Automated Case Base Creation and Management

Yang, Chunsheng; Orchard, Robert; Farley, Benoît; Zaluski, Marvin

Publisher's version / Version de l'éditeur:

Applications of Artificial Intelligence & Expert Systems (IEA/AIE-2003), 2003

National Research Council Canada
Conseil national de recherches Canada

Institute for Information Technology
Institut de technologie de l'information

NRC-CNRC

Automated Case Base Creation and Management *

Yang, C., Orchard, R., Farley, B., Zaluski, M.
June 2003

* published in Proceeding of International Conference on Industrial & Engineering. Applications of Artificial Intelligence & Expert Systems (IEA/AIE-2003). Loughborough, UK. June 22-26 2003. NRC 45812.

Canada

Automated Case Base Creation and Management

Chunsheng Yang[1], Robert Orchard[1], Benoit Farley[1] , and Marvin Zaluski[1]

[1] National Research Council of Canada, Ottawa, Ontario, Canada
{Chunsheng.Yang, Bob.Orchard, Benoit.Farley, Marvin.Zaluski }@nrc.ca

Abstract

In this paper, we report on a scheme for automated case base creation and management. The scheme aims at reducing the difficulty and human effort required for case creation. This paper provides an overview of the proposed scheme and outlines its technical implementation as an automated case creation system for the Integrated Diagnostic System. Some experimental results for testing the scheme and an interactive tool for evaluating the constructed case base are presented.

Keywords: case-based reasoning, case base maintenance, automated case creation, natural language processing

1. Introduction

Case base creation and management in case-based reasoning (CBR) systems have been recognized as the bottleneck issues that can determine whether a CBR system will be successful or not. To date a great deal of research effort has been devoted to case base maintenance [3][4][5][6][7][9] in CBR systems. This research has focused on a number of crucial issues such as the case life cycle [1], the optimization of the case indices [2] and so on. Some of the earliest case base maintenance works [4] [5] look at the development of maintenance strategies for deleting/adding cases from/to existing case bases. For example, in [4], a class of competence-guided deletion policies for estimating the competence of an individual case and deleting the case from a case base is presented. This technique has been further developed for adding a case to an existing case base [5]. Redundancy and inconsistency detection for case base management in CBR systems has also attracted a lot of attention from researchers [6]. In recent years, some new approaches based on automatic case base management strategies have been published. M.A. Ferrario and B. Smyth [8], introduced a distributed maintenance strategy, called collaborative maintenance (CM), which provides an intelligent framework to support long-term case collection and authoring. To automatically maintain the case base, L. Portinal et al [7] proposed a strategy, called LEF (Learning by Failure with Forgetting [9]), for automatic case base maintenance.

It is perhaps surprising that these works almost exclusively focus on maintaining case bases for runtime CBR systems and collecting cases from the on-line

problem-solving procedures. Relatively little work has focused on automated case creation at an earlier stage, using existing historic maintenance experience that can be collected from past maintenance operational data. In fact, a useful CBR system should provide the ability for a user to automatically create case bases from the recorded historic experience database at the initial stage and to automatically collect or author the cases at the on-line runtime stage. In order to reduce the effort required for case creation and overcome the difficulty of effective creation of high-quality cases, we propose a scheme for automated case creation and case base management that applies natural language processing (NLP) [11] and knowledge discovery technologies. The proposed scheme is presented in detail along with its technical implementation. Some experimental results from testing the effectiveness of the method and a case base evaluation tool are also discussed. The paper is organized as follows. Section 2 presents background information for automated case base creation; Section 3 describes the proposed scheme; Section 4 discusses the technical implementation of the scheme; Section 5 provides details on the tool developed for case base evaluation; and the final section discusses the conclusions.

2. Background Information

CBR is one component of the Integrated Diagnostic System (IDS[1]) [10], which was developed at the National Research Council of Canada. It is used to help refine solutions for aircraft maintenance by retrieving solutions to similar situations from the mechanic's historic experiences that have been stored in a case base. One important piece of data is the snag[2] message. A snag is a transcript of the hand-written notes describing a problem (reported by pilots, other crew or maintenance technicians) and the repair actions carried out to fix the problem. It is composed of well defined, fixed fields describing the date, the location, a unique snag identifier, etc. as well as unstructured free-text describing the problem symptoms, the pieces of equipment involved in the repair and the actions performed on them. Table 1 shows an example of a raw snag message. We can obtain a *clean* snag message (shown in Table 2) by preprocessing the raw message. This clean snag message contains the useful information for case creation. It is possible for someone to create a potential case (shown in Table 3) for the case base by combining the information in the cleaned snag message with information in the Fault Event Object (FEO) database. FEOs are created in the IDS runtime system that monitors the status of the aircraft. Onboard diagnostic systems record possible problems in the form of failure (FLR) and warning (WRN) messages that are delivered in real-time to the IDS system. These messages along with messages generated by the pilots are grouped according to the time they arrive and their relationship to each other (as determined by the aircraft troubleshooting manual) to form an FEO. This grouping of messages represents a set of symptoms that describe a potential or real problem. By matching a snag message to an FEO one can craft a case that describes the problem, identifies the symptoms present for this problem and shows

---

[1] *IDS is an applied artificial intelligent system that supports the decision-making process in aircraft fleet maintenance.*

[2] *A snag is a common term for an equipment problem in the aviation area. It is a record of the problem and the repair action.*

the repair action that was taken to fix the problem. By monitoring the aircraft after the fix is applied one can then determine whether the fix was successful or not (i.e. did the problem recur or not).

Table 1: An example of the raw maintenance data record

ACFT_MI_SEC:UNNNNNNNNNNNNNNNNNNNNNNYYYYYYYYYYYYYYYNYNNNNNNNNYYNN6615 437820001N**M1003286** 2312 2312ACA01058P28Q0CL6**YUL** ACA0646RT **RMA 27-93-2127 AVAIL. REPEAT E/W "F/CTL ELAC 1 FAU LT" "ELAC 1 OR INPUT OF CAPT ROLL CTL SSTU 4CE1". R 7.** I2000-09-23NNDEFN 0000000000000 0000000000000 0000000000000 00000000000000 40227AC 74577LNNS ORDER AC74577 **1998-01-22 14:07:00**6650 ACFT_MI_ACTN_SEC : INNNNNNNNNNNNNNNNNNNNYYYYYYNN 615437820002000 6889450001Y **REPLACED CAPTAINS SIDE STICK AND TESTED AS PER AMM 27-92-41-501** 42000-09-2506.36.00FIX**YWG** 26525AC 26525NNNNNN 000000000000 AC26525 **1998-01-30 16:00:00**.898990 ACFT_PART_RMVL_SEC:NNNNNNNNNNNNNNNNNNNNNNN66154378200020006889450001000100 01Y0000000010000NNNAC002FD 9W19XFEA 150000000042983622-9852-003 4V792 111AC26525 **1998-01-30 16:00:00**.89916023-80-0100 Y ACFT_PART_INST_SEC:NNNNNNNNNNNNNNYNYYNYNN6615437820002000 688945000 100010001 Y0000000010000NN AC002EA 150000000042983 1467 AC26525 **1998-01-30 16:00:00**.89921023-80-0100 Y

Table 2: A clean snag message obtained from the Table 1

Event Date & Time	*1998-01-22 14:07:00*
Report Station	*YUL*
Snag Number	*M1003286*
Problem Description	*RMA 27-93-2127 AVAIL REPEAT F/CTL ELAC 1 FAULT ELAC 1 INPUT CAPT ROLL CTL SSTU 4CE1*
Fin Number	*222*
Repar Station	*YWG*
Repair Date	*1998-01-30 16:00:00*
Repair Action	*REPLACED CAPTAINS SIDE STICK AND TESTED AS PER AMM 27-92-41-501*

Table 3: A potential case created from Table 2 and FEO database

Case ID	*Case-1*
Case creation date	*2002-04-05*
Event date time	*1998-01-22 14:07:00*
Snag number	*M1003286*
Case quality	*Success*
Success times	*1*
Failure times	*0*
Symptoms	*WRN321 FLR1188 WRN320 WRN340*
Problem description	*RMA 27-93-2127 AVAIL REPEAT F/CTL ELAC 1 FAULT ELAC 1 INPUT CAPT ROLL CTL SSTU 4CE1*
Fin number	*222*
Repar station	*YWG*
Repair date	*1998-01-30 16:00:00*
Repair actions	*Remove/Install (replace)*
Equipment (No)	*27-92-41-501*

3. A Scheme for Automated Case Base Creation

To alleviate the considerable human effort required in CBR applications such as IDS, we propose a scheme for automated case base creation and maintenance. The aim is to extract useful maintenance information for a solution to a problem and related symptoms from the historic maintenance databases, and to create the cases that document these historical relationships by applying NLP, CBR and free-text matching technologies. To describe the proposed scheme, we use the following notations. Let c denote a *case* and CB denotes a case base, then $CB \supseteq (c_1, c_2 ....., c_i, ..., c_n)$. A case c is defined as $c = ((p),(s),(m))$ where *(p), (s)* and *(m)* denote problem attributes (called symptoms), solution attributes to the problem and information for case base management respectively. *(m)* contains all attributes related to case base maintenance including redundancy, inconsistency, positive actions, and negative action. *(p)* could be single symptom or multiple symptoms, and *(s)* could be single action or multiple actions for fixing the problem *(p)*. If SB and FB denote the historic snag maintenance database and the FEO database respectively, then $SB \supseteq (snag_1, snag_2, ...snag_k)$ and $FB \supseteq (f_1, f_2, ...f_l)$. Our task is to create CB from SB and FB.

The scheme, shown as pseudo-code in Figure 1, automates the procedures for case base creation as three main processes:

- Preprocessing snag messages,
- Creating a potential case,
- Maintaining the case base.

The proposed scheme is expected to be suitable for maintenance domains other than aviation as long as they provide historic diagnostic maintenance records in a well-defined data format. We use dynamic attribute definitions for the number and type of attributes in the case. This will make it easier to apply the scheme to other domains. The step in which we preprocess snag message will likely need some adjustment to handle the raw data format for different application domains but the approach remains the same. Following are the details for the aircraft maintenance application domain.

3.1 Preprocessing Snag Messages

The raw snag messages like the one shown in Table 1 are processed to give the clean message as shown in Table 2. The parse is a simple since the various fields of the raw message are in a predetermined order of the fixed size. We extract the date, the place where the fix was done, a unique snag identifier, etc, as well as unstructured free-text describing the problem symptoms and the repair actions. The free-text contains many unnecessary symbols or words. To deal with this, we filter the unnecessary characters (such as '#', '.', '*' and so on) and using a list of "poor single" words, we remove some words as well. The list of poor single words are constructed by analyzing a large set of snag messages to see which ones were not helpful in matching the unstructured text FLR and WRN messages. For example, the free-text of problem description obtained from the raw snag message, *RMA 27-93-2127 AVAIL. REPEAT E/W "F/CTL ELAC 1 FAULT" "ELAC 1 OR INPUT OF CAPT ROLL CTL SSTU 4CE1". R 7.* after processing, results in *RMA 27-93-2127*

AVAIL REPEAT F/CTL ELAC 1 FAULT ELAC 1 INPUT CAPT ROLL CTL SSTU 4CE1, as shown in Table 2.

The free-text of the "repair action" field will be processed using NLP techniques discussed in the next section.

```
SchemeForAutomatedCaseCreationAndManagement (CB, SB, FB)
BEGIN
        FOR each snag_i in SB DO
        BEGIN
                // Preprocess the raw snag message
                Get-snag-data (snag_i );
                Filter-and-clean-free-text(snag_i);
                // starting to create a potential case from snag message
                IF not Identify-symptoms(input=FB, snag_i; output =(p) );
                THEN continue;
                ELSE
                   IF not NLP-identfy-solutions(input=snag_i, output=(s));
                  THEN continue;
                  ELSE
                        Create-potential-case(input=(p),(s); output = c_tmp);
                        IF not check-positive-case(input=FB; output=c_tmp);
                        THEN negative-case(c_tmp);
                        ELSE positive-case(c_tmp);
                        ENDIF
                 EENDIF
                ENDIF
                // starting case base management process
                FOR each case_j in CB DO
                BEGIN
                        IF not Detect-Redundancy-Inconsistency(c_j, c_tmp);
                        THEN add-new-case(CB,c_tmp);
                        ELSE maintain-case-bases(c_j, c_tmp );
                        ENDIF
                ENDFOR
        ENDFOR
END
```

Figure 1: The scheme for automated case base creation and management

3.2 Creating a Potential Case

This part of the scheme requires four main steps. The first step, symptom identification, is to identify the symptoms for the problem *(p)*; the 2nd step, repair action identification, is to find the solution information *(s)*; the 3rd step, case template creation, is to create a potential case C_{tmp}; and the 4th step, case quality identification, is to determine if the case is positive (a successful solution) or negative (an unsuccessful solution) by checking to see if the symptoms disappeared after the solution *(s)* is applied to the problem *(p)*. If the symptoms disappeared we say the case is positive, otherwise the case is negative. In CBR applications, both positive and negative cases are useful for decision-making

support. It is as important to know what will not fix a problem as to know what will fix it.

The symptom identification module finds a set of symptoms in the FEO database that match the problem described in the snag message. Identifying the symptoms for the problem is done using a free-text matching approach because the content of FLR and WRN message is described in formal (predetermined) text while the problem description in the snag message is unstructured free text. To match such free text to the formal text of the diagnostic messages, we use an N-gram algorithm. N-gram matching refers to a fragment of N consecutive letters of a text phase. For a given text phase of length ***L***, there are $L-N+1$ N-grams. Such matching algorithm helps to reduce the impact of misspelling, abbreviations and acronyms. After considering the trade-off between the algorithm performance and matching accuracy, we selected N to be 3 (tri-gram matching). For example, in the tri-gram matching algorithm, the text word "***diagnose***" could be disassembled into 6 tri-grams: $\{dia, iag, agn, gno, nos, ose\}$. If a text phase, "***diagnose****"* is matched to the misspelled one, "***diagnoes****",* the tri-gram will match them as two similar text phases.

The repair action identification module, called *NLP-identify-solutions* in the pseudo-code of Figure 1*,* extracts repair action and equipment information from the snag message using NLP techniques [11] [12]. In general, the free text of the repair action description of the snag message contains one or more "sentences" with extensive use of acronyms and abbreviations, omission of certain types of words (such as the definite article), and numerous misspellings and typographic errors. Extracting the required specific information, namely the pieces of equipment involved in the repair, the actions performed on the equipment (replace, reset, repair, etc.), and the results of those actions, from this free text is a typical natural language understanding procedure, consisting of the following main steps:

- dictionary and acronyms database creation,
- preprocessing of the free text message and morphological analysis,
- grammar and parsing , and
- semantic interpretation.

To carry out the NLP process for understanding the free-text maintenance messages, we have to build up a lexicon, which contains the words, acronyms and abbreviations used in the particular domain, and we have to create a knowledge base for interpreting these messages. For aircraft fleet maintenance, the lexicon and knowledge base were built from information in the snag databases [12]. The quality of the lexicon and knowledge base will directly affect of the ability to create good cases from the historic maintenance data.

In the natural language understanding procedure, the unstructured free text that describes the repair action is first preprocessed to determine the nature and properties of each word and token against the dictionary and acronyms database. Then the sequence of morphologically analyzed items is syntactically analyzed with a parser and checked against a grammar that describes the patterns of valid propositions. Finally the result of the syntactic parsing is semantically interpreted to generate the class of repair action and the equipment on which the action is performed. For example, the free-text that describes the repair action in the snag

message, **" *#1 EIU replaced*"**, is analyzed as follows: (1) If the part name is not found in the Airbus Illustrated Parts Catalog (IPC), part name is ***EIU #1*** and repair action is ***REPLACE.*** (2) If the part name is found in the IPC[3], the following values are assigned to the potential case, i.e. part name is ***EIU,*** part number is ***3957900612***, repair action is ***REPLACE,*** and part series number is ***3-25-8-2-40D*** ***(***detailed in [12]).

A new potential case is created by the case template creation module using the symptoms and repair actions extracted from the previous modules. Then the case quality identification module checks this case to determine if the symptoms related to the problem have disappeared or not during a period of time (window size) after the repair actions were taken. The window size is set by aircraft fleet maintenance requirements. We assume that if the symptoms of the problem disappear for the specified period (window size) that the repair was successful and the case is labeled as a positive case, otherwise it is labeled as a negative one.

3.3 Maintaining the Case Base

The case base maintenance process implements the basic functions for case base management. The first set of functionality includes detecting any redundancy or inconsistency for the potential case against the existing case base. In effect we determine whether this case is similar to cases within existing case bases or not. The inconsistency detection function also helps to detect historic data that may contain conflicting information for the same problem over time. The second set of functionality involves adding a new case to the case base, updating an existing case in the case base, deleting a case and merging multiple cases into a new case. If a potential case is new, it will be added to the case base and the case base management information will be refreshed. If it is similar to an existing case, we have to modify the existing case by updating the case management information *(m)* or merge them into a new case. For example, if we detected a similar case (c_i) in the existing case base against the potential case c_{tmp}, i.e. $(p)_i \cong (p)_{tmp}$ and $(s)_i \cong (s)_{tmp}$, then $(m)_i$ will is updated to reflect the effect of the repair action applied to the problem. If c_{tmp} is a positive case, then we increase the count of successful repair actions of $(m)_i$ otherwise we increase the count of unsuccessful repair actions of $(m)_i$.

4. Implementation and Experimental Results

The proposed scheme has been applied to the IDS project to create the cases from the aircraft fleet maintenance historic data (snag database) and the FEO database. We developed a Java-based CBR engine, and an automated case creation system (ACCS) which incorporates the CBR engine, natural language processing, free-text matching, and database technologies. The goal of the ACCS tool is to demonstrate that we can create an set of cases in an automated way that will enhance the decision making process of the maintenance technicians.

---

[3] IPC is a catalog of all the parts of the particular type of aircrafts. It describes the component makeup of an airplane with a list of all the part number and a keyword designating the part.

The ACCS, as shown in Figure 2, identifies the three main components: snag message preprocessing, potential case creation, and case base maintenance. The potential case creation component contains the four modules: symptom identification, repair action identification, case template creation and case quality identification. The case base maintenance component is supported by the Java-based CBR engine and the redundancy and inconsistency detection modules. We have used JDK2.0, Oracle7.0, and Prolog as development environments.

To test the effectiveness of our automated case creation, experiments were carried out using the ACCS with the aircraft fleet maintenance database from Air Canada and the FEO database created by our own IDS system. We used ACCS to create the cases from the 359 clean snag messages that were formed from January 1, 1998 to January 31, 1998 and the FEO database. The ACCS created 35 cases in 2 minutes. It is interesting that not each clean snag message contains the fully useful information for creating a potential case because ether the symptoms are not found from the FEO datable, or the fix does not exist the snag message. In the 35 constructed cases, 21 cases are created from single snag message and consist of positive case or negative case; 14 cases are linked to multiple snag messages, which recorded similar resolutions for similar problems or the same problem, and they contain information on the successful or failed repair action by the attributes of case base management (m). From the statistic result, 45 snag messages from 359 snag messages were linked to those 14 cases. Totally, 66 clean snag messages among 359 snag messages were useful for creating the cases.

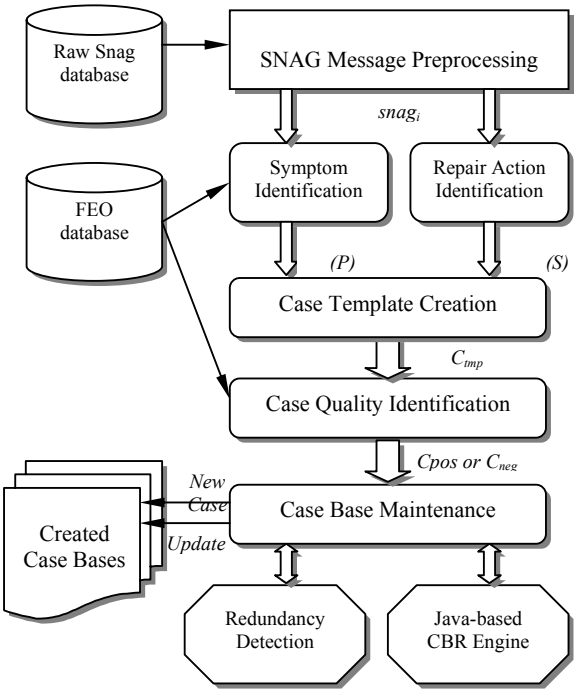

Figure 2: ACCS system implementation

5. Case Base Evaluation

Before the cases that have been automatically created are incorporated into CBR applications such as IDS, they must be validated by either a knowledge-based system or domain experts. The validation of cases by a knowledge-based system is a very difficult task and requires rich domain knowledge from experts. Therefore, we are providing the domain experts with a supporting tool to help them validate the case base. This interactive environment allows the user to browse the constructed case base and evaluate cases one by one, checking the original snag message, problem symptoms, problem description, repair action and so on. It also provides the basic support for the user to do case base maintenance operations such as modifying a case, deleting a case and merging multiple cases. Figure 3 shows the main window of the validation tool.

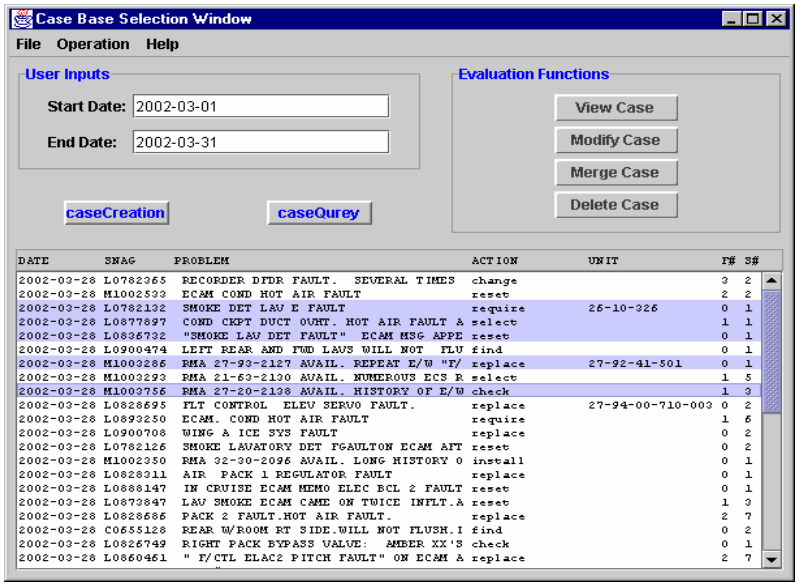

Figure 3: The main window of the case base evaluation tool

6. Conclusions

In this paper, we first presented the proposed scheme for automated case base creation and management in CBR systems, then we briefly described the system implementation, an automated case creation system for IDS (an application in the aircraft maintenance domain) and discussed the experimental results. We also presented an interactive tool for domain experts to evaluate the case base. From the experimental results, it can be pointed out that the proposed scheme is feasible and effective for automated case base creation and management in CBR systems and it can significantly reduce the human effort required for case creation. Currently the ACCS system is creating case bases off-line. The constructed case base will be

incorporated into IDS to provide the CBR support for aircraft fleet maintenance. The proposed scheme can be applied to other maintenance application domains by implementing specific preprocessing of snag messages and setting up a special lexicon and knowledge base corresponding to those application domains. Future work would be to integrate the system into IDS as an on-line component. This will be beneficial in providing a system for collecting and authoring the cases from real-time maintenance procedures.

Acknowledgements

Many people at NRC have been involved this project. Special thanks go to the following for their support, discussion and valuable suggestions: M. Halasz, R. Wylie, and F. Dube. We are also grateful to Air Canada for providing us the aircraft fleet maintenance data.

References

1. M. Minor and A. Hanft, *The Life Cycle of Test cases in a CBR System,* In Proceedings of Advances in case-based Reasoning: 5th European Workshop, EWCBR 2000, Trento, Italy, Setp. 2000, pp.455-466
2. D.W. Aha and L.A. Breslow, *Refining Conversational Case Libraies*, In Proceedings of Int'l Conference of Case-based Reasoning, RI, USA, 1997, pp.267-278
3. B. Smyth, *Case-Based Maintenance*, In Proceedings of the 11th Intl. Conference on Industry and Engineering Applications of AI and Expert Systems, Castellon, Spain, 1998.
4. B. Smyth, *Remembering to Forget: A Competence Persevering Deletion Policy for Case-Based Reasoning Systems*, In Proceedings of the 14th Intl. Joint Conference on AI, Morgan-Kaufmann, 1995, pp.377-382
5. J. Zhu and Q. Yang, *Remembering to Add: Competence Persevering Case-Addition Policy for Case-Base Maintenance*, In Proceedings of the 16th Intl. Joint Conference on AI, Stockholm, Sweden, 1999, pp.234-239
6. K. Racine and Q. Yang, *On the Consistency Management for Large Case Bases: The Case for Validation*, In Proceedings of AAAI-96 Workshop on Knowledge Base Validation, August 1996
7. L. Portinale and P. Torasso, *Automated Case Base Management in a Multi-model Reasoning System,* In Proceedings of Advances in case-based Reasoning: 5th European Workshop, EWCBR 2000, Trento, Italy, Setp. 2000, pp.234-246
8. M. A. Ferrario and B. Smyth, *Collaborative Maintenance—A Distributed, Interactive Case-based Maintenance Strategy,* In Proceedings of Advances in case-based Reasoning: 5th European Workshop, EWCBR 2000, Trento, Italy, Setp. 2000, pp.393—405
9. L. Portinale, P. Torasso, and P. Tavano *Speed-up, Quality and Competence in Multi-model Case-based Reasoning,* In Proceedings of 3rd ICCBR, LNAI 1650, Springer Verlag, 1999, pp. 303-317
10. R. Wylie, R. Orchard, M. Halasz and F. Dubé, *IDS: Improving Aircraft fleet Maintenance*, In Proceedings of the 14th National Conference on Artificial Intelligence, Calif, USA, 1997, pp.1078-1085
11. B. Farley, *From free-text repair action messages to automatic case generation*, In Proceedings of AAAI Spring Symposium: AI in Equipment Maintenance Service & Support, Technical Report SS-99-04, Menlo Park, CA,: AAAI Press, 1999, pp.109-118
12. B.Farley, *Extracting information from free-text aircraft repair notes,* Artificial Intelligence for Engineering Design, Analysis and Manufacture, Cambridge University Press 0890-0604/01, 2001, I5, pp.295-305