An Architecture for the Expert System for the Telecommunications Internetworking Design

조대연*

Cho, Dai Yon*

Abstract

CBR is a knowledge-based system that utilizes the previous knowledge or experience to solve the current problem. In previous CBR research, the emphases are mainly put on the development of more sophisticated indexing mechanism for past cases or the most similar case retrieving methodology out of a group of previous cases. In this paper, discussed is a CBR system that is able to take advantage of the case or knowledge that does not belong to the past in the telecommunications internetworking design area. And the architecture for such CBR system is proposed. Finally, the performance of the CBR system is shown through an ablation experiment.

Key words: Case-Based System, Expert system, Internetworking, Network design, Telecommunications

* 한동대학교 경영경제학부

Introduction

CBR (Case-Based Reasoning) has been rapidly emerging as one of the AI (Artificial Intelligence) problem-solving paradigms employed by expert systems. It is regarded as an alternative to such paradigms as rule-based reasoning (e.g., Buchanan & Shortliffe, 1984) and model-based reasoning (e.g., Hamscher, 1991). CBR has been recognized as a better problem-solving approach especially tothose problems from such domain as design that makes use of past experience heavily because it is based on the premise of experiential reasoning or analogical reasoning in which past cases are used to solve current problems (Kolodner, 1991).

In telecommunications network design area, however, there has been very little research has been done using the CBR approach. This paper discusses the appropriate architecture of the system that takes the CBR approach in the internetworking design.

The Knowledge-Based Approaches

Computer-assisted network design may be divided into two categories: the "calculator" model and the "logical search" model (Kan, 1986).

The calculator model computes or simulates the consequences of network design alternatives under the control of a human designer. The analytic approach for the network design using queuing models, Markovian models, Petri Net models, and the simulation-oriented approach based on these models, belong to the calculator model.

The logical search model first creates a "search space" that contains all the possible solutions to the problem, and then performs a search by using theorems or heuristic algorithms from operations research and graph theory to find the optimal solution. Gersht and Weihmayer (1990) proposed a heuristic computational procedure that jointly considers the problem of link capacity and link type selection, and routing as well as the problem of topological design.

The logical search approach is based on proven theorems and well-established algorithms. Many problems in the real world, however, cannot match the required theoretical conditions. These are the places where AI-based research can play a role.

Pierre and Hoang (1990) attempted to apply an AI approach to solve the design problem of determining the geographical locations of switching nodes, a backbone network design problem, in wide area networks. The authors proposed a general framework for integrating conventional, approximate methods and knowledge-based systems.

Feinstein et al. (1988) adopted a rule-based approach, and developed an expert system called XTEL for assisting the design of the military communication network. XTEL emulates a telecommunications system architects activities in designing a military communications network, and employs rules derived from telecommunications system design experts to develop design.

Thuraisingham (1989) suggested a framework for developing an expert network design system. He divides the network design problem into three smaller subproblems: modeling, analysis and

synthesis problems. His system is a planning system, that is based on an intermediate subgoal tree.

The above telecommunication network design systems that have been developed so far are all based on a knowledge-based approach, using methods of rule-based reasoning, constraint satisfaction, and goal-directed search. No system has the capability of taking advantage of previous design experience.

The Case-Based Approach

In this section, we discuss the knowledge representation and the architecture of the case-based expert system called CIDA (Joh, 1996) that captures salient features of the cognitive model of an internetworking design process. The CIDA was implemented using GoldWorks IIITM, a development environment based on Golden Common LISPTM, and run on a Pentium PC. It currently consists of approximately 8,000 lines of Lisp code, 500 functions, 200 frames, and 150 rules.

CIDAs architecture is centered around a case-based reasoning system. CIDA consists of 4 modules: a Problem Solver, a Case Manager, a memory, and a user interface.

1. Knowledge Representation

1.1. Case Representation: Frames

Cases, in CIDA, are instances of internetwork situations and their designs. Figures 1 shows CIDAs representation of an instance of an internetwork.

```
(DEFINE-INSTANCE CASE-87
  (:IS internetworking-design-case)
 (case-id abcnet-1987)
 (date 1987)
 (status actual)
 (media-bw-node-wc (rg58))
 (%-of-rg58-bw-node-wc 1)
 (%-of-utp-bw-node-wc 0)
 (media-bw-wc-be (multimode-fiber))
 (media-bw-be-hub (multimode-fiber))
 (media-bw-hub-hub (multimode-fiber))
 (media-bw-hub-super-hub ())
 (no-of-wc 160)
 (no-of-be 80)
 (no-of-hub-place 5)
 (no-of-bridge-segments 40)
 (no-of-router-segments 1)
 (no-of-wan 1)
 (type-of-wan t1)
 (no-of-network-nodes 2000)
 (bandwidth-req-for-backbone 10)
 (backbone-technology-used ethernet)
 (type-of-data asynchronous)
 (network-data-traffic-to-support
    case-s-87
    case-t-87))

(DEFINE-INSTANCE CASE-T-87
  (:IS data-traffic-case)
 (protocol-suits tcp/ip)
 (no-of-nodes 200)
 (type-of-lan (ieee-802.3 ethernet-v2))
 (Avg-bandwidth-for-node 0.1)
 (max-bandwidth-for-node 0.5))
```

[Figure 1] An Example of a Case for a Network

1.2. Design Heuristics: Daemons

The expert mentioned several rules of thumb that he uses to perform his design task. It can be seen that the rules are divided into two groups: general-level rules (the expert called them big rules) and detailed-level rules (the expert called them small rules). While general-level rules are closely related to the conceptual design phase, detailed-level rules have something to do with physical level design. The expert also claimed that both rules are acquired from his own experience. Examples of general-level rules are shown in Figure 2. The first rule indicates that the expert has a rule of putting a bridge in every Building Entrance that has more than 10 data lines.

IF	the number of cables coming in or going out from a Building Entrance is more than 10
then	the internetworking device at the Building Entrance should support the bridging function
IF	the number of cables coming in or going out from a Wiring Closet is less than 100
THEN	the internetworking device at Wiring Closet should support the repeating function
IF	the installed media at Wiring Closet is RG58.
THEN	the internetworking device at the Wiring Closet should support IEEE-802.3 MAC layer standard
IF	the installed media at Wiring Closet is RG58
THEN	the internetworking device at the Wiring Closet should support 10BASE2 Physical layer standard
IF	x number of RG58 cables are coming in or going out from a Wiring Closet
THEN	the internetworking device at Wiring Closet should support x number of BNC ports
IF	x number of optical fiber cables are coming in or going out from a Wiring Closet
THEN	the internetworking device at Wiring Closet should support x number of ST ports

[Figure 2] Examples of Design Knowledge Representation

Such design rules are represented as when-modified daemons in the frame that contains the information about the problem that is being solved. When information about the current problem is given to the system in the form of an instance of the frame, the daemons are triggered to generate constraints based on the given information, and to post them to another frame that contains the constraints on the network requirements. Using the constraints in that frame, appropriate internetworking devices for a conceptual solution are selected from the design component tree, described in the following section.

1.3. Adaptation Knowledge: Rules

When the solution to the retrieved case is not appropriate for the current problem, it is adapted using adaptation knowledge. Adaptation knowledge is represented in the form of rules. The IF part of a rule contains a description of the problem and its constraints, and the THEN part includes the actions needed to modify the old solution to the current problem. For example, Figure 3 shows an adaptation rule that adapt the previous design by not only increasing the number of the devices used but also by introducing a new device (a multiport transceiver) that is able to connect the previously existing devices and the newly added devices.

1.4. Design Component Tree: Object Hierarchy

In CIDA, design components are the internetworking devices that has the function of connecting subnetworks. These components form an object hierarchy in CIDA. At the top of the hierarchy is the internetworking device. At the next level down are a transceiver, a bridge, a router, a hub, and other internetworking devices.

IF	there are more incoming and outgoing cables than the internetworking devices from the retrieved case can support
THEN	increase the number of the internetworking devices
IF	more than one internetworking devices are required to support incoming cables AND there are outgoing cables
THEN	select a multiport transceiver AND connect the internetworking devices to the multiport transceiver AND connect the incoming cables to the internetworking devices AND connect the outgoing cables to the multiport transceiver

(Figure 3) Examples of Adaptation Rules

At the next lower level below a transceiver, for example, are more detailed categories of a transceiver: a Fiber Optic Transceiver that has one ST port and one AUI port, an Ethernet Transceiver that has one or more BNC ports and one AUI port, and an Ether Twist Transceiver that has several RJ-45 ports. Below the Ethernet Transceiver, there are two different types of devices: a Singleport Transceiver that has one AUI port and one BNC port and a Multiport Transceiver that has one AUI port and 8 BNC ports. Figure 4 shows part of the tree implemented in CIDA.

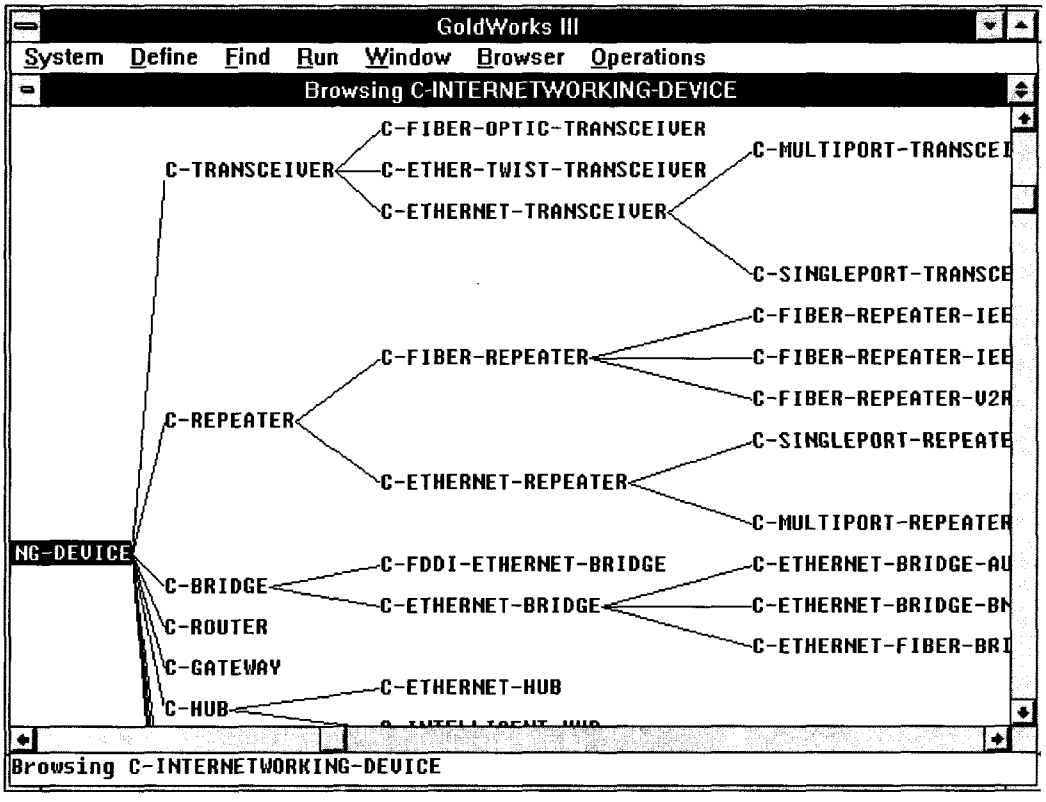
2. Architecture

2.1. Problem Solver

The Problem Solver consists of two components: the Constraint Poster and the Case-Based Designer.

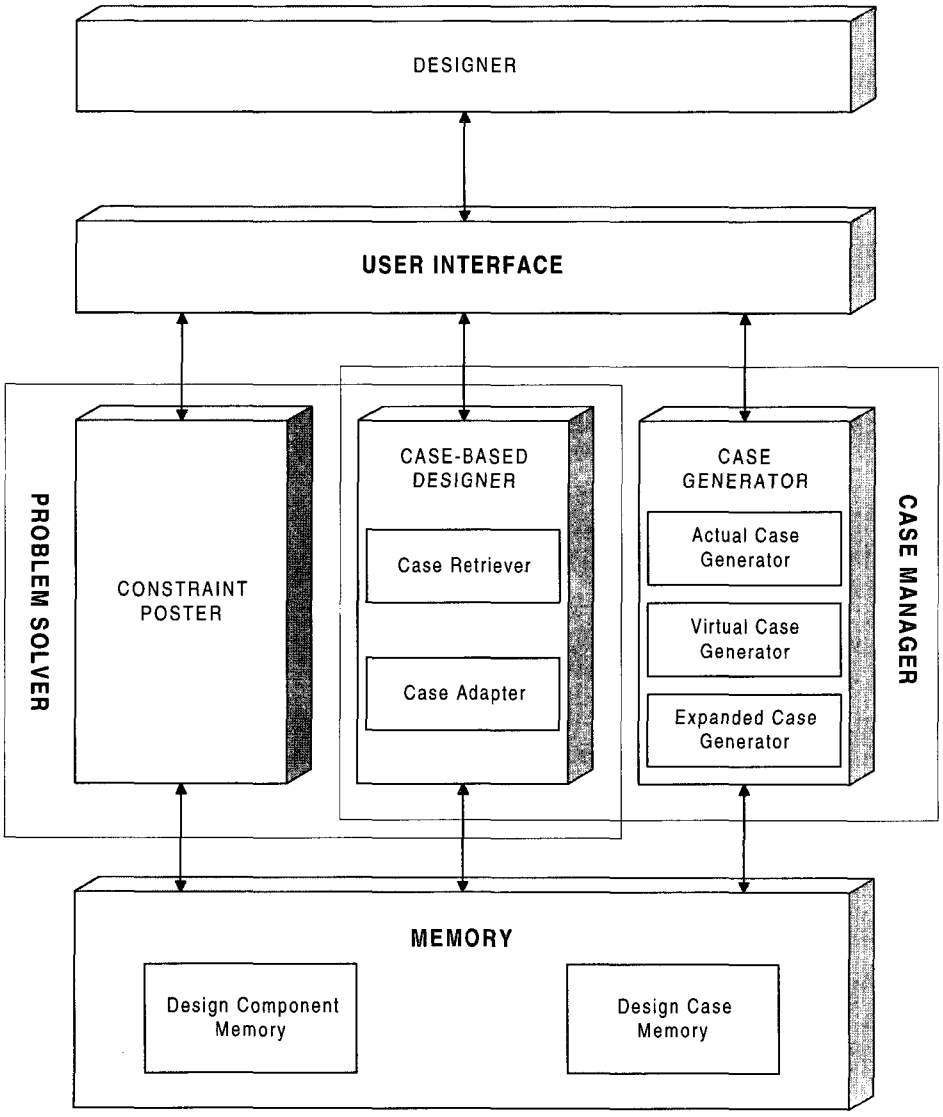
2.1.1. Constraint Poster

The constraint poster makes incremental commitments on design decisions as a function of previous decisions. Constraint posting (Stefik, 1980) consists of three operations on constraints:



(Figure 4) The Design Component Tree in CIDA

constraint formulation, constraint propagation, and constraint satisfaction. In CIDA, constraint formulation is implemented in the form of 'when-modified' daemons. When information about the current internetworking design problem is known to the system, the daemons are triggered to formulate constraints about the requirements. Some of the constraints formulated in this phase are propagated to other parts of the system to make later problem solving feasible. Constraint satisfaction is implemented by rule based reasoning. In order to find a device that satisfies



(Figure 5) CIDA System Architecture

the constrained description, a search is performed using rule based forward reasoning.

2.1.2. Case-Based Designer

The Case-Based Designer consists of the following components: a Case Retriever and a Case Adapter.

2.1.2.1. Case Retriever

CBR researchers have used a number of methods to select a case relevant to a new situation. In CBR systems (e.g., Kolodner, 1984; Ashley & Rissland, 1988), retrieval typically proceeds by indexing the features of the situation description into a memory organized as a discrimination net or decision tree. On the other hand, Kibler and Aha (1987) compare the featural description of the new case to that of every known case, and assign a score by counting the number of matching features. King et al. (1988) advanced Kibler and Aha's method by giving weights to features according to expert-assigned importance. Several other case-based reasoners implement versions of King et al's method, which is called nearest-neighbor matching (see Kolodner, 1993), including MEDIATOR (Kolodner & Simpson, 1989), PERSUADER (Sycara, 1988) and CBR shells on the market (e.g., Cognitive Systems, 1992). CIDA adopts King et al.'s method of giving weights to features according to expert-assigned importance.

2.1.2.2. Case Adapter

The case adapter is the module that adapts the retrieved design using adaptation rules. There are two types of constraints in CIDA: hard constraints that are either satisfied or violated, and soft constraints that can be satisfied to a certain degree (Sriram & Maher, 1986). While it may be difficult to relax a hard constraint, a soft constraint usually can be relaxed easily. The constraints that an internetworking device should satisfy can be classified into these two groups. MAC layer standards, physical layer standards, OSI functions, and port types that should be supported are hard constraints. On the other hand, the number of ports a device should have is a soft constraint. In internetworking design in CIDA, only the soft constraints can be adapted.

2.2. Case Manager

The Case Manager consists of the following components: a Case Generator and a Case-Based Designer.

2.2.1. Case Generator

The Case Generator consists of the following components: the Actual Case Generator, the Virtual Case Generator, and the Expanded Case Generator.

2.2.1.1. Actual Case Generator

The Actual Case Generator generates a case after an internetwork has been designed through problem solving. The generated case then is saved for future use.

2.2.1.2. Virtual Case Generator

The virtual case generator creates virtual cases whenever a new internetworking device is introduced into the system. Once a new device is introduced into the design component tree, CIDA

retrieves the design cases, and inserts the new device or the new device in combination with the other existing devices, into the solution parts of the retrieved cases to see whether the new device is compatible with the network situation of the cases. If the new device works fine in the network situation or a retrieved case, CIDA generates virtual cases, by replacing the solution part of the retrieved cases with the new device, or the new device with other existing devices, retaining the network description part of the retrieved case.

2.2.1.3. Expanded Case Generator

The expanded case generator makes expanded cases when a device chosen as part of a solution of a internetworking design has functionality beyond that . While a virtual case is generated when a new device is introduced to the system, an expanded case is generated when a design problem is solved. Once a design task is done, the system checks to see if any device has any extra functionality comparing to the current network situation. If the device has any functionality that is not used to solve the current problem, the problem description part of the current case is modified. The modified version of the current design problem and the solution forms an expanded case.

3. Ablation Experiment

An ablation experiment evaluates the contribution of a component to the performance of the overall system by removing the component (Cohen & Howe, 1988). I have performed an ablation experiment to determine how different components in CIDA affect its efficiency and competence.

In this experiment, the most interesting issue to look at is whether the addition of Virtual Case and Expanded Case generating capability to an existing CBR system could result in any efficiency comparing to just CBR system alone or CBR system and Constraint Satisfaction Problem Solving techniques combination.

To establish this, I first disabled all the functions but the CBR function and ran two internetworking design tasks: task #1 with smaller number of network nodes that requires rather simple internetworking devices and task #2 with a larger number of network nodes that requires internetworking devices with recent technologies. Under this circumstance, the system retrieved the most similar case solely based on the features of the case and the similarity calculation function. Once the most similar case was retrieved, the case was adapted to generate the solution for the current

〈Table 1〉 Number of Designs Generated under the Ablation Experiment

Tasks	CBR only	CBR + CS	CBR + VC + EC
Task #1	None	54	12
Task #2	1	260	17

task. This system could not solve task #1 because it was not able to retrieve a similar case from the case library. It is because the features of the current task is so different that none of the case in the case library could be matched with. For task #2, it could retrieve one case and adapted it as a solution.

Next, I added Constraint Satisfaction problem solving capability to the CBR system. When CBR system failed to solve the task, Constraint Satisfaction problem solving part is triggered to solve the problem; to find appropriate internetworking devices that satisfy the constraints that are calculated based on the network description. CIDA implements the Constraint Satisfaction capability only in a local sense. Once it finds design components for local places, such as Wiring Closet, Building Entrance, and Hub Place, it combines them to form a global solution without checking any further constraints. When CIDA calculates the constraints from the network description, the constraints that should be met in every local places are propagated so that there won't be any conflicts among constrains from different local places. Protocols and LAN standards used in designing an internetwork are the examples. Under such system, CIDA generates 54 solutions for task #1, and 260 solutions for task #2 (Table 1).

Finally, Virtual Case and Expanded Case generating function were added to the CBR system. Another words, the system has Virtual Cases that incorporates newly introduced internetworking devices as their solutions, Expanded Cases that have extra capacities of the internetworking devices selected as the solution in the form of network description, and the data dependency links between the outdated and the recently developed internetworking devices. With these components added, the CIDA generated 12 designs for task #1 and 17 for task #2 (Table 1). The reason CIDA with all components generated much smaller number of solutions than the CBR plus Constraint Satisfaction system did is as follows. It is observed that human expert solved the task around geographical locations and then combined them to generate the solution for the whole area (Joh, 1996). While combining local solutions into a global solution, the expert considered several constraints the global solution should meet: consistencies among internetworking devices, consistencies between the data processing capacities of devices, scaling constraints and growth related constraints. These features in CIDA made the solutions optimal in a more global sense while the Constraint Satisfaction function in CIDA made the solutions optimal only in a local sense.

Conclusion

In this paper we discussed the knowledge representation and the architecture of the CBR system that has the capability of taking advantage of future knowledge that has not been used before. A prototype system is constructed in the domain of internetworking design and the performance of the system is compared with that of the constraint satisfaction problem solver and the state-of-the-art CBR technology. It shows that with a proper

architecture that is able to use future knowledge as well as previous cases the performance of a CBR system could be enhanced considerably.

Reference

[1] Ashley, K. D. and Rissland, E. L., (1988). "Waiting on Weighting: A Symbolic Least Commitment Approach," In *Proceedings of AAAI-88*, 239-244.

[2] Buchanan, B. G. and Shortliffe, E. H. (1984). *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley Publishing Co., Menro Park, CA.

[3] Cognitive Systems (1992). *ReMind Developer's Reference Manual*, Boston.

[4] Cohen, P. and Howe, A. (1988). "How Evaluation Guides Research," *AI Magazine*, 9(4), 35-43.

[5] Feinstein, J. L., Siems, F., Popolizio, J., Bailey, D. and Wang, A. (1988). XTEL: An Expert System for Designing Theaterwide Telecommunications Architectures," in J. Liebowitz (Ed.) *Expert System Applications to Telecommunications*, New York, John Wiley & Sons, 161-199.

[6] Hamscher, W. C. (1991). "Modeling Digital Circuits for Troubleshooting," *Artificial Intelligence*, 51, 223-271.

[7] Joh, D. Y. (1996). *"Knowledge Revision in Case-Based Reasoning: A Cognitive Model of the Telecommunications Internetwork Design Process,"* Ph.D. Dissertation, Joseph M. Katz Graduate School of Business, University of Pittsburgh.

[8] Kan, K. (1986). "Expert Systems in Telecommunications Network Planning and Design," In D. Sriram and R. Adey (Eds.) *Application of Artificial Intelligence in Engineering Problems, Volume II*, Spring-Verlag, 1161-1164.

[9] Kibler, D. and Aha, D. W. (1987). "Learning Representative Exemplars of Concepts: An Initial Case Study," *In Proceedings of the Fourth International Workshop on Machine Learning*, 24-30.

[10] King, J. A., Klein, G. A., Whitaker, L. and Wiggins, S. (1988). "SURVER III: An Applicatio of Case-Based Reasoning," In *Proceedings of the Fourth Annual Aerospace Applications of AI Conference*, Dayton, OH.

[11] Kolodner, J. (1993). *Case-Based Reasoning*, Morgan Kaufmann Publishers, Inc., San Mateo, CA.

[12] Kolodner, J. and Simpson, R. L. (1989). "The MEDIATOR: Analysis of an Early Case-Based Problem Solver," *Cognitive Science*, 13(4), 507-549.

[13] Kolodner, J. and Simpson, R. L. (1989). "The MEDIATOR: Analysis of an Early Case-Based Problem Solver," *Cognitive Science*, 13(4), 507-549.

[14] Pierre, S. and Hoang, H. (1990). "An Artificial Intelligence Approach to Improving Computer Communications Network Topologies," *Journal of the Operational Research Society*, 41(5), 405-418.

[15] Sriram, D. and Maher, M. L. (1986). "The Representation and Use of Constraints in Structural Design," In D. Sriram and R. Adey

(Eds.) *Applications of Artificial Intelligence in Engineering Problems: Proceedings of the 1st International Conference*, Southampton University, U.K., 355-368.

[16] Stefik, M. (1981). "Planning with Constraints (MOLGEN: Part 1)," *Artificial Intelligence*, 16, 111-140.

[17] Sycara, K. P. (1988). "Using Case-Based Reasoning for Plan Adaptation and Repair," In *Proceedings: Workshop on Case-Based Reasoning (DARPA), Clearwater, Florida*, San Mateo, CA: Morgan Kaufmann, 425-434