situations where incremental knowledge acquisition plays an important role in AI systems:

- When knowledge is missing in the current context, the user adds new knowledge through explanations. Here, explanations enable the contextual knowledge to be proceduralised.
- When 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 proceduralised 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 judgement 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 only entering the expert's new rule 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 that 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 can thus provide a frame to fill, allowing some kind of syntactic checking and ensuring that all required parameters are given.

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 within this context. Thus knowledge cannot be generalised 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.

3.4 Context and explanation

Most researchers have focused on explanations considering them as a transfer of knowledge *from* the system *to* the user. Feedback is used by the system to tailor its explanation to user's needs. Thus, users might only rarely intervene in the generation of explanations. Conversely, the approach taken in SEPT (Brézillon, 1990) lets the user alone build 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 the problem jointly, and to co-construct an explanation for the solution, explanation generation being an intrinsic part of the task at hand.

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 of the current state of problem-solving. The differing interpretations will be compatible if the user and the system make proposals, explain their viewpoints and spontaneously produce information (Karsenty & 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 process. 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 process and the line of reasoning of the system may be modified by explanation (Brézillon & Abu-Hakima, 1995). This leads to cooperative problem solving. Here context is an extended version of the context in the previous approach because it also integrates direct information from users, mainly on the basis of the consequences of their actions on the system and on the real-world process.

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 the event occurs. 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 characterisation.

Turney (1996) distinguishes three types of feature: primary, contextual and irrelevant. Primary features are useful for classification even when they are considered in isolation, without regard to 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 colour 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 proceduralised context.

Karsenty (1994) argues that communication implies the sharing of a linguistic code and a context. The context is a set of bits of information that are accessed or built to give a meaning to a message. Thus explanation is a means of sharing the context that is needed for the actors' understanding. Its aim is to differentiate between an initial context, in which information is not understood or is misunderstood, and a target-context, in which the information becomes comprehensible. It is a way to make the implicit knowledge in a procedure explicit, and to interpret new information. Explanation

generation acts as a contextualisation process (Karsenty & Brézillon, 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 & 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 of pointing out the links between the problem at hand and shared knowledge of its current state.

3.5 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 practical viewpoint versus the theoretical (Brézillon & Abu-Hakima, 1995). The *cognitive science view* is that context is used to model interactions and situations, and context-modelling must be human-centred (Brézillon, 2003). 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 makes it possible to understand the contrasted views found in the literature (see e.g. Brézillon, 1999).

All work in knowledge representation considers a set of discrete contexts, and the effort relies on the crossing of contexts. Creating a context from existing contexts, as proposed by McCarthy (1993), 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 in another. Conversely, in cognitive science, researchers who do not 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 of context and the interest is in 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. In its dynamic aspect, part of the problem is linked to the changing nature of context in time, by elaboration and shift (Clancey, 1991). If it is (relatively) easy to represent the static aspect of context, the dynamic aspects of context must be considered during its use, say in problem-solving. Thus one must account for both the *static aspect* (knowledge that remains constant throughout 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 means it is possible to not say all the things in a story. For example, “At his birthday party, Paul blew out 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 by shared knowledge. For example, there is a French movie call *Le Chat* (“the cat”) which depicts the lives of a husband and wife who have been living together for 40 years. Knowing each other very well, they are able to reduce the amount they need to communicate. For instance, with a slight movement of his chin towards the cupboard, the husband means “Darling, can you please pass 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 amount of information 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.

3.6 Dynamic dimension of context

Beyond the two contrasted views discussed in the previous section, context possesses a dynamic dimension that poses some problems in modelling. This dynamic dimension of context arises from the interactions among agents. That is, without interacting agents, there would be no context. In communication, the context is considered as the history of all that has 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. 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. Mittal and Paris (1993) point out that communication, including explanations, and context interact with each other: the context of the situation triggers some actions, and this in turn modifies the context of the situation.

The dynamic dimension of context appears also at the level of the reasoning task (such as problem-solving and diagnosis). The dynamic dimension of the process is very important in diagnosis as well as control of complex systems (Hoc, 1996; Hoc & Amalberti, 1995). We think that it is important to understand not only the dynamic dimension of planning and action but also the dynamic dimension 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 of moving contextual knowledge into and out of the front-stage knowledge that we call proceduralised context hereafter.

3.7 Lessons learned in AI

We cannot speak of context out of its context. Context is something surrounding an item (e.g. the task at hand or the interaction) and giving meaning to this item. Giving meaning to an item, context acts then more on the relationships between items than on items themselves; it modifies their extension and surface. Contextual knowledge concerns the future and consequences of actions, and intervenes in a 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. organisational knowledge) and 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-organisational environment) and the domain in which we are. All these contexts are interdependent; that is, the interaction context is constrained by the knowledge, and dictates the model chosen to represent knowledge.

There are different representations of the context depending on whether context is considered either as knowledge or as a process of contextualisation (Edmondson & Meech, 1993). Context as knowledge implies that we must distinguish between the knowledge effectively used at a given step of problem-solving (what we are calling proceduralised context) and contextual knowledge (the knowledge constraining the proceduralised context at that step). Considering context as a process – a viewpoint close to the previous one – supposes a distinction between knowledge, information and data. Data becomes information through the contextualisation process on the basis of the available knowledge at the time of the observation.

The context makes it possible to guide the focus of attention; that is, 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) is seen as being the mutual context that already exists between the agents (Maskery & Meads, 1992). The relationships between context and focus of attention are at the centre of the concerns of the community on "context-aware applications".

4 Modelling context

4.1 Introduction

The understanding of context is somewhat better now than ten years ago. We know that context must always be considered in reference to its use, that there are different types of context according to what we consider, and that the representation of context is strongly dependent on the knowledge to be represented and the reasoning that can be done with it. 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.

It is 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 organise emergency operations in a station than in a tunnel. When the reasoning yields this type of knowledge, it is easily proceduralised 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 mobilised to understand a given situated decision problem. By "situated" we mean in given, dated and well-specified circumstances. Clancey (1991) introduced the word "situated" into artificial intelligence. In its artificial intelligence sense, "situated cognition" emphasises 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 my views.

In this section, I present a view on context. First, I describe three parts of context and discuss the interest of such a distinction in the SART application. Second, I give another example of movement from contextual knowledge to proceduralised context. Finally, I contrast this view of context with related works in the literature.

4.2 The three parts of context

At a given decision-making step, Brézillon and Pomerol (1999) consider three types of knowledge, context being considered 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 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 proceduralised. We call it the *proceduralised context*. Figure 1 illustrates the three types of context.

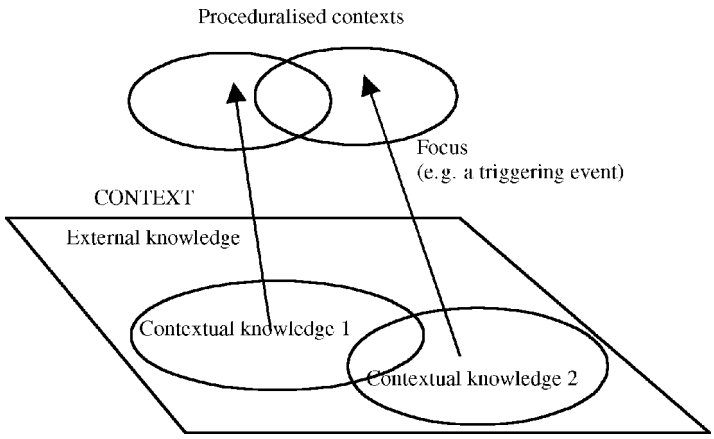

Figure 1 Different types of context

The proceduralised 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 proceduralised 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.

Contextual knowledge is more or less similar to what people generally have in mind about the term "context". 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. When the task becomes more precise, a large part of this contextual knowledge can be proceduralised 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 proceduralised. In our definition, the contextual knowledge is dependent on the situation (date, location and participants).

Contextual knowledge intervenes implicitly by constraining problem-solving. For example, operators that ensure the monitoring of the distribution of water in Paris noted that there was a peak in water consumption late each evening. The peak was reproducible every day but not predictable because it did not occur at exactly the same time. After an enquiry, they discovered that people use water for domestic needs (drinking a glass of water, washing dishes, watering flowers, going to the toilet and so on) during the advertisements introduced in the TV movie. The introduction of advertisements in the movie depends on the organisation of the movie scenario. Such knowledge (the link between peak consumption and advertisements on TV) has a contextual nature for water distribution. Contextual knowledge constrains a given step of problem-solving (water distribution at advertisement time in the example) without intervening in it explicitly.

Contextual knowledge appears back-stage, whereas the proceduralised context is front-stage, in the spotlight. It is noteworthy that, as far as engineering is concerned, only the proceduralised context matters, but contextual knowledge is necessary because this is the raw material from which proceduralised context is made. In a sense, the proceduralised context is the contextual knowledge activated and structured to make diagnoses, decisions and actions.

In the SART application, the goal was to support the operator responsible for a subway line when an incident occurs. Figure 2 gives a very partial view of the solving of the incident "Ill traveller in a

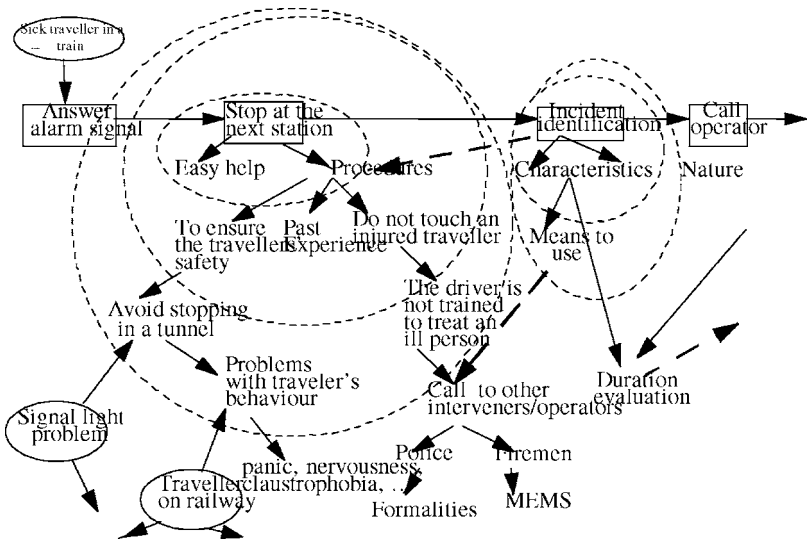

Figure 2 Context-based representation of the incident "Ill traveler in a train"

train." Ovals represent incidents and rectangular boxes represent steps in handling the incident (e.g. "Alarm signal").

Consider the step "Stop at the next station". This step – proceduralised context – is imposed on the driver because, for example, this corresponds to procedures. Procedures arise from operators' experience with similar incidents. For example, travellers' security is better ensured 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 because some travellers may suffer from claustrophobia and be inclined to leave the train to go on the railway (and thus may generate another type of incident) if the train stays in the tunnel for too long.

All the pieces of contextual knowledge are not at the same distance from the proceduralised context "Stop at the next station". For instance, "Procedures" is 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 2 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 proceduralised 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 the 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 during the solving.

Our context modelling using 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 realisation.
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 traveller") 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 such elements as "Past experience" and "Do not touch an injured traveller". 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.
4. 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 2 shows how such a relationship is established between "Ill traveller in a train" and other such 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 unrelated (e.g. "Ill traveller 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 2.

4.3 Movement between contextual knowledge and proceduralised context

In the SART application, the control operator faces the following concern in a normal situation:

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

In this task, the word *normal* has different meanings according to the context. The task can be regarded as routine and does not require special attention. Nevertheless, contextual knowledge about control is involved, such as:

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, which describes the environment of the problem with other pieces of knowledge, such as:

- 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 proceduralised context in which the incident is resolved.

At the announcement of an incident on the subway line, the piece k5 of contextual knowledge "incident-elimination know-how" enters the focus of attention and becomes a piece of the proceduralised context. Some pieces of external knowledge, such as the position of the incident on the line, also enter the focus of attention. The context also evolves to integrate such external knowledge as maintenance activity on the line, the number of trains on the line and help that is available. Thus the context of the diagnosis evolves from one step of the diagnosis to the next.

The operators' reasoning during 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, as 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 proceduralised context (entering the focus of attention). On the basis of this subset of proceduralised context pieces, they try to have a clear idea of all the alternatives, and make their decision that is made concrete as a sequence of actions.

Figure 3 represents how the proceduralised context is built from contextual knowledge during an interaction between two agents. The interaction context contains proceduralised context pieces in the

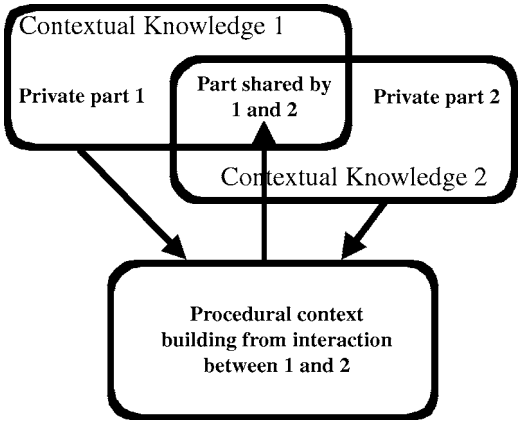

Figure 3 A representation of the interaction context

focus of attention of the two agents. These pieces of knowledge are extracted from the contextual knowledge of each agent; they are organised 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 proceduralised context contains all the pieces of knowledge that have been discussed and accepted (or at least made compatible, as noted in Karsenty & Brézillon, 1995) by all the agents. These pieces of proceduralised context then become part of the shared contextual knowledge of each agent, even if they do not remain within the focus of the proceduralised context.

The dynamic dimension of the context is essentially a movement between contextual knowledge and proceduralised context through a problem-solving or decision-making process. From one step to the next, a piece of contextual knowledge either enters the proceduralised context or becomes external knowledge. Conversely, a piece of proceduralised context may become either contextual knowledge or external knowledge. The proceduralised context may also evolve to integrate some knowledge that, up to now, neither has been proceduralised nor is contextual (i.e. external knowledge). Our view of context combines the two views in the literature (see the section on the two sides of context) and accounts also for the dynamic dimension of context.

4.4 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, allow 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. However, their approach does not integrate the dynamic dimension of context.

Gonzalez and Ahlers (1993, 1995, 1999) describe the development of a knowledge representation paradigm that is used to model the intelligent behaviour 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 an 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 incorporates 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 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 preliminarily 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 the light of a trainee's action. In general, tactical knowledge can be said to address time-critical 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 with 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, are stored in the context objects themselves. There are three levels of context that can be represented, and they are ordered hierarchically: 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 been recognised, and which has a prescribed set of procedures that must be carried out, sequentially, methodically, or arbitrarily. This work is quite close to my view of context, but contextual graphs go one step further as a unified formalism for representing knowledge, reasoning and context (see next section).

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) to make predictions about the situation, (2) to modulate the agent's behaviour, (3) to focus the agent's attention, (4) to influence an agent's choice of actions and (5) to determine how an agent should handle unanticipated events. An agent should be able to recognise 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 the most appropriate of these are retrieved and used to help the reasoner behave appropriately in its current context. Thus c-schemas contain information not only describing the context they represent, but also 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 combined as needed to adequately represent a particular situation. The major difference with case-based reasoning is how c-schemas are used: generalised cases are usually used as indexing structures, while c-schemas are problem-solving structures in addition to their role in memory organisation. Context is mainly considered as a way to cluster knowledge for search efficiency, to represent counter-factual or hypothetical situations, to circumscribe the effects of particular actions on particular situations and to direct an agent's focus of attention to salient features of a situation.

In this framework, Turner describes Context-Mediated Behaviour (CMB), an approach to context-sensitive behaviour developed over the past few years for intelligent autonomous agents. CMB is based on the idea that an agent should 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 recognised. In CMB, contexts are represented explicitly as contextual schemas (c-schemas). An agent recognises its context by finding the c-schemas that match it, and then the agent merges these c-schemas 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 behaviour 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. 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 makes it possible 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 the exclusion of 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 the 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 Glasgow (1997, 1998) proposed an algorithm that is based on a notion of relevance assessment and on a modified nearest-neighbour matching. Its modifications include: (1) grouping attributes 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 or 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 also 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 Glasgow, 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), 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.

5 Reasoning representation by contextual graphs

5.1 Introduction

This part of the work has been realised with L. Pasquier (Pasquier, 2002; Brézillon *et al*., 2003) in the framework of the SART application and extensions are given in Brézillon (2003). Initially we used a rule-based representation to describe incident-handling. This representation was not adapted to operators, and we then chose a tree representation. Decision trees were not the right kind of representation for operators either, but using trees allowed us to identify some particular points of the representation and the role of context. This operation has several main consequences for the structure of the representation and for the meaning of the model.

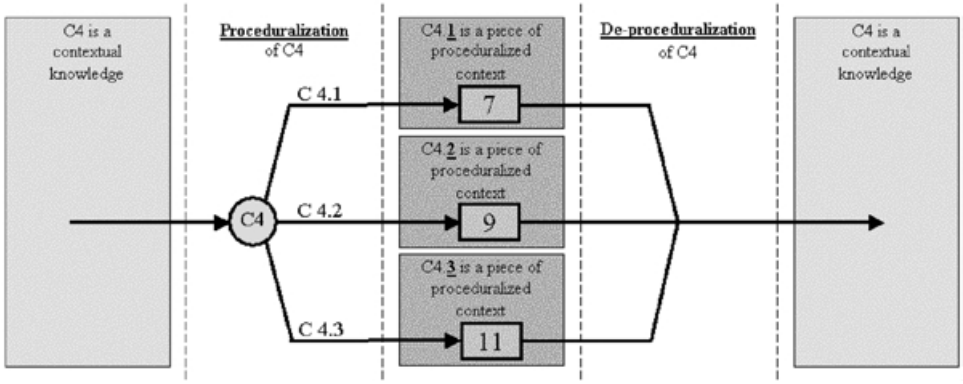

Figure 4 Proceduralisation and deproceduralisation

1. The contextual graphs are acyclic directed graphs, 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).
2. The size of the structure is easily controlled and consideration of a new contextual element will add some elements to the graph but not increase its size drastically as it would for a tree representation.
3. The structure introduces a dynamic model comparable to the dynamic model of the change between proceduralised context and contextual knowledge. Indeed, when two branches are merged, it means that the actions that are undertaken lead to a common situation from different contexts. The contextual elements attached to the different branches are proceduralised at the diverging node. They stay in this state for the different action sequences, because they intervene in the branch decisions. Finally, they are deproceduralised when the branches are merged. In this way we explicitly express the life duration of the contextual elements (see Figure 4).
4. Contextual graphs give a unified representation of procedures and practices. Moreover, they permit a learning process for integrating new situations by assimilation and accommodation. This learning process looks for the most similar incident-handling known and recombines the known elements to express the new incident-handling.
5. The sub-graphs 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. 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 with each other according to a given context for incident-solving.

Moreover, even in a part of a subgraph it may happen that the actions are only partially ordered. To represent this issue, we introduced the *parallel action grouping* symbols to represent action sequences that can be done in a different order or in parallel. This symbol is made of two parts: a divergent branching and a convergent branching; the parallel branches carry the temporally independent decision blocs.

Finally we obtain the following structure (Figure 5) that we call a *contextual graph*. (The explanation of the symbols used in the figure is given in the Annex.) This structure is called a contextual graph as a reminder 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 take to solve an incident; the different possible paths express the possible strategies according to the situation.

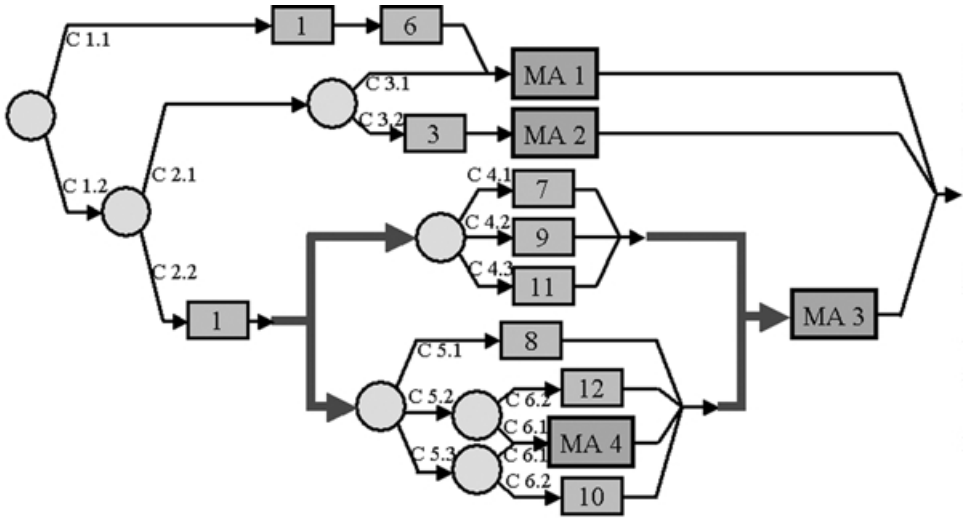

Figure 5 Contextual graph representing the official procedure for "lack of train power" incident

5.2 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 proceduralisation of the contextual knowledge is a process that makes the links explicit, especially the causal and consequential links, between contextual knowledge chunks; as such the links become a part of the proceduralised context. Thus the proceduralised 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 5) as representing the proceduralised context of actions, 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.

The arrows in bold in Figure 5 shows a parallel action grouping structure. 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. This structure can be thought of 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 is a general structure that can be found in the handling of different incidents.

For example, the block "Train aid" in Figure 6 is found in the handling 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).

This contextual graph reminds us that the structure makes the context and its dynamics explicit in order to improve decision-making. This representation is more compact than trees and seems to be well accepted by the operators.

5.3 Incremental acquisition of practices in the contextual graphs

Consider the sequence of actions 3–7–4–2 used by the operator for handling the incident (see Figure 7A). First, the system begins by identifying the path in the contextual graph that is concerned with the sequence of actions that have been entered. Second, the system tries to fit the sequence that has been entered to the sequence of actions on the path by associating actions 3, 7 and 4 (see Figure 7A). Then, the matching stops here because the system expects action 5 when the operator specifies action 2.

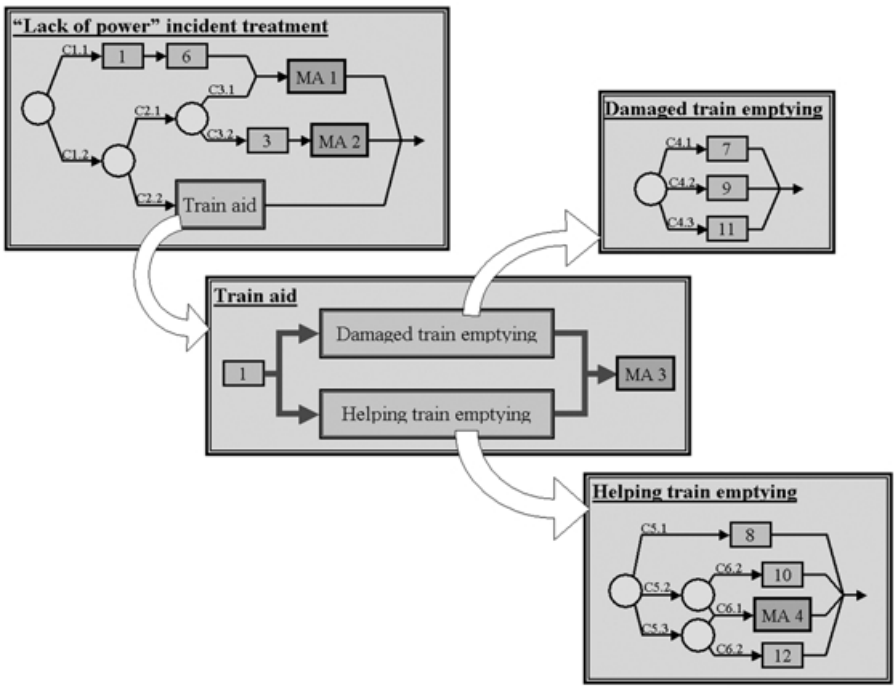

Figure 6 Set of contextual graphs used during "Lack of train power" incident resolution

Asking the operator for his reason, the system learns that a contextual element (C7 in Figure 7B) was not important until now because its instantiation (currently C7.1) did not matter. However, in the context of the current incident-solving, the contextual element C7 must be instantiated with C7.2. Thus, the system acquires the contextual element to take into account (C7, and the previous value C7.1 and the specific value C7.2) to distinguish later the two situations corresponding to actions 5 and 2.

5.4 Related notions

A more detailed presentation of this part is given in Brézillon (2002).

Scripts, frames and their relations provide a static description of the world. In such representations, the dynamics are given 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 to the representation of the procedures established by companies, procedures in which contextual considerations are not retained. Conversely, contextual graphs permit us to represent procedures, but also all the variants of the procedures that deal with contextual cues (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-graphs), except that a scene gives a static organisation of sub-graphs in a contextual graph, but does not represent a dynamic organisation of the sub-graphs. Moreover, there is no possibility of adding new paths in scenes as in contextual graphs.

The use of scenarios is a natural and efficient way to capture end users' needs in their context (Karat, 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. The context might include details about the workplace 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's 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

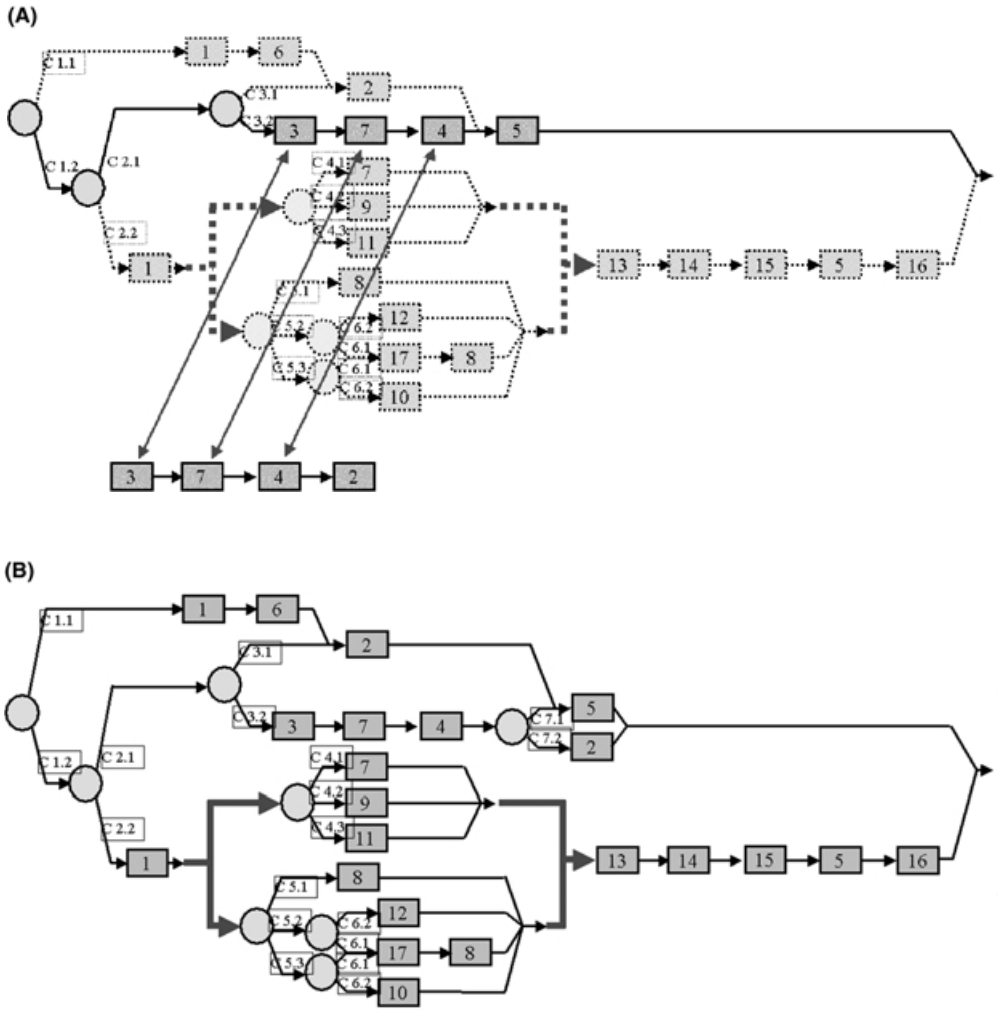

Figure 7 (A) The contextual graph at the initial phase of the learning process. (B) The contextual graph at the final phase of the learning process

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 details 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).

In design, patterns contain the results 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 us 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 spirit, to our notion of context when we consider a 3-tuple ⟨problem, context, solution⟩ as represented by contextual graphs.

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*. (1996) noted that every pattern is formulated as a rule that 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 contextualise knowledge.

A strong parallel can be made between contextual graphs and patterns, although that which 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 as well as dynamic aspects of problem-solving. However, in contextual graphs we try to use the dynamic aspects when patterns only give a representation of them with a weak means of using it. Third, both of them possess an organisation, in terms of items and sub-items, that appears as a kind of language. In contextual graphs the organisation presents the same organisation that permits the 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 at a level above that of a programming language. With respect to the notion of context, contextual graphs use it explicitly while patterns and frameworks introduce it implicitly. Globally, patterns help a designer to develop a computer system, when the contextual-graph formalism is to permit the computer system itself to reuse the experience.

The Unified Modelling Language (UML 2000, Eriksson and Penker, 1998) possesses some notions, such as activity graph, use case and scenario, that are quite similar to our approach based on contextual graphs. A first similarity relies on a structured description of problem-solving. Activities are organised 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 natural learning and explanation capabilities in the system. Both of them also authorise a representation of temporal branching. Beyond these shared properties, contextual graphs also have learning capabilities and possibilities of reorganisation 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.

The concept of place plays an important role in the affordance approach. 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. Affordances are what objects or things offer people to do with them (Gibson, 1977, 1979). The theory of affordances states that all the information necessary for correct perception is already present in the environment. Norman (1988) proposed the notion of perceived affordance such 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 in web browsers is that links are underlined and in blue, visited links are underlined and in 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 frustrating and annoying users. The perceived affordance is, in our interpretation, contextual knowledge and proceduralised context considered jointly. When we want to print a document, the printer icon is similar to the proceduralised context, and its associated function the contextual knowledge.

The notion of scheme was proposed first by Kant around 1800, with an emphasis on its temporal dimension (Eco, 1997). This notion plays an important role for structuring operators' activity; Béguin (1994) for drawers, Galinier (1996) for truck drivers and Duvenci-Langa (1997) for workers on tool machines have all identified schemes of activity. A scheme is composed of a structure of actions, but also other things such as the means used to accomplish the actions. 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 makes it possible to achieve its goal depending on the context. Each scheme organises the activity around an object and can call other schemes to complete specific sub-goals. A contextual graph, as a scheme, makes it possible to:

- represent and clearly organise operators' activities and all their variants (procedures and practices),
- include automatic learning and adaptation capabilities in a system and
- 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 database 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 latter 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 closest to the official procedure. It continues the path selection up to the end of the contextual graph and returns the path(s) found. It presents the possible sequences of action, representing the integrated schemes as expandable actions.

6 Conclusion

The two questions of contextual knowledge and knowledge-sharing have received wide consideration in recent years. In this paper I have addressed these two topics through intelligent assistant systems and, more specifically, by using my experience in the development of intelligent assistant systems for process control. As a result of these analyses, I stress the dynamic aspect of the contextual knowledge. I propose defining contextual knowledge as a possibly unlimited, personal and situated set of relevant knowledge involved in problem-solving. Part of this contextual knowledge is proceduralised to enable cooperation and this results in a shared, limited proceduralised 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 proceduralised 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 & Pomerol, 1998) is to use proceduralised context for situations which are close or similar to a reference situation (i.e. a kind of context-based reasoning). Using this definition I show how contextual knowledge is partially proceduralised in cooperative settings. The proceduralisation process itself is a cooperative process, and I show how proceduralised context is incrementally enriched, and I examine the role of explanation in this process.

My 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 with the AI community reinforces this; there is now another community, which is concerned with context-aware applications and interested in context in a more hardware-oriented way than in AI. Each community tackles ambitious challenges. However, it is clear that neither of the two communities is totally right. It is necessary to find a generalisation 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 centralised part and mobile parts attached to a user or group of users; and hardware and interaction among pieces of hardware.

From the observation of an operator's actions in accomplishing 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 person goes back home, the system can recall some tasks to do on the way with respect to the location. For example, the system can detect that the person will be soon at the level of the baker and tells the person that he needs to buy bread before going home. Moreover, having support from the operator in the accomplishment of a task may help the system to support him in the accomplishment of other tasks. Here a system will have to make compatible the context of the task (as in context-aware applications) and the context of the user (as in artificial intelligence).

Beyond the support that the system can bring to a user by managing his personal context in relation 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 in which an agent interacting with a user can require 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 me that an important point of making context explicit is the adjustment of all users' contexts to make compatible their interpretations and understandings on a given problem (Karsenty & Brézillon, 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.

The granularity of context is a kind of density measure that can be used as a function of the distance from 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 the Sun and the Earth?" Fisheye views (Pook *et al*., 2000) are one way of integrating the context and the focus into a single view. Some of the information surrounding the focus is shown following the rule that 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 from the focus of attention.

One sometimes 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 to the focus of attention and highly detailed, and (2) the "distant" context that is general and with less detail. This approach is also found in other domains. For example, van Dijk (1998) presents a similar position on political discourses with:

- A local or micro context (often 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), organisations, institutions and even whole states and nations.

We extend this notion of granularity in the situation of collaborative work where we distinguish different types of context at different levels. Movement from one context to another is ensured by the proceduralisation of a part of the knowledge from the first context to the second context. It is a way of managing 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.

References

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)* 351–362.

Alexander, C, Ishikawa, S, Silverstein, M, Jacobson, MI, Fiskdahl-King, S and Angel, S, 1996, *A Pattern Language* 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).

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.

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)* 61–69.

Brézillon, P, 1990, "Interpretation and rule packet in expert systems: application to the SEPT expert system" in S Ramani, R Chandrasekar and KSR Anjaneyulu (eds) *Knowledge Based Computer Systems* **444** 78–87. Springer-Verlag.

Brézillon, P, 1996, *Context in Problem Solving* Technical Report 96/29, LAFORIA, University Paris 6 (available at ftp://ftp.ibp.fr/ ibp/reports/ laforia.96/laforia.96.29.ps).

Brézillon, P, 1999, "Context in human-machine problem solving: a survey" *Knowledge Engineering Review* **14**(1) 1–34.

Brézillon, P, 2002, *Modeling and Using Context: Past, Present and Future* Research Report LIP6 2002/010, University Paris 6 (http://www.lip6.fr/reports/lip6.2002.010.html).

Brézillon, P, 2003, *Contextual Graphs: A Context-Based Formalism for Knowledge and Reasoning Representation* Research Report LIP6, University Paris 6 (forthcoming).

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

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

Brézillon, P and Pomerol, J-Ch, 1996, "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) 67–75.

Brézillon, P and Pomerol, J-Ch, 1998, "Using contextual information in decision making" in G Widmeyer, D Berkeley, P Brézillon and V Rajkovic (eds) *Context-Sensitive Decision Support Systems* Chapman & Hall.

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

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)* 209–222 (also available at http://www-poleia.lip6.fr/ brezil/Pages2/CONTEXT-97/index.html).

Brézillon, P, Pomerol, J-Ch and Pasquier, L**,** 2003, "Learning and explanation in a context-based representation: application to incident solving on subway lines" in C Faucher, L Jain and N Ichalkaranje (eds) *Innovative Knowledge Engineering* Springer-Verlag (forthcoming).

Carroll, JM, 1995, "Introduction: the scenario perspective on system development" in JM Carroll (ed.) *Scenario-Based Design: Envisioning Work and Technology in System Development* J Wiley & Sons.

Clancey, WJ, 1983, "The epistemology of a rule-based expert system: a framework for explanation" *Artificial Intelligence* **20**(3) 197–204.

Clancey, WJ, 1991, "Situated cognition: stepping out of representational flatland" *AI Communications* **4**(2–3) 109–112.

Compton, P and Jansen, B, 1988, *Knowledge in Context: A Strategy for Expert System Maintenance* Lecture Notes in Artificial Intelligence, Springer-Verlag.

de Kleer, J, 1987, "An assumption based truth maintenance system" in M Ginsberg (ed.) *Readings In Nonmonotonic Reasoning* Morgan Kaufmann.

Degani, A and Wiener, EL, 1997, "Procedures in complex systems: the airline cockpit" *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans* **27**(3) 302–312.

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.

Eco, U, 1997, *Kant et l'ornithorynque* Grasset.

Edmondson, WH and Meech, JF, 1993, "A model of context for human-computer interaction" *Proceedings of the IJCAI-93 Workshop on Using Knowledge in its Context* Technical Report 93/13, LAFORIA, University Paris 6, 31–38.

Erickson, T, 1995, "Note on design practice: stories and prototypes as catalysts for communication" in JM Carroll (ed.) *Scenario-Based Design: Envisioning Work and Technology in System Development* J Wiley & Sons 37–58.

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

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

Forslund, G, 1995, "Toward cooperative advice-giving systems: the expert systems experience" Ph.D. thesis 518, Linkoping University, Sweden.

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.

Gibson, JJ, 1977, "The theory of affordances" in R Shaw and J Bransford (eds) *Perceiving, Acting and Knowing* Erlbaum.

Gibson, JJ, 1979, *The Ecological Approach to Visual Perception* Houghton Mifflin.

Gonzalez, AJ and Ahlers, RH, 1993, "A context-based representation of tactical knowledge for use in simulation-based autonomous intelligent platforms" *Proceedings of the Interservice/Industry Training Systems Conference* 81–90.

Gonzalez, AJ and Ahlers, RH, 1995, "Context-based representation of intelligent behavior in simulated opponents" *Proceedings of the Computer Generated Forces & Behavioral Representation Conference* 489–493.

Gonzalez, AJ and Ahlers, R, 1997, "Context-based representation of intelligent behavior in training simulations" *Transactions of the Society for Computer Simulation International* **15**(4) 153–166.

Gruber, T, 1991, "Justification-based knowledge acquisition" in H Motoda, R Mizoguchi, J Boose and B Gaines (eds), *Knowledge Acquisition for Knowledge-Based Systems* IOS Press.

Guha, RV, 1991, *Contexts: A Formalization and Some Applications* MCC Technical Report ACT-CYC-423–91.

Hatchuel, A and Weil, B, 1992, *L'Expert et le système* Economica.

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

Hoc, J-M, 1996, *Supervision et contrôle de processus: La Cognition en situation dynamique* Presses Universitaires de Grenoble.

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

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

Jordan, T, Raubal, M, Gartrell, B and Egenhofer, MJ, 1998, "An affordance-based model of place in GIS" *Proceedings of the 8th International Symposium on Spatial Data Handling, SDH'98* 98–109.

Jurisica, I, 1994, "Context-based similarity applied to retrieval of relevant cases" *Proceedings of AAAI Fall Symposium Series on Relevance* 21–23.

Jurisica, I and Glasgow, J, 1997, "Improving performance of case-based classification using context-based relevance" *International Journal of Artificial Intelligence Tools* **6** 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* Vol. II, 535–546.

Karat, J, 1995, "Scenario use in the design of a speech recognition system" in JM Carroll (ed.) *Scenario-Based Design: Envisioning Work and Technology in System Development* J Wiley & Sons.

Karsenty, L, 1994, "L'Explication d'une solution dans les dialogues de conception" Ph.D. thesis, University Paris 8.

Karsenty, L and Brézillon, P, 1995, "Cooperative problem solving and explanation" *International Journal of Expert Systems With Applications* **4** 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, 115–124.

Laird, JE, Newell, A and Rosenbloom, PS, 1987, "Soar: an architecture for general intelligence" *Artificial Intelligence* **33** 1–64.

Leake, DB, 1992, *Evaluating explanations* Lawrence Erlbaum Associates Inc.

Leplat, J, 1985, "Erreur humaine, fiabilité humaine dans le travail" in A Colin *Collection Universitaire*.

McCarthy, J, 1993, "Notes on formalizing context" *Proceedings of the 13th IJCAI* 555–560.

McDermott, J, 1982, "R1: a rule based configurer of computer systems" *Artificial Intelligence* **19** 39–88.

Mark, B, 1988, "Explanation and interactive knowledge acquisition" *Proceedings of AAAI'88, Workshop on Explanation* 163–165.

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

Menzies, T, 1996, Assessing responses to situated cognition. *Proceedings of the Banff Knowledge Acquisition Workshop*. `http://citeseer.nj.nec.com/menzies96assessig.html`

Mittal, VO and Paris, CL, 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* 87–99.

Molière, 1670, *Le bourgeois Gentilhomme*, Acte 2, Scene 4.

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

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.

Pasquier, L, 2002, "Modélisation de raisonnement tenus en contexte: Application à la gestion d'incidents sur une ligne de métro" Thèse de l'Université Paris 6.

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). In: P Bouquet, P Brezillon, L Senatini, M Benerecetti, F Castellani (Eds) Springer, New York, 499–502.

Patil, RS, Szolovits, P and Schwartz, WB, 1981, "Causal understanding of patient illness in medical diagnosis" *Proceedings of the 7th International Joint Conference on Artificial Intelligence* 893–899.

Pomerol, J-Ch, 1997, "Artificial intelligence and human decision making" *European Journal of Operational Research* **99** 3–25.

Pook, S, Lecolinet, E, Vaysseix, G and Barillot, E, 2000, "Context and interaction in zoomable user interfaces" *AVI 2000 Conference Proceedings (ACM Press)* 227–231.

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

Schmidt, DC, Johnson, RE and Fayad, M, 1996, "Software patterns" *Communications of the ACM* (Special issue on patterns and pattern languages) **39**(10) 37–39 (`http://www.cs.wustl.edu/ schmidt/ CACM-editorial.html`).

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

de Terssac, G, 1992, *Autonomie dans le travail* Presses Universitaires de France.

Turner, RM, 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, 141–153.

Turner, RM, 1997, "Determining the context-dependent meaning of fuzzy subsets" *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)* 233–242.

Turner, RM, 1998, "Context-mediated behavior for intelligent agents" *International Journal of Human-Computer Studies* (Special issue on using context in applications) **48**(3) 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* 53–59.

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

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

Van Dijk, TA, 1998, "Cognitive Context Models and Discourse" in Maxim Stamenov (ed.) *Language Structure, Discourse and the Access to Consciousness* Benjamins.

Woods, DD, Roth, EM and Benett, K, 1990, "Explorations in joint human-machine cognitive systems" in S Robertson, W Zachary and JB Black (eds) *Cognition, Computing and Cooperation* Ablex.

Annex

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

Actions	
1	Residual traffic regulation
2	Damaged train continues with travellers
3	Damaged train continues with travellers until a steep incline
4	Damaged train restarts without travellers
5	Damaged train stays at end station
6	Repair damage
7	Exit of the travellers out of the damaged train
8	Exit of the travellers out of next train
9	Exit of the travellers out of damaged train via available cars
10	Exit of the travellers out of next train via available cars
11	Exit of the travellers out of damaged train via track
12	Exit of the travellers 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

Macro-actions		Action 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