A Web-based Intelligent Hybrid System for Fault Diagnosis

G. Jha, S.C. Hui and S. Foo

School of Applied Science, Nanyang Technological University
Singapore - 639798
gunjan@booch.sas.ntu.ac.sg, {asschui, assfoo}@ntu.edu.sg

Abstract

In traditional help desk service centres, service engineers provide a world-wide customer support service through the use of long-distance telephone calls. Such a mode of support is found to be inefficient, ineffective and generally results in high costs, long service cycles, and poor quality of service. With the advent of Internet technology, it is possible to deliver customer service support over the World Wide Web. This paper describes a Web-based intelligent fault diagnosis system, known as WebService, to support customer service over the Web. In the WebService system, a hybrid case-based reasoning (CBR), artificial neural network (ANN) and rule-based reasoning (RBR) approach is adopted for machine fault diagnosis. In this approach, ANN and RBR are incorporated into the CBR cycle (i.e. Retrieve, Reuse, Revise and Retain) instead of using traditional CBR technique for indexing, retrieval and adaptation. The hybrid approach has been implemented to provide efficient online intelligent machine fault diagnosis over the World Wide Web.

Introduction

A multinational corporation in Singapore manufactures and supplies insertion and surface mount machines widely used in the electronics industry. Its customer support department services its worldwide customers and provides installation, inspection and maintenance support for its customers. Insertion and surface mount machines are expensive and require efficient maintenance during machine down time. Although most customers have some engineers to handle day-to-day maintenance and small-scale troubleshooting, expert advice are often required for more complex maintenance and repair jobs. Prompt response to request from customers is needed to maintain customer satisfaction. Therefore, the multinational corporation has set up a hot-line service centre (or help desk) to answer frequently encountered problems. Service engineers from the hot-line service centre are responsible for answering inquiries from their customers via telephone calls and carrying out an on-site repair if necessary. At the end of each service, a customer service report is generated to record the reported problem and the proposed remedies or suggestions taken to rectify the problem. These service reports are then stored as service records in a customer service database.

In the traditional hot-line service support, the identification of machine faults relies heavily on the service support engineers' past experience or the information drawn from the customer service database. This method has a problem of training and maintaining a pool of expert service engineers. Thus, instead of relying on the knowledge of service engineers, an approach for intelligent fault diagnosis that extracts the knowledge from the customer service database to assist customers identify machine faults becomes extremely useful. The approach should be able to generate suggested remedial actions automatically or through user interaction based on the observed fault-conditions.

As a collaborative project between the multinational corporation and the School of Applied Science, Nanyang Technological University, Singapore, a Web-based intelligent fault diagnosis system, known as WebService, has been developed in order to support customer services over the Web. In this research, a hybrid case-based reasoning (CBR), artificial neural network (ANN) and rule-based reasoning approach is adopted as the intelligent technique for fault-diagnosis. The hybrid approach is operated as follows. Instead of using traditional CBR technique for indexing, retrieval and adaptation, ANN and RBR are incorporated into the CBR cycle (i.e. Retrieve, Reuse, Revise and Retain). ANN extracts knowledge from service records of the customer service database and subsequently retrieves the most appropriate service records based on user's fault description during the retrieval phase. RBR is then used to reuse the checkpoint solutions of the retrieved service record and guide the customer through a step-by-step approach to help diagnose the machine fault in the most effective manner. The machine problem and its checkpoint solution are revised with user feedback and the revised information are then retained by updating the relevant databases. The hybrid approach has been implemented into a Web-based system for intelligent fault diagnosis to deliver the customer service support over the World Wide Web.

Intelligent Fault Diagnosis

For the past few decades, there have been a proliferation of intelligent systems for fault diagnosis and related applications (Balakrishnan and Honavar 1998).

Traditionally, case-based reasoning has been successfully applied to fault diagnosis for customer service support or help desk (House 1994; Law et al. 1997; Liu and Yan 1997; Patterson and Hughes 1997; Shimazu et al. 1994). CBR systems rely on building a large repository of diagnostic cases (or past service reports) in order to circumvent the difficult task of extracting and encoding expert domain knowledge (Riesbeck and Schank 1989). It is one of the most appropriate techniques for customer service support as it learns with experience in solving problems and hence emulates human-like intelligence. However, the performance of CBR systems depends critically on the adequacy as well as the organization of cases and the algorithms used for retrieval from a large case database. Most CBR systems (Watson, 1997) use the nearest neighbor algorithm for retrieval from the flat-indexed case database, which are inefficient especially for large case database. Other CBR systems use hierarchical indexing such as CART (Breiman et al. 1984), decision trees (Quinlan 1986) and C4.5 (Quinlan 1993). Although this performs efficient retrieval, building a hierarchical index needs the supervision of an expert during the case-authoring phase.

The artificial neural network approach (Ficola et al. 1995; Ramden et al. 1995) provides an efficient learning capability from detailed examples. Supervised neural networks such as Learning Vector Quantization (LVQ3) (Kohonen 1990) are used when the training data consists of examples with known classes. LVQ3 stores the weight vectors as the code-book or exemplar vector for the input patterns. The matching is based on a competitive process that determines the output unit that is the best match for the input vector similar to the nearest neighbor. However, in LVQ3, the search space is greatly reduced because of the generalization of knowledge through training. In contrast, the CBR systems need to store all the cases in the case database in order to perform accurate retrieval that greatly increases the search space. The CBR systems that store only relevant cases for an efficient retrieval lack the accuracy as well as the learning feature of the neural networks. Hence, supervised neural networks become a very suitable complement for case retrieval.

Rule-based reasoning approach (Bowerman and Glover 1988; Klahr and Waterman 1986) is very effective and efficient for quasi-static systems operating within a fixed set of rules. Rule-based reasoning system follows the process of logical reasoning to make intelligent decisions. In modeling human problem solving approach, it can guide users in a step-by-step manner to arrive at solutions. For example, when a particular fault-condition is retrieved as the one closest to the user's problem description, a rule-based reasoning engine can guide the user in reusing its checkpoint solutions for that fault-condition to repair the machine fault in an effective manner.

Therefore, the incorporation of the artificial neural network and rule-based reasoning into the case-based reasoning cycle (Aamodt 1994) is deemed to be one of the most suitable intelligent techniques for fault diagnosis. Such a system behaves most similar to human beings in problem solving through recalling prior experience or case records, re-using and revising the checkpoint solutions through user feedback and subsequently learning through experience in solving the problems.

System Architecture

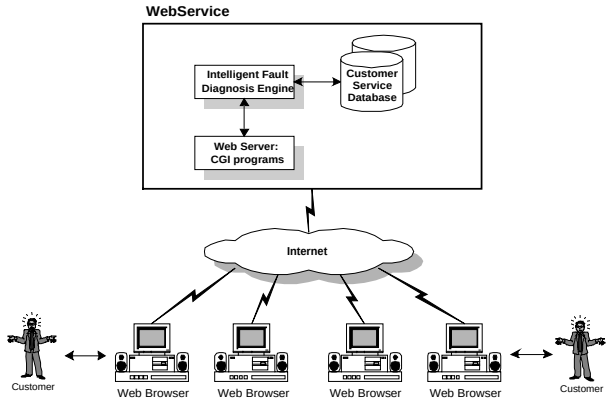

Figure 1: The WebService System

In this research, a Web-based system, which incorporates the integrated approach for online intelligent fault diagnosis, is developed. Figure 1 shows an overview of the system. The system is developed on the Windows NT environment. The Netscape Enterprise Server 3.0 is used as the Hypertext Transfer Protocol (HTTP) server. The Fault Diagnosis Engine has adopted the hybrid CBR-ANN-RBR approach and implemented in Microsoft Visual C++ as Common Gateway Interface (CGI) programs that are linked to the Web Server. The Microsoft Access database management system is used for storing the Customer Service Database. The CGI programs communicate with the database system through Open Database Connectivity (ODBC) to insert, delete or update information in the database. The CLIPS environment has been integrated into the CGI programs for rule-base management. Hypertext Mark-up Language (HTML) is used to create the user interface as Web pages to accept the user's fault description. The user can access the Web-based system via any Web browsers such as Netscape Navigator or Microsoft Internet Explorer.

Fault-condition	3008 PCB CARRY MISS ERROR. PCB WAS NOT TRANSFERRED BY THE CARRIER DURING LOADING BU STAYED AT THE DETECTION POSITION OF PCB DETECTION SENSOR 2.	
Checkpoint group : AVF_CHK007		
Priority	Checkpoint description	Help file
1	CONFIRM WHETHER THE CARRY GUIDE PINS ARE IN LINE WITH PCB.	AVF_CHK007-1.GIF
2	CONFIRM WHETHER THE PCB IS IN CORRECT DIRECTION.	AVF_CHK007-2.GIF
3	CONFIRM THE POSITION OF THE GUIDE LOWER LIMIT SENSOR. (I/O 0165)	AVF_CHK007-3.GIF
4	CONFIRM THE TIMING FOR PCB 2 DETECT SENSOR.	AVF_CHK007-4.GIF
5	CONFIRM THE TIMING FOR THE CARRIER START TIMING.	AVF_CHK007-5.GIF

Figure 2: Fault-condition and checkpoints information of a service record

Customer service database

To develop the integrated approach for online intelligent fault diagnosis, it is necessary to look into the structure of the customer service database. Service records (or reports) are currently defined and stored in the customer service database to keep track of all reported problems and remedial actions. Each service record consists of customer account information and service details. Service details contain two types of information: *fault-condition* and *checkpoint*. Fault-condition contains the customer's description of the machine fault. Checkpoint indicates the suggested actions or services to be carried out to repair the machine into normal condition based on the occurred fault-condition given by the customer. Checkpoint information contains checkpoint group name, and checkpoint description with priority and an optional help file. The checkpoint group name is used to specify a list of checkpoints that defined under the group. Each checkpoint is associated with a priority that determines the sequence in which it can be exercised and a help file that gives visual details on how to carry out the checkpoint. An example of fault-condition and its checkpoint information for a service record is given in Figure 2.

Currently, two sets of service records are maintained. The first set contains all the service reports the service engineers have reported during their on-site repair for the past few years. As the same fault-conditions and its corresponding remedial actions can appear repeatedly in this set of service records, the second set is then created as a subset of the first set which contains service reports with unique fault-conditions, thereby no repetition of service reports are stored. As will be discussed later, the first set of the customer service database (or training set) will be used for training purposes, while the second set (unique set) will be used for the initialization process of the neural network and rule-base generation. Currently, the unique set of customer service database is used and searched by the service engineer of the hot-line service centre to provide or suggest remedial actions for a reported fault-condition by the customer.

Hybrid approach

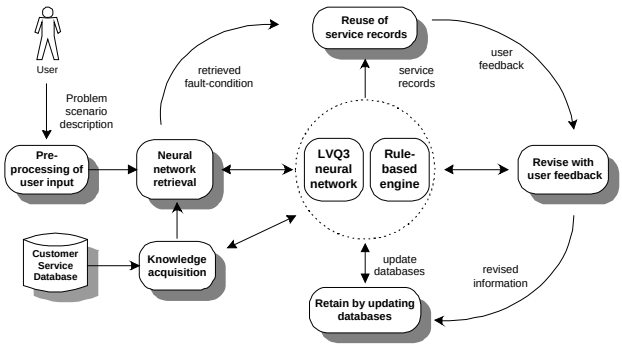

Figure 3: Fault diagnosis process

Figure 3 shows the intelligent fault diagnosis process. It accepts user's description as input, maps the description into fault-conditions of the closest faults reported before, and retrieves the corresponding checkpoints the user can try to resolve the problem. As discussed earlier, the integrated CBR, ANN and RBR approach has been adopted as the intelligent technique for fault diagnosis. Apart from the traditional four phases of CBR cycle, the Learning Vector Quantization (LVQ3) neural network and the rule-based engine (CLIPS) (Johnson 1998) have been incorporated into the CBR cycle to improve the efficiency, accuracy and effectiveness of the fault diagnosis process.

The diagnosis process starts by extracting expert knowledge from the customer service database through neural network training and rule-base generation. The fault-conditions are pre-processed into a suitable format, and the LVQ3 neural network is then trained based on the fault-conditions. A rule-base is subsequently generated using the checkpoints of each fault-condition from the unique set of the service records.

Knowledge acquisition

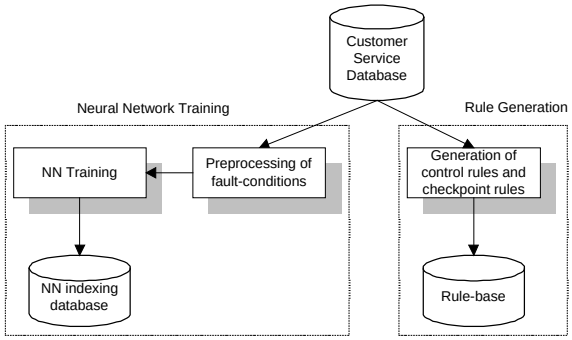

Figure 4: Knowledge acquisition

Figure 4 shows the knowledge acquisition phase. The LVQ3 neural network (Kohonen 1990) is trained to index the service records using the fault-conditions. In order to perform training using the fault-conditions, the pre-processing process is carried out to obtain a set of consistent keywords from the fault-conditions, which is in turn used to initialize a neural network indexing database to perform retrieval. The neural network indexing database consists of the list of the extracted keywords, a neural network weight matrix and the linking information between keywords and the corresponding fault-conditions. A CLIPS rule-base (Johnson 1998) is also constructed from the fault-conditions, checkpoints and checkpoint priority information of the customer service database through rule generation. The rule-base contains two types of rules, namely, control rules and checkpoint rules.

The *Preprocessing Fault Conditions* process preprocesses fault conditions of service records from the customer service database in order to extract keywords and key phrases. This involves the preprocessing of sentence(s) representing a fault condition. When parsing each fault condition, the redundant non-alphanumeric and punctuation characters are first eliminated so that the sentence is converted into a sequence of words separated by blanks. Then, technical abbreviations from the machine domain are extracted as keywords using a list of abbreviations. For examples, NC, PTR and N.G. are used as abbreviations in the fault conditions. Subsequently, all the words in inflectional forms are morphed to their stem form. For examples, "insertion", "insertions" and "inserted" are various inflectional form of the stem word "insert". Antonym form of words, that should not be matched to the original word, are identified and converted to the form of "not + verb" using a logical-not list. For examples, "not insert" and "cannot insert" are converted to "not + insert". In addition, there are several verb phrases in the English language having special meanings. Examples of such phrases are "flip flop", "turn on" and "shut up". These phrases are extracted using a verb-list. Finally, the redundant or stop words, such as "to", "the" and "after", are eliminated using a stop-list. The preprocessing process is implemented using verb-list, stop-list and algorithms from Wordnet (Miller 1990).

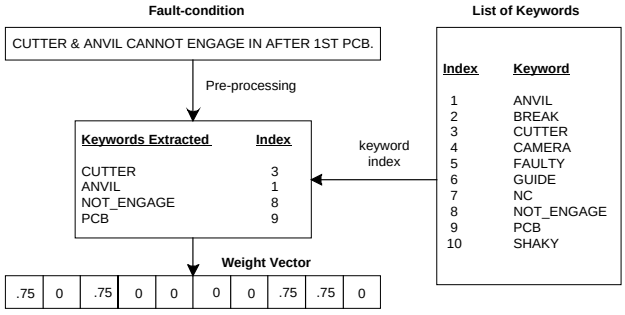

Figure 5: An example to show how a weight vector is obtained

The *Neural Network Training* process is needed to *initialize* and *train* the neural network before it can be used for retrieval. During the initialization phase, input nodes are connected to the output nodes through links. The weights of these links are assigned by considering one output node at a time and updating the weight vector corresponding to the links associated with it. During the *Preprocessing Fault Conditions* phase, each fault-condition from the customer service database (the unique set) is preprocessed and its keywords are extracted to generate a list of keywords. The indices of keywords from a fault-condition in the keyword list are used to form the weight vector as shown in Figure 5. The process is repeated for all the fault-conditions in order to initialize the weight matrix. The initial weight matrix is a sparse matrix of order equal to the number of keywords times the number of unique fault-conditions and consists of the values '0' and '0.75' only. The value '0.75' has been chosen instead of '1' to provide the learning capability of the network. After the neural network has been initialized, the network will undergo a training process. The training set of the customer service database will be used for training. The weights are updated using the LVQ3 supervised learning algorithm. The learning rate is set to 0.4. For each fault-condition, the less frequent keywords will have lower weights whereas more frequent keywords will have higher weights. The training algorithm is presented in Figure 6, which is based on the LVQ3 network (Kohonen 1990).

Neural Network Training Algorithm

For each training record from the customer service database, do steps 1-7.

1. Identify the target fault-condition of the training record from the unique set of fault-conditions.
2. Preprocess the fault-condition of the training record to form a numerical vector.
3. Match the input vector with weight vectors of the output nodes and compute the winner output node as the one with minimum Euclidean distance.
4. If the winner output node corresponds correctly to the target fault-condition, update weight vector corresponding to the winner node, w_c, as

$$w_c(t+1) = w_c(t) + \alpha\,[x - w_c(t)]$$

where w_c is the weight vector closest to the input vector, t refers to the time instance, α is the learning rate and x is the input vector. Go to step 1.

5. Otherwise, update weight vector, w_{c1}, of the winner node as

$$w_{c1}(t+1) = w_{c1}(t) - \alpha\,[x - w_{c1}(t)]$$

6. Compute the 2nd winner output node with the corresponding weight vector w_{c2}. If the 2nd winner node is the same as the target fault-condition, adapt w_{c2} positively to the input vector as

$$w_{c2}(t+1) = w_{c2}(t) + \alpha\,[x - w_{c2}(t)]$$

provided

$$\min\left[\frac{d_{c1}}{d_{c2}}, \frac{d_{c2}}{d_{c1}}\right] > (1-\varepsilon)(1+\varepsilon)$$

where d_{c1} and d_{c2} denote the distances of the two weight vectors with the input vector and ε is a constant typically set to 0.2.

7. If new keywords (i.e. keywords not in the existing keyword list) have been identified in the preprocessing process, add new input layer nodes corresponding to the new keywords and update the weight matrix.

Figure 6: The neural network training algorithm

A *rule-base* is generated consisting of control rules and the checkpoint rules. Control rules are coded manually and provide diagnostic control towards solving a machine fault problem, whereas checkpoint rules are generated automatically from the information contained in the service records and provide specific diagnostic instructions towards solving a particular machine fault. Control rules will specify the procedure for the firing of checkpoint rules so that the checkpoints can be exercised one by one according to their priorities. Using these two types of rules, the rule-based inference engine under the CLIPS environment can provide a step-by-step guidance to the user in diagnosing a fault-condition. Priorities of checkpoints, which will affect the checkpoint rules' firing sequence, are assigned initially by the service engineers according to the importance of the checkpoints in solving the fault-condition. The checkpoint priorities are pre-defined in the customer service database as shown in Figure 2. Therefore, a rule-base generator has been developed to read in the service records from the customer service database and generates the checkpoint rules automatically. An example of a checkpoint rule, based on the service record given in Figure 2, is shown in Figure 7. A total of 15 control rules are defined to regulate the firing of the checkpoints. The rule-based engine seeks user's feedback, fires the rules and subsequently accesses the customer service database to display checkpoints through a Web-based interface. The firing of next rule and the stopping condition for firing is based on user feedback.

```
(defrule MAIN::chkpt_rule-7
   (phase accept.fault.cond)
   (fault.cond AVF_CHK007)
   =>
   (assert (check-seq AVF_CHK007-1 AVF_CHK007-2
    AVF_CHK007-3 AVF_CHK007-4 AVF_CHK007-5))
   (assert (help-seq AVF_CHK007-1.GIF
    AVF_CHK007-2.GIF AVF_CHK007-3.GIF
    AVF_CHK007-4.GIF AVF_CHK007-5.GIF))
)
```

Figure 7: Checkpoint rules for a particular fault-condition

Preprocessing of user input

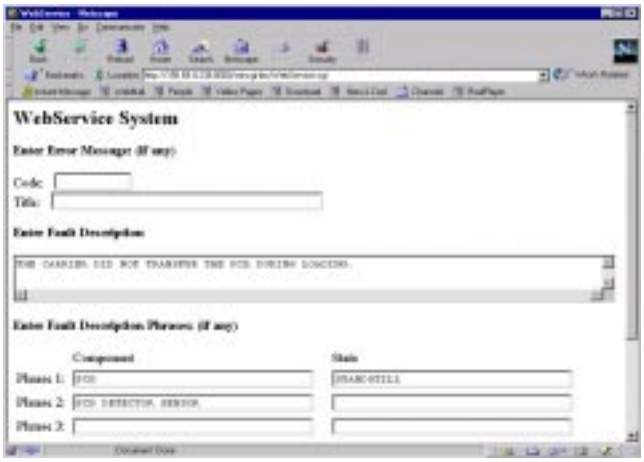

Figure 8: User interface to accept fault description

The user input to the fault diagnosis engine is in the form of fault description. In the WebService system, the user fault description is accepted in the form of natural language. However, for the convenience of some users, the system also accepts other forms of inputs such as the error code and error title, and the name of fault components and their states. The user may choose the input mode or a combination of them. Figure 8 shows the Web-based interface for accepting the user input of the fault description.

Firstly, the user can enter the error code and error title if available. These information appear on the monitor of the NC (numerically controlled) machine when the problem occurs. Error code uniquely represents the corresponding fault condition. A list of error codes and its corresponding fault conditions is maintained for efficient retrieval. If the error code is known, then no other information is required from the user for further processing, the corresponding fault condition can be identified and its checkpoints can be retrieved. Otherwise, the fault description can be entered in natural language or as a set of keywords. User can also provide the names of machine components and their states as input. If the user input contains keywords that are not in the keyword list, synonyms of these keywords will be retrieved for user confirmation as input keyword. User fault input is then preprocessed to form the input vector for retrieval.

Neural network retrieval

The neural network retrieval process recalls similar fault-conditions experienced in the past and ranks them based on the score that signifies the closeness of the retrieved fault-condition to the user input. The retrieval algorithm is based on the matching process discussed in the training algorithm. The process of computing winners can be extended to compute successive winners for the input provided by the user. Each of the winners should have a matching score above a pre-defined threshold. The number of matches to be retrieved is based on this threshold. Then, the fault-conditions corresponding to the winner nodes are presented to the user for feedback. Figure 9 shows the fault-conditions retrieved by the neural network when the user enters the fault description as shown in Figure 8. It can be observed that the fault-condition displayed at the top of the screen is the one closest to the fault description provided by the user.

Figure 9: Retrieval results based on the specified user input

Reuse of service records

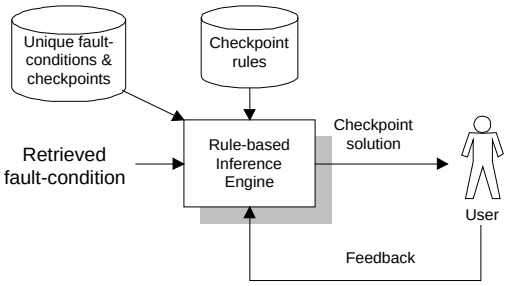

Figure 10. Case Reuse.

This process reuses the checkpoint solutions of the fault-conditions retrieved during the retrieval process. In the neural network retrieval process, the displayed fault-conditions that match most closely to the input fault description provided by the user are ranked according to their matching scores. The lower portion of the display in Figure 9 also shows a checkpoint solution of the step-by-step guidance given to the user. The checkpoints are presented according to the rules fired from the rule-based inference engine as shown in Figure 10. The rules operate in a competitive manner to display the checkpoints in the order of their priority in solving the fault-condition. Help can be obtained for exercising a particular checkpoint by clicking the “Help” option. This will load the help file for the corresponding checkpoint, as shown in Figure 11, to help the user to carry out the remedial task.

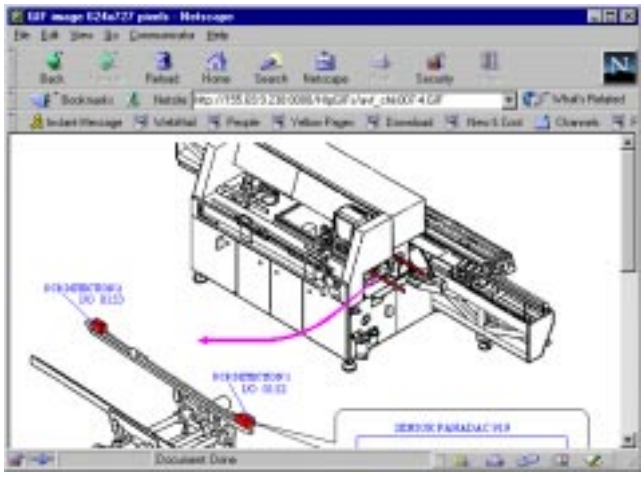

Figure 11: Help information to assist user carry out the remedial task

Revise and retain with user feedback

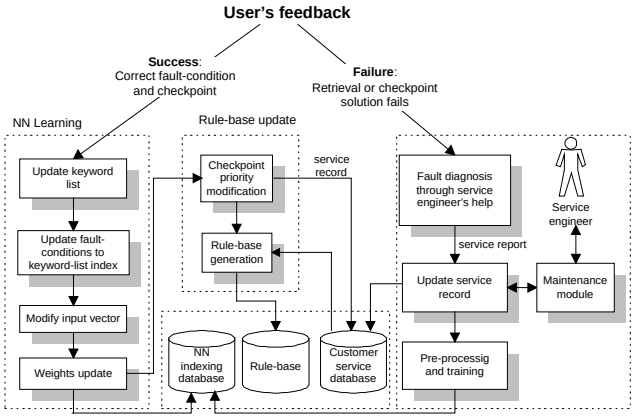

Figure 12: Revise and retain with user feedback

In this phase, the neural network indexing database, the checkpoint rules and the service records in the customer service database are updated based on user feedback on the effectiveness of the fault diagnosis process. The input problem description and its past checkpoint solutions are revised through the user feedback and updated into the relevant databases. As shown in Figure 12, the user provides feedback on whether the problem is solved or not. If the problem is resolved, then the neural network indexing database and the checkpoint rule-base are updated. If the problem is not resolved after trying all the checkpoints of the retrieved fault-conditions, then the user will make a failure report and the problem will be further investigated by service engineers. The service report generated subsequently will be taken into account for updating the customer service database and the neural network using the maintenance module as shown in Figure 13.

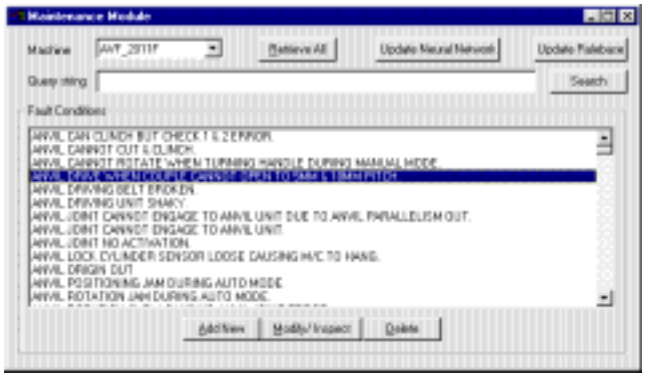

Figure 13: Customer service database maintenance

Conclusion

Traditional help desk service centres are inefficient and ineffective in providing customer service support. The paper has presented a Web-based intelligent hybrid system that uses hybrid CBR, artificial neural network and rule-based reasoning approach to support online intelligent fault diagnosis over the Web. Users can access WebService through any Web browsers such as Netscape Navigator or Microsoft's Internet Explorer. The performance of the hybrid approach has been evaluated and compared with traditional CBR approach using nearest neighbor algorithm in terms of efficiency and accuracy (Jha, Hui, and Foo 1998). This approach has performed better than the traditional CBR approach.

References

Aamodt, A., and Plaza, E. 1994. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, AICom - Artificial Intelligence Communications, IOS Press, 7(1):39-59.

Balakrishnan, K., and Honavar, V. 1998. *Intelligent Diagnosis Systems*. Forthcoming in the Journal of Intelligent Systems, Vol. 8.

Bowerman, R.G., and Glover, D.E. 1988. *Putting Expert Systems into Practice*. Van Nostrand Reinhold Company. New York.

Breiman, L., et al. 1984. *Classification of Regression Trees*. Monterey CA: Wadsworth & Brooks/Cole.

Fausett, L. 1994. *Fundamentals of Neural Networks*. New Jersey: Prentice-Hall.

Ficola, A.; Cava, M. La; and Magnino, F. 1995. An application of neural networks in a fault diagnosis scheme for dynamic systems. In proceedings of the International Symposium on Intelligent Robotic Systems, Bangalore, India.

House, W.C. 1994. Automating Help Desk Operations

Using Case-Based Reasoning: A Practical Application of Expert Systems Technology. In Proceedings of the Annual Conference of the International Association for Computer Information Systems, 100-106. Washington, DC, USA.

Jha, G.; Hui, S.C.; and Foo, S. 1998. WebService: A Hybrid CBR-ANN-RBR System for Intelligent Fault Diagnosis, Technical Report, JHF-98-06, School of Applied Science, Nanyang Technological University, Singapore.

Johnson Space Centre's homepage 1998. CLIPS (C Language Integrated Production System). Online document available at <URL: http://www.jsc.nasa.gov/~clips/CLIPS.html>.

Klahr, P., and Waterman, D.A. 1986. *Expert Systems: Techniques, Tools and Applications*. Addison-Wesley Publishing Company.

Kohonen, T. 1990. The Self-Organizing Map. In Proceedings of the IEEE, 78(9), part I, 464-480.

Law, Y.F.D.; Foong, S.W.; and Kwan, S.E.J. 1997. An Integrated Case-Based Reasoning Approach for Intelligent Help Desk Fault Management. *Expert Systems with Applications* 13(4):265-274.

Lees, B., and Corchado, J. 1997. Case Based Reasoning in a Hybrid Agent-Oriented System. In Proceedings of the 5th German Workshop on Case-Based Reasoning, 139-144.

Liu, Z.Q., and Yan, F. 1997. Fuzzy Neural Network in Case-Based Diagnostic System. *IEEE Transactions on Fuzzy Systems* 5(2):209-222.

Muller, N.J. 1996. Expanding the Help Desk through the World Wide Web. *Information Systems Management* 13(3):37-44.

Papagni, M.; Cirillo, V.; and Micarelli, A. 1997. A Hybrid Architecture for a User-Adapted Training System. In Proceedings of the 5th German Workshop on Case-Based Reasoning, 181-188.

Patterson, D.W.R., and Hughes, J.G. 1997. Case-based reasoning for fault diagnosis. *The New Review of Applied Expert Systems*. Vol. 3, 15-26.

Quinlan, J.R. 1986. Induction of Decision Trees. *Machine Learning*, Vol. 1, 81-106.

Quinlan, J.R. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufman Publishers.

Ramden, T.; Krus, P.; and Palmberg, Jan-Ove 1995. Fault Diagnosis of Complex Fluid Power Systems Using Neural Networks. In Proceedings of the 4th Scandinavian International Conference on Fluid Power, 27-29 Sept, Tampere, Finland.

Riesbeck, C.K., and Schank, R.C. 1989. Inside Case Based Reasoning. Lawrence Erlbaum Associates Inc.

Shimazu, H.; Shibata, A.; and Nihei, K. 1994. Case-Based Retrieval Interface Adapted to Customer-Initiated Dialogues in Help Desk Operations. Proceedings of the Twelfth National Conference on Artificial Intelligence, Seattle, WA, USA, Vol. 1, pp. 513-18.

Watson, I.D. 1997. *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. Morgan Kaufman Publishers.