Expert Systems with Applications 37 (2010) 2427–2438

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A rule-based CBR approach for expert finding and problem diagnosis

Yuan-Hsin Tung [a,c], Shian-Shyong Tseng [a,b,*], Jui-Feng Weng [a], Tsung-Ping Lee [a], Anthony Y.H. Liao [b], Wen-Nung Tsai [a]

[a] *Department of Computer Science and Information Engineering, National Chiao Tung University, Taiwan, ROC*
[b] *Department of Information Science and Applications, Asia University, Taiwan, ROC*
[c] *R&D Supporting Department, Telecommunication Lab., Chunghwa Telecom Co., Ltd., Taiwan, ROC*

ARTICLE INFO

Keywords:
Rule-based CBR
RBR
CBR
Expert finding
Role-based access control
Problem diagnosis

ABSTRACT

It is important to find the person with right expertise and the appropriate solutions in the specific field to solve a critical situation in a large complex system such as an enterprise level application. In this paper, we apply the experts' knowledge to construct a solution retrieval system for expert finding and problem diagnosis. Firstly, we aim to utilize the experts' problem diagnosis knowledge which can identify the error type of problem to suggest the corresponding expert and retrieve the solution for specific error type. Therefore, how to find an efficient way to use domain knowledge and the corresponding experts has become an important issue. To transform experts' knowledge into the knowledge base of a solution retrieval system, the idea of developing a solution retrieval system based on hybrid approach using RBR (rule-based reasoning) and CBR (case-based reasoning), RCBR (rule-based CBR), is proposed in this research. Furthermore, we incorporate domain expertise into our methodology with role-based access control model to suggest appropriate expert for problem solving, and build a prototype system with expert finding and problem diagnosis for the complex system. The experimental results show that RCBR (rule-based CBR) can improve accuracy of retrieval cases and reduce retrieval time prominently.

1. Introduction

It is important to find the person with right expertise and the appropriate solutions in the specific field to solve a critical situation in a large complex system such as an enterprise level application. To obtain the high reliability and availability of the information system in the enterprise environment, the multi-domain architecture has been applied to system design extensively to enhance performance, flexibility and scalability of the information system (Eckerson, 1995; Schussel, 1995). However, it increases both the complexity of the system and the difficulty of problem diagnosis. Moreover, once the complex information system goes wrong, domain experts usually get together to look for solutions to fix the problem as soon as possible. In the enterprise environment, the expert finding and problem diagnosis of complex system is mission critical. In this paper, the customer relationship management system (CRM system) for a telecom company is designed to receive and resolve customer complaints. The CRM system provides 24/7 services for about 3000 daily service calls by approximately 300 online operators. As shown in Fig. 1, the CRM system, a typical multi-domain system for daily operation, connects several component applications to provide services of billing management system, human resource management and network maintenance system. To ensure the CRM system works well in daily operation, experts in different domain (e.g., DBAs, system maintainers, system administrators and developers etc.) should participate in maintaining the system. Therefore, how to utilize domain knowledge and the profiles of experts during problem diagnosis processes to find the right persons to fix the problem of the complex system (multi-domain) has become an important issue.

As shown in Fig. 2, the common scenario of problem diagnosis processes in the multi-domain information system can be represented as a multi-domain architecture, where the experts who have the specific domain knowledge and experiences on the occurred problem are requested to maintain the corresponding layers of complex system. Basically, the problem diagnosis and solution retrieval is an iterative process which should be repeatedly performed stage by stage until the problem is solved. When a system hanging error occurs, the related experts are asked to solve it, and the root cause of problem may be diagnosed by the rule of thumb of domain experts in the first stage, diagnosis phase. Experts can find out the appropriate layer that the causes may lie in; e.g., application table locks, database system process timeout, and physical disk space full may lie in the application layer, the system layer, or the hardware layer, respectively. After identifying the layer of

[*] Corresponding author. Address: Office of the Vice President, Asia University, Taiwan, ROC. Tel.: +886 4 2332 3456ext1088; fax: +886 4 2331 6699.

E-mail addresses: yhdong.cs94g@nctu.edu.tw (Y.-H. Tung), sstseng@asia.edu.tw (S.-S. Tseng), roy@cis.nctu.edu.tw (J.-F. Weng), kelvin.lee@sti.com.tw (T.-P. Lee), liao@mail.isa.asia.edu.tw (A.Y.H. Liao), tsaiwn@csie.nctu.edu.tw (W.-N. Tsai).

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.07.037

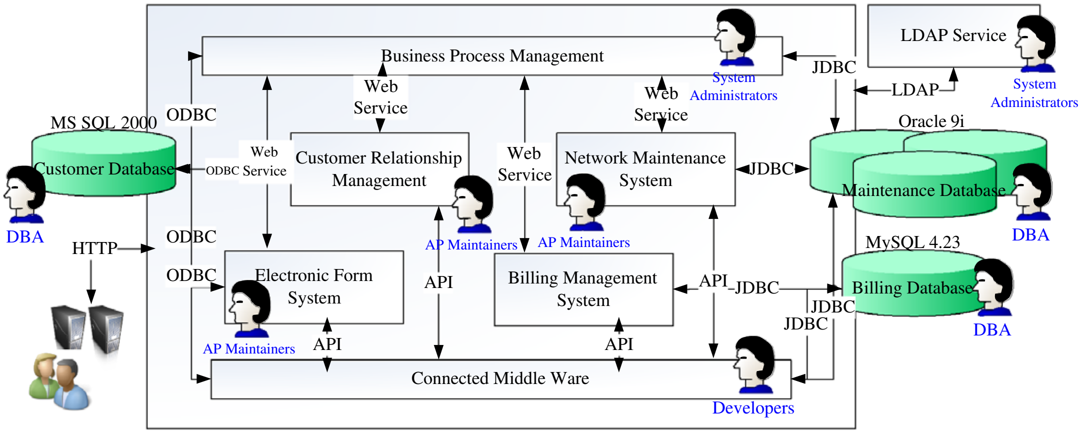


Fig. 1. System architecture of CRM system.

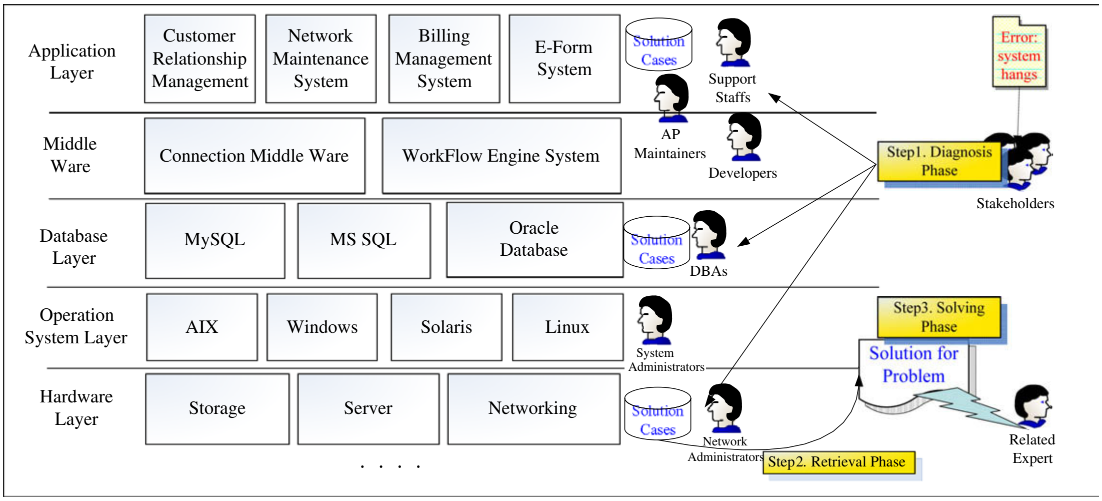


Fig. 2. A scenario of problem diagnosis and solution retrieval when system hanging error occurs.

problem, experts come up with a solution obtained from existing similar solution cases in the second stage, retrieval phase. In the third stage, solving phase, the used solution is found out and domain experts are requested to solve the problem. Based upon the processes of problem solving mentioned above, we can extract the idea that the efficiency of searching can be enhanced by adopting experts' knowledge in diagnosis phase because categorizing similar problem types in advance is a good way to narrow down the searching spaces. Afterward we can locate a problem according to the classified categories in retrieval phase. Finally, based on domain knowledge of personal profile, we can find an appropriate expert to solve the problem. In this paper, we have to face two main challenges: one is how to imitate the problem solving processes of experts to mitigate the loading of experts, and the other is how to incorporate the appropriate domain experts into processes of problem diagnosis.

As we have known, case-based reasoning (CBR) is an approach that solves new problems by retrieving existing successful solutions of similar problems from a knowledge source of cases, the so-called "case-base". CBR has been broadly applied in various areas such as problem diagnosis, solution retrieval, help desk, assessment, decision support, design, and planning (Carswell, Wilson, & Bertolotto, 2002; Spalazzi, 2001). However, the process of case-based reasoning is very time-consuming and the result might not be accurate when the case base is likely a large coarse-grained case base. Searching through the whole "case base" for a solution in a sequential way is rather inefficient. Moreover, it is important to recommend an appropriate expert to solve the problem based on her/his domain knowledge, technical skill, experiences, and so on. Since role-based access control model can be used to solve such requirements of problem diagnosis and solution retrieval, we combine it with a hybrid case-based reasoning approach, the rule-based case-base reasoning (RCBR) methodology, to apply to the high-level knowledge for problem diagnosis and the concrete-level knowledge for solution retrieval. The high-level knowledge which is extracted by rule-based reasoning (RBR) can locate the problem in a specific category, and the concrete-level knowledge can retrieve solution from the specific case base with case-based reasoning (CBR).

In this paper, the problem diagnosis and solution retrieval system based upon a three-phase RCBR framework have been implemented. By using this system, we further construct a prototype system to assist on-duty IT employee in trouble-shooting. The experimental results show that RCBR can improve accuracy of retrieval cases and reduce retrieval time dramatically. The rest of this paper is structured as follows. In Section 2, we depict the preliminary of RCBR methodology. Section 3 describes the architecture of rule-based CBR. Section 4 introduces the system implementation based upon RCBR methodology. In Section 5, the experimental result demonstrates the efficiency of our approach. Section 6 presents the conclusion and proposes future work.

2. Preliminary

In the following, we will brief the preliminaries of the related work.

2.1. Problem diagnosis and solution retrieval

In (Rish et al., 2005), probabilistic reasoning techniques were used to solve the distributed system problem diagnosis. Other related approaches such as applying data-mining algorithm with ontology-based approach to fault diagnosis and analysis (Hou, Gu, Shen, & Yan, 2005) based on neural network to detect the operating machine fault (Zhang, Dai, Zheng, Zhang, & Mu, 2000; Hayashi & Zhang, 2002) were proposed. However, these approaches can not work well without providing appropriate solutions of the problem. Therefore, previous researches provide approximate solutions which are the solutions of similar problems by CBR (Aamodt & Plaza, 1994; Smyth, IEEE Computer Society, Keane, & Conningham, 2001; Wang & Liu, 2004; Chang, Wang, Hu, & Zheng, 2004; Kumar, Gopalan, & Sridhar, 2005; Lambert-Torres, Martins, Rossi, & da Silva, 2003; Gu, Tong, & Agnar, 2005; Hajar & Lee, 2005), but the system performance is poor due to the lack of a proper decision making mechanism.

Dunker (1945) classified the broad sense of trouble-shooting tasks into the following four basic steps: (1) detection, (2) the testing of fault reason, (3) testing, and (4) maintenance and evaluation. By our observation, when system goes wrong, experts usually solve problems in two steps. In the first step, according to the error logs and some system status information, experts use their domain knowledge to reduce the error space. In the second step, they compare the similarity among the current error instances with the existing solution cases which they solved before. Hence, the idea of developing a hybrid problem diagnosis and solution retrieval system based upon RBR and CBR approach is proposed.

2.2. Role-based access control model

The RBAC model (Ferraiolo, Sandhu, Gavrila, Kuhn, & Chandramouli, 2001; Sandhu, Coyne, Feinstein, & Youman, 1996; Enokido & Takizawa, 2008) is known as an ideal access control model for enterprise environment. The main concept of RBAC is to prevent users from accessing company information by direction. RBAC allows us to model the relationships of roles and responsibilities, users and roles. The major advantages of RBAC are the assignment of user and permissions to roles. Change in a user's responsibility or role within an organization can be managed efficiently by assigning her/him a new role and revoking her/his assignment to any previous roles. Access rights are associated with roles in which users are assigned to appropriate roles. The RBAC model proposed by Ferraiolo et al. consists of four basic components: a set of users Users, a set of roles Roles, a set of permissions Permissions, and a set of sessions Sessions as shown in Fig. 3. A role is a collection of permissions needed to perform a certain function within an organization. A user can be a human being or an autonomous agent. A permission refers to an access mode that can be exercised on an object in the system and a session relates a user to possibly many roles. In this paper, to incorporate domain experts into processes of problem diagnosis and solution retrieval, we applied the role-based access control (RBAC) model to our methodology in the assignments of users and responsibilities.

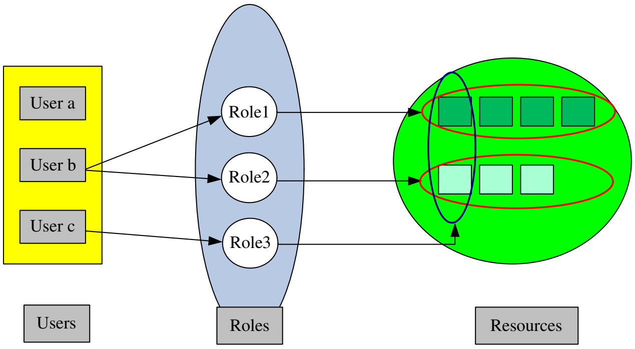


Fig. 3. RBAC elements: users, roles and resources.

2.3. Ontology

In recent years, many researches have been carried out to investigate the use of ontology to represent domain knowledge, and source data can be stored in an unstructured, semi-structured, or fully structured format. Ontology is a knowledge representation model that specifies the concepts and relations of knowledge and has been used in various research domains, such as knowledge engineering, natural language processing, knowledge management, etc., to facilitate knowledge sharing and reuse. Aktas, Pierce, Fox, and Leake (2004) developed a well-defined ontology to support CBR's case representation. Yang and Chen (2006) constructed the organization memory knowledge model with ontology. The ontology will remove the ambiguity, thus it can be used for the searching the browsing. An ontology can be used to represent the concepts and how they are related; it addresses the conceptualizations, no technical knowledge required and mostly with hierarchy of concepts structure. However, in the artificial intelligence area, there are many definitions of ontology. In this paper, we use ontology to define the experts' knowledge of error type.

3. The architecture of rule-based CBR

In this section, we introduce the processes of problem diagnosis and main components of RCBR approach. As shown in Fig. 4, the RCBR approach contains three main phases, knowledge bases constructing phase, rule-based reasoning phase and case-based reasoning phase. In knowledge bases constructing phase, diagnosis rule base is built for problem diagnosis and solution case base is built for solution retrieval. Furthermore, we construct user rule base and user base from user profiles provided by human resource department. Knowledge engineers and domain experts acquire knowledge ontology of system error types and RBAC ontology of expertise. Applying EMCUD and DRAMA technology to the transforming algorithm (Hwang & Tseng, 1990), knowledge and embedded rules of problem diagnosis can be acquired by rule-based format. In rule-based reasoning phase, the error type of query case can be determined by rule inference and then used to narrow down the case space to increase the performance in similarity calculation. After splitting case space into several small case bases by rule inference, the efficiency and accuracy of solution retrieval in the case-based reasoning phase can be improved and then we can retrieve most similar solutions from case bases.

3.1. Knowledge acquisition with ontology generation

Similar to the concept of object-oriented programming, we could treat all the entities in the real world as concepts and it is natural for us to model the world using concepts hierarchy. In knowledge acquisition phase, knowledge engineers acquire error type ontology with domain experts. The ontology is divided into two layers, the abstract layer ontology describes abstract categories of error types, and the concrete layer ontology describes error spaces of the specific domain. In Fig. 5, knowledge ontology of

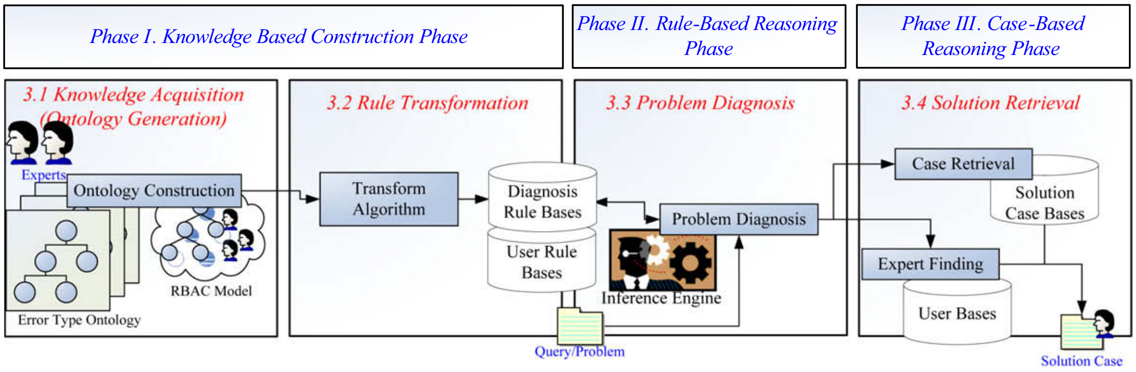


Fig. 4. The architecture of RCBR approach.

error spaces is excerpted from the oracle database that is designed by cooperation of the domain experts and knowledge engineers. The knowledge classes that include oval and rectangularity represent the concepts from domain experts. As shown in Fig. 5, the knowledge class "oracle DB", consists of two KCs (knowledge classes), "instance" and "database", and the rectangle stands for knowledge class of the error type of cases. Two types of relationships are used in error type ontology to describe relationships of problems. The first one is the "trigger" relationship between concepts. Some rule class is triggered when some specific conditions are satisfied. It means that a problem may be transformed into another problem. For example, "system error" can transform to "databases error" when the root cause of error is identified in DB layer. The second one is "acquire" relationship, which could be used to describe the sub-problem may be solved by acquiring another rule class. For example, a "control file error" may acquire the expertise of "DB diagnosis".

In addition, we model the relationships of knowledge classes and domain experts by modified RBAC model with expert profile that contains expertise, privileges and schedule arrangements. As shown in Fig. 6, we create the technical role-hierarchy ontology to reflect the physical technique map of experts' expertise and professional experiences. We link error category of error type ontology to corresponding role class of role-hierarchy ontology by relationship "Acquire". For example, to solve the "DB crash" error, we may need expertise of "DB design", "DB diagnosis" or "SQL tuning". In the modified RBAC model, we map user to corresponding concept classes with domain knowledge, schedule arrangements and privileges. Peter and Dan are related experts in expertise "DB diagnosis". Afterward we can recommend corresponding experts by considering how the user profiles are related to the requested problem. In user base, work responsibility and schedule arrangement are stored in role-to-permissions table and role-to-user table in user databases as additional information for system.

The relationships of roles and permissions are illustrated in Table 1, RPT. The technical roles on the horizontal edge of Table 1 are excerpted from technical role hierarchy. The value in the matrix stands for responsibilities of roles to reflect assignments of on-duty experts, PR is "primary responsibility", SR is "secondary responsibility", and N is "notify".

The relationships of roles and permissions are illustrated in Table 2, role-to-user table, RUT. Each element of the row represents a role and each element of the column represents a user. Also, the grids could be filled up with schedule classes, and schedule class S is a set of schedule arrangements of user, $S = \Sigma s_i$. Moreover, schedule arrangement s_i is a duration expression [begin_time, end_time], e.g., [01/01/2008 08:00:00, 01/31/2008 17:30:00], it means that the role is authorized to work in time period between 8:00 am and 17:30 pm.

3.2. Ontology-to-rule algorithm

By acquiring knowledge from experts, error type ontology describing the relationships among system modules, system

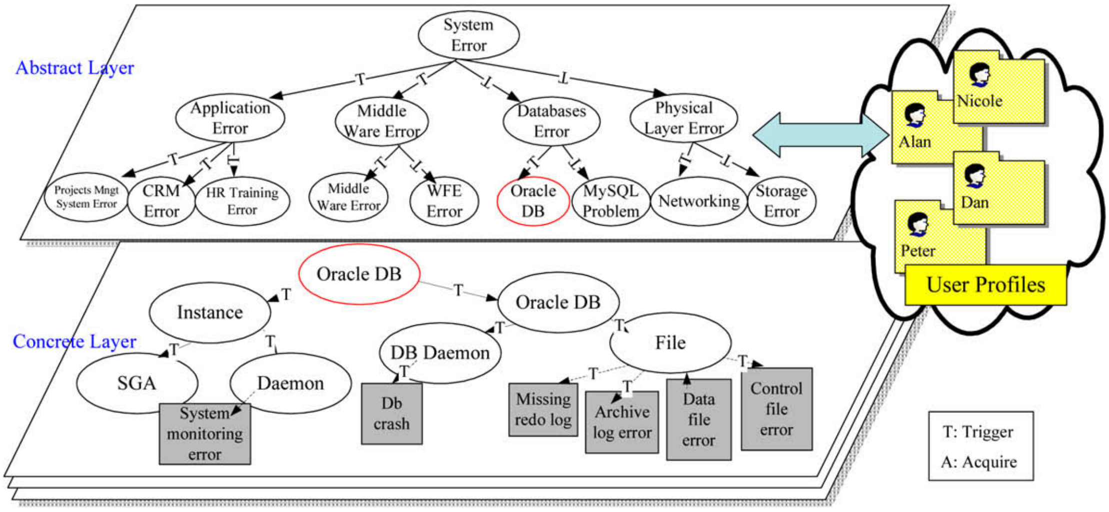


Fig. 5. The error type ontology in oracle database and rule-based knowledge representation.

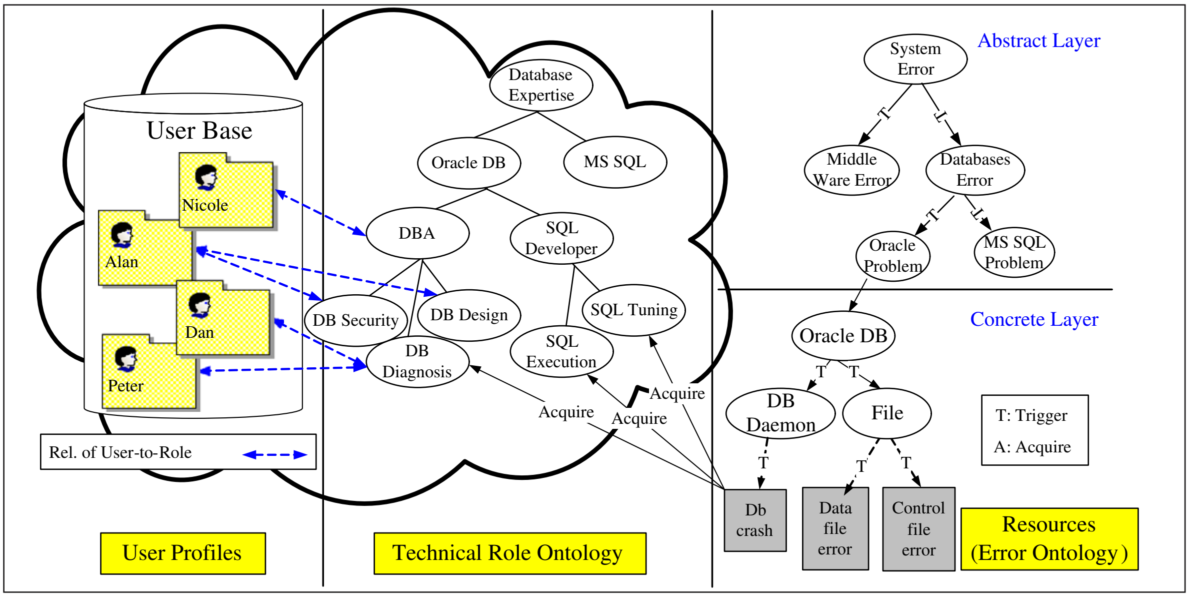


Fig. 6. Role-based access control model for expert responsibilities assignment management.

Table 1
Example of a role-to-permissions table, RPT.

	DB design	DB diagnosis	DB security
Nicole	*N*	*N*	*N*
Alan	*PR*		*SR*
Dan		*PR*	
Peter		*SR*	
...	...		

PR: primary responsibility, *SR*: secondary responsibility, *N*: notify.

Table 2
Schedule arrangement in user-to-role table, URT.

	DB design	DB diagnosis	DB security
Nicole	S_{n1}	S_{n2}	S_{n3}
Alan	S_{a1}		S_{a2}
Dan		S_d	
Peter		S_p	
...	...		

$S = \Sigma s_i$.

characteristics, system applications, and system status could be constructed. The ontology owns “trigger” and “acquire” relationships between rule classes depending on rule class features, where NORM (new object-oriented rule-base model) (Tsai, 2002; Tsai, Tseng, & Wu, 1999; Wu, 1999) is used as knowledge representation and EMCUD is applied to elicit embedded meaning of the error type.

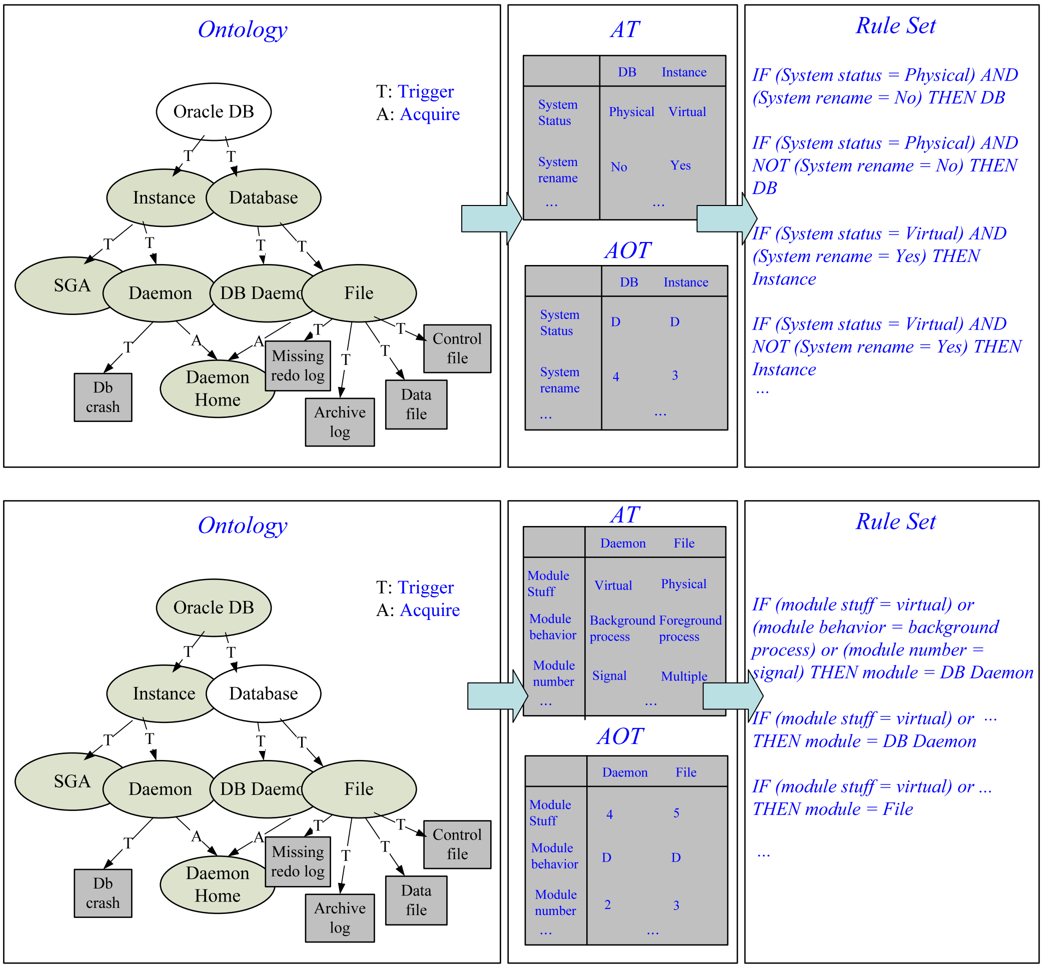
Algorithm 1. Ontology-to-rule algorithm

```
Input: Error Type Ontology, RBAC Ontology.
Output: Error Diagnosis Knowledge Class with facts and rules
    Step 1. Transfer Error Type Ontology concepts into
    Knowledge Classes: for each ontology concept, we
    could transfer the concept into the KC
    Step 2. Choose exemplary attributes that could characterize the domain
    Step 3. Define or identify the relationships between the
    KCs. For each relationship T:
        Step 3.1. Interview domain experts to build the
        corresponding attribute tables, AT and AOT, for
        each knowledge classes
        Step 3.2. Acquire domain experts' inference with
        AT and AOT, and generate rules with facts
        Step 3.3. Generate the Certainty Factor of the rule
    Step 4. For each KCs, if the KC has corresponding Technical Role. For each relationship T:
        Step 4.1. Acquire domain experts' inference with
        AT and AOT, and generate rules with facts
        Step 4.2. Generate the Certainty Factor of the rule
    Step 5. Verify the rules by domain experts and modify
    the rules if needed
```

The ontology-to-rule algorithm which can generate rules from knowledge ontologies is illustrated as follows. Acquisition table (AT) and attribute ordering table (AOT) of EMCUD (Hou et al., 2005) describe the relationships of objects and facts, the objects are put in columns, and facts are put in rows to build objects values and relationships. AT is a repertory grid of multiple data types to identify the relationships of objects and attributes. The corresponding object and fact have a value to identify the value of the object feature. Besides, relative importance of each attribute to each object is represented as attribute ordering table (AOT). In AOT table, D stands for dominate the relationship, X represents no relationship, and Integer represents the strength of relationship (from 1 to 5, 5 is strongest relationship). We apply ontology-to-rule algorithm to transform knowledge ontology to rule sets, and the rules are generated by AT and AOT. When algorithm transforming the knowledge into the rules, the rules are generated by the format: if <condition> then<result>.

Example 1. (Ontology-to-rule:)

In this example, rules are generated from root class “oracle DB” by ontology-to-rule algorithm. In the left part of Fig. 7, the ontology is excerpted from oracle knowledge ontology. The root class “oracle DB” in ontology contains two classes, “instance” and “database”. Based on the features of the class “oracle DB”, we acquire objects and attributes from domain experts and construct AT and AOT in the middle part of Fig. 7. Objects in columns contain “database” and “instance”, and facts in rows include “system



AT

	DB	Instance
System Status	Physical	Virtual
System rename	No	Yes
…		…

AOT

	DB	Instance
System Status	D	D
System rename	4	3
…		…

Rule Set

IF (System status = Physical) AND (System rename = No) THEN DB

IF (System status = Physical) AND NOT (System rename = No) THEN DB

IF (System status = Virtual) AND (System rename = Yes) THEN Instance

IF (System status = Virtual) AND NOT (System rename = Yes) THEN Instance
…

AT

	Daemon	File
Module Stuff	Virtual	Physical
Module behavior	Background process	Foreground process
Module number	Signal	Multiple
…		…

AOT

	Daemon	File
Module Stuff	4	5
Module behavior	D	D
Module number	2	3
…		…

Rule Set

IF (module stuff = virtual) or (module behavior = background process) or (module number = signal) THEN module = DB Daemon

IF (module stuff = virtual) or … THEN module = DB Daemon

IF (module stuff = virtual) or ... THEN module = File

…

Fig. 7. Rule generation from rule class oracle DB.

status" and "system rename". The input values physical, virtual, no, and yes in AT are listed from top to bottom, left to right in sequence to describe the feature values between each object and fact. Input values are D, D, 4, 3 in AOT to describe the relationships strength between each object and fact.

Each meaning-embedded rule extracted by EMCUD has a certainty factor (CF), which is based upon a general repertory-grid-analysis method. For the inference process of each rule, the result may be affected by the rules in child knowledge objects. In other words, the CF of the inferred rule may be affected by the CFs of the rules in child knowledge objects. Therefore, a new formula of calculating CF based upon hierarchical grids is defined as follows. Since embedded rules with weak acceptable CF values (the CF values below a user defined threshold) usually mean domain experts might lack strong confidence, objects matching weak embedded rules may be the candidates of new evolved objects.

Each embedded rule is assigned a certainty sequence (CS), the sum of each AOT values of the ignored attributes, and the CF calculated by formula (1) which is between 0 and 1 can represent the degree of certainty for each embedded rule. Each of them is assigned a CF between 0 and 1, and the CF value approaching to 1 means more important.

$$CF(R_i) = UB(R_a) - \left\{\frac{CS(R_i)}{MAX(CS_i)} \times (UB(R_a) - LB(R_a))\right\} \qquad (1)$$

The R_a is original rule, R_i and $CS(R_i)$ are the ith embedded rule of the object and the CS values, respectively. The $MAX(CS_i)$ is the maximum CS value of the embedded rules generated from object. To decide the CF of each embedded rule R_i, the upper bound $(UB(R_a))$ and the lower bound $(LB(R_a))$ CF values of the object have been firstly defined for accepted embedded rules. Accordingly, CF values of each rule can be automatically determined by the mapping function, formula (1). Therefore, the useful embedded rules with corresponding CF values could be used to cover more uncertainty cases.

In the right part of Fig. 7, the rules generated from AT and AOT are shown as follows.

$$\begin{aligned}
&IF\ (System\ status\ =\ Physical)\ AND\ (System\ rename = No)\\
&\qquad THEN\ Oracle\ DB,\ \ CF = 0.8\\
&IF\ (System\ status\ =\ Physical)\ AND\ NOT(System\ rename = No)\\
&\qquad THEN\ Oracle\ DB,\ \ CF = 0.4\\
&IF\ (System\ status\ =\ Virtual)\ AND\ (System\ rename = Yes)\\
&\qquad THENInstance,\ \ CF = 0.4\\
&IF\ (System\ status\ =\ Virtual)\ AND\ NOT(System\ rename = Yes)\\
&\qquad THEN\ Instance,\ \ CF = 0.6
\end{aligned}$$

3.3. Rule-based error type inferring for problem diagnosis

Rule is a natural knowledge representation, in the form of the "If<condition> Then<result>" structure and rule-base system (RBS) is popular for real applications among expert systems. RBS has many advantages (Reichgelt, 1991). The first is naturalness of expression since experts rely on rules rather than textbook

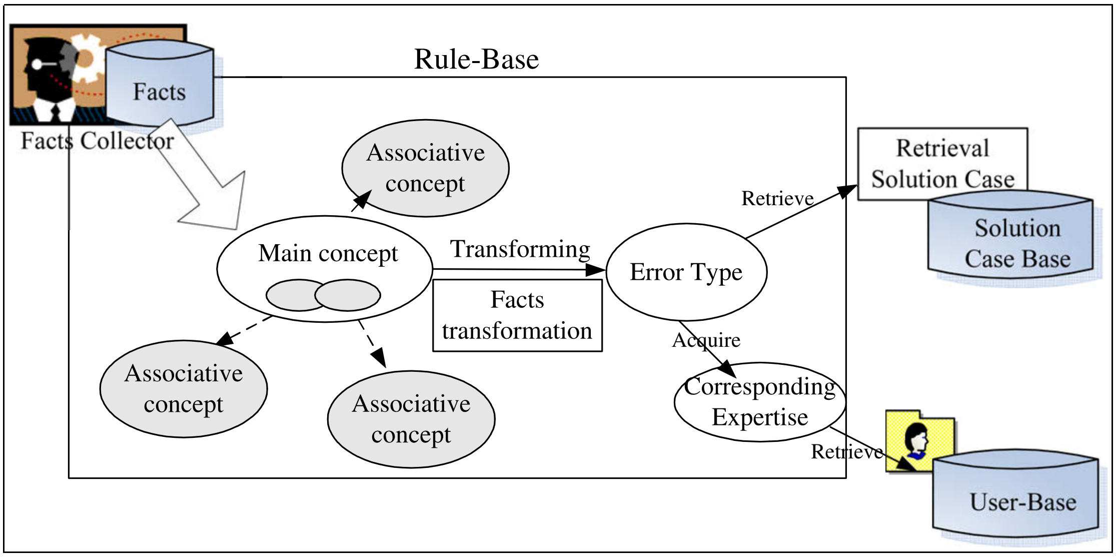


Fig. 8. The behavior of pondering over known information in rule base.

knowledge. The second is modularity that permits RBS easy to construct, to debug, and to maintain. Restricted syntax and ability of explanation are also the advantages of RBS. Consequently, we applied rule-based reasoning approach to problem diagnosis processes for the representations of experts' knowledge.

As shown in Fig. 8, when facts are collected through sensors or other input sources, the facts will be inferred from a specific concept in a domain and other three concepts can be associated according to their relationships. Nevertheless, people may not consider all relevant knowledge at the same time since too much effort is required to solve the problem. Some inference skills are widely used in human thoughts to improve the performance of knowledge inference. The inference process for problem diagnosis is described as follows. The first step is to select a rule base from multiple rule-bases. Because a knowledge system cannot contain all types of domain knowledge, it is necessary to specify a knowledge domain before inference. The second step is to collect the facts and specify a knowledge class (KC) (Lin, Tseng, & Tsai, 2003) containing the corresponding control knowledge for the problem to be solved. According to the specified KC, the inference engine will perform the reasoning process. Finally, interesting and useful information can be obtained from final fact value. Furthermore, the order of fired rules is decided by *CF* values, and the lower priority rules have weak *CF* values. After the inference processes, the error type of the problem can be identified.

3.4. Case-based reasoning for solution retrieval

After the error type of the problem is diagnosed, we retrieve the solution from corresponding case base with case-based reasoning approach. Case-based reasoning (CBR) is an approach that solves a new problem by recalling a previous similar situation and reusing information and knowledge of that situation. A process model of the CBR cycle may be described by the four processes: RETRIEVE the most similar case, REUSE the information and knowledge in that case to solve the problem, REVISE the proposed solution, and RETAIN the parts of this experience which it's likely to be useful for future problem solving (Chen et al., 2002). A flexible combinatorial strategy makes RCBR possible to solve the multi-domain problems without the need for huge case bases of complex mechanisms. Searching through the whole "case base" for a solution in a sequential way is rather inefficient. Therefore, the efficiency of searching can be enhanced by adopting categorized cases base in such situation since categorizing similar problem types in advance is a good way to narrow down the searching spaces. Based on the classified categories, we can locate a solution retrieval problem according to experts' experiences and only the required attributes are adopted. By searching solution cases in split case spaces, we can increase efficiency of inference in case-base reasoning.

Generally speaking, in typical retrieval systems, information is retrieved by a search engine in response to submitted queries. Traditionally, a query is represented as a set of keywords that are used to specify the intended information, and almost all search engines treat the search keywords equally. However, it is sometimes not exact what the users want. To improve precision and efficiency of retrieval, we represent each categorized document with a specific keyword set. Based upon the characteristics of separate error spaces, we define the case features to represent the solution cases of each error space. Each error space has its own features set, called local solution case features set, and the universal solution case features set is the union of all local feature sets. With local feature set, we can reduce keyword set for solution representation and compare the similarity between query and cases using case-based reasoning for solution retrieval.

Definition 1. The universal solution case features set (Σ)

$$\sum = \bigcup_{i=1}^{n} F_i$$

where n is number of error spaces, Σ is universal solution case features set, and Fi is local solution case features set.

Example 2. In the case bases, the original solution documents of error instances obtained from the experts and technical forums are retained as the attribute-based solution cases with attributes error type, subject, module, version, platform, publisher, date, and solution statement as described in Table 3. It is the example case of "redo log error", and the case is represented as case vector by local solution case feature of error type "redo log error".

Based upon local solution case feature of "redo log error", the solution case can be represented as case vector.

"Dropping redo logs not possible" Vector = {"redo log error", "dropping redo log", "8.1.7", "solaris", "online redo log", "corrupt

Table 3
Example case of dropping redo log not possible.

Attributes	Description
Error type	Redo log error
Subject	Dropping redo logs not possible
Application	File
Version	8.1.7
Platform	Solaris
Description	Could not drop the redo logs which may be needed for instance recovery The online redo logs could not be dropped if: 1. There are only two log groups 2. The corrupt redo log file belongs to the current group
Solution	The error ORA-1624 will be produced, since an online redo log file with status = CURRENT or status = ACTIVE in v$log could not be cleared. The command erases all data in the logfile Please note that ’alter database clear logfile’ should be used cautiously. If no archived log was produced, then it is impossible to conduct a complete recovery. Perform a backup immediately after completing this command

redo log file”, “ORA-1624”, “status = current”, “status = active”, “logfile”, “alter database clear logfile”,...}

The attributes subject, application, platform and version are used to compute the similarity between query and case in case base according to formula (2).

$$S(q,c) = \alpha S_s(q,c) + (1-\alpha)S_a(q,c), \tag{2}$$

where $S(q,c)$ is the similarity between query and case, $S_s(q,c)$ is the similarity between query and case in subject and problem description, $S_a(q,c)$ is auxiliary similarity for the application that contains application, platform, and version, and α is a control variable. The control variable α can be decided by domain experts. The similarity $S_s(q,c)$ is defined in Formula (3), and the auxiliary similarity $S_a(q,c)$ is defined in formula (4) and formula (5).

In formula (3), N_q is subject keyword set of the given query, and N_c is subject keyword set of a case. As subject is the description of the solution case, meaningful keyword list is extracted from “subject” and “description attributes”.

$$S_s(q,c) = \frac{2|N_q \cup N_c|}{|N_q| + |N_c|} \tag{3}$$

$$S_a(q,c) = \frac{1}{n}\sum_{i=1}^{n} S_a^i(q,c) \tag{4}$$

$$S_a^i(q,c) = \frac{D_{\max}^i - D_{qc}^i}{D_{\max}^i} \tag{5}$$

Example 3. The similarity of formula (3) is computed based upon the set of keywords of cases. Assume there are a query and two cases, Query = {timeout, space, resource, enqueue}, Case 1={timeout, space, management, resource, In queue}, and Case 2={resource, busy, NOWAIT, speified}. The Query is similar to Case 1, since they have the same set of keywords (timeout, space, and resource). Based upon Formula (3). Similarity of Query and cases are shown as below.

$$\text{Similarity of Case 1: } S_s(q,c_1) = \frac{2|N_q \cup N_c|}{|N_q|+|N_c|} = \frac{2\times|3|}{5+4} = 0.66$$

$$\text{Similarity of Case 2: } S_s(q,c_2) = \frac{2|N_q \cup N_c|}{|N_q|+|N_c|} = \frac{2\times|1|}{5+4} = 0.22$$

4. System implementation based on RCBR

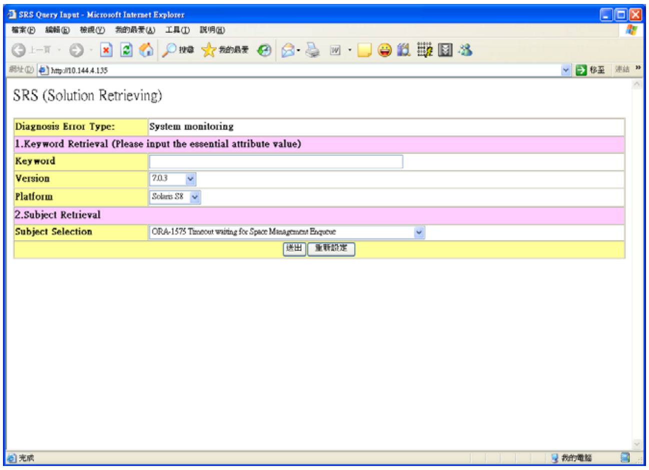
4.1. System implementation

Based upon RCBR approach, we have constructed the prototype system of knowledge management for IT employees, called Solution Retrieval System (abbrev. SRS). SRS contains two main modules, problem diagnosis module (rule-based reasoning) and solution retrieval module (case-based reasoning). The SRS system is a help desk platform to provide problem diagnosis and expert suggestion for IT employees and beginner employees. When problem comes, IT employee can diagnose error type of the problem and retrieve the appropriate solution by SRS easily. We applied SRS system to CRM system that contains several sub-systems and complex system architectures as shown in Fig. 1. In SRS, users can input query case by keyword, system logs or error description (Chang & Hsieh, 2004; Wai & Lau, 2003) and receive the response with the diagnosed error type and solution case from the system, as shown in Fig. 9. In the following paragraph, we will describe the operation of SRS with the case of “database pending error”.

After submitting the query, the solution lists will be displayed in descending order of the similarity. Each solution contains subject, brief solution description, similarity, and suggested expert, as shown in Fig. 10. And users can choose the desired solution case according to similarity. Eventually, the solution will show the know-how and assist users step by step to solve system problems efficiently, as shown in Fig. 11, and the suggested user profile is shown in Fig. 12.

4.2. Case maintenance

Case maintenance includes case retention and case revision. The solution case contains multi-attributes based upon characteristics of problem domain, e.g., in oracle DB error type, the attributes in Table 4 are chosen as the representation of solution cases. The descriptions of oracle DB error type are shown in Table 5. In

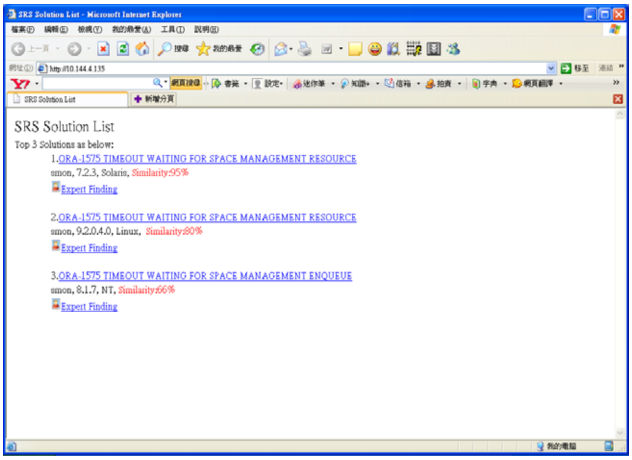

SRS Query Input - Microsoft Internet Explorer
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
上一頁
搜尋
我的最愛
網址(D) http://10.144.4.135
移至
連結
完成
我的電腦

SRS (Solution Retrieving)

Diagnosis Error Type:	System monitoring
1.Keyword Retrieval (Please input the essential attribute value)	
Keyword	
Version	7.0.3
Platform	Solaris S8
2.Subject Retrieval	
Subject Selection	ORA-1575 Timeout waiting for Space Management Enqueue

送出 重新設定

Fig. 9. SRS user interface in query.



SRS Solution List

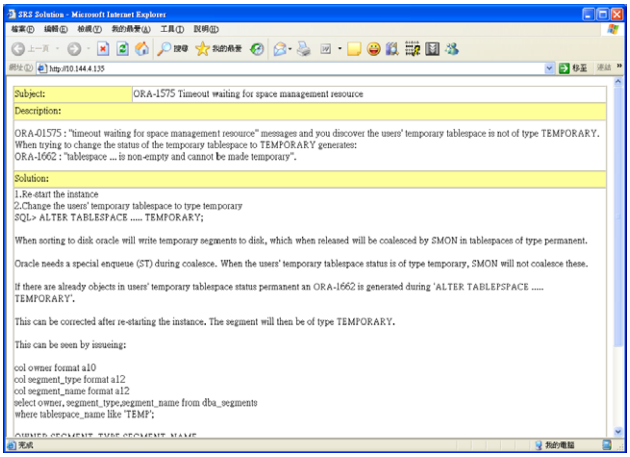
Top 3 Solutions as below:

1.ORA-1575 TIMEOUT WAITING FOR SPACE MANAGEMENT RESOURCE
smon, 7.2.3, Solaris, Similarity:95%
Expert Finding

2.ORA-1575 TIMEOUT WAITING FOR SPACE MANAGEMENT RESOURCE
smon, 9.2.0.4.0, Linux, Similarity:80%
Expert Finding

3.ORA-1575 TIMEOUT WAITING FOR SPACE MANAGEMENT ENQUEUE
smon, 8.1.7, NT, Similarity:66%
Expert Finding

Fig. 10. Solution list in SRS.


SRS Solution - Microsoft Internet Explorer
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
上一頁
搜尋
我的最愛
網址(D)
http://10.144.4.135
移至
完成
我的電腦

Subject: ORA-1575 Timeout waiting for space management resource

Description:

ORA-01575 : "timeout waiting for space management resource" messages and you discover the users' temporary tablespace is not of type TEMPORARY.
When trying to change the status of the temporary tablespace to TEMPORARY generates:
ORA-1662 : "tablespace ... is non-empty and cannot be made temporary".

Solution:

1.Re-start the instance
2.Change the users' temporary tablespace to type temporary
SQL> ALTER TABLESPACE ..... TEMPORARY;

When sorting to disk oracle will write temporary segments to disk, which when released will be coalesced by SMON in tablespaces of type permanent.

Oracle needs a special enqueue (ST) during coalesce. When the users' temporary tablespace status is of type temporary, SMON will not coalesce these.

If there are already objects in users' temporary tablespace status permanent an ORA-1662 is generated during 'ALTER TABLEPSPACE .....
TEMPORARY'.

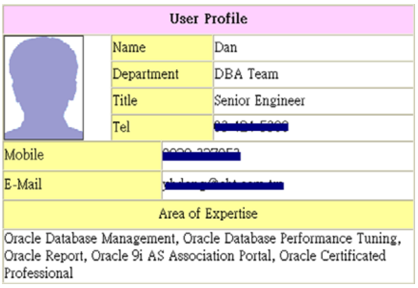
This can be corrected after re-starting the instance. The segment will then be of type TEMPORARY.

This can be seen by issueing:

```
col owner format a10
col segment_type format a12
col segment_name format a12
select owner, segment_type,segment_name from dba_segments
where tablespace_name like 'TEMP';

OWNER SEGMENT_TYPE SEGMENT_NAME
```

Fig. 11. Solution description in SRS system.



User Profile		
	Name	Dan
	Department	DBA Team
	Title	Senior Engineer
	Tel	[redacted]
Mobile	[redacted]	
E-Mail	[redacted]	
Area of Expertise		
Oracle Database Management, Oracle Database Performance Tuning, Oracle Report, Oracle 9i AS Association Portal, Oracle Certificated Professional		

Fig. 12. User profile in SRS system.

Table 6, an example case of the redo log error in case-base contains four attributes (subject, app, version, and platform), and has corresponding description and solution. Domain experts retain cases with complete descriptions and solutions to fulfill case attributes requirements and collect the cases into case bases. They also retain revised value into case base when the case is modified.

Table 4
The attributes of solution case of oracle DB error types.

Attribute name	Data type	Description
Error type	*Text*	The error type of solution case
Subject	*Text*	The subject of case
Module	*Nominal*	The root cause module of system error
Version	*Numerical*	The version of installed system
Platform	*Nominal*	The description of installed operating system
Publisher	*Nominal*	The publisher of solution statement
Date	*Date*	The date of solution document
Description	*Text*	The symptom description of error instance
Solution statement	*Text*	The error recovery standard operation process (SOP) description as solution

Table 5
Descriptions of oracle DB error types.

Error type	Description
DB crash error	DB crash means database crashed and unable to startup
Redo log error	Redo log is a buffer saving data in memory or log files for data rollback
Archive log error	Archive log is for providing 24-h availability and guaranteeing complete data recoverability
Data file error	Data file is a physical file for saving data records that are committed in database
Control file error	Control file records all database information and related database parameters setting
System monitoring error	System monitoring (SMON) performs instance recovery after an instance crash

Table 6
Example case of redo log corruption.

Attributes	Description
Error type	Redo log error
Subject	Redo log corruption
Application	File
Version	8.1.7
Platform	Solaris
Description	Redo log corruption errors in one of the redo log files while the database is open The redo log corruption could be any of these errors: ORA-16038 log%s sequence#%s cannot be archived ORA-354 corrupt redo log block header ORA-353 log corruption near block<num> change<str>time<str> ORA-367 checksum error in log file header ORA-368 checksum error in redo log block
Solution	If an online redo log file has been corrupted while the database is open, the 'alter database clear logfile' command can be used to clear the files without turning off the database Clear the logfile having the problem: Syntax: alter database clear<unarchived> logfile group<integer>; alter database clear<unarchived> logfile '<filename>'; ie: alter database clear logfile group 1; alter database clear unarchived logfile group 1

5. Experiments and evaluation

In this section, we try to evaluate the performance of the novel approach. The purpose of the proposed approach, RCBR is to support problem diagnosis and solution retrieval. The SRS system is proposed based upon rule-based CBR approach to provide solution retrieval service. The experimental results of SRS system compared to the original solution retrieval system called KM center, have been implemented based upon case-based reasoning approach which is a key component in knowledge management system. We have further defined six error categories, and extracted about 10,800 error inference rules, 360 real cases, and 27 expert profiles in SRS system. Besides, two experiments have been designed and implemented to evaluate the accuracy and efficiency of both rule-based CBR approach and case-based reasoning approach, where five domain experts have participated in our experiments by inputting the query to both systems and then evaluating the results. In Experiment 1, we evaluated the accuracy in solutions and expert suggestion between the KC center and the SRS. In Experiment 2, we calculated the efficiency of system in average query times.

Experiment 1: accuracy of solution retrieval and expert suggestion

To evaluate the retrieval accuracy, in Experiment 1, 28 error problems have been dispatched to experts randomly for judging the correctness of suggested solutions from both systems, KC center and SRS, and the evaluated results are shown in Table 7. In addition, the SRS system suggested appropriate expert to solve the problem. In Evaluation 1, the average accuracy rate of RCBR (82.14%) is better than that of CBR (60.71%) as shown in Fig. 13. In Table 8, the SRS system suggests domain expert for each test case. The average accuracy rate of RCBR is 78.57%.

Experiment 2. Efficiency: average query times in system diagnosis and solution retrieving

In Evaluation 2, with predefined 28 questions of six categories, the average times are listed in Table 9, where the query efficiency of SRS system is quicker than that of KC Center and the average query times of RCBR is 2.10, and CBR is 4.93. The diagram of Table 9 shown in Fig. 14 describes the comparison result between SRS and KM center for system diagnosis and solution retrieval in efficiency aspect.

6. Conclusion

The issues of information system problem diagnosis and expert finding are very important for IT service management. Most of novices are unable to find an efficient way to solve the problem even though the relevant document center and the search engine are available. In this paper, we designed and implemented solution

Table 7
Accuracy evaluation between RCBR and CBR.

	Error types						Average hitness
	Db crash	Redo log	Archive log	Data file	Control file	System monitoring	
Test cases	5	4	6	4	4	5	28
Accuracy of KM center (CBR)	3	3	4	2	2	3	17
Accuracy rate of KM center(CBR)	60%	75%	66%	50%	50%	60%	60.71%
Accuracy of SRS (RCBR)	4	4	5	3	3	4	23
Accuracy rate of SRS (RCBR)	80%	100%	83%	75%	75%	80%	82.14%

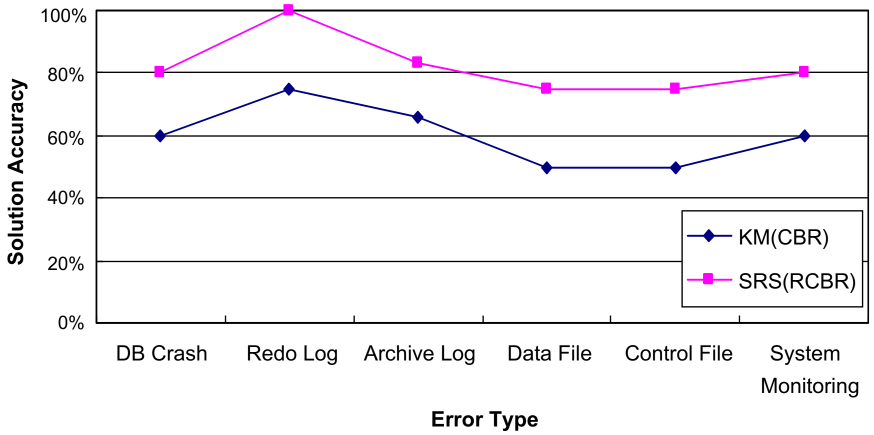


Fig. 13. Accuracy evaluation between RCBR and CBR.

Table 8
Accuracy evaluation of expert suggestion.

	Error types						Average hitness
	Db crash	Redo log	Archive log	Data file	Control file	System monitoring	
Test cases	5	4	6	4	4	5	28
Accuracy of expert suggestion	3	4	4	3	3	5	22
Accuracy rate of expert suggestion	60%	100%	67%	75%	75%	100%	78.57%

Table 9
Efficiency evaluation between RCBR and CBR.

	Average times in solution retrieval						Average times
	Db crash	Redo log	Archive log	Data file	Control file	System monitoring	
SRS (RCBR)	1.60	2.20	2.00	1.20	2.40	3.20	2.10
KM center (CBR)	4.40	5.40	4.20	5.00	4.60	6.00	4.93

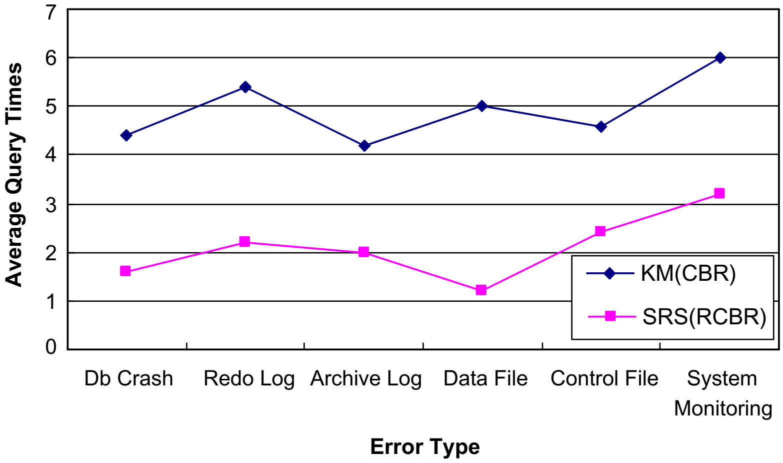


Fig. 14. Efficiency evaluation between RCBR and CBR.

retrieval system (SRS) to assist employee to discover and solve problem in CRM system. Our main contributions are: (1) We proposed the RCBR methodology to improve the accuracy and efficiency of solving multi-domain solution retrieval problem. (2) We proposed RCBR methodology by providing hybrid architecture to solve system diagnosis problem in which hierarchical knowledge ontology is very easy to maintain. (3) We defined the architecture of RCBR for solution retrieval system to enhance process of diagnosis. (4) We incorporate domain expert profiles into our methodology with RBAC model.

According to the experimental results, the paradigm of using RCBR methodology and RBAC model to build SRS system works well and effective. RCBR will benefit the inference on problem diagnosis, and incorporate domain experts into retrieval system with RBAC model by constructing expertise ontology. It is assumed that the same approach could be adaptively modified to other problem domains for knowledge base and user database construction.

In contrast to CBR, our proposed RCBR methodology using rule-based inference and case-based reasoning can refine rule base and revise case base very quickly. In the near future, we will try to enhance the ability of the system to handle more complex problem, e.g., multi-category problem diagnosis. With imitating thinking models of experts, the architecture of RCBR system will also be enhanced to solve similar problems in different category. Furthermore, in our experiments, RCBR system can be supported by online or mobile service.

Acknowledgement

This work was partially supported by National Science Council of the Republic of China under contracts NSC96-2752-E009-006-PAE.

References

Aamodt, A., & Plaza, E. (1994). *Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications* (Vol. 7 (5)). IOS Press. pp. 39–59.

Aktas, M. S., Pierce, M., Fox, G. C., & Leake, D. (2004). A web based conversational case-based recommender system for ontology aided metadata discovery, grid computing. In: *Proceedings of the fifth IEEE/ACM international workshop, 8 November* (pp. 69–75).

Carswell, J. D., Wilson, D. C., & Bertolotto, M., (2002). Digital image similarity for geo-spatial knowledge management, advances in case-based reasoning. In: *Proceedings of the sixth European conference, ECCBR, Aberdeen, Scotland, United Kingdom, September 4–7* (pp. 58–72).

Chang, Y. I., & Hsieh, W. H. (2004). An efficient scheduling method for query-set-based broadcasting in mobile environments, distributed computing systems workshops. In: *Proceedings of the 24th international conference* (pp. 478–483).

Chang, C. G., Wang, D. W., Hu, K.Y., & Zheng, B. L. (2004). Two stage case-based reasoning application research on steel-making dynamic scheduling. In: *Proceedings of the thrid international conference on machine learning and cybernetric, Shanghai* (pp. 2116–2121).

Chen, Y. F., Huang, H., Jana, R., John, S., Jora, S., Reibman, A., & Wei, B. (2002). Personalized multimedia services using a mobile service platform. In: *Wireless communications and networking conference, 17–21 March, WCNC2002, IEEE* (Vol. 2, pp. 918–925).

Dunker, K. (1945). On problem solving. *Psychological Monographs, 58*.

Eckerson, W. W. (1995). Three tier client/server architecture: Achieving scalability, performance, and efficiency in client server applications. Open Information Systems, Vol. 10, 1 (January 1995): 3(20).

Enokido, T., & Takizawa, M. (2008). Role-based concurrency control for distributed systems. In: *Proceeding of the IEEE ICDCS, June* (pp. 24–29).

Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., & Chandramouli, R. (2001). Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security, 4*, 224–274. August.

Gu, M., Tong, X., & Agnar, A. (2005). Comparing similarity calculation methods in conversational CBR, information reuse and integration, conference. In: *IRI-2005 IEEE international conference, 15–17 August* (pp. 427–432).

Hajar, M. J., & Lee, S. P., (2005). Applying machine learning using case-based reasoning (CBR) and rule-based reasoning (RBR) approaches to object-oriented application framework documentation, information technology and applications. In: *ICITA, third international conference, 4–7 July* (Vol. 1, pp. 52–57).

Hayashi, S., Asakura, T., & Zhang, S. (2002). Study of machine fault diagnosis system using neural networks, neural networks. In: *IJCNN '02, proceedings of the 2002 international joint conference, 12–17 May* (Vol. 1, pp. 956–961).

Hou, X., Gu, J., Shen, X., & Yan, W. (2005). Application of data mining in fault diagnosis based on ontology. In: *Proceedings of the third international conference on information technology and applications (ICITA'05)*.

Hwang, G. J., & Tseng, S. S. (1990). EMCUD: A knowledge acquisition method which captures embedded meanings under uncertainty. *International Journal of Man–Machine Studies, 33*, 431–451.

Kumar, F. R., Gopalan, S., & Sridhar, V. (2005). Context enabled multi-CBR based recommendation engine for e-commerce, e-business engineering. ICEBE 2005. In: *IEEE international conference, 12–18 October* (pp. 237–244).

Lambert-Torres, G., Martins, H. G., Rossi, R., & da Silva, L. E. B. (2003). Using similarity assessment in case-based reasoning to solve power system substation problems, electrical and computer engineering, IEEE CCECE. In: *Canadian conference, 4–7 May* (Vol. 1, pp. 343–346).

Lin, Y. T., Tseng, S. S., & Tsai, C. F. (2003). Design and implementation of new object-oriented rule base management system. *Expert Systems with Applications, 25*, 369–385.

Reichgelt, H. (1991). *Knowledge representation, an AI perspective*. Norwood, NJ: Ablex Publishing Corporation.

Rish, I., Brodie, M., Ma, S., Odintsova, N., Beygelzimer, A., Grabarnik, G., et al. (2005). Adaptive diagnosis in distributed systems. *IEEE Transactions on Neural Networks, 16*(5). September.

Sandhu, R. S., Coyne, E. J., Feinstein, H. L., & Youman, C. E. (1996). Role-based access control models. *IEEE Computer, 29*, 38–47. February.

Schussel, G. (1995). *Client/server past, present, and future*. <http://news.dci.com/geos/dbsejava.htm>.

Smyth, B., IEEE Computer Society, Keane, M. T., & Conningham, P. (2001). Hierarchical case-based reasoning integrating case-based and decompositional problem-solving techniques for plant-control software design. In: *IEEE transactions on knowledge and data engineering* (Vol. 13, pp. 793–812).

Spalazzi, L. (2001). A survey on case-based planning. *Artificial Intelligence Review, 16*, 3–36.

Tsai, C.-F. (2002). *Design and implementation of new object-oriented rule base management system*. Master thesis, Department of Computer and Information Science, National Chiao Tung University.

Tsai, C. J., Tseng, S. S., & Wu, Y. C. (1999). A new architecture of objected-oriented rule base management system. In: *Proceeding of international conference on TOOLS 31, Nanjing, China* (pp. 200–203).

Wai, Y. L., & Lau, F. C. M. (2003). User-centric content negotiation for effective adaptation service in mobile computing. *Software Engineering, IEEE Transactions, 29*(12), 1100–1111. December.

Wang, H., & Liu, Y. (2004). Hierarchical case-based decision support system for power system restoration. *Power Engineering Society General Meeting, 1*, 1115–1119.

Wu, Y. C. (1999). *An approach to object-oriented rule base management system*. Master thesis, Department of Computer and Information Science, National Chaio Tung University.

Yang, K. J., & Chen, Y. M. (2006). Ontology-based knowledge retrieval in organizational memory, innovative computing, information and control. In: *ICICIC'2006, 30 August* (Vol. 1, pp. 566–569).

Zhang, D., Dai, S., Zheng, Y., Zhang, R., & Mu, P. (2000). Researches and application of a hybrid fault diagnosis expert system, intelligent control and automation. In: *Proceedings of the third world congress, 28 June to 2 July* (Vol. 1, pp. 215–219).