Knowledge Acquisition and Explanation for Diagnosis in Context

P. Brézillon and S. Abu-Hakima

Knowledge Acquisition and Explanation for Diagnosis in Context

Patrick Brézillon
LAFORIA-IBP
University Paris 6
75252 Paris Cedex 05
France
brezil@laforia.ibp.fr

Suhayya Abu-Hakima
Knowledge Systems Lab.
Institute for Information Technology
National Research Council of Canada
Ottawa, Ontario, Canada
suhayya@ai.iit.nrc.ca

Abstract

This report, which is submitted in a review, describes the use of context in problem solving with knowledge-based systems. It describes how context can be represented, reasoned with, explained and acquired. It also describes how context can be used to better represent, reason with, explain and incrementally acquire knowledge. Three different real-world diagnostic applications are described to illustrate the practical application of context. The three applications are: the troubleshooting of an aircraft engine using the Jet Engine Troubleshooting Assistant (JETA), the J85-CAN-15 powers the CF-5 trainer of Canadian Forces pilots; the diagnosis and explanation through simulation of power plants using SEPT at Electricité France, and the diagnosis of a Xerox workstation using RATIONALE. A method of incremental knowledge acquisition is presented as a cooperation between a user who is a domain expert and the system. For the system to succeed in incrementally acquiring knowledge, it must represent, reason with and explain context.

Résumé

Ce rapport (soumis dans une revue) décrit l'utilisation du contexte dans la résolution de problème par un système à base de connaissances. Il décrit comment le contexte peut être représenté, expliqué, acquis, et comment on peut raisonner avec et permettre d'acquérir incrémentalement des connaissances. Trois applications industrielles sont décrites pour illustrer une utilisation concrète du contexte. Ces trois applications sont : le "Jet Engine Troubleshooting Assistant" (JETA) pour les Forces Canadiennes, le système SEPT de diagnostic et de simulation pour EDF, et le diagnostiqueur RATIONALE de stations de travail Xerox. Une méthode d'acquisition incrémentale des connaissances est présentée comme une coopération entre un utilisateur qui est un expert du domaine et le système. Le succès de cette méthode demande que l'on puisse représenter, raisonner avec et expliquer le contexte.

1.0 Introduction

The hypothesis of this work is that context can be represented, reasoned with and explained. This is supported by the fact that context must be made explicit so that it improves tasks where the user has a crucial role. In this paper, we focus on the role of context in diagnostic tasks and their relation to explanation and incremental knowledge acquisition in reasoning. This is further motivated by the hypothesis that the representation, reasoning and explanation of context facilitates the incremental acquisition of knowledge. In addition, context can significantly improve the explainability of the knowledge represented and reasoned with.

Thus, the objective of this paper is to describe the use of context in problem solving with knowledge-based systems (KBS). Such problem solving systems are not oracles to be questioned by the user [Karsenty et al. 94]. Problem solving is a cooperative task between a user and the system. Four types of users can be identified: the expert whose heuristics help create the knowledge, the knowledge engineer who models the expertise, the end user who gains practical experience by frequently using the KBS (may not be the expert), and the occasional end user (novices, users from other departments that occasionally consult the KBS, etc.). Such a distinction amongst users of a KBS permits one to identify users who are authorized to modify the knowledge base. In this paper, a user who is authorized to modify the knowledge base is the authorized user.

During problem solving, explanations are cooperatively constructed, and knowledge is incrementally acquired as part of the task at hand. Explanations and incremental knowledge acquisition are highly context-dependent. There are two aspects to the problem of context:

(1) The elaboration on the representation, reasoning, explanation and acquisition of knowledge through context.

(2) The description of how context can be used to better explicitly represent, reason with, explain and incrementally acquire knowledge cooperatively with the authorized user.

This novel method of incremental knowledge acquisition (IKA) arises from the cooperation between an authorized user and the system. As the system reasons and the user asks for explanation, the user may discover that in some cases the context of reasoning of the system does not match the user's expectations. In such cases, the user enters a clarification dialogue with the system which may result in the acquisition of new knowledge for either the system or the user.

To facilitate the description of context and show its practicality, three real-world applications are described: the troubleshooting of an aircraft engine of the CF-5 of the Canadian Forces using the Jet Engine Troubleshooting Assistant (JETA - [Halasz et al. 92]), the diagnosis and explanation through simulation of the behavior of equipment components in extra-high voltage substations operated in power plants using SEPT at Electricité France [Brézillon et al. 90], and the diagnosis of a Xerox workstation using RATIONALE [Abu-Hakima 88; 90].

In the remainder of the paper: section 2.0 defines the required knowledge representation for context, section 3.0 introduces the three practical applications used in the paper to elaborate on context, reasoning with context is detailed in section 4.0, the explanation of context is described in section 5.0, section 6.0 describes incremental knowledge acquisition through the use of context and explanation, and section 7.0 concludes the paper with future real-world applications requiring context.

2.0 Knowledge Representation Strategies for Context

To establish context in reasoning, context must be explicitly acquired and represented in the knowledge-based system. This can be achieved by explicitly representing context with three relations.

1) The first relation defines a specific context.This relation requires three explicit representations that define a context.

i) Pre-conditions of a specific context in the form of variables that are instantiated dynamically. These could be encoded as antecedents of a meta-rule relating a group of rules or a slot activated by the relevant variables in a frame.

ii) Actions in-context during problem solving (reasoning). These actions additionally include variables that are dynamically instantiated within the context. They could be represented as a group of rules triggered by a meta-rule, screening clauses triggering a rule or a set of slots in a frame that are activated by the traversal of the reasoning algorithm.

iii) Consequences of the specific context. These could be encoded as the post-conditions of a meta-rule that relates the group of in-context rules or as one or more slots in the frame which is activated on exiting a frame.

2) The second relation of context defines the movement and relation of one context to another. This is essentially the knowledge used dynamically in reasoning to navigate a knowledge base representation. It could be in the form of meta-rule consequences triggered on exiting a context or it could be in explicit parent and child slot relations navigated in reasoning in a KBS frame-base.

3) The third relation defines the movement through time of a context and its relation to another context as a function of time. This is closely related to relation 2 but contains explicit temporal knowledge. It is representable as continuous time variable functions embodied in a rule or as slots in a frame.

If these three relations are present in current knowledge-based systems, they are generally implicit or exist as ad hoc relations and do not adequately make use of context. In some cases these relations are partially but not completely defined. This is evident in examining complex real-world systems such as JETA and SEPT which are described in section 3.0. These two systems have an explicit knowledge representation and reason with the relations of context without identifying them as such. This explicit representation aids the system in its diagnostic problem solving task. However, both systems do not explicitly make use of context in explanation and in their monolithic rather than incremental knowledge acquisition. Mechanisms for extending them for incremental knowledge acquisition are described in section 5.0.

It is also interesting to note that in currently published literature [Giunchiglia 93; McCarthy 93; Lespérance 93; Guha 91], some context formalisms based on first order logic are presented but are too simple to adequately cover the problems of knowledge representation in developing real-world complex systems such as JETA or SEPT. These formalisms also concentrate on the movement of one context to another with some temporal representation but largely ignore the explicit representation of, entering, acting within or leaving a context.

3.0 Three Real-World Diagnostic Systems

The three diagnostic applications, JETA [Halasz et al. 92], SEPT [Brézillon et al. 90] and RATIONALE [Abu-Hakima 88; 90], are good illustrations of explicit representation and reasoning in diagnostic systems. They differ in their objectives, knowledge representation, reasoning and explanation to the user.

JETA combines functional and fault decomposition in its hierarchical representation of engine fault, test, component repair and replacement nodes. JETA also represents its nodes with frames. SEPT uses explicit structural and functional decomposition of a power network to organize its knowledge hierarchically with the leaf nodes representing the expected and faulty function of a component. SEPT uses packets of rules to represent the nodes in its hierarchy [Brézillon 90]. RATIONALE organizes its knowledge as a classical fault hierarchy of frames with its terminal nodes representing component failures.

It has been the experience of these authors that knowledge should be structured and well organized in any real-world knowledge-based system. Six key reasons exist in support of this:

(1) A large rule-based system will result in a combinatorial explosion in attempting to determine inter-rule propagation for chains of reasoning when a new rule is added.

(2) Rule ordering has been shown to directly affect the firing of rules and produce incorrect results in system reasoning [Clancey 86]. Thus, a new rule cannot simply be added without propagating its effects so that its position in the rule base may be determined. In systems such as GUIDON [Clancey 86], screening clauses are added to the rule base to limit the effects of triggered rules. The same conclusion has been reached in the SEPT project [Brézillon 88; 90]. In SEPT, screening clauses were replaced by rule packets which were triggered by the antecedent of a rule since screening clauses did not adequately address the ease of frequently changing the knowledge base due to the continous changes required by the domain expert. However, screening clauses and rule packets do not adequately address the problem of a flat (rule-based) versus a structured (frame-based) knowledge base, although that the introduction of rule packets is a step in the right direction.

(3) The addition of new rules generates global distributed effects in the rule base whereas the addition of new frames creates localized effects [Chandrasekaran 83]. Distributed effects are much more difficult to track than local effects.

(4) Causality cannot be adequately explained in rule-based systems without significant partitioning of the rule base and the addition of explicit attachments to rules so that they may be explained [Abu-Hakima 88].

(5) Rule bases mix control knowledge with data required for reasoning whereas frame-based systems provide more separation through the use of explicit slots that are used for control and others that are used as data for reasoning [Halasz et al. 92].

(6) The inference engine used for flat rule bases may be augmented with domain expertise to effect the right results. Such hidden knowledge is not explainable and should be made explicit (for example, as a slot in a frame).

The three applications also differ in the method in which they support end users. JETA uses classical contextual help and hypermedia links to provide relevant help [Abu-Hakima et al. 93]. SEPT uses explanations generated through a simulation to explain to the user its reasoning strategies once the results of a diagnosis are known. RATIONALE generates dynamic explanations on its reasoning in context that include why, how, why-not and what-if explanation types. RATIONALE also allows the user to browse the static knowledge and ask for explanation on the reasoning strategies of the system for any aspect of its explicitly represented knowledge.

In the three diagnostic problem solvers the unit representing context is either a frame (JETA, RATIONALE) or a packet of rules (SEPT). These units are instantiated dynamically while reasoning to provide a context which is related by a combination of nodes representing faults or components, relations between nodes and other dynamically instantiated variables relevant to a particular aspect of the problem solving.

To highlight the differences of the three systems, the objectives, the specific application, and the problem solving strategy constraints related to context are methodically described for each system. The constraints are based on the required explicit definition of the three relations of context: context function, movement from one context to another, and the movement of one context to another as a function of time. These relations are manifested in the knowledge organization, knowledge representation, explicit control strategy of reasoning and the dynamic control of the reasoning strategy.

JETA

Objective of KBS

The objective behind the development of the Jet Engine Troubleshooting Assistant (JETA) was to develop a tool that can be used to troubleshoot complex electromechanical devices [Halasz et al. 92].

Application of KBS

JETA's first application is the troubleshooting of J85-CAN-15 aircraft engines manufactured by General Electric and used by the Canadian Forces pilots on the CF-5 fighter aircraft. JETA’s knowledge base is designed to mainly troubleshoot aircraft engine acceleration faults. Such faults involve most of the subsystems of the aircraft engine and are the most difficult for technicians to troubleshoot.

Problem Solving Strategy Constraints related to Context

1) Knowledge Organization

In JETA domain knowledge is organized into a network of nodes that lead to other network nodes. The knowledge is organized into a directed network graph which starts off hierarchical in nature and as a result is sometimes refered to as a hierarchy, rather than a network. Reasoning proceeds by activating nodes whose root node contains the most general diagnosis of the system. Guided by a reasoning algorithm, the network nodes activate more specific nodes that are triggered by the more specific parameters provided by the user.

JETA’s knowledge network of approximately 200 nodes starts as a hierarchy which is organized into two main branches. The root of the hierarchy is a J85-CAN-15 jet engine fault. This splits into a fault during phases of engine operation or a ‘snag’. Snag is a term used by the manufacturer to indicate a fault that is more symptomatic in nature. The phases of engine operation faults branch across seven engine operation paths such as, engine start-up, acceleration, steady state, deceleration, shut-down faults or indication faults. J ETA’s snags are refinable to thirteen snag nodes. These nodes are further refined to indicate specific component faults. At the lowest level of JETA’s diagnostic hierarchy are 78 repair or replacement nodes [Abu-Hakima 94a; 94b; 94d].

2) Knowledge Representation

JETA's knowledge representation includes three types of frames, diagnostic network, parameter, and glossary frames. The parameter and glossary frames explicitly represent knowledge required for reasoning and are refrenced directly by the diagnostic network frames. The relevant parameter and glossary slots are activated dynamically during reasoning [Abu-Hakima et al. 93a]. The key frame in JETA is the diagnotic network frame which represents a node in the troubleshooting network. As shown in Figure 1, the frame has 14 slots which are activated dynamically when JETA reasons. The slots include the type of node, its parental (child-node-of) and child (child-node-ranking) relations. In addition, it includes entry-actions (in-context actions), after-fixup (in-context actions) and completion-actions (part of consequences of the context of visiting a JETA node).

3) Control Strategy

JETA's reasoning strategy is very flexible. It interprets locally encoded commands in the diagnostic network frame. This allows for context-sensitive branching depending on values of the different parameters that apply at each node. Advice displayed at a node is tailored for special circumstances, such as the mode of entry into the node or the path taken in the network to get to the node. This allows JETA to behave flexibly and does not force the user through fixed sequences of questions. Logical jumps in the network are accommodated so that a current line of reasoning can be aborted should new observations lead to other nodes becoming more probable. Postponement of certain procedures is allowed when a technician may not have time to carry them out, or a parts shortage exists and observations are not conclusive enough for the particular failure, or the technician simply has a hunch that a path is not where he/she really wants to be. The possible next moves in the JETA network are represented as the children of a node. JETA’s reasoning strategies navigate its network in three possible ways:

• Through sibling node refinement by traversing the network from left to right and top down which is the means of implicitly organizing the knowledge. Any node can have multiple parents which permits more than one cause for a component malfunction or several means of entering a node.

• Through activating conditions (rules in a slot) that allow the reasoner to jump around in the network.

• Through entry actions which require the evaluation of various parameters that in turn activate new nodes not immediately related to the current node.

4) Dynamic Control while Reasoning

JETA's troubleshooting knowledge is encoded at each node or decision point through the use of a custom command language. The reasoning strategy, while interpreting this language, allows for the many aspects inherent in complex diagnosis. Some of these are to dynamically:

- Block entry to a particular node if certain pre-conditions have not been cleared first.
- Jump to another node if intermediate results point to another strategy.
- Activate nodes by rules dependant on network status and relevant parameter values.
- Postpone a test or procedure because of intermediate results or user discretion.
- Provide dynamic help, graphics and hypertext to describe nodes or terminology.
- Allow for instrument mis-readings or failures.
- Revisit a node if it seems that a faulty component has been used in place of another.
- Change parameters as the diagnosis extends over time. For example, temperature will change during the day making initial values invalid for later tests requiring the same information.
- Control looping in the network by setting and monitoring node states to repeat tests.
- Let the user select from a list of prioritized tests and failure modes. The list order is based on the difficulty of a test and the likelihood of the failure. It can be dynamically re-ordered depending upon observed snag combinations.

defnode:	node name (not a slot)
display-name:	displayed node name for user
time-estimate:	time estimate for recommended actions (displayed on node entry)
line-of-maintenance:	severity of repair (1st, 2nd or 3rd line)
help-info:	help information on node including hypertext markers
node-type:	type of frame {general (default), all-ok, snag, functional test incorrect test setup, repair,replace,repair/replace}
child-node-of:	kinship to parent nodes
activating-observation:	rule expression which could be used to activate node
child-node-ranking:	relative child-node priority list (default = equally likely)
blocking-if-user-selected:	check to see if other conditions which may block entry to node (if user uses direct selection to move in network).
entry-actions:	sequence of operations carried out upon node entry. For example: - general advice, - advice which depends upon path through network, status of other nodes or parameter values, - asking for intermediate parameter values, - intermediate servicing of another node, - setting states of other nodes, - ask user to make selections from the list of sub-nodes.
after-fixup-actions:	sequence of operations to be executed after having carried out entry-actions.
completion-actions:	sequence of operations to be executed before returning to the invoking node
functional-tests-required:	validation tests to check that repair was carried out correctly
component-affected:	component in engine parts database with serial numbers, etc.

Figure 1: JETA's diagnostic frame template representing explicit fault knowledge

SEPT

Objectives of SEPT

The overall objectives of the SEPT project are twofold: to provide a diagnosis as quickly as possible; and to subsequently explain the diagnostic results (SEPT is the French acronym for the diagnosis of equipement in an Extra High Voltage (EHV) substation.).

Three versions of SEPT exist.

1) A flat rule-based mock-up [Brézillon et al. 88].

2) An Industrial version certified at Electricité de France implemented in the Knowledge Craft Toolbox from Carnegie Group Inc. [Fauquembergue et al. 88]. This has been in operation since 1990 in two extra high voltage substations.

3) A prototype with rule packets that partition the knowledge base and a simulator for generating explanations based on the diagnosis [Brézillon et al. 90].

The paper focuses on the third version of SEPT.

Application of KBS

The SEPT project aims to develop a tool to support an operator diagnosing types of equipment in an extra high voltage substation at Electricité de France, the French national power utility company.

The types of equipment permit the operator to rapidly isolate a faulty element of the power network in order to insure a continuous distribution of electricity. The SEPT knowledge base is designed to detect non-functioning or functional faults in a more systematic manner than a human operator.

Problem Solving Strategy Constraints related to Context

1) Knowledge Organization

The French power network is a meshed network of power lines, whose nodes are Extra High Voltage substations. A substation is composed of several busbar sections or transformers which are interconnected by cut-off systems (COSs). Power lines are connected to the busbar sections by COSs. Power network components (known as the controlled system) are often perturbed by faults such as short-circuits. The role of the COSs (also known as the control system) are to diagnose and eliminate faults by automatically opening certain circuit breakers. A COS consists of two relays, a circuit breaker, etc.

A fault is detected not only by the two COSs located respectively at both ends of the faulty component, but also by the many nearby COSs through electrical feeding paths. This redundancy allows the elimination of a fault by opening different sets of circuit breakers depending on all competing COSs behavior (normal or abnormal). Most operations are recorded chronologically as signal events on a central printer in the substation.

An incident is analyzed by a sequence of logical deductions based on experience. Such a diagnosis is established on five successive levels: the signals, the equipment, the cut-off systems, the busbars and the substation. The successive aggregations of information leads to the localization of the fault and, later, to the determination of the faulty equipment at the substation level.

2) Knowledge Representation

SEPT is aimed at the association in a single system of the tasks of diagnosis and simulation. The language in which SEPT is described is based on first-order predicate logic within a formalism using rule packets and the notion of interpretation [Brézillon 90; 91]. The system has approximately 700 rules partitioned accross 150 rule packets. Rule packets allow an explicit structured knowledge representation and are managed dynamically (a rule packet is called in the action part of a rule and some instances of variables in the antecedent of a meta-rule may be set in the rule packet).

Rule packets are used to explicitly represent the structural and functional organization of the domain knowledge. For instance, there is a rule packet describing a cut-off system. Part of the rules in this rule packet are meta-rules that call other rule packets describing parts of equipment (e.g., a protective relay, a circuit breaker, etc.) in the cut-off system, and other rules check coherence between parts of equipment (e.g., to verify that the circuit breaker opens at the right timing given by the protective relay). This schema (the study of an entity is led from the study of its elements and the study of the coherence in their behaviors) is used recursively, an element being itself an entity composed of sub-elements.

Since it was impossible to produce explanations because of the strategy of attaching explanation text to individual rules, the operator is provided with the ability to explore the line of reasoning of SEPT through varying levels of detail: a result of a synthesis, a trace, details at a chosen step in reasoning, levels of explanatory detail. The notion of interpretation permits one to select actions to execute when a rule is fired. It keeps each unit of the expertise in a single rule, and this rule can be fired for different purposes, as, for instance, various levels of detail in the presentation of the results. An example of the types of explanation of a given rule in SEPT is:

IF the tripping order of a protective relay is emitted in two phases,
THEN
action 1: conclude an abnormal functioning of the protective relay;
action 2: send a warning message;
action 3: present the instantiated premisses;
action 4: explanation level 1 :
  An opening on two phases means probably that the fault concerns three phases and that therefore the protective relay is faulty in one phase.
action 5: explanation level 2 :
  Opening of a circuit breaker can only be on 1 or 3 phase
action 6: explanation level 3 :
  Opening on 1 phase preserves the actual state of the network, whereas the opening on 3 phases isolates the faulty line.

Knowledge in SEPT is partitioned accross four knowledge bases: the domain knowledge (i.e., structural and functional knowledge); the know-how of the simulation (compiled functional knowledge); the know-how of the diagnosis (another compiled functional knowledge); the management of interaction among tasks and with the user.

3) Control Strategy

An exhaustive and systematic diagnosis of equipment faults is used in SEPT. This strategy differs considerably from that of the operator who first checks for circuit breakers that are opened and attempts to locate the fault. The operator may eventually detect faulty equipment that originated the problem but this is not his/her main objective.

The main goal is to find any faulty equipment types (for example, a busbar, a line, a controller, a cut-off system, relays, circuit breakers, etc.). The location of the fault, although important, is not the main concern. This approach has generated considerable interest in the commercial version, but has been less accepted by the operators who had trouble following the reasoning of the system. This has been alleviated somewhat by the explanation module.

4) Dynamic Control while Reasoning

The diagnosis of the control systems has three aspects:

(1) The identification of abnormal behaviour in the control systems based on observations which include conflicting and redundant information.

(2) The diagnosis of the controlled system.

(3) The inter-relations between the controlled system and the control system.

Some heuristics are used to speed up the diagnostic process. They are used to check if behaviours corresponding to correct function of the control system are present and if so, this knowledge is used to skip part of the diagnostic path.

The diagnosis of the control system relies on the redundancy of information. One of the problems is that equipment may send false information. For example, a protective relay may not detect the fault in the correct section of the network. Thus, the protective relay may be declared as the culprit once the location of the fault is known, but the location of the fault is derived from the information originated by the protective relays. To resolve this problem of uncertainty, a vote among the protective relays is taken which minimizes the false alerts [Fauquembergue et al. 88].

Points of interruption are introduced in the structural and functional hierarchy in order to permit the user to explore aspects of the diagnostic reasoning at various levels of detail through the simulator. These points of interruption may be activated and inhibited. Encountering an active point of interruption, the inference engine asks the user to define the context of the desired observation. The user can choose a combination of elementary interpretations which are explanations attached to the action part of each rule within a rule packet in the current context of reasoning. Then, it executes the corresponding part of the reasoning and returns to the point of interruption. This cycle ends when the user inhibits the point of interruption.

RATIONALE

Objective of KBS

The objective of the development of RATIONALE was to implement a tool that can be used to reason by explaining to the user [Abu-Hakima 88; 90]. The system tracks all user reasoning and maintains a dynamic trace which is filtered for explanation. It supports dynamic explanation of its reasoning and static explanation of its methodologies of reasoning given a particular fault. Its dynamic explanations include why, why-not, how and what-if explanations in an already established context while reasoning.

Application of KBS

RATIONALE's first application is the diagnosis of the 1109 Xerox workstation from Xerox Inc.. Its knowledge includes knowledge of communication, power, software and hardware faults of the workstation. These faults are diagnosed to the computer card level.

Problem Solving Strategy Constraints related to Context

1) Knowledge Organization

The knowledge in RATIONALE is organized into a fault hierarchy with the root node representing a workstation fault. The hierarchy includes approximately 30 nodes with 10 terminal nodes. The root is refined into one of the four general types of faults, hardware, software, power or communication problems. Each general fault is subsequently refined into more specific faults. Faults may share parents and children within the four branches of faults. The faults are eventually refined to terminal nodes which indicate repairs, adjustments or replacements of worstation cards or components on cards such as memeory or cpu chips for example.

2) Knowledge Representation

RATIONALE's knowledge representation includes two types of frames, top goal and subgoal frames. RATIONALE was designed as a tool that allows a user to define multiple knowledge bases, thus, the top goal is the entry point into a specific knowledge base. Currently, only the workstation diagnosis topgoal exists. The subgoal frame is shown in Figure 2.

```
add_object( SubG_name,                              {not a slot}
            Refined_subG_name,                      {not a slot}
                [subG(refined_from,refined_SG)              {parental knowledge}
                 subG_name(refined_SG),
                 novice_explanation(NE),
                 experienced_explanation(EE),
                 expert_explanation(EXE),
                 explanation_detail(Complexity_of_subgoal,Importance_ranking),
                 enabling_conditions(Enable_symptoms, Enabled_degree),
                 invalidating_conditions(Invalid_symptoms, Invalid_degree),
                 tolerance_conditions(Special_case_symptoms, Tolerance_degree),
                 alternate_subG(SubG_name,shared_symptoms(SS), Alternate_degree),
                 urgency_indication(Urgency),
                 refinement_subG(Refine_to_subG_name,Suff_symptoms,Refine_degree)]).{children}
```

Figure 2: RATIONALE's subgoal frame template

RATIONALE explicitly encodes the parental node knowledge and in its refinement_subG slot, the children of a fault (child knowledge is implicit context consequence knowledge). RATIONALE has explanation slots which are essentially contextual help slots locally defined in a node for a novice, experienced or expert user. These form part of the text output as the reasoner generates an explanation for a user question. The explanation detail slot weighs the complexity of a subgoal as to its importance in a chain of reasoning so that the explanation facility can tailor its output for novice users and not overwhelm them with details. The enabling_conditions slot activates a particular node (this can be taken implicitly as part of the pre-conditions for entering a context represented by that node). The invalidating_conditions slot is used to override enabling_conditions so that a user may not skip the diagnosis of a fault before another (this is an implicit mechanism to control the reasoning through the nodes which in RATIONALE represent contexts). The tolerance_conditions are used to further override the invalidating_conditions for special cases (the invalidating and tolerance conditions form the in-context actions of the fault context in RATIONALE).

The alternate_subG slot allows a loosely related node to be introduced in the chain of reasoning thus changing the organization of knowledge in RATIONALE from a strict hierarchy to a directed network as in the case of the JETA knowledge (this is implicit knowledge of context consequences). Such a strategy is needed in real-world systems since faults are not neatly organized and multiple faults may indicate problems with loosely related subsystems or components.

3) Control Strategy

RATIONALE interprets locally encoded commands in the diagnostic network frame as JETA does. This permits context-sensitive branching in reasoning depending on values of the different parameters encoded in its slots. The diagnosis output to the user is based on locally encoded text which exists in the frame as explanation slots. This facilitates the user's understanding of the local context and the explanation of the fault in that context. RATIONALE allows the user to select a problem to diagnose or as in the case of JETA, select unknown so that the reasoner may search for the parameters that allow it to enable a particular fault. Jumps in the network are accommodated through the alternate faults encoded explicitly in a frame slot. The possible next moves in the network are represented as the refinements (children) of a node. RATIONALE's reasoning strategies navigate its network in two possible ways:

• Through node refinements by traversing the network in a manner similar to JETA's, mainly, from left to right and top down which implicitly relates its fault nodes. Any node can have multiple parents which permits more than one cause for a fault .

• Through alternate subgoal conditions that allow the reasoner to jump around in the network.

4) Dynamic Control while Reasoning

The reasoner dynamically traverses the network. RATIONALE has less dynamic flexibility in reasoning than JETA has, however, the reasoner in RATIONALE can:

• Block entry to a particular node if certain pre-conditions have not been met.
• Jump to an alternate node if intermediate results point to another strategy.
• Postpone a test or procedure because of user discretion.
• Provide contextual explanations of dynamic system reasoning.
• Let the user select from a list prioritized according to difficulty and likelihood of a test or failure. The list can be dynamically re-ordered depending upon observed fault combinations.
• Allow the user to select unknown and have the system identify the pre-conditions that would identify a fault context.

4.0 Reasoning with Context

In a typical diagnostic hierarchy structure such as that shown in Figure 3 (see [Halasz et al. 92] for JETA's), the root node indicates the most general problem with a device. In SEPT, the organization of nodes is directly related to the structural and functional hierarchy of the domain. In JETA and RATIONALE, the nodes organize the faults of a device and are refined to leaf nodes that represent components. The node is subsequently refined along device operational modes. These are further refined until they indicate specific behavioural faults on the components of a device [see Abu-Hakima 94a; 94b; 94d; 93b]. Navigation in the hierarchy is accomplished along two implicitly organized axes. The x-axis implies that movement from left to right represents the most likely and least expensive faults to resolve. This knowledge is based on the expertise of the technicians who make use of these heuristics as they troubleshoot. Their heuristics tell them what faults consume the least amount of time to check and correct. The y-axis organizes the knowledge top-down and is taken as the overall navigation strategy which is again based on implicit knowledge of heuristics for the likelihood of faults. Both the left-to-right and top-down control strategies are used by the reasoning algorithm in JETA for determining the next node(s) that represent faults that the user is advised on.

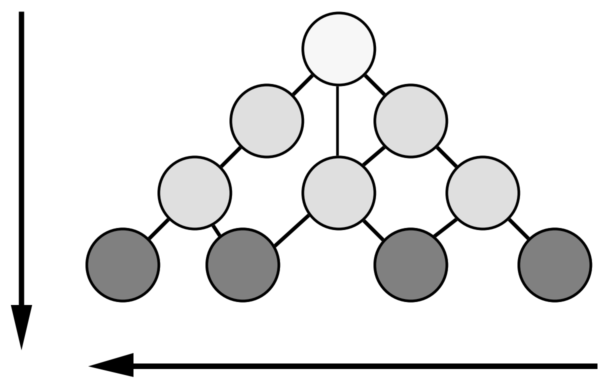

Figure 3: Typical diagnostic hierarchy with implicit reasoning control axes

In SEPT, the rule packets partition and structure the knowledge and can be considered to have the same role as nodes do in JETA and RATIONALE. The rule packets differ in their knowledge content and how they are used by SEPT's inferencing mechanism. The first rule in a rule packet represents the conditions for the correct operation of an equipment type. This rule is used to check pre-conditions of entering a context. If the rule indicates that the operation is correct, the context is exited. The remaining rules in the packet represent the various types of failure of that component (in-context actions). Thus, within each SEPT context, entered by traversing the rule packets, a combination of operational and fault knowledge is reasoned with.

In SEPT diagnosis is performed first from specific contexts (context of each type of equipment) to more general ones (context of the cut-off system, context of the busbar on which several cut-off systems are connected, and context of the substation). The function of equipment may appear correct at one level and incorrect at a higher level. The diagnosis presents three main results:

(1) a list of equipment not functioning properly in the control system;

(2) the location of the fault on the controlled system; and

(3) the incoherence between equipment function and the location of the fault.

The movement through the nodes representing the network by a reasoning algorithm is essentially movement through the various implicit diagnostic contexts. Although these contexts are not made explicit they essentially represent a diagnostic window into the knowledge whose movement is dictated by satisfying pre-conditions which move one from one context to the next. These pre-conditions are in the first relation that defines context, the function of the context, and are generally represented as the contents of a slot in a frame-based knowledge representation or as rule pre-conditions in a meta-rule system (or the first rule in a SEPT rule packet). These must be explicitly defined as establishing and moving context from one node to another.

As the pre-conditions of a context are satisfied, the actions in-context dictate problem solving (reasoning) itself. For example in JETA, once a node (context) is entered various tests and data acquisition is performed. These result in a decision as to what node (context) one moves to next. This is the consequent of completing the problem solving in a context and leads to the consequences of a context. In some knowledge representations, such as JETA's and RATIONALE's, a slot in the frame provides an explicit order for the movement from one node to the next based on locally established conditions. Some of these conditions eliminate or activate the traversal along various branches in the diagnostic knowledge. Such consequences also form part of the first relation that defines the context function.

In SEPT, the movement along a path of the reasoning is insured by the order in which rules are written in a rule packet (RP). An eventual consequence implies that a RP may be entered or not based on its first rule which determines if the detected operation is within bounds or not. If it is, the RP is exited without triggering all its in-context actions.

Such consequences lead to the second defined relation for context, namely, the movement from the current context to the next in problem solving. In the case of JETA, finishing with a diagnostic node may lead one to its activated children or siblings. In both JETA and SEPT, an activated node or context may trigger the activation of other nodes.for various test or repair actions. Control is returned to the originating node when the actions required are completed. Movement in both JETA and SEPT from one context to the next is governed by the change from one activated frame or triggered rule packet to another while reasoning. In SEPT,and JETA there is a systematic exploration of the diagnostic tree, and sometimes part of the diagnostic path is suspended if the pre-conditions of a context are not met. For example, in SEPT, a part is found to be operational so the RP is exited and in JETA, the activation conditions of a frame are not satisfied so a frame is not visited. The movement through a diagnostic hierarchy can be equated to movement through a semantic network or a conceptual graph defined by Sowa [Sowa 84]. Figure 4 graphically illustrates the equivalency of a diagnostic hierarchy with a semantic network.

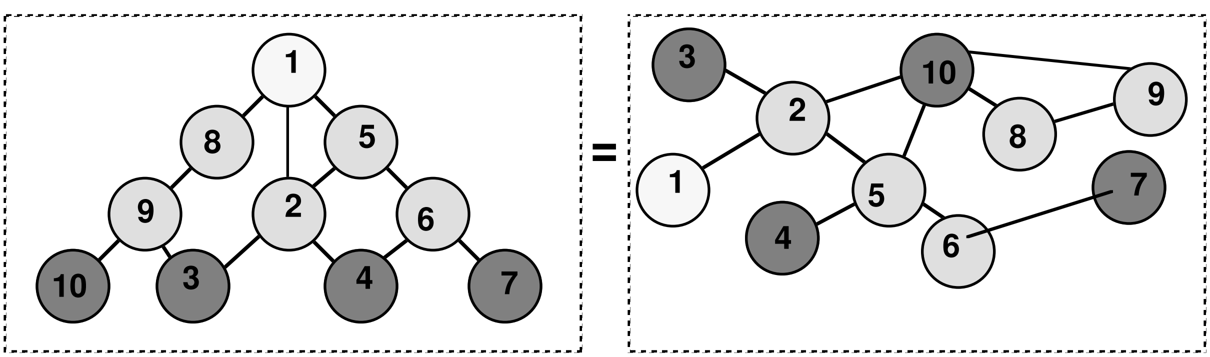

Figure 4: Illustrating the equivalence of a diagnostic hierarchy & a semantic network

As shown the nodes can be organized in any manner, however, one must make explicit that the axes of their organization represents the movement in problem solving from one context to another, the second relation that defines context.

As a system reasons, the static or passive knowledge represented is instantiated as dynamic or active knowledge. This active knowledge has in-context variables which are also instantiated to represent context in time. Some of these in-context variables are continous while others are discrete. This is the third relation defined for context, namely, the explicit representation of context in time.

Continuous time can be mapped to discrete movements as illlustrated with the example of Figure 5. A typical modelling strategy, traditionally used by control engineers, for time-variable components that have feedback is to examine some input variable with respect to its output variable at time t+1 [Abu-Hakima 94d]. Thus, between time t1 and t2, an input function may be constant but between time t2 and t3, it may rise proportionately with time. At time t3, the output may reach a maximum which implies that the input must be maintained as a constant until t4, where it is again allowed to increase proportionately with time and then it is held at some input maximum. In all cases the input has a direct relation to the output function whose charactersitics are fed back in time. Thus, the output function is fed back in time and it directly affects the value of the input function.

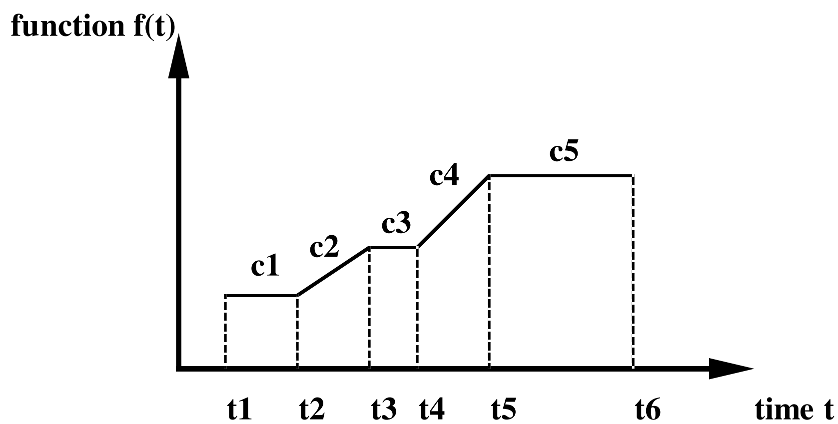

Figure 5: Representing the behaviour of time variable components in reasoning

In a similar manner, context can be modelled with respect to time so it may smoothly represent a continuous function. Thus, at time t1, the context c1 may be entered. Between time t1 and t2 the context is constant and does not exhibit any time-dependent behaviour.

At the end of t2, a second context c2 is entered. Between time t2 and t3, the context is a time-variable context. At the end of t3, a third context is entered, c3. Between t3 and t4, c3 is another context which is not time variable. At the end of t4, the context c4 is entered and this new context is time variable until t5. At the end of t5, the non-time variable context c5 is entered and continued with until t6. Thus, the two context relations of movement from one context to another and movement through time are represented in a manner that allows the dependence of one context on another as one moves through time.

Again, this relation of movement through time is implicitly represented in JETA by the slots local to a particular node that indicate the priorities of the movement to the next nodes. Dynamically, these node lists are tempered by the collected evidence at an instance in time so that a decision may be taken as to which node(s) to move to next. In the case of a conflict in the evidence collected, the implicit control strategy of moving to the leftmost node in a top-down manner is used to resolve the decision. If a conflict remains and the nodes are at the same level and are at the same leftmost position, the reasoner simply selects the first one in the list.

The described movement from contexts c1 to c5 is based on the consequences of the individual contexts. Multiple contexts can result from the consequences of a context. In diagnosis, a fault may lead to multiple faults. But the reasoner will prioritize them and move to one fault or through one context at a time.

In SEPT, the movement through time is represented by realized and potential events which can be explicitly represented because of the encoded structural and functional knowledge [Bau et al. 92]. An event is realized when the time of the simulation is equal or higher than the time at which an event must occur. An event is potential when the time of the simulation does not match the time at which the event must occur. Thus, a potential event may be never realized. This may be due to:

(1) another event inhibiting the realization of the potential event (e.g., a fault disappears after the protective relay sends a trip-order to the circuit breaker and before the time at which the latter must open),

(2) there is a functional fault (e.g., a circuit breaker does not open at the correct time), and

(3) there is a non functioning component (e.g., the circuit breaker does not open at all when the fault is present).

The diagnosis in SEPT may be understood as the identification of the context of a faulty event from observations first, and the identification of the fault second.

5.0 Explaining with Context

As described in the introduction of the three relations that define context, all system decisions that affect context must be made explicit. A system decision will lead to a system action which in turn will lead to an effect on context. Thus, all assumptions and predictable effects about movement from left to right and top-down in the reasoning of a system such as JETA or SEPT traverse a context, and must be represented explicitly. In the same manner, all assumptions and effects about time dependencies in movement from one context to the next must be explicit.

In the case of RATIONALE [Abu-Hakima 88; 90; 93a], as one moves from one context to another one tracks all context pre-conditions, in-context actions and context consequences. Such a path is graphically illustrated in Figure 6. Thus, as one moves from one context to the next, one has an accurate record of movement through the diagnostic nodes that represent the context and all their implications which can be used to generate detailed explanations. These explanations include the ability of a user to ask about movement through hypothetical chains of reasoning that would modify the context. Thus, if an explicit path of reasoning is not well understood by the user, the user can ask a what-if question and as a result generate a second hypothetical path based on the user's request which has an explicit set of pre-conditions, in-context actions and consequences. Traditional explanations in KBS do not usually include hypothetical explanation capabilities but should.

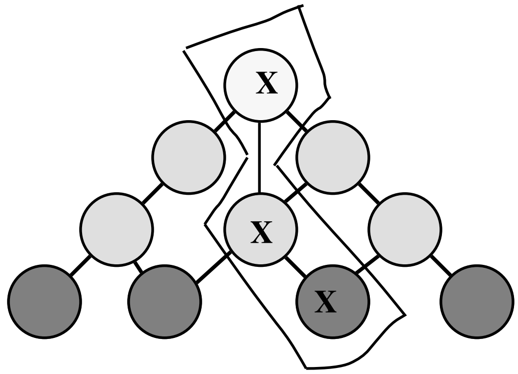

Figure 6: A path in RATIONALE's diagnostic reasoning hierarchy

The notion of interpretation, used in SEPT, is a manner for the user to define a window (context) on a part of the diagnostic reasoning. In this window, the user may observe that part of the reasoning with various levels of detail. Thus, the diagnosis is performed in two steps. SEPT first gives the results of the diagnosis (faulty equipment parts, location of the fault on the controlled system, incoherence between equipment behavior and the fault location). SEPT then replays the diagnostic reasoning in a context defined by the user. This context corresponds to the observation of a slice of the reasoning by the user with varying levels of detail which are obtained through successive simulation of the specified slice of reasoning. The user has the option to modify the timing (e.g., the delay between sending a trip order by a protective relay and the opening of the corresponding circuit breaker) and observe the consequences on the result of the diagnosis. This opportunity permits the user to check the knowledge, and, eventually, correct the real timing.

In many knowledge-based systems, contextual help is generated for the user as a form of describing system behaviour. Contextual help is not explanation but it supports the user in the session based on static rather than dynamic knowledge. In the case of JETA, hypertext links are used to chain all user-explainable knowledge in context [Abu-Hakima et al. 93a]. Thus, any boldfaced text indicates a hypertext link which provides the user with in-context help. This method of helping the user allows the user the flexibility of choosing when and what to ask for contextual help on.

RATIONALE shortens the chains explained to the user based on previously published strategies put forth by Wallis where explanations for a novice/intermediate/expert user are provided based on a prioritization of the importance of a concept in the reasoning chain with respect to the complexity of its explanation [Wallis 82]. Other work has gone further in attempting to identify the parameters most sensitive in diagnosis and select the chain of reasoning affected by those parameters to explain to the user [Alvarez 92]. Such a strategy allows a user to further explore hypothetical lines of reasoning to confirm the contextual decisions of the system.

SEPT does not take build explanations in a manner similar to RATIONALE's. However, SEPT offers the user various mechanisms through which to explore the diagnostic tree. The user may use the various points of interruption mechansim to suspend the reasoning and look for various events. The user is free to activate or not to activate these points of interruption. When the system reaches such an action point, it asks the user to indicate the context of observation (the pieces of equipment concerned, the level of detail, etc.). Then, SEPT replays the part of the reasoning in the specified context and returns to the point of interruption until the user accepts to move on to another path in the diagnosis.

6.0 Use of Context for Incremental Knowledge Acquisition through Cooperative Explanation

Typically, current knowledge-based systems acquire knowledge in a monolithic manner. The faults in a diagnostic hierarchy or netwrok are defined and their basic form of knowledge representation (frames, meta-rules, etc.) is used as a template to acquire knowledge. The knowledge is tested only when the system reasons. If a problem with the knowledge is detected, the developer is forced into a monolithic network to modify the knowledge, re-load (re-compile or re-interpret) and then test the reasoning again. It would be much more useful if the developer or the domain expert had the ability to incrementally modify the knowledge. This section describes the new approach of incremental knowledge acquisition.

The knowledge that is often missing in SEPT (i.e., the knowledge coming from the human expert) is knowledge that the operator possesses. Such knowledge differentiates between what is planned and what really exist. This includes the state changes of equipment parts that are normally recorded on a central printer. (SEPT works from this set of signals.) For instance, when a fault disappears from the controlled system, each protective relay must send a signal indicating the halt of its operations. This is the knowledge that the human expert has introduced in SEPT. However, in practice, these signals are not always recorded in some substations, and, expecting this missing information, SEPT indicates a fault on the protective relays. SEPT could detect such a recurrent situation with successive diagnoses of these incidents, and ask the user the specific reasons to explain this situation. Thus, SEPT could acquire the knowledge that these signals are effectively not recorded and then are not to be checked if it had the ability to incrementally acquire knowledge

RATIONALE acquires and represents explicit knowledge so that it may generate a range of explanations for an end user. MYCIN generates explanation in a classical manner for knowledge-based systems introduced with earlier expert systems [Scott et al. 77; Clancey 86] . It aligns its explanation facility directly with the reasoning paths that define movement accross contexts of the diagnostic system. Thus, all traces of reasoning that represent the contexts moved through are kept and their contents provided to the user for explanation. Unlike RATIONALE, MYCIN is not selective in its construction of explanations. In addition, MYCIN has a flat rule base which is not the well structured network that RATIONALE reasons with. As a result, causality in MYCIN is more difficult to discern than in RATIONALE or any other well-structured KBS.

In later systems such as that described by Wick [Paris et al. 88; Wick et al. 92], an explanation facility is aligned only periodically with the reasoning of the system. This differs from both RATIONALE and MYCIN which are classical in their explanation generation since they continously align the explanation facility with the reasoner. In Wick's system, only some of the contexts that the system reasons with are explained to the user. The differences between these two strategies are illustrated in Figure 7. In the second case additional explanatory knowledge may be used to generate enhanced explanations. This implies that the explanation path diverges or separates from the path of reasoning to produce effective explanations. This can be likened to a teacher clarifying a concept to a student by using simplistic knowledge without minimal detail so as not to overwhelm the student. Even though the explanation may not be absolutely correct, the student may find it easier to grasp the richly explained concept. One problem with such an approach is that it may be unsuitable for critical or real-time applications such as those whose results may affect the safety of processes and people. For example, Wick's approach would be unacceptable for the generation of explanations in an aircraft engine troubleshooting system such as JETA. An aircraft mechanic is held accountable for repairs and adjustments to subsystems and must be able to understand the consequences of any actions. If the explanation does not truly follow the path of reasoning, the mechanic may be mislead into believing that an action produces a desired result when it truly does not.

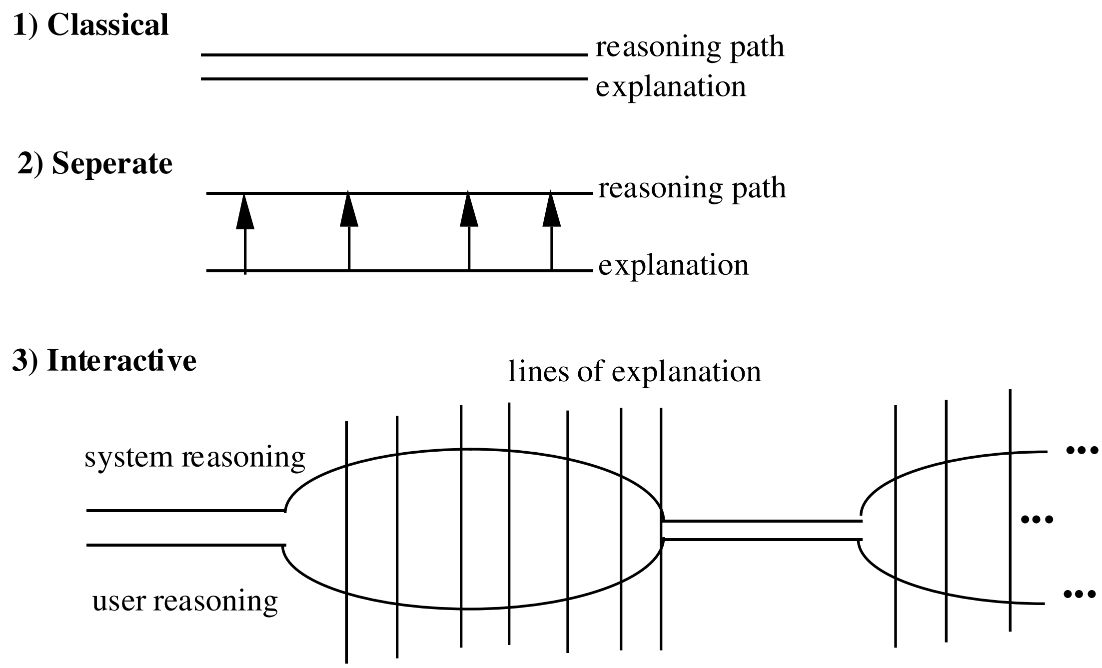

Figure 7: Explanation and paths of reasoning

A new approach for explanation is to accept that the reasoning of the system is often different than that of the user. In order to align the system's reasoning with that of the user and vice versa, explanation in the form of clarification dialogue can be used. This clarification dialogue may uncover inconsistencies or gaps in the knowledge of either the system or the user.

Another consideration for incremental knowledge acquisition is that the user may have part of an explanation that the system does not have. Thus, an explanation could be co-constructed by the system and the user. Early KBSs contruct the explanation for the user in a non-interactive manner, they may eventually check the effect of the utterance by requesting user feedback. SEPT acts in a reverse but related manner, by giving the user control over the point of explanation. However, SEPT does not allow the user to incrementally add knowledge that is absent.

For missing system knowledge, the user should be provided with the capability to add new knowledge through explanation. This form of explanation becomes more of a cooperation between the user and the system [Karsenty et al. 94]. This is cooperative problem solving between the user and the system so that they may cooperate to improve the decisions made by the system through the correction and addition of new knowledge.

As a result of introducing new knowledge to the system, the system must have the ability to validate acquired knowledge. The definition of validation adhered to here is that put forth by Laurent [Laurent 92] to imply the verification and evaluation of all propagated changes to the system. Laurent further defines validation as a combination of objective and interpretive validation, where the formal and pseudo-formal specifications of the system are evaluated. In the case of interactive knowledge acquisition through explanation, a user introduces a pseudo-formal specification for a change in the knowledge, thus, this change is evaluated with respect to the other knowledge in the system. Such incremental knowledge acquisition is essential in Intelligent Assistant Systems designed to aid users in problem solving [Brezillon 94].

For the user to interactively add knowledge the system must be able to acquire the knowledge in a structured and general manner. For this reason, it is essential that the knowledge in the system be structured and tractable. Structured and tractable systems are key in traditional software engineering systems which use modules to compartmentalize information (whether it is data or procedures) and localize side effects in executing software. All knowledge-based systems practicioners have yet to accept the obvious need for adopting some of the traditional and very successful software engineering methods. However, some KBS developers have argued for the explicit and structured representation of knowledge [Chandrasekaran 83; Halasz et al. 92; Swartout 81; Abu-Hakima 88; Abu-Hakima 94a].

For the system to acquire knowledge incrementally, the context of reasoning must be made explicit and explainable. This context represents the traversal of nodes through various actions in a particular context. Thus, a single context may contribute to reasoning about one or more nodes. There are four distinct steps that would allow a user to add new knowledge to a structured knowledge base:

1) acquire new context functional knowledge (including pre-conditions, in-context actions and consequences),

2) assimilate new knowledge,

3) propagate effects of new knowledge and

4) create any additional supporting knowledge.

To allow the user to add knowledge interactively, a generalized knowledge browser/editor could be used such as that developed for JETA [Abu-Hakima et al. 91]. In a session the user would be interacting with the main screen of the knowledge-based system shown in Figure 7 on the left. A user could ask to add new knowledge through a menu selection or in the course of being provided with a contextual explanation. This would pop up the browser/editor and start the 4-step process of interactive knowledge acquisition.

In step one, the user is given the opportunity to graphically introduce a new node in relation to the nodes that are grouped to provide the current context as illustrated in Figure 8. Here the user wishes to add node 5' after node 5.

Once the user indicates where the node should go, a frame template for node 5' would be popped up. In this template, the user is prompted for the frame name of node 5' as well as the type of node in slot 4. In a structured knowledge representation such as JETA's the node type indicates whether a node is a terminal node (repair, replace or adjust a component) or an intermediate node (general or test). In either case the user is moved to step 2 to assimilate the newly acquired knowledge interactively. In the case of an intermediate node, the user is prompted for the child(ren) of the node. Again, these can be indicated graphically or textually in the frame template. The user is also prompted for sibling nodes in both the terminal and intermediate node case. In the case of a node with parental knowledge, the user is asked if any of the specifics of that knowledge should be automatically updated in the new node frame.

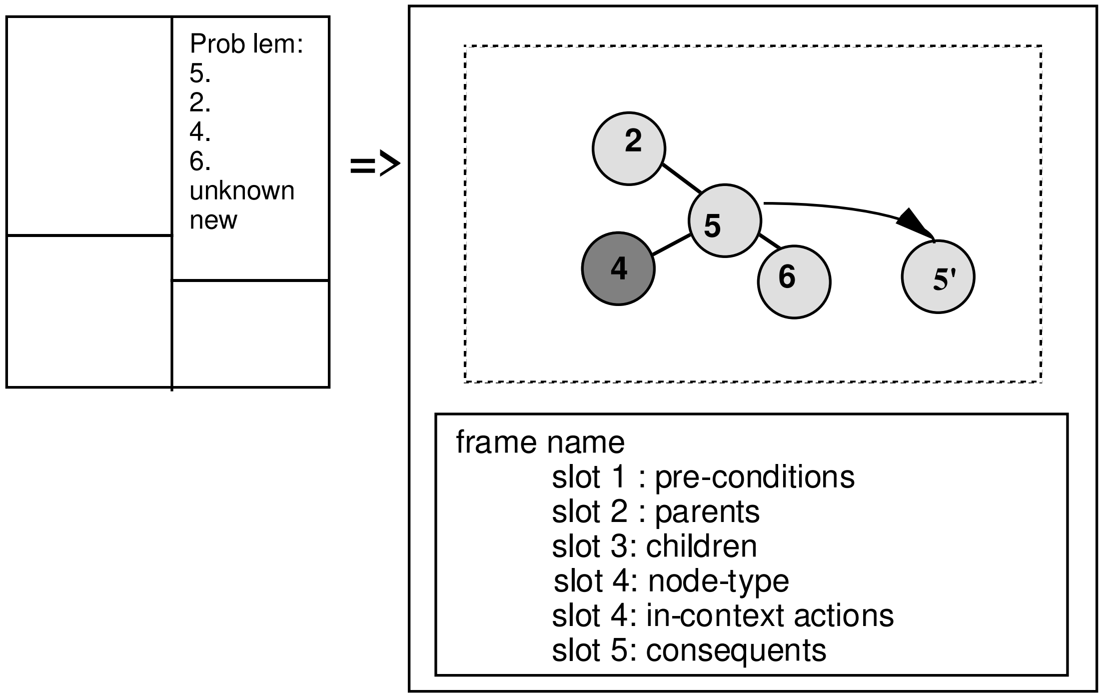

Figure 8: Browser/Editor window

Once the node and its links are added and any new knowledge is textually put into the respective node frame the user continues with step 3, propagate new knowledge. This is the validation step where the new knowledge is verified and evaluated based on the user's pseudo-formal specification. Any syntactic errors are flagged and any clashes or inconsistencies with the existing knowledge are detected.

In step 4, the results of step 3 are used to guide the user as to any additions or deletions that should be made. For example, if the new node clashes with an existing node, the user is given a set of constraints from which to select handling them.

In this manner, the system is given the ability to acquire knowledge incrementally. Through the cooperation between the system and the user and the explicit definition and use of context, new knowledge is acquired and validated. This knowledge is acquired through clarification dialogue. Thus, if the system can identify gradually through its dialogue with the user, the target knowledge, it can verify if this knowledge is missing. Thus, knowledge is acquired when needed. In some cases this knowledge may be present elsewhere in the knowledge base but is missing a link to the current context. This link could be added in the form of a consequence of the current context.

7.0 Future Real-World Applications of Context and Conclusions

Several real-world applications that require context will be addressed in our future work. The first of these is the servicing of diesel engines on highway trucks. The second is the diagnosis of aircraft operations. The third is the generalization of SEPT so that it may be re-used for substations with differing environmental constraints. The fourth is the use of context to aid DNA sequencing in the Genome project. The fifth is the development of a cooperative process between the operator and the process in a nuclear power plant. These four applications are described further below.

Diesel Engines

There are many models of highway trucks. Each model has a variety of subsystems that include a diesel engine, brakes, transmission, etc. For each model of a truck there are additionally many differently manufactured subsystems. As a result, the problem of servicing a truck is highly reliant on the model, manufacturer and subsystem specifics. If one takes only the diesel engine of a truck one has 50 implementations of a similar engine. Context can be used to determine the implementation and aid in the diagnosis of diesel engines of highway trucks. One can start with a generalized knowledge base that encodes the basic subsystem faults of the diesel engine. Context could then be used to tailor the representation, reasoning and explanation of the KBS. Thus, for the 50 types of implementations of the same basic engine, one could develop a generalized knowledge base that can then be specialized on the basis of the context of the engine being diagnosed. As gaps or inconsistencies in the knowledge are found, the knowledge can be corrected incrementally through clarification dialogue with the domain expert and the system.

Diagnosis of Aircraft Operations

Context could be used in decision support to determine the dimension of diagnostic decisions for aircraft engines. Currently, commercial airlines monitor the subsystems of the aircraft periodically and based on the severity of a problem, an aircraft may be taken out of service so that it may be repaired.

This is what has been referred to as "just in time diagnosis" [Halasz 92]. The dimensions of the diagnosis decision may include:

- the operational cost of taking the aircraft out of service, this is measured in dollars based on the loss of paying passengers,
- the time any repair is expected to take based on the expertise of the technician,
- the availability of inventory parts to service the aircraft,
- the most tolerable delay in waiting for the repair so that it minimizes the aircraft downtime (this may be gauged on being able to perform the repair at a central maintenance location versus a remote location where personnel are not available), and
- the danger to passenger safety in delaying a repair.

Context can be used to aid in the decision of effecting a repair on the aircraft. Each of the dimensions of the diagnosis decision can be represented explicitly in the function relation of context. They can be used as pre-conditions for entering a specific context. In addition, the in-context actions and consequences can reflect each of the dimensions of the aircraft repair decision explicitly.

Facilitating the Portability of Knowledge Bases

A system being developed for Electricité de France is KA7 (Knowledge Acquisition for SEPT - SEPT means seven in French) [Perrot et al. 93]. As previously mentioned, the Knowledge-Craft version of SEPT is operational in two substations in France.

Although SEPT contains theoretical knowledge that would apply to all substations, environmental constraints lead to the re-writing of SEPT for each substation. KA7 aims to ensure the portability of the KBS to other substations. It acts as a configurator of knowledge and automatically generates a clone of SEPT that is tailored to the specifications of a given substation. KA7 acquires these specifications by interacting with the operator of the substation.

KA7 is a system that permits the customization of a SEPT clone to the context of a given EHV substation. In this case, the context is defined from specifications of the substation (e.g., the number of lines, the type of equipment, the timing rules, etc.).

Use of Context in sequencing DNA for Genome Project

Software currently exists that analyzes DNA sequences from image of gel autoradiograms. Due to experimental conditions some ambiguities appear on the images. The software acknowledges the presence of an ambiguity but it cannot resolve it. Thus, a biologist must intervene to solve the ambiguity on the basis of additional pieces of knowledge on the context of the experiment (temperature of the solution, sensibility of the film used for the autoradiograms, the position of other nucleotic acids around the ambiguity, etc.). Such pieces of knowledge constitute the context of solving the ambiguity.

A system that will acquire this contextual knowledge of ambiguities and re-using it after for automatically solving similar ambiguities is under development [Brézillon et al. 93]. The system is called SAIC that stands for Sequence Acquisition In Context. SAIC is designed to interwork with existing software used by biologists.

The acquisition of the context in which a user solves a problem is a means for the system to intelligently and cooperatively support the user.

Development of a Cooperative Process in a Nuclear Power Plant

An operator is in charge of monitoring a process in the treatment of combustibles in nuclear plants [Minault and Pénalva 93]. Due to continuous automation of the process, the operator faces complex and numerous pieces of information that are difficult to understand, organize, integrate and interpret.

An Intelligent Assistant System (IAS) is planned that will provide the operator with information that is significant in the current context, and explanations on the various alternatives. The IAS is expected to work in two modes.

- A waking state in which the IAS will take the initiative to alert the operator to act when necessary;
- A cooperative state in which the IAS will participate in the problem solving in response to an operator request.

The IAS would use models of the operator-in-action, of the process and of the current situation. In this application, the relationships between two contexts (e.g., the waking state and the cooperation state) will be studied. The contextual mechanisms to tranfer information and knowledge from one context to another one will also be analysed. Explaining the context of the cooperation will be an important element as it should be with any IAS.

Conclusions

This paper has described how context can be represented, reasoned with and explained so that it may facilitate the incremental acquisition of knowledge. Additionally, if context is represented such as in RATIONALE or SEPT, it can improve the explainability of the knowledge represented and reasoned with.

The main concern of this paper has been to elaborate on the representation, reasoning, explanation and acquisition of knowledge through context. Context has been described in diagnostic problem solving with knowledge-based systems. Three relations that define context have been presented. First, the function of context embodied in the pre-conditions for a context, actions in-context and consequences of a context have been described. Second, the representation and reasoning for the movement from one context to another has been described. Third, the movement of context in time for both time variable and non time variable contexts has been described.

The second key concern of the paper has been to describe how context can be used to better explicitly represent, reason with, explain and incrementally acquire knowledge cooperatively with the domain expert. This has been addressed by the introduction of a new method for acquiring knowledge through cooperative problem solving between the domain expert and the system. Thus, as the system reasons and the user asks for explanation, the user discovers that the context of reasoning of the system does not match what the user expected. Thus, the problem solving becomes the attempt to align the reasoning of the system with that of the user through clarification dialogue. This clarification dialogue with the system may result in the acquisition of new knowledge.

To facilitate the description of context and show its practicality, three real-world applications, JETA, RATIONALE and SEPT were described. In addition, five future real-world applications that require context were introduced.

Acknowledgements

Mike Halasz and Sieu Phan who were principals on the JETA project team are gratefully acknowledged. The SEPT project is the subject of a contract between LAFORIA, Electricité de France and Airial Inc.. Patrick Brezillon recieved funding from the GDR 1029 of CNRS for the application in Genetics. We would also like to acknowledge the University of Paris VI, CNRS and the National Research Council of Canada for supporting this collaboration.

References

[Abu-Hakima 94a] Abu-Hakima, S., Automatic Extraction of Device Models from Fault Knowledge: the Diagnostic Remodeler Algorithm. To be published in the Proceedings of the AAAI-94 Workshop on Device Function. To be held in August 1994, Seattle, Washington, USA. AAAI Press(1994).

[Abu-Hakima 94b] Abu-Hakima, S., Automated Model Acquisition: the Diagnostic Remodeler Algorithm. Submitted to the Fifth International Workshop on Principles of Diagnosis. To be held in October 1994, New Paltz, New York, USA. (1994).

[Abu-Hakima 94c] Abu-Hakima, S., Diagnostic techniques in knowledge-based systems: a review of approaches, applications and issues. 100 p. NRC ERA (1994).

[Abu-Hakima 94d] Abu-Hakima, S.,, PhD thesis, Carleton University, Ottawa, Ontario, Canada (1994).

[Abu-Hakima 93a] Abu-Hakima, S., Automatic Knowledge Acquisition in Diagnosis. Proceedings of DX-93, Fourth International Workshop on Principles of Diagnosis. Aberystwyth, Wales. 236-250. (1993).

[Abu-Hakima 93b] Abu-Hakima, S., Halasz, M., and Phan, S., An Approach to Hypermedia in Diagnostic Systems. Chapter 10 in Intelligent Multimedia Interfaces, Edited by Mark Maybury, AAAI Press (1993).

[Abu-Hakima 93c] Abu-Hakima, S., The Use of Context in Diagnostic Systems. IJCAI-93 Workshop on the Use of Context, Chambery, France (1993).

[Abu-Hakima et al. 91] Abu-Hakima S., Halasz M., Phan S. and Kornelson K., A graphical knowledge browser for the jet troubleshooting assistant (JETA), Proceedings of the Workshop on Expert Systems, May 1991.

[Abu-Hakima 90] Abu-Hakima, S., and Oppacher, F., Improving explanations in knowledge-based systems: RATIONALE. Knowledge Acquisition Journal (2). 301-343. December 1990.

[Abu-Hakima 88] Abu-Hakima, S., RATIONALE: A Tool for Developing Knowledge-Based Systems that Explain by Reasoning Explicitly. Masters thesis, Carleton University, Ottawa, Canada. May 1988.

[Alvarez 92] Alvarez, I., Explication morphologique, un mode d'explication fondée sur la géométrie, Thesis, University Paris 6, November 1992.

[Bau et al.92] Bau D.Y. and Brézillon P., Model-based diagnosis of power station control systems: the SEPT experiment", IEEE Expert, February 1992, pp. 36-44.

[Brézillon et al. 88] Brézillon P., Fauquembergue P., & Hertz A., SEPT, an expert system approach for the monitoring of EVH substation control equipment, Proceedings of the Symposium on Expert Systems Application to Power System, Stockholm-Helsinki, 1988, pp.6.9-6.13.

[Brézillon et al. 90] Brézillon P., Bau D.Y., Fauquembergue P., Hertz A. & Maizener A. (1990) Elaboration of the SEPT expert system as a coupling of a simulator and a diagnostician, Proc. of the Third International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems,Charleston, South Carolina, Vol.I, pp. 54-60.

[Brézillon 90] Brézillon P., Interpretation and rule packet in expert systems. Application to the SEPT expert system, Lecture Notes in Artificial Intelligence, N° 444, Knowledge Based Computer Systems, S. Ramani R. Chandrasekar K.S.R. Anjaneyulu (eds.), Springer-Verlag, 1990, p. 78-87.

[Brézillon 91] Brézillon P., METAL: a language for structured knowledge-based systems, IJCAI-91 Workshop on Software Engineering for Knowledge-Based Systems, Sydney, 1991, pp. 11-22.

[Brezillon et al. 93] Brézillon P , Zaraté P & Saci F (1993) "AI approach in analysis of DNA sequences", Biochimie, Special Issue on "Informatique et Génome", 75(5): 337-345.

[Brezillon 94] Brézillon P., Design of an intelligent assistant system from various applications, International Conference on Expert Systems for Development, Bangkok, Thailand, 1994, pp. 228-233.

[Chandrasekaran 83] Chandrasekaran, B. and Mittal, S., “Deep versus compiled knowledge approaches to diagnostic problem-solving”. International Journal of Man-Machine Studies, 19(5), 425-436.(1983).

[Clancey 86] Clancey, W.J., “From GUIDON to NEOMYCIN and HERACLES in twenty short lessons: ORN final report 1979-1985”. The AI Magazine, pp. 40-60,(1986).

[Fauquemberg et al. 88] Fauquembergue P., Brézillon P., & Harmand Y., Synthesis of events in an EHV substation : an expert system approach. Proceedings of the IFAC Symposium on Power Systems, Modelling and Control, Applications, Brussels, 1988, pp. 16.6.1- 16.6.5.

[Giunchiglia 93] Giunchiglia F., Non-omniscient belief as context-based reasoning, Proceedings of the 13th IJCAI, 1993, pp. 548-554.

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

[Halasz et al. 92] Halasz, M., Davidson, P., Abu-Hakima, S., and Phan, S., JETA: A Knowledge-based Approach to Aircraft Gas Turbine Engine Maintenance. Journal of Applied Intelligence, 2, pp. 25-46, Kluwer Academic Publishers, NRC 31832, (1992).

[Halasz 92] "In time Diagnosis", personal communication with Mike Halasz at the National Research Council of Canada, Ottawa, Canada, (1992).

[Karsenty and Brézillon 94] Karsenty L. and Brézillon P., Cooperative problem solving and explanation. International Journal of Expert Systems With Applications, 1994 (to appear).

[Laurent 92] Laurent J. - P., A proposal for a valid terminology in KBS validation. ECAI-92. 10th European conference on Artificial Intelligence. Edited by B. Neumann. Published by John Wiley & sons. 1992, pp. 829-834.

[Lespérance 93] Lespérance Y., An approach to modeling indexicality in action and communication, Research Repport 93/13 of LAFORIA, April 1993, pp. 75-85.

[McCarthy 93] McCarthy J., Notes on formalizing context, Proceedings of the 13th IJCAI, 1993, pp. 555-561.

[Minault and Pénalva 93] Minault S. and Pénalva J.M., Vers une approche coopérative de l'aide à l'opérateur d'exploitation, Actes de la Journée "Explication et Coopération: vers la co-construction d'explications", Rapport de Recherche 105, Laboratoire d'Ergonomie, CNAM, 41 rue Gay Lussac, F75005 Paris, France, juin 1993, pp. 56-62.

[Paris et al. 88] Paris C.L., Wick M.R. and Thompson W.B., The line of reasoning versus the line of explanation, Proceedings of AAAI Workshop on Explanation, 1998, pp. 4-7.

[Perrot et al. 93] Perrot L., Brézillon P., & Fauquembergue P., Towards automatic generation of knowledge bases for diagnosis systems in the field of power systems, Expert Systems Application to Power System,ESAP'93, 1993.

[Scott et al. 77] Scott, A.C., Clancey, W.J., Davis, R. and Shortliffe, E.H., “Explanation capabilities of production-based consultation systems”. Stanford University, Dept. of Computer Science, Report No. STAN-CS-77-593,(1977).

[Sowa 84] Sowa J.F., Conceptual Structures: Information Processing in Mind and Machine, Addison-Wisley, Reading, MA, 1994.

[Swartout 83] Swartout W.R., XPLAIN: a system for creating and explaining expert consulting programs, Artificial Intelligence, 1983, 21, pp. 285-325.

[Wallis 82] Wallis J.W. and Shortliffe E.H. Explanatory power for medical expert systems: studies in the representation of causal relationships for clinical consultations. Methods of Information in Medicine (Germany), 21, 127-136.

[Wick 92] Wick M.R. and Thompson W.B., Reconstructive expert system explanation, Artificial Intelligence, 1992, 54, pp. 33-70

ERROR: undefined
OFFENDING COMMAND: ÿ

STACK: