ELSEVIER

European Journal of Operational Research 136 (2002) 290–298

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Reasoning with contextual graphs

Patrick Brézillon *, Laurent Pasquier, Jean-Charles Pomerol

Laboratoire d'Informatique de Paris 6, Case 169, Université Paris 6, LIP6, 4, Place Jussieu, 75252 Paris Cedex 05, France

Received 15 December 1999; accepted 15 October 2000

Abstract

Decision trees allow the modeling of event-dependent reasoning, but do not consider the dynamics of contextual changes in reasoning. In the framework of the SART project, which aims at the design and development of an intelligent support system for subway regulators, we have to model highly contextual reasoning. We introduce the notion of contextual graph to take into account temporal and context-based reasoning. This model relies on observed reasoning modes in which the context and its dynamics are essential. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Artificial intelligence; Decision support system; Transportation; Context-based reasoning; Contextual graphs

1. Introduction

Incident management on a subway line is a twofold task. When an incident occurs, the operator has to: (i) solve and clear the incident; and (ii) simultaneously manage the train traffic on the part of the line that is not concerned by the incident. To achieve these two tasks, the decision maker's reasoning must include many different parameters and contextual information.

The SART project (French acronym for support system for traffic control) aims at the design and development of an intelligent system to support the operators, who are responsible for the traffic on a subway line, when an incident occurs (Brézillon et al., 1997, 1998). SART has a multi-agent architecture and is composed now of three agents: a line configurator, a traffic simulator and an incident manager. This paper focuses on results obtained for the incident manager for which we have to make the context explicit.

Contextual information plays an important role in the management of incidents. Brézillon and Pomerol (1999) point out three types of context, namely, the external context, the contextual knowledge and the proceduralized context. These three types of context allow to model the various information needed at each step of the incident resolution.

The modeling of operators' reasoning, including the contextual knowledge used, is a difficult task because the operators use a number of contextual clues for the application of the procedures.

* Corresponding author. Tel.: +31-4427-7008; fax: +31-4427-70900.

E-mail address: patrick.brezillon@lp6.fr (P. Brézillon).

PII: S0377-2217(01)00116-3

Our model associates case-based reasoning and decision trees and includes an explicit representation of contextual knowledge. In Section 2, we introduce the notions used to model the context in the reasoning. Starting from the representation of the reasoning as a decision tree in Section 3, we present in Section 4 our work on contextual graphs and its application to subway line control. Section 5 gives few words about the implementation phase before the conclusion.

2. Context and reasoning

Let us just remind some of the main proposals reviewed in Brézillon (1999). We do not quote the numerous works originating from semantic studies (e.g., Rastier, 1996). At the boundary between cognitive science and engineering several ideas have been put forward. At least two ways were followed, on the one hand, a contextual information is seen as a chunk of knowledge: script (Schank and Abelson, 1975), scheme (Schank, 1982; Dyer, 1990; Stacey et al., 1996), on the other hand, as a net of tangled concepts (Sowa, 1984) the notion of frame being at the crossroad between the two ways.

Recently, for the sake of engineering applications, we have tried to understand and model the role of context in reasoning (Pomerol and Brézillon, 1999). To do so, we worked on the control of a subway line with Zanarelli (1999) (Pasquier, 2000). Starting from these observations, we defined *contextual knowledge* as the part of the context that is relevant in a given situation for a given operator (Brézillon and Pomerol, 1999). This may be seen as the subset of the context in which the operator can find every chunk of knowledge for reasoning about or interpreting and explaining the situation. The complementary part in the context is called external context.

Brézillon and Pomerol (1999) also defined the *proceduralized context* as the proceduralized part of the contextual knowledge, which is considered explicitly, with causal and consequential links, *at a given step of the problem solving*. Indeed, the operators do not consider at each moment all the contextual pieces of knowledge during an incident solving. The proceduralized context is the part of the contextual knowledge on which the operators are focusing on at a given step of the incident solving (see Fig. 1). This notion is relative to each operator, to the current situation and to the moment at which the operator is working. The proceduralized status of a chunk of context is not permanent since a contextual piece of knowledge is proceduralized when the operator focuses on it and goes back to its initial status of back-stage contextual knowledge when it is no longer mobilized in the reasoning.

At the RATP, most of the incidents are common (object on the track, lack of power supply, etc.). Thus, the company has established mandatory procedures for incident solving on the basis of its experience. For example, Brézillon et al. (1997) discuss the procedure and the knowledge involved in the incident "Sick traveler in a train" (see Fig. 2).

In this case, the driver must continue up to the next station because travelers are more easily rescued 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 travelers may have behavioral troubles such as claustrophobia and could leave the train to wander about on the track (and thus can generate another type of incident such as "Person on the track"). These pieces of contextual knowledge, which are not necessarily expressed, result in more or less proceduralized actions that are compiled as parts of the proced-

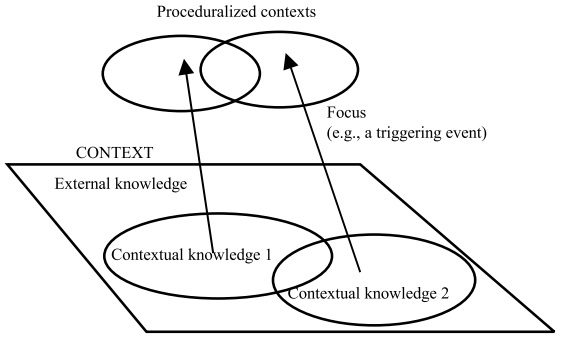

Fig. 1. The three types of context (from Brézillon and Pomerol, 1999).

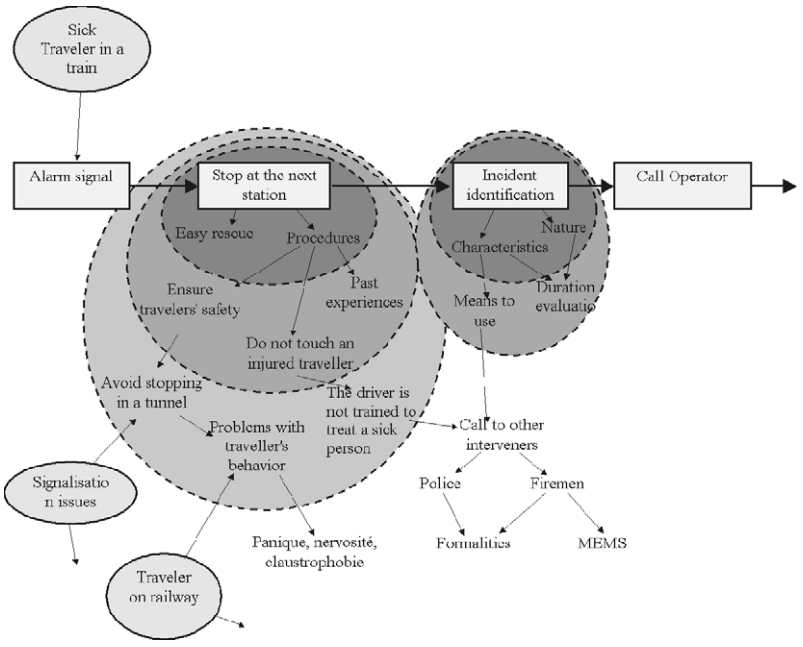

Fig. 2. Context-based representation of the incident "Sick traveler in a train".

uralized context. Very often many pieces of the proceduralized context are structured into comprehensive knowledge pieces about actions. The generation of relevant explanations implies that the system must acquire all the pieces of contextual knowledge intervening in the proceduralized context and the way in which they are structured. This is one of the main problems of knowledge acquisition. As such, mandatory procedures can be interpreted as proceduralized contexts.

For simple incidents, a unique procedure is sufficient to solve the incident. For example, the incident "fallen object on the track", implies the procedure: power cut-off, an employee of the RATP goes down on the trackbed to remove the object and once gone back on the platform, the power is reestablished.

However, there is no global procedure for complex incidents, only a set of procedures, each procedure solves a part of the incident. For example, when a train cannot move in a tunnel, there are procedures to evacuate the travelers at the nearest station, for moving the damaged train by another train, etc.

Some procedures are sequential, but others may be accomplished in any order. For example, when a train must push a damaged train, both trains must be empty but the order in which the travelers of the two trains are evacuated is not important and mainly depends on the context in which each train is. What is important is that the two actions must be accomplished before the damaged train clearing of the track.

As a consequence, each operator develops his own practice to solve incidents, and one observes almost as many practices as operators for a given incident solving because each operator structures the set of procedures in order to take into account the current context, which is, of course, particular and specific. For example, is the next station a connecting one or not? (In a connecting station, travelers can escape from an incident on a subway line by using another line.) Thus, cases that are similar in one context may be totally dissimilar in another as already quoted by Tversky (1977).

Another problem concerns non-written rules with which it is difficult to cope. In many working

processes, human beings develop genuine procedures to reach the efficiency that decision makers intended when they have designed the system. Some parts of this practice are not coded (Hatchuel and 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 reasoning followed. The validation of these unwritten rules is linked more to the result than to the procedure, de Terssac (1992) spoke of logic of efficiency.

In our application, in addition to all these points, we have to consider that even if the decision making in the incident solving belongs to the operator who is concerned, there is a collective solving of the incident because other operators in the control room follow the incident solving and intervene, say, by recalling to their colleague something not to forget, by suggesting to ask information on a specific point. This part of the incident solving is not recorded and is difficult to deal with in modeling.

3. Decision tree representation of the reasoning

We initially modeled the possible sequences of actions by structures inspired from decision trees (Raïffa, 1968). The nodes represent the possible states of nature (circles in Fig. 3, see also Table 1 for the meaning of the symbols) and the actions to undertake are on the corresponding branches depending on the events (square and rectangular boxes in Figs. 3 and 4).

Common sub-sequences of actions are grouped into macro-action structures (the large rectangular boxes in Fig. 4) and almost all the actions are postponed at the end of the branches (Brézillon and Pomerol, 1999). For example, Fig. 4 presents the decision tree obtained in case of a failure of one or several motor coaches. (A typical five-cars train is made up of three motor coaches and two trailers.) In such a case, the driver has to try to restart the failed motors. If all motor coaches restart, the train continues its run. If not all motor coaches work but enough engine power is available, the train continues its run only if there is no steep incline, otherwise it restarts without travelers. Finally, if

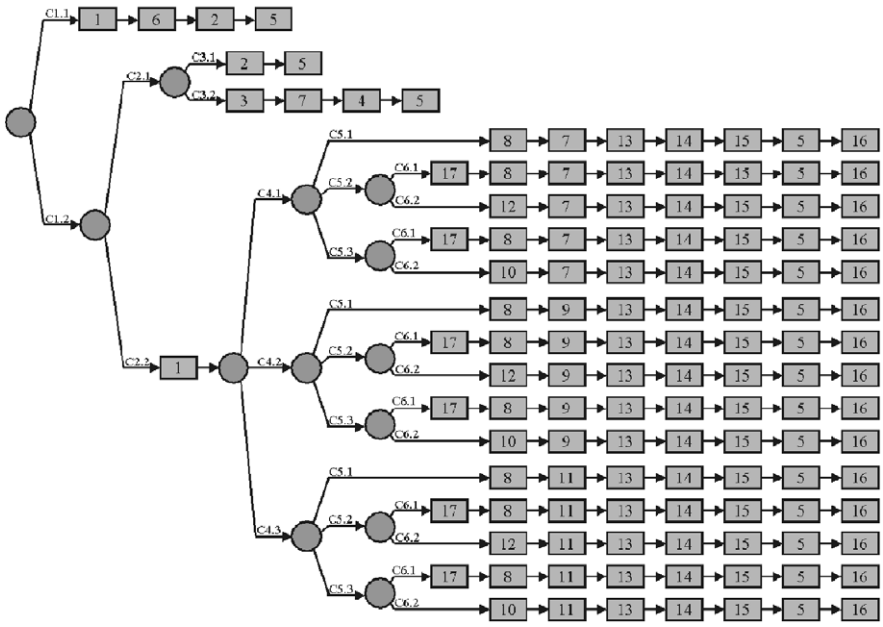

Fig. 3. A tree in which many actions are postponed to the end of the branches.

Table 1
Meaning of the symbols

Actions		Events	
1	Residual traffic regulation	C11	Immediate repair possible
2	Damaged train continues with travelers	C12	Immediate repair impossible
3	Damaged train continues with travelers until a steep incline	C21	Enough motor coaches available
4	Damaged train restarts without travelers	C22	Not enough motor coaches available
5	Stable damaged train at end station	C31	No steep incline between damaged train and end station
6	Repair damage	C32	Presence of steep incline until end station
7	Exit of the travelers out of the damaged train		
8	Exit of the travelers out of next train	C41	Damaged train at station
9	Exit of the travelers out of damaged train via available cars	C42	Damaged train under tunnel
10	Exit of the travelers out of next train via available cars	C43	Damaged train partially at station
11	Exit of the travelers out of damaged train via track		
12	Exit of the travelers out of next train via track	C51	Next train at station
13	Next train joins damaged train	C52	Next train under tunnel
14	Link both trains	C53	Next train partially at station
15	Convoy return to end station		
16	Disassemble convoy	C61	Presence of a station between damaged train and next train
17	Next train goes to next station	C62	No station between both trains

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

there is not enough motor coach power available, a train (usually the next one) pushes the damaged train up to the end-station.

In our application, the final goal of all branches of the decision tree is the same: ''Return as quickly as possible to a normal operating mode'' and the branches express only different means to reach the final goal according to the states of nature. Actually, the operators very often try to recover familiar procedures or actions so that it is frequent that different branches in the tree can be merged. For example many situations result into ''emptying a train'', which entails that the action ''emptying a train'' is a convergence point of several branches carrying out actions that are necessary before emptying a train (e.g., switching off power).

We arrive now to a main point of our representation. Each event at an event node carries on a part of the uncertainty of the situation. For example, C1, in previous figures, means that either an immediate repair would be possible or not. There are two ways to manage this uncertainty either assess some probabilities for each event, which is the usual view in decision theory or consider that, anyway, the two events are possible depending on the circumstances. Our graph must provide an answer for any possible circumstances whatever the probability is. The problem of an operator is to get an adapted answer as soon as he knows what the circumstances are. Thus, the main problem is to diagnose the exact situations according to the contextual information reaching the operators. For this reason, we considered that each branch actually describes a contextual knowledge, which becomes more and more accurate as long as the branch is followed. We are no

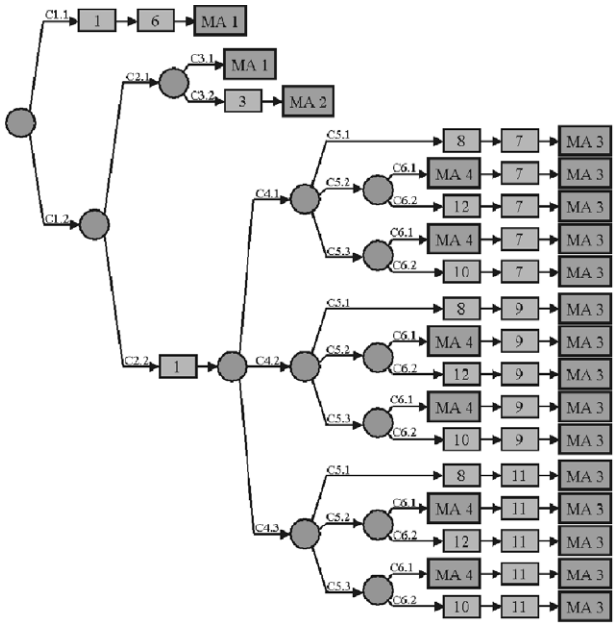

Fig. 4. An example of decision tree.

longer interested by the probability of a branch but by the possibility to determine, as soon as possible, on which path the operator is, to determine what is the next action to undertake. In other words, we come back to the original idea of Savage (1954), namely, that each nature state (a sequence of events is a nature state) describes a state of the world or using our words, a context for action.

At this step there is no probabilities on the events, the main purpose is to describe, with the maximum of parsimony, all the possible contexts in which the decision has to be made. For example, the branch with {C12, C21, C32} describes a context in which there is no immediate repair possible, but enough motor coach power is still available and a steep incline. For this reason *we will talk hereafter of context nodes instead of event nodes and of context instead of nature's state*. The action postponement observed in Figs. 3 and 4 amounts to relating each decision to a state of nature, here the context described in the branch. Thus, our representation tends to stick to operators' behaviour. In some cases, especially at the beginning of the tree, making decision under uncertainty is probably interesting, but by trying to gather relevant information the operators endeavor to make decision under certainty. This means that when undertaking an action in the tree they consider that, due to the contextual information they got, the state of nature between the root and the action undertaken is the true state of nature.

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

4. 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 event (see the annex). The contextual graph is intended to represent the part of the context that we have denoted proceduralized context, i.e., *context chunks ready to be used for action.*

The proceduralization of the contextual knowledge is a process that makes explicit the links, especially the causal and consequential links, between contextual knowledge chunks and as such the links become a part of the proceduralized context. Thus, the proceduralized context appears as a kind of compiled knowledge that the system will have to decompile to explain its reasoning. Consequently, one can regard the above contextual tree (Fig. 4) as representing the proceduralized context, because for each context represented by a sequence of contextual nodes the implicit reasoning about causes and consequences implies that the action to undertake is defined without ambiguity. In other words, each sequence of contextual nodes along a branch triggers some actions due to rationales which are not represented on the graph but are generally known and compiled in the operators' mind (proceduralized knowledge). Thus, the contextual graph explicitly represents the reasoning involved in the proceduralized context (Fig. 5).

We have already mentioned that, after a remedial action, different paths of the graph converge in a unique path. In some sense, these actions, if they are successful, abolish the problem related to a given proceduralized context. For example, the proceduralized contexts C4.1 and C4.2 in Fig. 6 are respectively solved by actions 7 and 9. Hereafter, the contextual element C4 is deproceduralized. Thus, our representation takes into account the dynamics of the proceduralization.

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

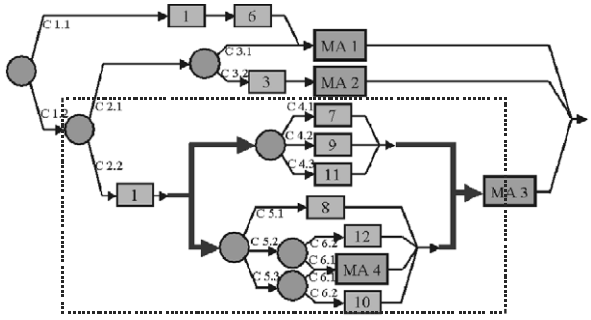

Fig. 5. The contextual graph corresponding to Fig. 1.

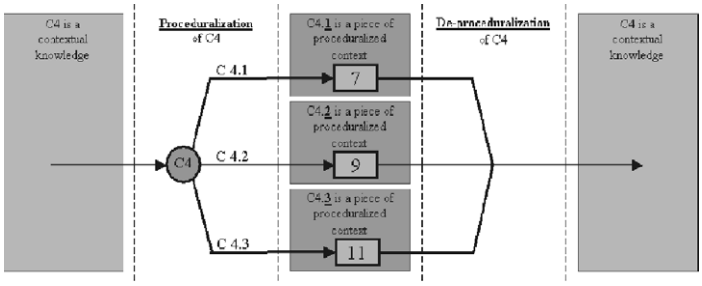

Fig. 6. Converging contexts.

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

This structure is shown in Fig. 5 (in the slashed square). One can see the opening and closing square bracket in dashed lines between action 1 and macro-action MA3. Both branches are composed of a contextual sub-graph. The upper one tells how to empty the damaged train, the lower one shows how to empty the helping train. The actions of both sequences are locally and independently carried out. To detect such temporal branching in real applications, one has to detect sequences of actions that may actually be indifferently carried out in an order or another. For example, if you see that sometimes the operators decide to do a sequence "A–B" and sometimes the sequence "B–A," one can suppose that actions "A" and "B" are independent, but must be accomplished before a given step. Such situations are easy to detect automatically from records.

In Fig. 5, the structure in the slashed square has its own signification. It can be thought as an independent plan named "assistance to a damaged train". This plan has a goal (to push a damaged train with another train up to the end-station) and an explanation about the way to carry it out. This explanation results of the proceduralization of the context contained in the contextual sub-graph shown in Fig. 5. This sub-graph could be found again in other contextual graphs. Each sub-graph can be considered for its own, by operators as well as by designers. This is a chunk of knowledge that can be considered as an elementary piece of knowledge. We have elsewhere argued that, on the one hand, such an elementary sub-graph can be considered as an atomic task or a scheme of action (Pasquier et al., 2000). On the other hand, our representation is reminiscent to Sowa's conceptual graphs (Sowa, 1984) with their mechanisms of aggregation and expansion. Macro-actions are then considered as a particular case of action structure in which the contextual graphs are reduced to sequences of actions not intertwined with context nodes. However, one must notice that our representation is simpler than Sowa's one because our graph is oriented and the path gives the reasoning followed by the designer according to the different contexts.

5. Validation and implementation

As already mentioned, this representation is inspired by operators' representation in an intelligent assistant system for the control of a subway line (SART). At the beginning, we tried to represent the operators' knowledge by decision trees, but a further study (Zanarelli, 1998) quickly shown that the problem was not really a problem of decision under uncertainty but a problem of diagnosing, as soon as possible, the real state of nature. This was done by merging all contextual chunks relevant to a situation. This is the reason why we introduced the notion of contextual graph. This representation proved rapidly that it was perfectly accepted and understood by the operators (Zanarelli et al., 1999) and that it describes accurately the intertwining of contextual clues gathered by the operators and of actions.

The notion of contextual graph and sub-graph heavily simplifies the representation. The significance of the sub-graph meets operators' practice and facilitates re-using. Based on this representation, the incident manager module is now almost completed.

6. Conclusion

By changing event nodes into context nodes, the contextual graphs allow a clear representation

of multiple actions depending on context. Moreover, the contextual graph representation takes into account the dynamics of the proceduralization. Another advantage of contextual graphs is their ability to deal with large structure usually encountered in industrial applications. This representation is easily understood by the operators (and designers) and well accepted by them and their hierarchy. The division into sub-graphs makes sense. This flexibility and modularity is another advantage of the contextual graphs. We do believe that this representation is well adapted to engineering and especially in troubleshooting contexts.

Acknowledgements

The SART project enters an agreement between the University Paris 6 (France) and the Federal University of Rio de Janeiro (FURJ, Brazil), a contract between the University Paris 6 and the French company of public transport in Paris (RATP), and another contract between the FURJ and Metrô-Rio that manages the subway in Rio de Janeiro. Grants are provided by the RATP, the COFECUB and the French Foreign Ministry in France and CAPES in Brazil. We also thank J.-M. Sieur at RATP, C. Gentile, I. Saker and M. Secron, Ph.D. students working on the SART project.

References

Brézillon, P., 1999. Context in problem solving: A survey. The Knowledge Engineering Review 14 (1), 47–80.

Brézillon, P., 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., Secron, M., 1997. SART: A system for supporting operators with contextual knowledge. In: Proceedings of the First International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97), pp. 209–222.

Brézillon, P., Pomerol, J.-Ch., Saker, I., 1998. Contextual and contextualized knowledge: An application in subway control. Using Context in Application (special issue). International Journal on Human–Computer Studies 48 (3), 357–373.

Dyer, M., 1990. Intentionality and computationalism minds machines. Journal of Experimental and Theoretical Artificial Intelligence 2, 303–319.

Hatchuel, A., Weil, B., 1992. L'expert et le Système. Economica, Paris.

Pasquier, L., 2000. Modélisation de raisonnements tenus en contexte et application aux agents d'aide à la gestion d'incidents de SART. LIP6 Report, Paris, France.

Pasquier, L., Brézillon, B., Pomerol, J.-Ch., 2000. From representation of operational knowledge to practical decision making in operations. In: Carlson, S., Brézillon, P., Humphreys, P., Lundberg, B., McCosh, A., Rajkovic, V. (Eds.), Decision Support through Knowledge Management. ISBN 3-901882-11-1, pp. 301–320.

Pomerol, J.-Ch., Brézillon, P., 1999. Dynamics between contextual knowledge and proceduralized context. Modeling and Using Context, CONTEXT-99. In: Bouquet, P., Serafini, L., Brézillon, P., Benerecetti, M., Castellani, F. (Eds.), Lecture Notes in Artificial Intelligence, vol. 1688. Springer, Berlin, pp. 284–295.

Raïffa, H., 1968. Decision Analysis. McGraw-Hill, New York.

Rastier, F., 1996. Le problème épistémologique du contexte et le statut de l'interprétation dans les sciences du langage. In: Proceedings of the CES. Rome, Italy.

Savage, L.J., 1954. The Foundations of Statistics. Wiley, London.

Schank, R., Abelson, R., 1975. Scripts, Plans, Goals and Understanding. L. Erlbaum, Hillsdale, NJ.

Schank, R., 1982. Dynamic Memory: A Theory of Learning in Computers and People. Cambridge University Press, Cambridge, UK.

Stacey, M., Sharp, H., Petre, M., 1996. A representation scheme to support conceptual design of mechatronic systems. In: Gero, J.S., Sudweeks, F. (Eds.), Artificial Intelligence in Design. Kluwer Academic Publishers, Dordrecht.

Sowa, J.F., 1984. Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading, MA.

de Terssac, G., 1992. Autonomie dans le travail. Serie Sociologie d'Aujourd'hui. PUF, Paris.

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

Zanarelli, C., 1998. Identifier des classes de situations caractéristiques: Quels critères de différenciation des situations sont utilisés par les chefs de régulation et les chefs de départ. RATP, Paris, France.

Zanarelli, C., 1999. Réflexion sur l'évolution des modes de régulation d'un réseau ferroviaire urbain: Application de l'approche instrumentale par identification de classes de situations caractéristiques. In: Acte du XXXIVe Congrès de la SELF. Caen, France, pp. 485–494.

Zanarelli, C., Saker, I., Pasquier, L., 1999. Un projet de coopération ergonomes/concepteurs autour de la conception d'un outil d'aide à la régulation du trafic du métro. Ingénierie des Connaissances (IC'99). Association Française pour l'Intelligence Artificielle, Palaiseau, France, pp. 161–170.