THE FUNDAMENTALS OF CASE-BASED REASONING: APPLICATION TO A BUILDING DEFECT PROBLEM

Salha binti Abdullah
Department of System Science and Management
Faculty of Information Science and Technology
National University of Malaysia
43600 Bangi, Selangor D.E.
Tel: 03-8292922
Fax: 03-8256732
email: salha@pkrisc.cc.ukm.my

ABSTRACT

Traditional expert systems model human problem solving as a deductive process. They construct a solution by applying general rules to the description of a problem. Recently, however, it has become apparent that human experts rely heavily on memory of past cases when solving problems. This is the focus of systems using case-based reasoning (CBR) approach which is rapidly emerging as a powerful expert system approach that can overcome some weaknesses of traditional systems. The aim of the article is to demonstrate to the construction industry, how CBR can be applied in its sector. It presents an overview of CBR, depict characteristics that make a problem domain suitable for CBR and describe briefly the implementation of a CBR system and its findings.

Keywords: ***CBR, Case-based reasoning, expert system, diagnosing building defects***.

1.0 INTRODUCTION

Traditionally, reasoning in artificial intelligence (AI) had been using model-based reasoning (MBR) for generating solutions [1]. MBR systems are based on some general explicit model of knowledge of the domain in which they operate. It depicts many types of knowledge concerning concepts in the application domain of the expert systems. During any problem solving, an expert system that uses MBR reasoners, will try to apply the general knowledge of the domain model to the current situation. There are several methods of representing the knowledge of MBR systems. Examples of such knowledge representations are production rules, frames, logic, semantic networks and object oriented method. These representations are categorised under MBR because their methods of reasoning use general domain models to generate specific instances of concepts and their interrelationships [2].

Recently, researchers have begun utilising an alternative reasoning paradigm and computational problem solving called case-based reasoning (CBR) [1, 3, 6]. CBR is a relatively new approach compared to MBR. CBR emphasises on the use of a set of examples of prior events (case histories) in its problem solving. The focal part of its knowledge is contained in its instantiated cases instead of an explicit and complete domain model.

2.0 THE THEORY OF DYNAMIC MEMORY

The development of CBR was based on Schank's theory of dynamic memory [4]. The theory of dynamic memory deals with the relationship between knowledge structures in memory and the implications of the structuring for the enhancement of matching processes. These knowledge structures are called memory organisation packets (MOPs). MOPs are used to represent knowledge about classes of events, especially complex events, which illustrates the formation of episodic long-term memory. The episodic memory provides the ability to generalise about episodes or events from a particular experience.

According to this theory, understanding of a new episode (case), includes finding the best knowledge in memory that can be used to make predictions from it. Finding this knowledge is equivalent to integrating the new episode with what is already on the memory, i.e., prior knowledge. As reasoning is going on, i.e., when new experiences demand it, for example, on the basis of mismatches or failures, memory is constantly being explored, adjusted and updated. Thus, the episode will get better integrated and better knowledge can be used in making predictions about the new episode being derived, i.e., it can learn.

There are consequences of this view of dynamic memory from the point of view of AI such as expert system. Often, programmers tried to build complete and error-free programs. However, observation on the past examples on expert systems indicated that they are not fully robust [5]. First, they do not have memories [6] and second, their algorithms did not include any attempt to find the most relevant related story in their experiences to aid in processing [3]. Therefore, expert systems should have dynamic memory structures that change each time the program is called into use. There is a difference between what people say they know and how what they said is grounded in their experience. People are modified by their experience and so the program should also be modified in the same way.

3.0 AN OVERVIEW OF CBR

In brief, CBR "remembers" previous problems and either adapts their solutions or uses their outcomes to evaluate new cases [7, 8, 21]. Past experiences or previous solutions, as represented by "cases" are stored in a knowledge based library or case base. The cases are to be recalled by the user when a new case with similar parameters are encountered. The solutions of the old case are then applied to the new case. Successful solutions are tagged to the new case and are stored in the case base. When a search fails to locate a similar case, the search itself becomes the basis for a new case, in effect "learning from experience". The result is a continuously expanding yet increasingly refined knowledge base. Fig. 1 illustrates a simple CBR structure.

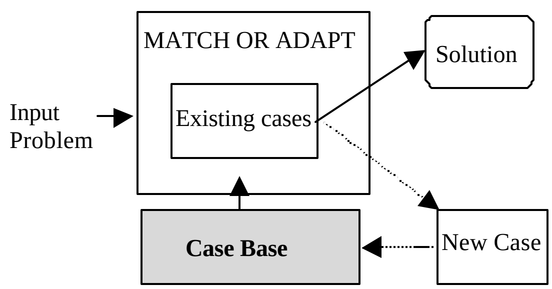

Fig. 1: A simple structure of CBR

There are two types of case-based reasoners: problem solvers and interpretive reasoners [1, 7].

- **Problem solver**

Problem solver CBR is used for problem solving task such as designing or planning. It modifies and adapts past solutions to new problems. This may lead to three different results:

1. a full solution that is almost accurate or;
2. a previous case may address only part of the new problem and require further reasoning to solve the rest or;
3. the previous case may suggest an abstract solution that must be further refined.

It may also warn of any plausible errors or failures. This method can be used as a "standalone" problem solver. However, it can also be used interactively with users who actually solve the problem, applied in 2. and 3. above. It shifts much of the burden of adapting old solutions from the reasoner to the user.

- **Interpretive reasoner**

This type is usually found in strategic planning and legal reasoning. It uses cases to evaluate or justify new solutions. Previous cases are used as argument for categorising new solutions. It can evaluate solutions when no clear-cut model-based techniques are available. It can also clarify situations whose boundaries are ambiguous. In essence, in interpretive CBR, the reasoner is attempting to better understand a situation.

What the domain experts and a CBR system have in common is a ready access to a variety of cases. Both the experts and the system have an organised memory of cases that represent encounters with problem situations that commonly occur, as well as, those that are rare and unusual, requiring unique problem solutions. It is this knowledge that is brought to apply to new situations.

Consequently, the first requirement is related to deciding what to represent about a case in the memory. It is vital, since it is the content of the retrieved cases that provides the knowledge that the system needs to solve new problem. A case is a unit of information such as instruction, graphics, diagrams, code, multimedia sequences, hypertext links to on-line documentation or even an automatic phone dialler to call a support person [8, 15].

A case should contain two kinds of information [8]. First there is information that helps to index the case so it can be found. Second is the information that will be useful to end user once the case is located.. It can represent both a physical object and a process that are stored in a case base. A case is an example that has occurred in the real world, i.e. a problem that occurred and has been solved and justified by problem solving mechanism such as human expert or expert system. Therefore, a case should contain:

- the problem description,
- the solution for the described problem
- the justification of the solution

CBR system uses two types of cases: prototype cases and episodic cases [10, 23].

- **Prototype cases**

Prototype cases express generalisation over a set of related cases that have been carefully analysed by experts. Most new input cases will be seen as variations of the prototypes.

- **Episodic cases**

They comprise formation created during a particular problem solving experience. These cases are similar to one or more prototype cases but need to be treated differently because of some particular characteristics.

The number of cases required is very subjective. In a domain such as design, it is more important to have a few high quality cases rather than many lower quality cases [2, 22]. Nevertheless, a case base whose cases cover more of a domain is better than one that cover less and one whose cases cover successes and failures is more helpful than one that covers only successes.

The structure of cases that are to be stored in the case base has an influence on the knowledge acquisition process. If the structure is simple, for example, flat attributes or list of values, then case acquisition is almost trivial. On the contrary, if the structure is more complex, for example, if cases contain causal relations, then case acquisition is more demanding. It may require application of knowledge acquisition and machine learning techniques similar to those used in rule-based system.

4.0 CBR CYCLE

The basic problem solving cycle which characterises the CBR model is described in Fig. 2 [9, 13].

STEP 1 **Acquire cases of the domain**. Store them in the case base (case memory or library of cases).

STEP 2 **Carry out a preliminary analysis of the current (input) situation**. This includes identifying the appropriate features.

STEP 3 **Retrieve "best" cases from case base**. They should share similar descriptions or features to the current situation and have the potential to make relevant predictions about the current situation. The relevant cases had to be classified and organised so that features of input situations can be used to find them. These features are called indexes.

STEP 4 **From the collection of cases in STEP 3, select the most promising case or cases to the current situation**. The indexes will retrieve a set of potentially relevant old cases. Matching the cases against the input is carried out to reject cases that are too different from the input situation and to determine which of the remaining cases is most similar to the input situation.

STEP 5 **Analyse the differences between the case and the current situation**. The case should be the best match that had been determined earlier.

STEP 6 **Use the differences to modify the solution in the case**. This process is called adaptation. The amount of adaptation to be done rely on the nature of the differences between the input and the retrieved case. Sometimes, what was done in a retrieved case will be the right thing to do again but, other times, some changes need to be performed.

STEP 7 **Apply the modified solution to the current solutions**. This can be done by using some adaptation techniques.

STEP 8 **Store and update by storing the modified solution as a new case into case base**. If the adaptation has been successful, the completed case can be incorporated into the case base. Thus, if the same problem occurs again, it can be directly solved by retrieving this case and applying its stored solution.

STEP 9 **Repair the wrong solution**. If the adaptation process has not been successful, i.e. when the selected case mismatches the input to such as extent that the case base reasoner will be unable to make the needed fixes, it can be stored as a negative example to warn the problem solver not to go to this direction if the same problem is encountered. Additionally, if the system (or human expert) can find out the cause of the failure, it might be able to correct the wrong solution.

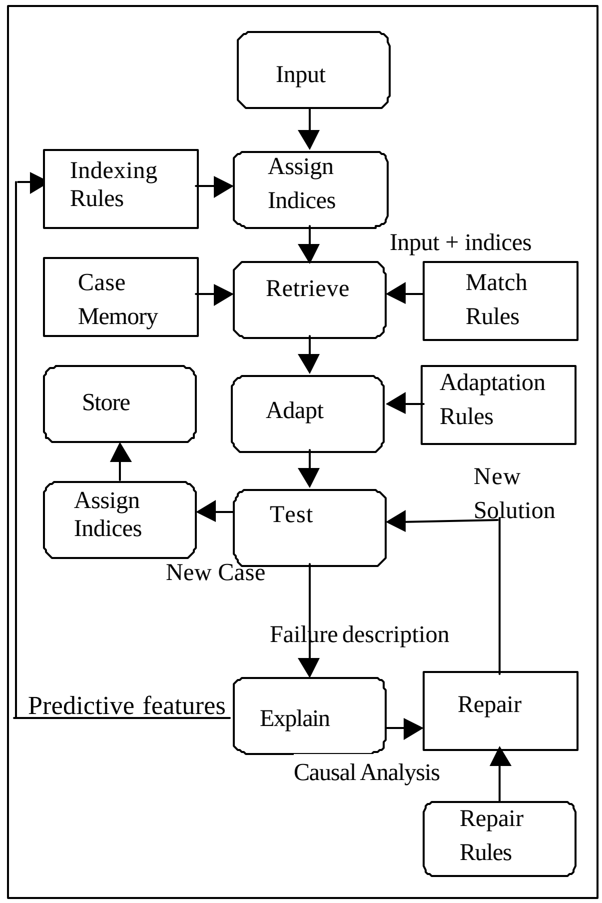

Fig. 2: CBR flow chart (adapted from [3])

5.0 THE DOMAIN

This aim of this article is to illustrate the CBR approach to the development of an expert system by presenting the development of such a system to aid building surveyors to diagnose building defects. Commonly, the building defect comes up for investigation when a symptom is reported. It may be a damp patch, a crack or a roof tile that fell off. The location of the visible symptom is usually known, as are the materials involved. Thus, the starting point will be the symptom that triggers the investigation, the location of the symptom and the material involved. This information with another related details will be used to diagnose the cause of the defect and to give advice on how to cure the defects once they have been identified.

5.1 The Relevance of CBR to The Domain

Fault diagnosis is a classic expert system domain and thus the diagnosis of building defects would seem ideal domain for an ES. However, the diagnosis of building defects is complicated by the following factors [23]:

- numerous materials are used in numerous combinations within a building,
- similar symptoms can be caused by different problems,
- there is limited causal information for why materials failed, and
- the cause of a defect may not be discovered until the destructive tests are performed.

All these factors suggest that implementing a diagnostic ES for this domain would be non-trivial. However, the description of CBR is a good match to the thought process of diagnosing building defects. Furthermore, researches [9, 16] have identified several characteristics that are suitable for certain CBR task orientations and the diagnosis of building defects seem to satisfy these criteria.

- **Experts know what they mean by a case**.

A case for this domain would be a prior knowledge and experience of the diagnosis of a building defect.

- **Domain experts deduce inferences by comparative thinking across related cases, i.e., comparing a current problem to prior cases**.

The domain experts reach their decisions based on sound technical knowledge. This knowledge comes from lessons or experiences of construction defects in the past that are often relevant to present situations.

- **Experts adapt cases to solve new problems**.

Historical records of defects allow experts to use the results and apply them wherever possible to current situations.

- **Cases are available in a data base, for example, published sources, in experts' memories, or can be recorded as new solutions are attempted**.

In the diagnosis of building defects, it appears, there is a lot of information relating to past records. As mentioned above, these sources of information can be obtained from completed maintenance feedback reports. Besides this, many building periodicals devote regular features to the subject and several publications have appeared dealing exclusively with building defects.

- **There are procedures in the domain to assign a result to case, explain it and consider it a success or failure**.

Surveys or investigations of the buildings, on-site or off-site testing and commissioning of defect elements and, feedback from tenants or complainers are ways the defect can be tested and evaluated. Success of the actions may be appraised by diminishing complaints, depleting frequency of occurrences and total elimination of recurrence of symptoms.

- **Cases can be generalised to some extent. Features that make them relevant can be abstracted**.

Defects of buildings can be generalised into various groups, for example, dampness and structural defects. The specific features that turn the group identity can, then, be extracted to make them applicable to a particular problem and arrive at a solution.

- **Comparison and adaptation can be achieved with some level of effectiveness**.

Case comparison and adaptation can be done through a system discussed in section 6.0.

- **Cases are acceptable, i.e. stable, for relatively long time periods**.

Throughout the last decade, the methods and the type of materials for constructing buildings remains conservative. The data for building defects occurring in most buildings are still derived from similar failures, consistent along this period. Hence, the domain knowledge is predicted to be constant and will hold true for a long time. In this respect, the system would be a reliable one.

- **The domain may, or may not, have a strong model, i.e. its concepts are fuzzy with many exceptions that are not easily summarised or understood**.

Defects are often caused by combinations of factors. Particular circumstances may have produced the exceptional defect but the same symptoms can be caused by different faults, which can cause confusion. Sometimes, even the most obvious diagnosis of a defect may still give a wrong solution or action. Likewise, many failures that occur result from the shortcomings of new materials, components and techniques that are not currently fully understood.

- **Cases, which contain past knowledge and experience, are used to train professionals within the domain**.

In higher institutions, students are taught through text and reference books. The information compiled from these materials, i.e. distinctive symptoms, causes and solutions of previous cases constitute the relevant content. Building surveyors will make reference to hundreds of past records, which may be possessed by the organisation in which they work. This would assist them in their training and evaluation of new jobs.

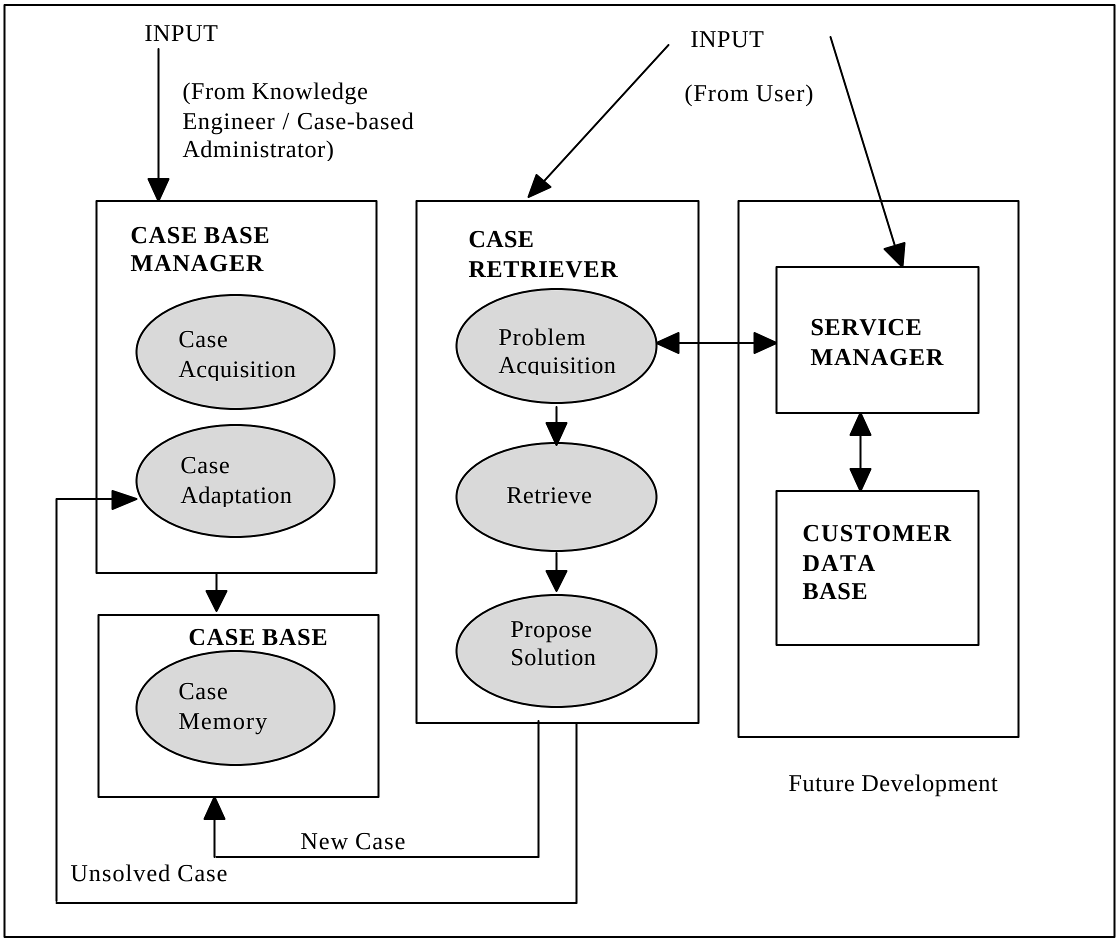

Fig. 3: Architecture of PAKAR

6.0 PAKAR'S ARCHITECTURE

In this article, a prototypical case-based expert system, called PAKAR [23], was built to assist building surveyors in the housing association or local authority to diagnose building defects. The tool chosen for the development of the prototype was *CBR Express* by Inference Corporation [20].

The prototype consists of five components, as illustrated in Fig. 3. They are Case Base, Case Base Manager, Case Retriever, Service Manager and Customer Data Base.

- The **Case Base** acts as the knowledge base or library of all the former cases that are available for comparison.
- The **Case Base Manager** acts as a case acquisition unit of resolved reports. This component plays an important part to the success of PAKAR. The designing of PAKAR's case base done by the case base administrator, i.e. building, extending, revising and debugging cases in the Case Base, is performed in the Case Base Manager.
- The **Case Retriever** uses the current problem description as an index to retrieve cases that are closely related from the Case Base and display the actions of the retrieved cases. Even though, the layout does not entirely duplicate the CBR problem solving cycle as illustrated in Fig. 2, part of the basic cycle, i.e. "input a problem and find a relevant old solution,", is being utilised in the Case Retriever.
- The **Service Manager** and **Customer Data Base** have not been exploited by the case base administrator, but they are included in the architecture for future work.

These components are described in more detail in the following sections.

6.1 Knowledge Elicitation

In practice, a substantial case base on building defects may be available from a data base of previous cases. This data base held by an individual organisation, would contain detailed historical records of defects. However, in this article, the knowledge elicitation was achieved through published construction materials [17-19]. The published materials contain articles that deal with specific aspects of building defects such as dampness and cracking. Analyses of building failures from the published materials are not catalogued as occurrences based on specific buildings that occurred at certain date and time. They, however, address contemporaneous records of what caused the defects and the remedial actions taken, based on wider and general scale of incidents.

6.2 Case Base

One of the key factors influencing the applicability of CBR to a domain is the existence of a historical case base. The two procedures undertaken to establish PAKAR's Case Base are knowledge acquisition and knowledge organisation.

6.2.1 Knowledge Acquisition

The first requirement to build PAKAR's Case Base would be to decide what to represent about a case and how it would look like in a case base. These features are important, in order to find similarities between each case. In theory, the structure of the diagnosis of building defects cases that are to be stored in the Case Base is relatively simple. The structure of the cases should, in some way, reflect the structure of the simple defect feedback report [18]. Each report would represent one case. Hence, the case acquisition is a straight forward data transferring exercise. However, since, in this prototype, the case base administrator did not acquire the cases directly from any organisation data base, but from published materials, the knowledge acquisition was approached in a reversed manner.

Each reference in the literature may contain more than one case at a time. This contradicts with the structure of information that contains in a typical defect report, where normally one defect is assigned to a report. In order to carry out the knowledge acquisition, a simple paper knowledge base, with its content, was produced (Fig. 4). These are the associated attributes that will contain in a case.

CASE NO.	
ELEMENT/CODE	
TITLE	
SYMPTOMS	
CAUSES	
REMEDIES	
QUESTIONS	

Fig. 4: Structure of Paper Model.

The right information from the published materials has to be extracted and mapped into the paper model, which depicts the Case Base structure. Each paper model would represent a case that can be entered into PAKAR's Case Base. The paper model reflects in some way a typical defect feedback report and also the interface of *CBR Express*. One could argue that since the paper model reflects the interface of *CBR Express*, one could almost escape the need for a paper model and therefore, minimise the time involved. The data from literature may be entered straight into the Case Base. Although special knowledge engineering is not required, there is still a demand for planning and structuring. Having in mind that the amount of information to be handled is large, the Case Base creation need to be approached as a structured and well-organised task, with cases being added in accordance with an overall plan.

The data within the published materials are usually compiled together. A few descriptions of the symptoms are combined that seem to appear as one case reference, although they actually can be arranged into several other cases. To make the matter more complicated, these symptoms can be generated by a number of different causes. An example is explained below.

"*The opening light does not sit against the frame on all edges. This can lead to draughts and rain penetration, either of which may be related to certain wind conditions. In severe cases, the light or frame is so distorted that the two parts do not fit properly. The light can then not be properly shut and secured*."

This statement can be organised into several cases.

1. Opening light does not sit against the frame on all edges that subject to certain wind condition and lead to draught.
2. Opening light does not sit against the frame on all edges that subject to certain wind condition and lead to rain penetration.
3. Light or frame is so distorted that the two parts do not fit properly and cannot be properly shut, may still lead to draught.
4. Light or frame is so distorted that the two parts do not fit properly and cannot be properly shut, may still lead to rain penetration.

Each of these cases, then can be diagnosed into several causes.

1. Weather stripping defective.
2. Weather stripping non-existent.
3. Moisture movement.
4. Paint build up.
5. Paint defective.

Therefore, one reference in the literature review can create confusion with regard to which symptom and diagnosis should be included in each case. They cannot be combined since they would have different remedies. Each of these possibilities should be written as a distinct case. Hence, the need for the paper model becomes prominent.

6.2.2 Knowledge Organisation

By adopting the procedure, there exists a standard for generating the problem description. Consequently, the process of retrieving cases may be made easier. The reason is that each case is reorganised and stored into the Case Base. All the cases in the Case Base are evaluated by using the elements or components of a building.

Prior to the decision to select elements of building as key terms, the case base administrator had considered other options to be used. The main one is considering the types of defects as the main context. However, there appeared to be several disadvantages. Most importantly, there seems to be uncertainty on the number of types of defects. Their list may be indefinite compared to elements of building. They could run from the most common type such as dampness and cracking, to the least such as discolouration and degradation.

The defect can be viewed by many under different perspectives. Misunderstanding between defects themselves and their causes may also arise. For instance, "Is rain penetration a defect?" since a building that allows rain to penetrate may be a defective building; or "Can rain penetration be considered as the source of the building defects?'. Where at this instance, many building defects such as timber decays are caused by rain penetration.

The form of presentation of many existing cases takes elements as the context. To change the order, one has to reorganise every case and reclassify them under the defects family, which as aforementioned may not be easily identified. The Maintenance Manuals for Buildings [19] itself, operates by grouping the defects according to elements and components. The overlapping interpretation of defects and what cause the defects can be best avoided by selecting building elements as key terms. The division between one element and another can be distinguished

6.3 Case Base Manager

The Case Base Manager allows the case base administrator to perform case acquisition by storing new, dispatched and resolved case, compiled in the paper model. It also permits the case base administrator to execute any case adaptation when a search operator, who is the end user, cannot resolve a particular problem. The Case Base Manager can be accessed through Maintenance Mode. The Case Panel of PAKAR provides four different fields and a graphic browser, to store a case.

- *Title* field enables a developer to record the name of the case that is used only for recognising a case.
- *Description* field contains textual description of a problem to be utilised for case matching.
- *Question* field contains a set of questions to channel the search process starting from the search operator's input.
- *Action* field contains the action taken and any additional instruction or information.
- *Graphic browser* will open an external *ToolBook* when a search operator requests a browse of this case from the Search Panel.

There are two methods of defining new cases.

1. Define the questions and actions before building a case.
2. Proceed straight to Case Panel and enter the title and description of a case. Then, build the questions and actions as needed for that particular case.

The method that was adopted by the case base administrator is the second one. Even though, the first method is more simple than the second one [20], the case base administrator believes that the first method may create confusion. This is because the number of questions and action will keep growing and there may be some difficulties when it comes to choose the questions and action, to define a case.

In the second method, PAKAR allows the case base administrator making a brief excursion to the Question Panel and Action Panel as needed, then returning to the Case Panel and automatically inserting the newly created questions or action into the case definition. The case base administrator believes that this method is more systematic compared to the other. The next few sections, i.e. section 6.3.1 to 6.3.5, will address the strategies of designing each particular field.

6.3.1 Title

The features that uniquely describe each case become the case titles and are used to index the cases. The problem, the cause and the proposed solution were placed in the title. The cryptogram of the title was approached by placing brief information in the following order:

Secondary Code/Symptom/Material/Location/Diagnosis

By placing symptom, material, location and diagnosis, the case base administrator seeks to differentiate one case from another. Some of the defects have very similar descriptions. The only factor that differentiates them may be either material or location. Furthermore, they are often the key points of any investigation of building defect. The solution is excluded because in this domain, one case may have more than one action. It is not feasible to identify all the actions in the title (Fig. 5).

Title:
33A: Vertical crack/masonry/away from corner/swell of clay.

Fig. 5: Example of a title

Description:
External wall. Masonry. Any position, place. Close, near to opening. Within 300mm. Show, appear on internal surface, inside of external wall. Isolated damp surround window. Develop, spread, quickly, fast. Apparent, clear, spread in wet, rainy weather. (END)

Fig. 6: Example of a description

6.3.2 Description

The case description is the crucial text that form the essence for the initial search from the Search Panel. The matching algorithm used by PAKAR, will score the search operator's search description. It will base on how much information described by the search operator is included within the case description. The description contains a normal report of symptoms that normally trigger any defect investigation (Fig. 6).

The descriptions for all the cases were reviewed after finishing entering them into the Case Base. Synonyms were entered to cover the variation that the search operator may present or describe of particular situation. Since each case's description would constitute 20% of the total score of each case, this could enhance the initial searching process.

6.3.3 Questions

The initial search executed using the case description may be insufficient to arrive at the solution. To supplement this, PAKAR poses questions to the search operator. Basically, a question is an interrogative statement that helps define and differentiate cases. The list of questions and answers is another way of indexing the cases. It is vital because by answering the questions, the searching process will continue. This will rapidly narrow down the possibilities. However, it does need good planning and an efficient enquiry strategy. The way to do this is by clustering the questions. This method will orderly separate and define the population of questions into smaller and smaller group, i.e. distinguish a group of related cases from one another. The questions were divided into three groups (Fig. 7).

- **Context question**

Basically, a context question is the question that distinguishes group of related cases from one another. It is used to identify the element of building where the defect is occurring. This question will always be the first question displayed in the Search Panel and therefore prompt the search operator to answer it first. It will assist steering the search process to the right cluster of cases. An example of the context question can be viewed in Fig. 8.

- **Sub-context questions**

The function of the sub-context question is very similar to the context question. They should be displayed after the context question. There are three sub-context questions in the Case Base assigned to each element. These questions are related to identification of defect, i.e., symptoms, location of defect and material of the element of building where the defects arise.

Example of these sub-context questions is as the followings:

Question : What type of material has been used in the *wall* construction??(-)

Answer : Masonry
Solid masonry
Clay brickwork
Unfilled cavity wall
Filled cavity wall
Others

These questions are allocated to each of the elements defined in the context questions (Fig. 8). They are designed as such, to avoid the answer for one element being mixed up with another.

- **Confirmation questions**

Confirmation questions separate individual cases from a small group of similar cases. These questions are normally be answered after the context and sub-context questions have been answered. In this manner, the searching process will be more accurate since the context and sub-context questions had already narrowed down the possibilities and pointed out to the most appropriate area.

Questions for each case are set up during knowledge acquisition, after some planning has been carried out. Often the questions in the Case Base reflect and confirm the information entered in the description field. This is to prevent the description being left out, especially if it is used by a novice search operator or non-domain expert. All the

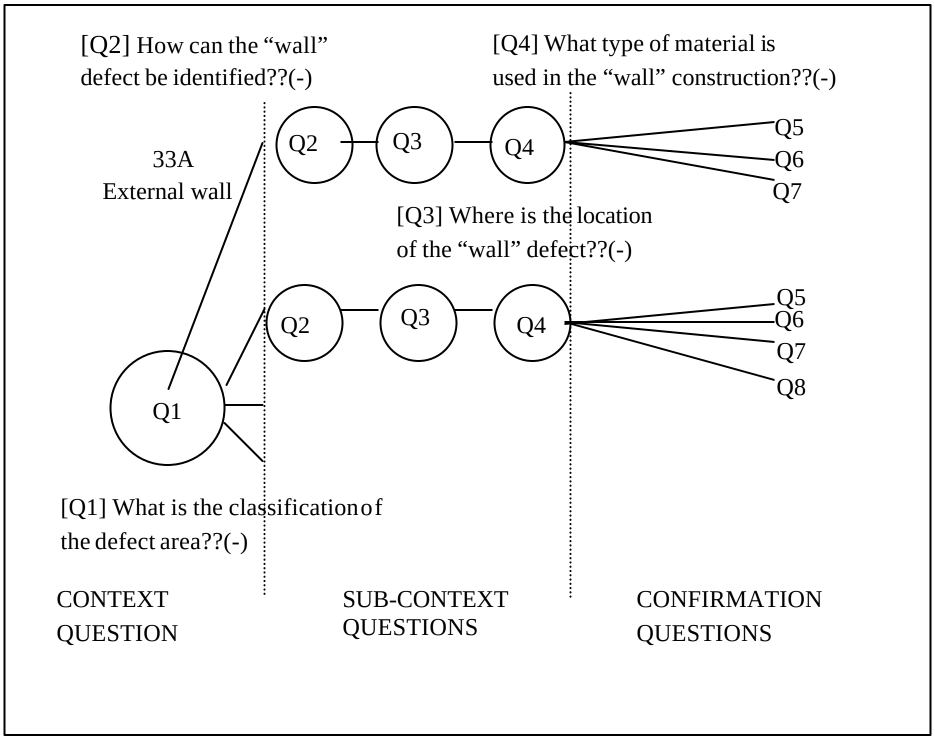

Fig. 7: Structure of questions

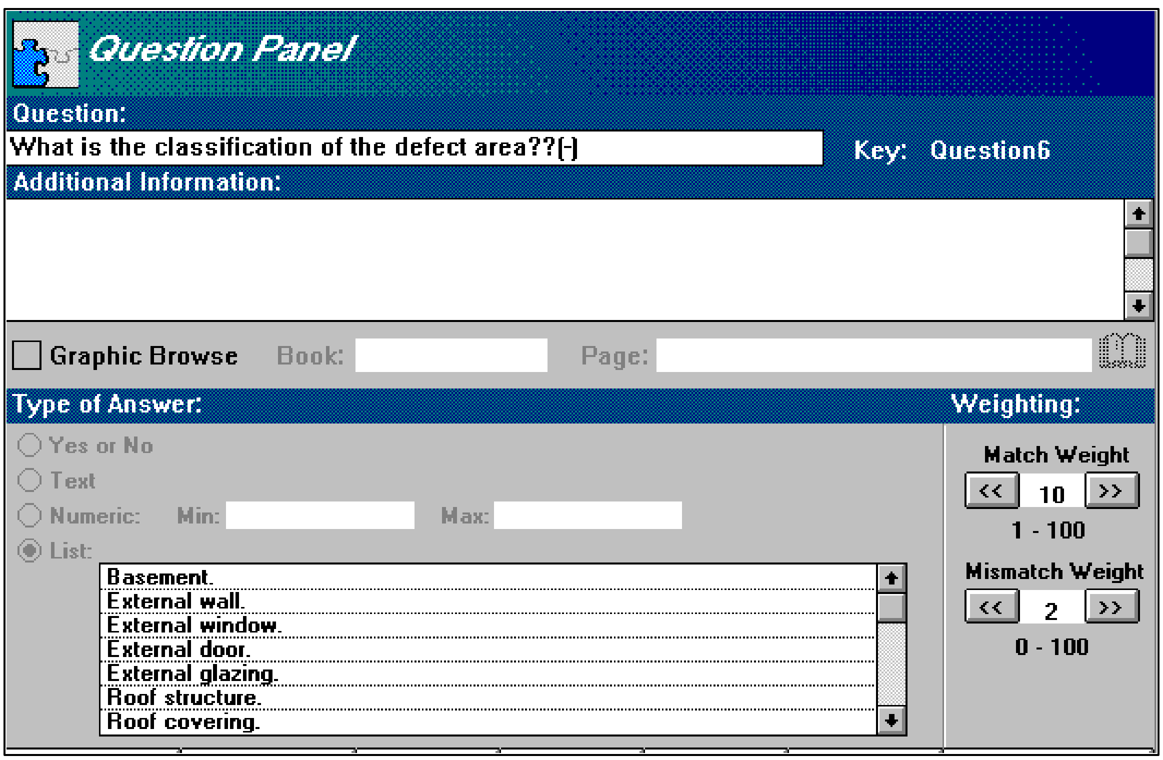

Fig. 8: Context question in the Question Panel

questions were placed in a logical and consistent order as possible. The case base administrator had placed first the context questions, then sub-context questions and finally all the confirmation questions, as illustrated in Fig. 7.

6.3.4 Actions

Designing an action is a straight forward process. Basically, an action is something associated with case. An action could be any of the following.

- Text instruction on how to fix a problem.

Text description of anything at all that is useful to associate with the case.

- A program to run, for animation, video instruction or even letter writing.

In PAKAR, an action supplies information regarding the recommended remedies or cure for building defects as in Fig. 9.

6.3.5 Graphic Browser

One problem that arose during the development is the layout or interface of *CBR Express*. It does not accommodate any flexibility to change the space allocation. In this domain, there are three most important factors that would have to be appraised, i.e. symptoms, diagnosis and remedy. The interface in Case Manager does give direct allowance for symptoms and remedy but not the diagnosis of building defects. Therefore, the option is to utilise the graphic browser. If this checkbox is checked, PAKAR will open an external *ToolBook* file when the search operator requests a browse of this case from the Search Panel.

The external program that was used to support the supplementary information is *Windows Write.* Frequently, the diagnosis can be perceived better with some illustrations. Hence, in some cases, drawings that described the defects are drawn by using *AutoSketch for Windows*.

6.3.6 An Overview of Case Base Manager

Fig. 11 illustrates an overview of a case in the Case Panel that is complete.

6.4 Case Retriever

This component utilises the Search Panel of PAKAR to search prior cases with three small windows, i.e., description, questions and actions or matching cases. The window for description receives general description of a current problem. The description is in form of unconstrained natural language. The first step in performing a case base search is to write a natural description of a problem in the description window. There is no syntax restriction. A good description is about one sentence in length. A very short description does not always contain enough information for a good match.

When the search operator enters the description of a defect in this window, the Case Retriever conducts an initial search on this description alone, looking for prior cases relevant to the current problem. The retrieval process can identify partial matches. Hence, the system can work in the presence of almost correct or incomplete information from the search operator.

After a search, the matching cases' window returns with cases that are similar to the search description. A default number of cases retrieved at this stage is five. Each case is displayed with its matching score, ranging from 0 to 100 with a score of 100 representing a perfect match. This indicates how closely a retrieved case matches the current problem description. The higher the score, the closer the case is to the problem. These scores are located on the far left of each case in the Matching Case window (Fig. 12).

In PAKAR, the case description itself constitutes 20% out of the total score assigned to a case. The rest would come from the questions' answers. In addition, the Case Retriever retrieves a set of questions from the retrieved cases and displays the combined set of questions in the question window. A set of questions is used to sharpen the focus of the search and help to differentiate among the competing cases. Incorrect answer to these questions will eliminate cases from the list.

The search operator, then, answers the questions. The search operator can decide which questions to be answered and in what order. Hence, leaving more control in the hand of the search operator. Nevertheless, the search operator is advised to answer all the context and sub-context questions. Normally, the questions listed repeat and confirm the problem entered by the search operator in the description window (Fig. 12).

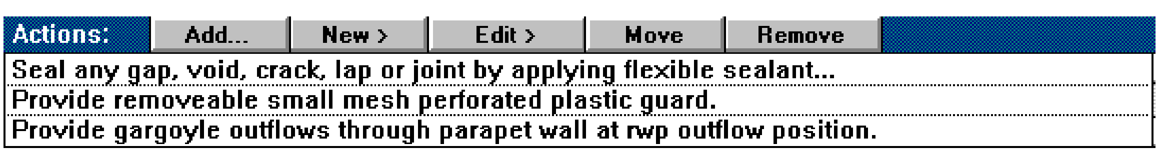

Fig. 9: Example of a case with several actions

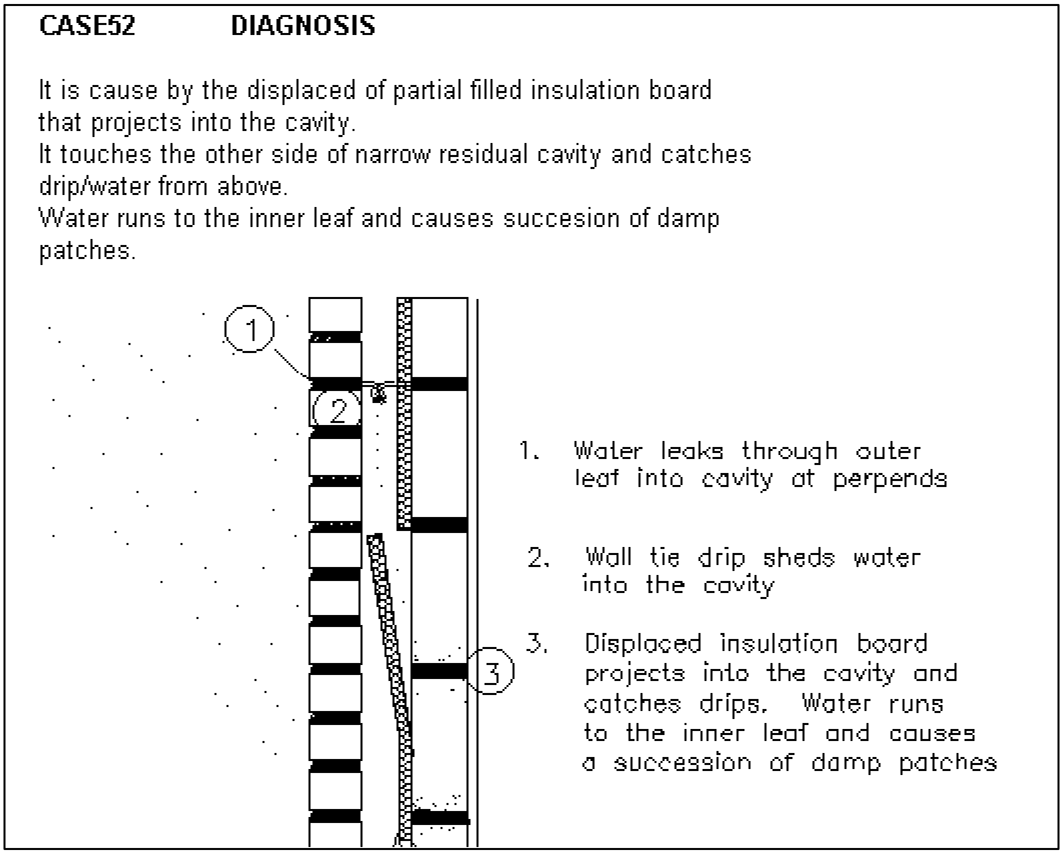

Fig. 10: Example of a diagnosis in *Write*

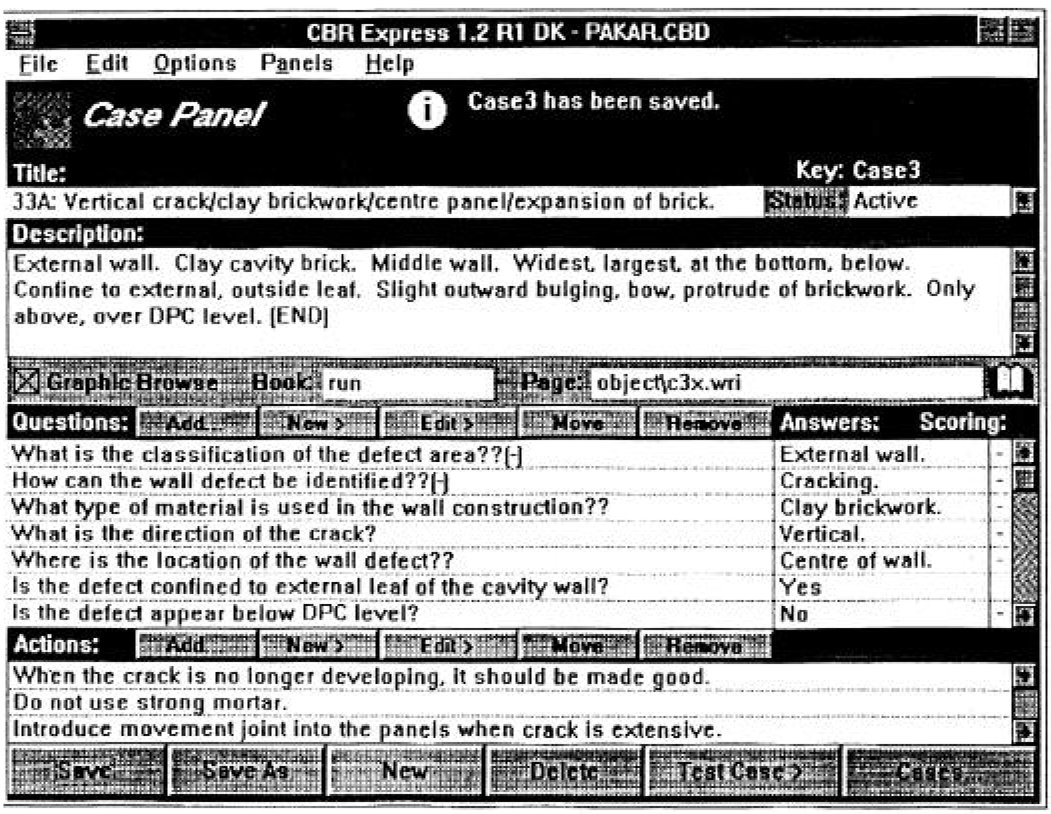

Fig. 11: A completed case in the Case Panel

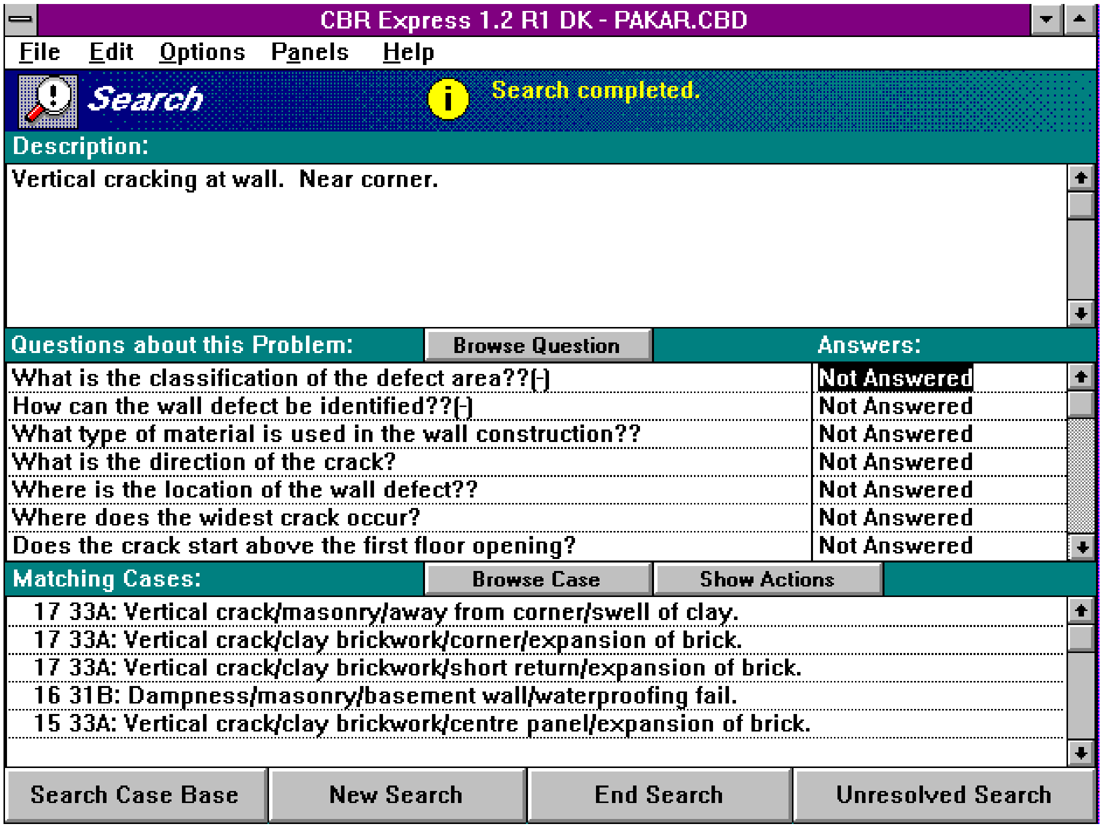

Fig. 12: Result of Initial Search

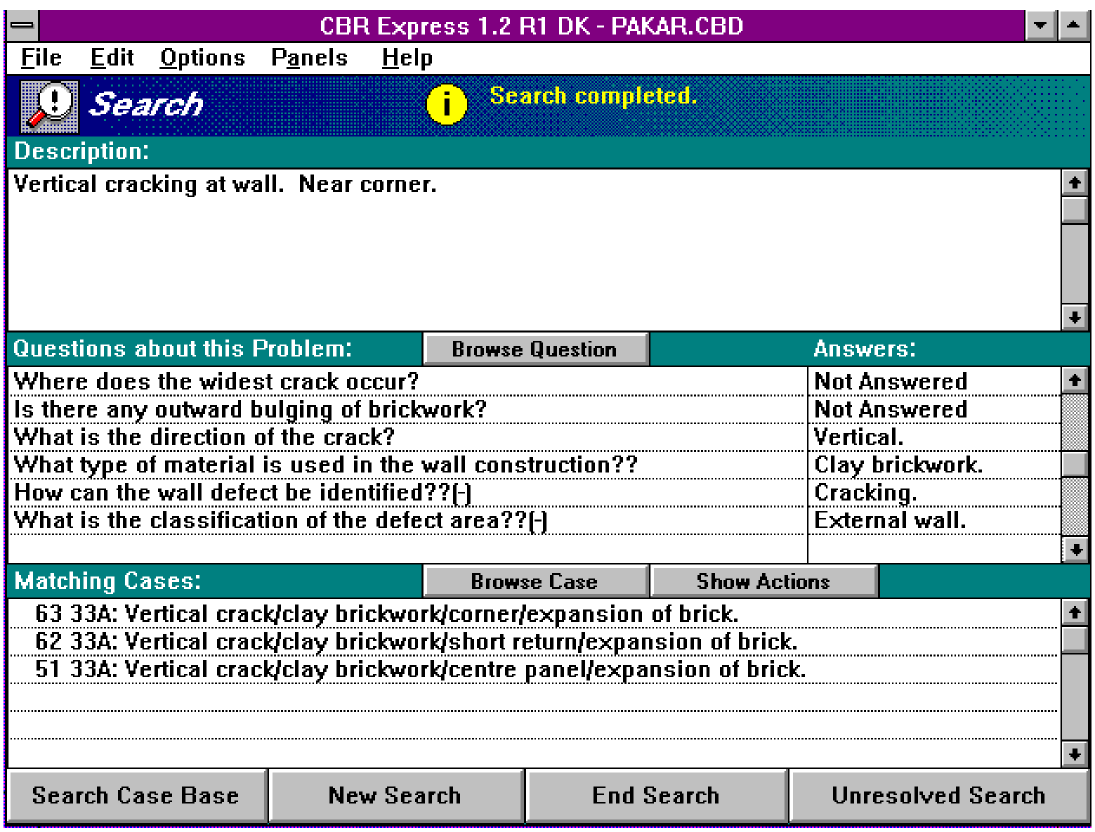

Fig. 13: A more accurate list of potential cases

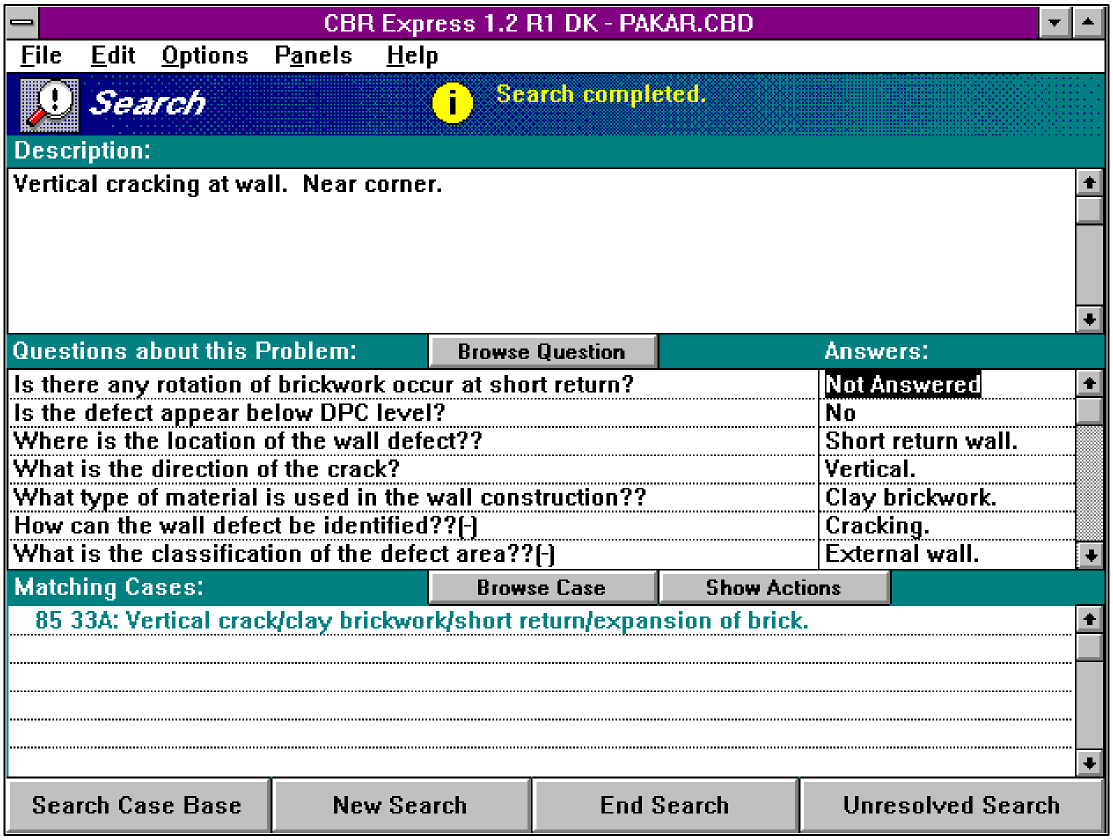

Fig. 14: The Final Matching Case

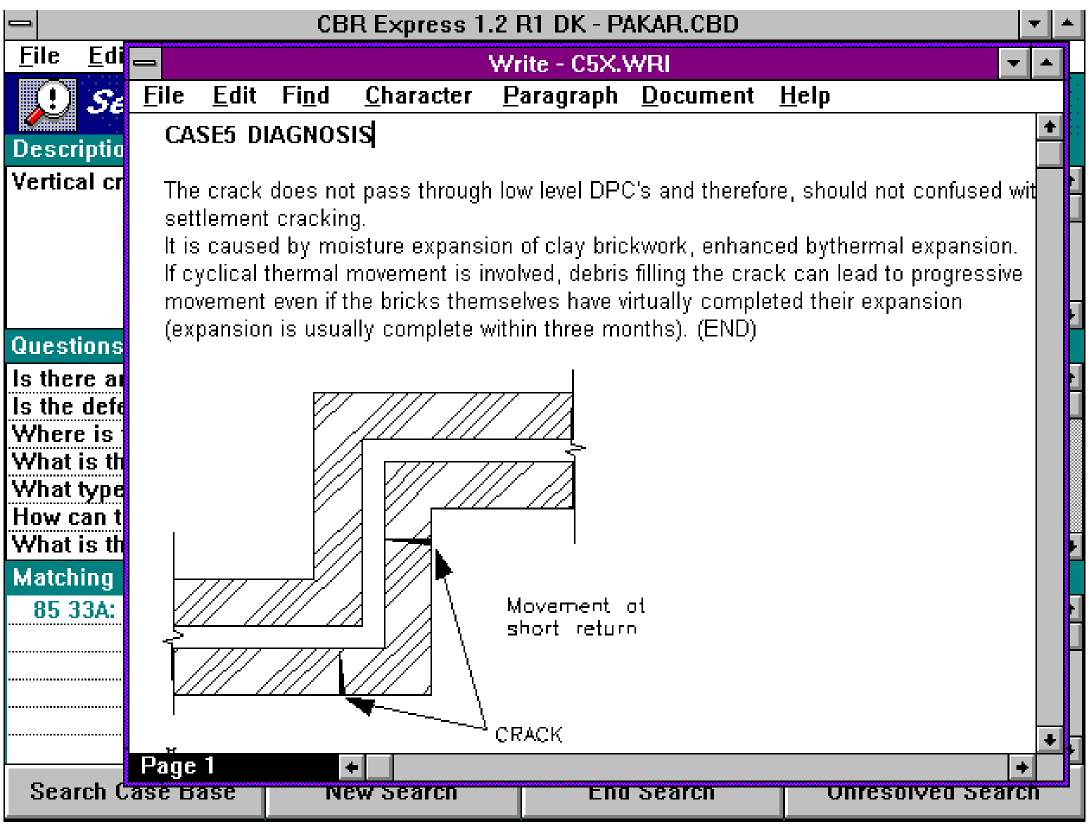

Fig. 15: The diagnosis of a case

The search operator supplies the information available and then launches another search. Here, the search is done automatically. The Case Retriever conducts a search after each question is answered on the Search Panel. Another set of cases will be retrieved again. The list of cases is normally similar to the first but with revised matching scores. The search operator may continue to answer questions and search until one of the cases shows an acceptably high score or until all the information has been used.

In PAKAR, the colour of the case and the action will change when the score exceeds the acceptance colour threshold, i.e., when it reaches 85 (Fig. 14). This case and action would represent the solution to the problem since they have the highest matching score. Then, the search operator may browse the case and its recommended actions to view additional information (Fig. 15).

When the search operator encounters a situation where the Case Base does not perform satisfactorily, he may record the state of the Search Panel. The purpose is that a subsequent evaluation can be done on the case by the case base administrator since it may represent flaws in the Case Base or a case that have been overlooked. Therefore, a new case will be added to the Case Base. The unresolved case will be saved as an entry in the Case Base, i.e., as an unindexed case. The case base administrator will browse the unresolved case using the Case Panel in the Case Manager. He will make any editing or adaptation necessary to the unresolved case. In PAKAR, since it depends on the case base administrator to perform the evaluation, it is called manual adaptation. After some adaptation has been made, this case will be resaved. It will represent a new case and will be added to PAKAR's Case Base as a new indexed case.

6.5 Service Manager and Customer Data Base

Both of the components have not yet been developed. Service Manager will enable a search operator to maintain a Customer Data Base, which not only store customer records but also service request by customer in a chronological order. The process is briefly described in this section.

The customer is assumed to be a tenant of a building where a defect has occurred.. When a defect is reported by a tenant, the details are usually immediately keyed in. The details will be entered into the Customer Data Base through the Service Manager. The Case Retriever will be activated to search for relevant cases. When the search is completed, the search operator returns to the Service Manager with all the information obtained from the search. The search operator uses this information to determine the recommended remedies. Once they are determined, the search operator can send out the information to those who will carry out the work. Once the work has been resolved, the service case is closed and then stored in Customer Data Case.

7.0 LESSONS LEARNED

In developing PAKAR, several practical lessons were identified about building case based systems to support real world decision making.

7.1 Knowledge Acquisition

CBR supports better knowledge transfer and justification of solutions from system to domain experts. This is because cases can be considered as expertise since they constitute a library of prior experience. Unlike rule-based system which requires knowledge in an ‘if-then’ format, in this application, domain experts can present knowledge directly to the CBR shell. Instead of painstakingly generating sets of rules, the cases are kept in their original forms. Hence, the problem of the knowledge acquisition bottleneck is less severe [11]. It also lets the domain expert define cases in almost natural language, so that the system can be used, compiled and maintained by domain experts instead of knowledge engineers and programmers.

7.2 Incomplete Case Base

PAKAR helps the case-base administrator get the application running, even though there is an incomplete original case library. An incomplete rule-based system rarely provides much value [12]. Since, it rigidly matches rules to a problem description, a missing rule will halt the reasoning process. The problem will not be recognised and cannot be solved. Therefore, it is not robust. However, partial matching and "best guessing" are built into the case-based strategy, because it is seldom found that two complex situations conform with every aspect [6]. Other cases in a nearby neighbourhood can compensate for a missing case. Therefore, the system can find and adapt near solutions.

The case base will grow as the prototype is being used, i.e., during problem solving. The case base grows with "episodic" cases. These cases are similar to the seed cases but differ in some particular characteristics. The "episodic" cases are initially identified as unresolved cases by utilising the Unresolved Search option. The case base administrator will evaluate the unresolved cases. These cases may form new cases after the case base administrator had made some adaptations to the differed characteristics. The case base will be used for finding and capturing new cases.

7.3 Future Case Base

If an expert system, which uses DBMR knowledge representations such as production rules, is to be distributed to different organisations, all the systems distributed would be the same as the first, i.e., the rules would remain the same [2, 24]. Only the developers of the system can change or increase the rules within the system. However, CBR system will grow. If the system, which initially contains the "prototype" cases, is circulated among different organisation, after some time, each system will grow with other "episodic" cases. These "episodic" cases will make each system different. The case base will develop according to the demand by that particular organisation. For example, organisation A may encounter many wall defects while organisation B may encounter many roof defects. The case base grows in different directions (Fig. 16). Therefore, in future, each CBR system will not be the same as when they were initially distributed. It will reflect the whole experience of each individual organisation. This experience will, then, be retained within the organisation in a structured way, termed "knowledge asset management".

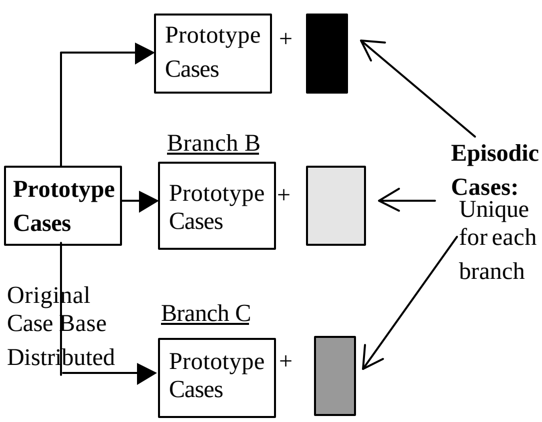

Fig. 16: Distribution of case base

7.4 Validation

In general, many development tools' applications require specification of system functionality, i.e., given some input and specify the output, so that the results produced by applications can be validated. For DBMR systems, explanation of the tools' functionality can be done through a decision tree or a list of rules. This knowledge is typically at low enough level to be independently evaluated. A CBR system such as PAKAR associates a problem description with a solution via historical correlation. If it works before under similar condition, then it should work again. This explanation can be very powerful because while rules express the theory that a solution should work, cases express the reality that a solution did work.

The overall PAKAR's system can only be validated by validating the solution of each case. This proved to be more difficult than validating a single rule, since rule typically represents a lower, simpler level of knowledge than does a case. For most DBMR system, the functionality is static. However, each problem solved with PAKAR can add new cases. Therefore, it will inherently change the overall system. In PAKAR, validation of any new case is done manually by the case base administrator.

8.0 CONCLUSION

CBR is a powerful approach to the problem of organising a large amount of information and to the problem of learning from experience. Its strength cannot be perceived as a technological advancement but more of a methodological approach whose success depends on how several problems have been solved previously in the area of memory. It induces generalisation based on its ability to detect similarities between cases.

CBR is more pertinent than traditional expert systems where cases are important knowledge source within the underlying domain and the available experts reason from cases. Within such domains, there are often many cases that can be easily adapted to solve a problem. CBR is not suited for a domain, which has cases that can only be treated using common sense knowledge or in-depth reasoning of domain knowledge.

PAKAR, the developed CBR system for diagnosis of building defects, has verified that CBR can be applied for use in developing an expert system in the construction industry. Basically:

- the domain is already "case-based" in its overall structure.
- remedial actions and diagnosis of building defects depend heavily upon the outcomes of past records.
- the availability of a data base of previous cases could be used to seed the case base.

Therefore, CBR, which has demonstrated its capability for applying past experience, is natural for decision making in this domain.

PAKAR's architecture is based largely upon *CBR Express* and it may need be modified if another CBR building tool is used. PAKAR has shown:

- how a domain expert can add knowledge to a CBR without having to write rules, but using unconstrained natural language. It can be implemented by professionals with construction background who do not have any programming background.
- that *CBR Express* can be linked with other development tools to achieve more satisfactory results.

The main operations that the case base administrator performs are:

- determining the behaviour of the case base during searches to produce consistent and economical searches.
- designing each individual case, question and action in a clear and unambiguous manner.

- refining the cases to produce intuitive, accurate and useful searches.

PAKAR learns primarily by collecting and indexing cases. New cases will give more familiar contexts to solve problem or evaluate situations. PAKAR becomes more efficient:

- by increasing its collective memory of past solutions and adapting them, i.e., the case base grows.
- because the answers can be accessed more readily and quickly, instead of having to deduce a new answer each time a problem is raised.
- it is easier to maintain since there is no requirement to edit a rule set or to reconstruct a decision tree; the system easily absorbs new experiences.

PAKAR has indicated that the development of expert system could be given a whole new lease of life. Its simple approach combined with ample flexibility and its proficiency for overcoming the separation between learning and problem solving gives a very promising sign to the future generation of expert system in the construction industry.

REFERENCES

[1] E. L. Rissland, J. Kolodner and D. Waltz. "Case-based Reasoning". *Proceedings of the DARPA Workshop on Case-based Reasoning. May 31-June 2, 1989, Florida*, pp. 1-13.

[2] A. M. Dearden and D. G. Bridge. "Choosing a reasoning style for a knowledge-based system: Lessons from supporting a help desk". *The Knowledge Engineering Review*. September 1993, Vol. 8, No. 8, pp. 201-222.

[3] C. K. Riesbeck and R. C. Schank. *Inside Case-Based Reasoning*. Lawrence Erlbaum Associates Publishers. Hillsdale, NJ, 1989.

[4] R. C. Schank. *DYNAMIC MEMORY: A Theory of Reminding and Learning in Computer and People*. Cambridge University Press, USA, 1982.

[5] P. Buta and R. Barletta. "Case-based Reasoning for Market Surveillance", in *Proceedings of the First international Conference on Artificial Intelligence Applications on Wall Street. October, 1991, New York*, pp. 116-121. IEEE Computer Society Press.

[6] S. Slade. "Case-based Reasoning for Financial Decision Making", *in Proceedings of the First international Conference on Artificial Intelligence Applications on Wall Street. October, 1991, New York, pp. 232-237*. IEEE Computer Society Press.

[7] J. Kolodner and W. Mark. "Guest Editors' Introduction: Case-Based Reasoning". *IEEE EXPERT Intelligent System and Their Applications*. October, 1992, Vol. 7, No. 5, pp. 5-6.

[8] C. Williams and B. D. Clayton. "Case Base Retrieval", *Microsoft Internet Explorer*, Inference Corporation.

[9] K. Sycara, D. N. Chandra, R. Guttal, J. Koning and S. Narasimhan. "CADET: A Case-Based Synthesis Tool for Engineering Design". *International Journal of Expert Systems*. 1991, Vol. 4, No.. 2, pp. 157-188.

[10] C. K. Riesbeck. "An Interface for Case-based Knowledge Acquisition", in *Proceedings of the DARPA Workshop on Case-based Reasoning. San Mateo, CA. May 10-13, 1988*.

[11] L. M. Lewis, D. W. Minior and S. J. Brown. "A Case-Based Reasoning Solution to the Problem of Redundant Engineering in Large Scale Manufacturing". *International Journal of Expert Systems*. Vol. 4, No. 2, 1991, pp. 189-200.

[12] D. Hennessy and D. Hinkle (1992). "Applying Case-based Reasoning to Autoclave Loading". In *IEEE EXPERT Intelligent System and Their Applications*. October, 1992, Vol. 7, No. 5, pp. 21-26.

[13] S. Abdullah (1993), *The Potential of Case-based Reasoning in Construction Industry*, University of Salford, UK.

[14] G. U. Gupta and C. E. Brown (1994). "Applying Case-based Reasoning to the Accounting Domain". In *International Journal of Intelligent Systems in Accounting Finance and Management*, Vol. 3, No. 3, August 1994, pp. 205-221.

[15] D. Subar (1992). "Case-based Reasoning: A familiar story". In *Release 1.0*. January 31, 1992, Vol. 92, No. 1, pp. 1-10. EdVenture Holdings.

[16] M. Pearce, A. K. Goel, J. L. Kolodner, C. Zimring, L. Sentosa and R. Billington. (1992), "Case-based Design support: A case study in architectural design". *IEEE Expert Intelligent Systems and their applications*. Oct. 92, Vol. 1, No. 5, pp. 14-20.

[17] Property Service Agency (PSA) & Directorate of Building Development (1989), *Defects In Buildings*, HMSO, London, UK.

[18] L. Addelson (1989), *Building Failure: A Guide to Diagnosis, Remedy and Prevention*, Butterworth Architecture, UK. 1989.

[19] I. H. Seeley (1987), *Building maintenance*, MacMillian, UK.

[20] Inference Corporation (1992), *CBR Express: Designing CBR Express case Base*, Development Kit, Version 1.2, Berkshire, UK.

[21] I. D. Watson & F. Marir (1994), *Case-based reasoning: A review*, The Knowledge Engineering Review, Vol. 9, No. 3.

[22] R. E. Oxman (1990), *Prior knowledge in design: A dynamic knowledge-based model of design and creativity*. Design studies, Jan, Vol. 11, No. 1, Butterworth.

[23] I. D. Watson & S. Abdullah (1994), *Developing case-based reasoning systems: A case study in diagnosis building defects.* Case-based Reasoning: Prospects for Application, IEEE Digest No 1994/057, pp. 1-3.

[24] J. Bachant & J. McDermont (1984), *RI revisited: Four years in the trenches*, The AI Magazine 5(iii).

BIOGRAPHY

Salha Abdullah obtained her Master of Science in Information Technology in Property and Construction from University of Salford in 1994. Currently, she is a lecturer in the Department of System Science and Management, Faculty of Information Science and Technology, National University of Malaysia. Research interests include intelligent systems, multimedia and construction estimating.