Context Dynamic and Explanation in Contextual Graphs

Patrick Brézillon

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

Abstract. This paper discusses the dynamic of context through the use of a context-based formalism called contextual graphs that has been initially developed in the SART application for the development of a support system in incident solving on a subway line. First, we present the formalism of contextual graphs through its new implementation. Second, we discuss the dynamic of context in contextual graphs. Third, we present two characteristics of contextual graphs as they relate to the dynamic of context, the incremental knowledge acquisition and the explanation generation. We conclude by a discussion of the key properties and the potential of contextual graphs for other applications.

Keywords: Contextual graphs, explanation, visual explanations, context dynamic, applications

1 Introduction

Brezillon [1, 2] defined context as a collection of relevant conditions and surrounding influences that make a situation unique and comprehensible. Based on this initial work, Pomerol and Brezillon [19] showed strong relationships between context and knowledge. Pasquier et al. [16] gave an example of the application of these ideas in the SART application in the monitoring of a subway line. A large volume of knowledge (about trains, electricity, people reaction, and so on) contributes to make each situation unique, while some more particular conditions about the time, the day, the weather and so on, influence many decisions. Brezillon and Pomerol [7] proposed three types of context called external knowledge, contextual knowledge and proceduralized context.

At a given step of a decision process or of the accomplishment of a task, we distinguish between the part of the context which is relevant at this step, and the part which is irrelevant. The latter part is called external knowledge. The former part is called contextual knowledge, and obviously depends on the individual agent and on the decision at hand. Moreover, there is a part of the contextual knowledge that is proceduralized at this step, which we refer to as the proceduralized context. The proceduralized context is invoked, structured and situated according to a given focus.

An important issue is the passage from contextual knowledge to proceduralized context. This proceduralization process [17, 18] is task-oriented and provides a consistent explanatory framework to anticipate the results of a decision or an action.

P. Blackburn et al. (Eds.): CONTEXT 2003, LNAI 2680, pp. 94–106, 2003.

This point is particularly salient when a company establishes procedures and its employees contextualize these procedures to develop efficient practices.

Companies establish procedures that are collections of safety action sequences permitting to solve a given problem in a wide set of circumstances. These procedures are supposed to cover large classes of problems whatever the conditions in which problems must be solved. This is a kind of uniformization in problem solving but it often results in sub-optimal solutions for problem solving. Conversely, each operator develops their own practice, tailoring the procedure in order to take into account the current context, which is particular and specific.

The modeling of operators' reasoning (practices) is a difficult task because operators use a number of contextual elements, and because procedures for solving complex problems have some degree of freedom. Thus, it would be better to store advantages and disadvantages rather than the complete decision.

This discussion points out that if it is relatively easy to model procedures, the modeling of the corresponding practices is not an easy task because they are as many practices as contexts of occurrence. Moreover, it is not possible to establish a global procedure for complex problem solving, but only a set of sub-procedures for solving different parts of the complex problems.

Based on the design of the contextual graphs for the SART application (e.g. see [19]), we present in this paper a new development of our context-based formalism that (1) goes beyond the SART application, (2) is relevant for problems dealing with procedures, practices and context, and (3) presents new functionality in terms of incremental acquisition of practices and explanation generation. Hereafter, the paper is organized in the following way. First, we present the formalism of contextual graphs through its current implementation. This version of the contextual graphs differs of the version presented previously [15] because we suppress assumptions concerning storage and update of data that darken the expressiveness of the formalism about the dynamic of context, the incremental acquisition of practice and the capacity of explanation generation. Second, we discuss the dynamic of context as represented in contextual graphs as a movement of elements between the contextual knowledge and the proceduralized context, with introduction of new elements from the external knowledge when a new practice has to be acquired. Third, we introduce the types of explanation on practices and problem solving that can be generated from contextual graphs. We conclude with a discussion of the properties and potentialities of contextual graphs.

2 Contextual Graphs

2.1 Introduction

The contextual-graph formalism has been developed initially for an application for incident solving on a subway line ([8], http://www.lip6.fr/SART/). The general observation is that the company establishes procedures for incident solving, and the operator in charge of a subway line adapts the procedure for solving an incident to the context in which each incident occurs. This contextualized procedure is called a

practice, and thus there are as many practices as different contexts encountered by the operator.

In our tests operators appreciated the easy understanding of the system's behavior through the use of contextual graphs in which knowledge and reasoning are used in a manner very close to the manner in which they solve incidents [15]. An extension of this project could be for the training of the future operators, thanks to the expressiveness of the contextual graphs, their manipulation (aggregation and expansion of parts of the contextual graph, etc.) and the possibility to replay some incident solving to study potential variants. However, the use of contextual graphs is not limited to the SART application, but is relevant in all domains where operators' reasoning deals with the need to contextualize "official" procedures in order to develop efficient practices by accounting for the context in which the practice is elaborated.

A contextual graph (also noted hereafter CxG) allows a context-based representation of a given problem solving for operational processes by taking into account the working environment [3]. The initial structure of a CxG (its skeleton) is defined by the procedure that is established by the company. The CxG is then progressively enriched by the practices used by operators by applying the procedure in different contexts.

A path in a contextual graph represents a practice in which operator's actions are intertwined with the contextual elements considered explicitly by the operator. A practice differs generally from another one by few actions that are discriminated by a contextual element that has different instantiations for the two practices. Once the divergence between the two practices disappears, the two practices are recombined in a unique path.

2.2 Elements of a Contextual Graph

A contextual graph is an acyclic directed graph with a unique input, a unique output, and a serial-parallel organization of nodes connected by oriented arcs. A node can be an action, a contextual node, a recombination node, or a sub-graph (an activity).

2.2.1 Actions and Activities

An action is an executable method. An activity is a complex action assembling different elements such as a contextual graph with a unique input and a unique output. Mechanisms of aggregation and expansion, as in conceptual graphs, allow users to have different views on a contextual graph and transform an activity into action.

An activity is identified as such by operators as a recurring structure observed in different contextual graphs. The identification of an activity is interesting because a change in an activity appears automatically in all the contextual graphs where the activity has been identified. Activities are organized in a directed hierarchy, an activity possibly calling sub-activities, to maintain the status of acyclic directed graph to the structure.

2.2.2 Contextual Nodes and Recombination Nodes

A contextual element is represented by two types of node, namely a contextual node and a recombination node. A contextual node corresponds to the explicit instantiation

of the contextual element. For example, a contextual element could corresponds to be in a hurry with the instantiations "yes" and "no." A contextual node is represented by C(1, n) where n is the number of exclusive branches corresponding to known practices. The associated recombination node R(n, 1) corresponds to the abandon of the instantiation of the contextual element once the action on the branch is accomplished. Then, there is a convergence of the different alternatives towards the same action sequence to execute after.

Thus, at the contextual node, a piece of contextual knowledge becomes instantiated and enters the proceduralized context. At a recombination node, that last piece entered in the proceduralized context goes back to the contextual knowledge. Thus, a change in the context correspond to the movement of a piece of contextual knowledge into the proceduralized context, or conversely from the proceduralized context to the contextual knowledge.

Contextual and recombination nodes give to contextual graphs a general structure of spindle or series of spindles, with a divergence of branches at contextual nodes initiated by a diagnosis, and a convergence at recombination nodes, thanks to actions or activities realized.

2.2.3 Sub-graphs

A sub-graph represents a local reasoning (a diagnosis/action structure) corresponding to intermediate goals. A sub-graph can be an action, a sequence of actions, or a pair of contextual and recombination nodes. A sub-graph is itself a contextual graph, directed, acyclic, with one input and one output. If a sub-graph contains on a branch a contextual node, it contains necessarily its recombination node on the same branch. Conversely, if a subgraph is on a branch, it contains at most all the items on the branch.

2.2.4 Parallel Action Grouping

A parallel action grouping represents a set of m steps in a problem solving that can be realized in parallel or in any order but all must be accomplished before to continue. For example, a coffee preparation requires to take coffee, filter and the reservoir, these actions can be executed in any order but must be accomplished before to switch on the machine, the order in which these three actions must be executed does not matter. The activity is judged globally with respect to a high-level goal. For example, the type of coffee machine generally does not appear explicitly in the example of the coffee preparation, when it would aloow to order the previous actions (e.g. if the place where to put the filter is fix on the coffee machine). The ordering of the actions to execute in a parallel action grouping depends on contextual elements that does not appear in the contextual graph because they are not at the same level of description and constitutes a dense net of contextual nodes leading to few solutions (see [4] for a discussion on this point). This is a way to deal with the incompleteness or complexity of the local information.

2.2.5 An Example

Figure 1 gives an example of contextual graph. An action is represented by a square box. A contextual node is represented by a large circle and Cj.k is the instance k of the contextual node Cj (1, n). A recombination node Rj is represented by a small

black circle. (Subgraph and parallel action grouping are not represented and discussed in this example.)

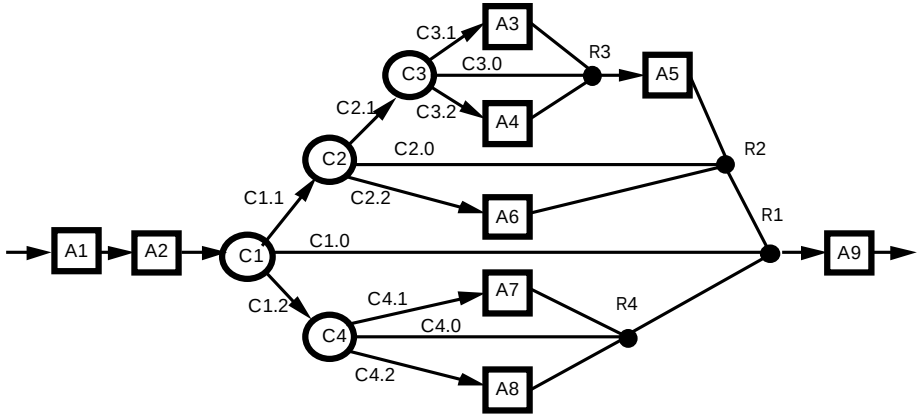

Fig. 1. An example of contextual graph

The operator provides a practice as a sequence of actions such as {A1, A2, A3, A5, A9}. The corresponding path is given by the sequence of actions intertwined with contextual and recombination nodes as {A1, A2, C1.1, C2.1, C3.1, A3, R3, A5, R2, R1, A9} on the upper path in Figure 1. A sub-graph can be an action (e.g. A3), a sequence of actions (e.g. A1-A2), a pair of contextual and recombination nodes and all the items between them (e.g. C3-A3/A4-R3), all the branches between a contextual node and recombination node (e.g. the upper branch of C2 for the value C2.1 with C3-A3/A4-R3-A5).

2.3 Practical Aspects: Implementation

We developed a software for exploiting the formalism of contextual graph. This implementation is realized actually as a prototype written in Java, with a storage of all the data in a database. It presents usual functionality as: switching between different language (French and English at any moment of a session), identification of the user (two types of users, namely the “super-user” who can create a new graph and the “user” who can only enrich the graph with new practices), enrichment and correction of all texts (immediately visible), an online help, different types of visualization (graph resizable according to the dimensions of the window, aggregation and expansion of parts of the graph), comparison of graphs, coloring sub-graph (e.g. for identifying an activity found in different contextual graphs), visualization of the growth of the graph (an addition after the previous one, or the series of additions), comparison of action sequences, explanation on all the items (history, contextual information, etc.), identification of the context of each action, etc.

2.4 Related Works

The contextual graph approach presents some common points with the Ripple Down Rule (RDR) technique [9, 10]. The RDR technique is a hybrid case-based and rule-based approach in which context is an important aspect of RDR, captured in associated cases and the exception structure. In the RDR technique, a case can fall under only one rule, and an error may be corrected by adding one exception rule taking into account only cases that previously fell under the rule to be corrected. There is a two-way dependency relation between rules (with if-true and if-false) such that rule activation is investigated only in the context of other rule activation. Ripple down rules form a binary decision tree that differs from standard decision trees in that compound clauses are used to determine branching, and these clauses need not exhaustively cover all cases so that it is possible for a decision to be reached at an interior node. The RDR technique relies on the fact that people cope with the acquisition and maintenance of complex knowledge structures by making incremental changes to them within a well-defined context such as the effect of changes is locally contained in a well-defined manner [11]. Thus the knowledge that is introduced is highly contextualized. The recommendation given by an expert depends on the context in which it is given and does not consist of a description of the expert's thought processes but is a justification of why this recommendation was made.

Gonzalez and Ahlers [12] describe a knowledge representation paradigm to model the intelligent behavior of simulated agents in a simulator-based tactical trainer. Their hypothesis is that tactical knowledge is highly dependent upon the context (i.e. the situation being faced) and proposed a system called context-based reasoning (CxBR). CxBR encapsulates knowledge about appropriate actions and/or procedures, as well as possible new situations, into contexts. This paradigm has been tested in an application for submarine tactical officers on a patrol mission. Tactical knowledge is required in order to endow autonomous intelligent agents with the ability to act, not only intelligently, but also realistically, in light of a trainee's action. Gonzalez and Ahlers' work is based on the idea that by associating the possible situations and corresponding actions to specific contexts, the identification of a situation is simplified because only a subset of all possible situations are applicable under the active context.

Turner [22] developed a system--an adaptive reasoner--to make context explicit for autonomous underwater vehicles to tackle unanticipated events in complex environments. Contextual knowledge is represented as a set of contextual schemas (c-schemas), then retrieving the most appropriate of those and using them to help the reasoner behave appropriately for its current context. Turner describes context-mediated behavior (CMB) that is based on the idea that an agent have explicit knowledge about contexts in which it may find itself, then use that knowledge when in those contexts. CMB is implemented in the Orca program, an intelligent controller for autonomous underwater vehicles.

3 Movement between Contextual Knowledge and the Proceduralized Context

3.1 The Three Types of Context in a Contextual Graph

Brézillon and Pomerol [18] proposed three types of context, namely external knowledge, contextual knowledge and the proceduralized context. This distinction is expressed in the formalism of contextual graphs in the following way. External knowledge is the knowledge that does not intervene in the contextual graph (i.e. belong to another contextual graph or does not exist in the database). This is a source of contextual knowledge through the incremental acquisition of new practices, as discussed below. Contextual knowledge exists in the CxG (the context of the CxG is composed of all the contextual elements in the graph) but is not considered through an instantiation. At the level of a practice (a given path in a contextual graph), all the contextual elements that are not on the path represent contextual knowledge. At the level of an action, contextual knowledge corresponds to contextual nodes out of the path where is the action, when the contextual elements belonging to the path are ordered in a sequence and considered through their instantiations, and thus constitute the proceduralized context. The proceduralized context is an ordered sequence of instantiated contextual elements on the path.

3.2 Context at a Step of a Practice Execution

The context of the contextual graph in Figure 1 is given by the elements {C1, C2, C3, C4}. The context of the action A3 is composed of two parts: the contextual elements used on the path from the input to the action and the other elements. The later elements are contextual knowledge (e.g. C4). The former contextual elements are instantiated, C1 with the value C1.1, C2 with the value C2.1 and C3 with the value C3.1. Thus, the context of the action 3 is defined by:

- The proceduralized context: {C1 with the value C1.2, C2 with the value C2.1, C3 with the value C3.1}, supposing that the actions A1 and A2 are realized.
- The contextual knowledge: {C4}

The context of the action A3 is described in a fixed and static way.

Consider now the context of the path where is the action A3 Once the action A3 is executed, the value C3.1 of C3 does not matter anymore (i.e. at the recombination node R3). The contextual element C3 leaves the proceduralized context at the recombination node R3 and goes back to contextual knowledge. Thus, the context of the action A5, which follows the recombination node R3, is described by:

- The proceduralized context: {C1 with the value C1.2, and C2 with the value C2.1}, and
- The contextual knowledge: {C3, C4}.

The context of the action A5 is also described in a fixed and static way. It differs from the context of action A3 by the contextual element C3 that moved from the proceduralized context to the contextual knowledge at the level of the practice. Thus, during the progress of the practice execution from action 3 to action 5, the context of the practice evolves when the contexts of A3 and A5 are static.

3.3 Context Evolution During the Progress of a Practice Execution

The dynamics of the context appears at the practice level when the focus of attention moves. The contextual knowledge and the proceduralized context evolve during the progress of a practice execution (along a path). For example, consider the upper path in Figure 1: {A1, A2, A3, A5, A9}. Its context presents the following dynamic along the practice execution (each line of the Table represents a step in the application of the practice, a step corresponding to a change in the context):

Table 1. Dynamics of the context along the upper path in Figure 1

L	Context from	Contextual Knowledge	Proceduralized context
1	0	{C1, C2, C3, C4}	{Ø}
2	C1	{C2, C3, C4}	{C1.1}
3	C2	{C3, C4}	{C1.1, C2.1}
4	C3	{C4}	{ C1.1, C2.1, C3.1}
5	R3	{C3, C4}	{C1.1, C2.1}
6	R2	{C2, C3, C4}	{C1.1}
7	R1	{C1, C2, C3, C4}	{Ø}

The movement between the contextual knowledge and the proceduralized context follows the rule “last in, first out.” The contextual elements are instantiated in the order C1, C2, and C3 (in the proceduralized context) and return to the contextual knowledge as C3, C2, and C1. The progress of the practice execution until an item itself is an element of the context. Thus, two contexts having the same contextual knowledge and proceduralized context (as at lines 3 and 5 of the Table 1) are different by their history in the practice.

3.4 Incremental Knowledge Acquisition in Contextual Graphs

An important part of our system is the identification of a sequence of actions used by the operator for a problem solving. This is realized by interaction between the operator and the system through a graphical representation of the current state of the contextual graph. Once a problem is solved, the operator reports the problem solving by providing the system with the action sequence used for the problem solving. Then, the operator tells the system which known practice is the closest of the entered sequence of actions. The entered sequence can be a known sequence or not. This is determined by the system that matches actions of the sequences in an ordered way. Once a discrepancy is detected (an action is different, new or missing between the two sequences), the system ask the operator the reason of the difference. The operator provides the system with the missing contextual element (definition), its location (position of the contextual and recombination nodes on the path), its instantiations for the known practice and the entered practice. The contextual element that is added, generally comes from the external knowledge. The reason is that the instantiation of this contextual element was not relevant before, but is instantiated in the new practice in a specific way. Thus, the movement from the external knowledge to the contextual knowledge of a contextual graph goes through its use in a proceduralized context.

This is a way to tackle the infinite dimension of the context because the external knowledge is considered only when needed [14].

Thus, contextual graphs have the capacity of evolving by accommodation and assimilation of practices. A new practice differs of a known practice in the CxG by few elements (generally an action). As a consequence, a contextual graph will possess more and more of practices as a kind of corporate memory. The acquisition of a new practice corresponds to the addition in a contextual graph of the minimum number of elements (generally one pair contextual node – recombination node and an action). Never there is to copy a large part of the contextual graph as in a decision tree [15]. (The complete algorithm is under study now.)

4 Explanation Generation in Contextual Graphs

Explanation generation was based on the domain knowledge, i.e. the task at hand, the actions (definition, input, output) in our case. Since about ten years it is known that such explanations bring few to the user and nothing to operators because of the lack of consideration for the context [6]. In contextual graphs, context is represented explicitly, the knowledge is acquired in its context of use, and thus an explanation can be generated from all the items in a contextual graph (contextual elements, actions, activities, their ordering, etc.)

The explanation of a practice is mainly the presentation of the different contextual elements intervening along the path, the order in which they intervene, their temporary instantiations and the temporal chronology in which they have been incorporated in the contextual graph. The explanation of an action in a practice relies mainly on the proceduralized context of this action. The system can thus explain the reasoning hold in the practice (until this action) by presenting (1) the contextual elements explicitly used in the practice until the action, (2) the instantiations of these contextual elements, (3) the order in which are instantiated the different contextual elements, and, the most important, (4) the order in which (and the reasons why) the contextual elements have been introduced in the contextual graph. These two last points are a way to take into account in the explanation the context dynamics that leads to the action to explain. This leads to view explanation as a process in progress along the reasoning progress, rather than deriving it from known and static factors. Our position is close from Leake's position [13] about explanation in case-based reasoning, but the explanation generated in a contextual graph is at different levels of detail (the proceduralized context or the order and the reasons of the introduction of each contextual element).

As the system and the user interact on the same contextual graphs, each one can provide the other with relevant explanation. In the case of explanations provide by the user, the system enters a phase of incremental acquisition of practices that will improve later its reasoning. In this way, the task at hand, the incremental acquisition and the generation of explanations must be intertwined, an important issue in cooperation [5]. Conversely, explanations enable the contextual knowledge to be proceduralized at the right place by supporting the process of incremental acquisition of practices.

Moreover, the mechanisms of aggregation and expansion, as in conceptual graphs [20, 21], allow the user to focus on one part of the contextual graph or another

according to his focus of attention. For example, it is possible to study parts of a reasoning and all the variants (the practices). This is particularly interesting to understand the differences between two practices, the role plays by a contextual element in the choice of an action instead of another, etc.

In conclusion, the context-based formalism called contextual graphs gives a uniform representation of actions series and contextual elements. Thus explanations are easier to produce because knowledge on which explanation relies is explicit in the representation. Contextual elements explains the reasons of the choice of an action on another action. With the history of changes in a contextual graph, it is possible to produce different types of explanation.

At the level of a practice, explanation is a way to present the progress of the application of a practice, the movement between the contextual knowledge and the proceduralized context, the changes between the practice and the previous one, the variants added after. It is possible to generate explanations at another level to present contextual information as the creation date of the practice, the author, the problem solving requiring this change for the first time, etc.

With the graphical interface for representing contextual graphs, the system can generate visual explanation on the path from the source to a given element, the ways in which a practice has been progressively specialized, the growth of a contextual graph (with the incremental addition of practice, the practices introduced by a given operator, etc. This aspect, thanks to the incremental practice acquisition, is a new way to generate explanations.

This shows that the task at hand, the incremental acquisition of practices and explanation generation are three aspects of the same thing (the task at hand in the large).

5 Discussion

Context-based formalisms allow a representation of knowledge and reasoning in a way that is directly comprehensible by users. The structures in a contextual graph put at the same level actions and activities (complex action structures). Thus, two people having to interpret the same activity at different levels can understand each other. For example, “Empty the train of travelers” is interpreted as a simple action by the operator who is responsible of the subway line and a complex activity by the driver (stop at the next station, announcement to travelers to leave the train, go and check that nobody is still in the train, close the doors and leave the station).

At the action level, making explicit contextual elements allows to explain the reasons for the choice of an action on another one. Thus, information in contextual graphs is useful and useable for operators.

After using a contextual graph for a while, most of the possible practices would be recorded. By analyzing the whole contextual graph (the initial procedures and all the practices), this would allow the company to improve its procedures, and thus reinforce the value of operators’ practices on too general procedures. Another consequences of the expressiveness of the practice representation and explanation capability in contextual graphs is for training purpose (1) of future operators by discussing subtlety of the task accomplishment, and (2) of operators by exchanging and discussing experiences.

In contextual graphs, incremental acquisition of practices and explanation generation are naturally intertwined with the task at hand. Moreover, the progressive evolution of the contextual graph and its graphical representation interact continuously and thus can not be separated according to the rules of the software engineering (separation of the interface from the program).

There are however some limits in this context-based representation at the level of the parallel action grouping. In the example of the coffee preparation (see [4]), we observe a dense net of contextual nodes for the selection of one of the three actions "Take reservoir", "Take filter" and "Take coffee" that can be executed in a sequence that depends on factors such as the places where are the items (e.g. filters are in the cupboard and coffee in the refrigerator) and the user's preferences (if I take coffee box in second, I have just to keep in my hand the coffee box to put it in the filter that will be made ready just before). The type of the machine also may intervene. For example, in some machines the filter is put on the reservoir, when on other machines it is fixed on the body of the machine. In the former case, it is necessary to first pour water in the reservoir, install the filter on the reservoir, and then put coffee in the filter. In the same spirit, it could be important to make explicit the number of persons interested by the coffee preparation (the choice of the machine depends on it), the place where are things and their relative location (place: home or at work; filter: in the cupboard or near the coffee machine; coffee: near the coffee machine or in the refrigerator; reservoir: near the coffee machine or in the "kitchen or wash room"), and the relationships between things. For example, is the filter on the receptacle or not? Is the cup a part of the receptacle itself? Must we take all the coffee machine to fill the reservoir with water? The availability of the resources in the operating environment (water, coffee, electricity source), etc. also intervene implicitly at the level of the parallel action grouping. All these factors constitute a set of choices that can be different from one day to another one (optimization of movement, of the duration of the operation, of the number of operations).

Another weakness of the context-based representation concerns the representation of time. Now, time is represented by the fact that contextual graphs are directed, and that there is an ordering of the actions to execute. However, it is not possible to represent the fact that an action must be accomplished, say, ten minutes before starting the execution of another one. For example, one needs to stick successively two objects, the second object once the first one is definitely stuck.

However, even now contextual graphs present some potentialities to exploit. It is clear that the more a system based on contextual graphs is used, the more it will preserve corporate memory. As a side-effect, it is possible to revise the procedures according to all the variants developed by operators. The new procedures would be more robust. As a contextual graph could describe all the ways in which something can be used (say, as the access to a server), it could be possible then to determine secure and sensible paths of access and forbid sensitive ones after to identify what a user is doing. We are currently studying such lines of use of contextual graphs.

Acknowledgments. We thank Laurent Pasquier, then Ph.D. student with which we have developed contextual graphs for the SART application, and Juliette Brézillon that has improved the software under several aspects. We also thank the French Foreign Ministry that provides grants for the SART application.

References

1. Brézillon, P.: Context in problem solving: A survey. The Knowledge Engineering Review, **14(1)** (1999) 1–34
2. Brézillon, P.: Modeling and using context: Past, present and future. Rapport de Recherche du LIP6 2002/010, Université Paris 6, France. http://www.lip6.fr/reports/lip6.2002.010.html (2002)
3. Brézillon, P.: Using context for Supporting Users Efficiently. Proceedings of the 36th Hawaii International Conference on Systems Sciences, HICSS-36, Track "Emerging Technologies", R.H. Sprague (Ed.), Los Alamitos: IEEE, CD-Rom (2003a)
4. Brézillon, P.: Contextual graphs: A context-based formalism for knowledge and reasoning in representation. To appear in Research Report, LIP6, University Paris 6, France (2003b)
5. Brézillon, P., Cases, E.: Cooperating for assisting intelligently operators. Design of Cooperative Systems (COOP-95). INRIA (Publisher) (1995) 370–384
6. Brézillon, P., Pomerol, J.-Ch.,: User acceptance of interactive systems: Lessons from Knowledge-Based and Decision Support Systems. International Journal on Failures & Lessons Learned in Information Technology Management **1(1)**(1997) 67–75.
7. Brézillon, P., Pomerol, J.-Ch.: Contextual knowledge sharing and cooperation in intelligent assistant systems. Le Travail Humain, **62(3)** (1999) 223–246
8. Brézillon P., Pasquier L., Pomerol J.-Ch.: Reasoning with contextual graphs. European Journal of Operational Research, **136(2)** (2002) 290–298
9. Clancey W.J. Model construction operators. Artificial Intelligence, **53** (1992) 1–115
10. Compton, P., Jansen, R.: Knowledge in context: a strategy for expert system maintenance. Barter, C.J. and Brooks, M.J., Ed. AI'88: 2nd Australian Joint Artificial Intelligence Conference, Adelaide Australia, November 1988, Proceedings. Berlin, Springer (1990a) 292–306
11. Compton, P., Jansen, R.: A philosophical basis for knowledge acquisition. Knowledge Acquisition **2(3)** (1990b) 241–258
12. Gaines, B.R., Compton, P.: Induction of Ripple-Down rules applied to modeling large databases. Journal of Intelligent Information Systems, **5(3)** (1995) 211–228
13. Gonzalez, A.J., Ahlers, R, Context-based representation of intelligent behavior in training simulations. *Transactions* **15**(4) (1997) 153–166
14. Leake, D B,: Case-based reasoning: Experiences, lessons, and future directions. Chapter I. *CBR in context: The present and future* Menlo Park: AAAI Press/MIT Press (1996)
15. McCarthy, J,: Notes on formalizing context *Proceedings of the 13th IJCAI* **1** (1993) 555–560
16. Pasquier L.: Modélisation de raisonnement tenus en contexte. Application à la gestion d'incidents sur une ligne de métro. Thèse de l'Université Paris 6, juillet (2002)
16. Pasquier, L., Brézillon, P., Pomerol J.-Ch.: Context and decision graphs in incident management on a subway line. Modeling and Using Context (CONTEXT-99). In: Lecture Notes in Artificial Intelligence, N° 1688, Springer Verlag (1999) 499–502
17. Pomerol J-Ch.,: Decision Making Biases and Context, Brussels DSS Conference, *Journal of Decision Systems* (2003, to appear)
18. Pomerol J.-Ch.., Brézillon P. Dynamics between contextual knowledge and proceduralized context, in *Modeling and Using Context* (CONTEXT-99), Lecture Note in Artificial Intelligence n° 1688, Springer Verlag, (1999) 284–295
19. Pomerol J.-Ch., Brézillon P. About some relationships between knowledge and context. In: P. Bouquet, L. Serafini, P. Brézillon, M. Benerecetti, F. Castellani (Eds.): Modeling and Using Context (CONTEXT-01). Lecture Notes in Computer Science, Springer Verlag, N° 1688, (2001) 461–464. (Full paper at http://www-poleia.lip6.fr/~brezil/Pages2/Publications/CXT01/index.html)
20. Sowa, J.F.: "Conceptual Structures: Information Processing in Mind and Machine", Addison Wesley Publishing Company (1984)

21. Sowa, J.F.: Knowledge Representation: Logical, Philosophical, and Computational Foundations, Brooks Cole Publishing Co., Pacific Grove, CA, ©2000. Actual publication date, 16 August 1999. 594 + xiv pages; ISBN 0-534-94965-7
22. Turner, R M,: "Context-mediated behavior for intelligent agents" *International Journal of Human-Computer Studies* Special Issue on Using Context in Applications **48**(3): (1998) 307–330
23. Walker, R.J.,d Murphy, G.C.: Implicit context: easing software evolution and reuse. In: D.S. Rosenblum (ed.) Proc. of the ACM SIGSOFT Eighth Int. Symp. on the Foundations of Software Engineering (FSE-8). ACM Press, (2000) 69–78