An Ontological Template-supported Interface Agent for FAQ Services

SHENG-YUAN YANG
Dept. of Computer and Communication Engineering
St. John's University
499, Sec. 4, TamKing Rd., Tamsui, Taipei County 251
TAIWAN
http://mail.sju.edu.tw/~ysy

Abstract: - This paper describes an Interface Agent which works as an assistant between the users and FAQ systems. It integrates several interesting techniques including domain ontology, user modeling, and template-based linguistic processing to effectively tackle the problems associated with traditional FAQ retrieval systems. Our work features an ontology-supported, template-based user modeling technique for developing interface agents. Our preliminary experimentation demonstrates that user intention and focus of up to eighty percent of the user queries can be correctly understood by the system.

Key-Words: - Interface Agents, Ontological Template-supported Processing, User Modeling.

1 Introduction

With increasing popularity of the Internet, people depend more on the Web to obtain their information. Especially the use of the World Wide Web has been leading to a large increase in the number of people who access FAQ (Frequently Asked Questions) knowledge bases to find answers to their questions [8]. The websites which provide FAQs organize user questions and expert answers about a specific product or discipline in terms of question-answer pairs on the Web. Each FAQ is represented by one question along with one answer and is characterized, to be domain-dependent, short and explicit, and frequently asked. People usually go through the list of FAQs and read those FAQs that are to his questions. This way of answering the user's questions can save the labor power for experts from answering similar questions repeatedly. The problem is after the fast accumulation of FAQs, it becomes harder for people to single out related FAQs. Traditional FAQ retrieval systems, however, provide only little help, became they fail to provide assistance and guidance for human-machine interaction, personalized information services, flexible interaction interface, etc. [2].

In order to capture true user's intention and accordingly provides high-quality FAQ answers to meet the user requests, we have proposed an Interface Agent acquires user intention through an adaptive human-machine interaction interface with the help of ontology-directed and template-based user models [9,10,11]. It also handles user feedback on the suitability of proposed responses. The agent features ontology-based representation of domain knowledge, ontological interaction interface, and personalized information filtering and display. Specifically, according to the user's behavior and mental state, we employed the technique of user modeling to construct a user model to describe his characteristics, preference and knowledge proficiency level, etc. We also used the technique of user stereotype [5] to construct and initialize a new user model, which helps provide fast-personalized services for new users. We built domain ontology [4] to help define domain vocabulary and knowledge and based on that to construct user models and the Interface Agent. We extended the concept of pattern match [3] to query template to construct natural language based query models. This idea leads to the extraction of true user intention and focus from his query posted as nature language understanding, which help the agent to fast find out precise information for the user. Our preliminary experimentation demonstrates that the intention and focus of up to eighty percent of the users' queries can be correctly understood. The Personal Computer (PC) domain is chosen as the target application of our Interface Agent and will be used for explanation in the remaining sections.

2 Basic Techniques

2.1 Domain Ontology

The concept of ontology in artificial intelligence refers to knowledge representation for domain-specific contents [1]. Although development of an ontology for a specific domain is not yet an engineering process, we have outlined a procedure for this in [9] from how the process was conducted in existent systems. By following the procedure we developed an ontology for the PC domain in Chinese using Protégé 2000 [4], but was changed to English here for easy explanation, as the fundamental background knowledge for the system.

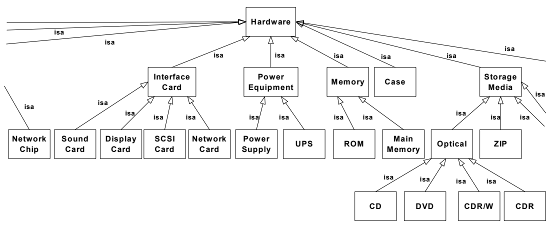

Fig. 1 Part of PC ontology taxonomy

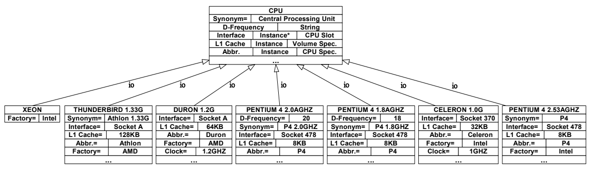

Fig. 2 Ontology of the concept of CPU

Fig. 1 shows part of the ontology taxonomy. The taxonomy represents relevant PC concepts as classes and their parent-child relationships as *isa* links, which allow inheritance of features from parent classes to child classes. Fig. 2 exemplifies the detailed ontology of the concept of CPU. In the figure, the root node uses various fields to define the semantics of the CPU class, each field representing an attribute of "CPU", e.g., interface, provider, synonym, etc.

The nodes at the lower level represent various CPU instances, which capture real world data. The arrow line with term "*io*" means the instance of relationship. The complete PC ontology can be referenced from the Protégé Ontology Library at Stanford Website (http://protege.stanford.edu/download/download.html).

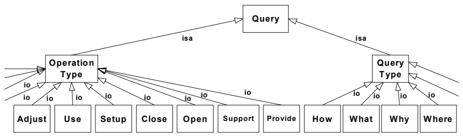

Fig. 3 Part of problem ontology taxonomy

We also developed a Problem ontology to deal with query questions. Fig. 3 illustrates part of the Problem ontology, which contains query type and operation type. Together they imply the semantics of a question. Finally, we use Protégé's APIs to develop a set of ontology services, which provide primitive functions to support the application of the ontologies. The ontology services currently available include transforming query terms into canonical ontology terms, finding definitions of specific terms in ontology, finding relationships among terms, finding compatible and/or conflicting terms against a specific term, etc.

2.2 Ontological Templates

Table 1 Examples of query patterns

Question Type	Operation Type	Intention Type	Query Pattern
是否 (If)	支援 (Support)	ANA_CAN_SUPPORT	<S1 是否 支援 S2>
		GA-7VRX 這塊主機板是否支援 KINGMAX DDR-400？ (Could the GA-7VRX motherboard support the KNIGMAX DDR-400 memory type?)	
如何 (How)	安裝 (Setup)	HOW_SETUP	<如何 在 S1><安裝 S2>
		如何在 Windows 98SE 下，安裝 8RDA 的音效驅動程式？ (How to setup the 8RDA sound driver on a Windows 98SE platform?)	
什麼 (What)	是 (Is)	WHAT_IS	<S1 是 什麼>
		AUX power connector 是什麼？ (What is an AUX power connector?)	
何時 (When)	支援 (Support)	WHEN_SUPPORT	<S1 何時 支援 S2>
		P4T 何時才能支援 32-bit 512 MB RDRAM 記憶體規格？ (When can the P4T support the 32-bit 512 MB RDRAM memory specification?)	
哪裡 (Where)	下載 (Download)	WHERE_DOWNLOAD	<S1><哪裡 可以 下載 S2>
		CUA 的 Driver CD 遺失，請問哪裡可以下載音效驅動程式？ (Where can I download the sound driver of CUA whose Driver CD was lost?)	
為什麼 (Why)	列印 (Print)	WHY_PRINT	[S1]<S2 無法 列印>
		為什麼在 Win ME 底下，從休眠狀態中回復後，印表機無法列印。 (Why can I not print after coming back from dormancy on a Win ME platform?)	

Table 2 Query template for the ANA_CAN_SUPPORT intention type

Template_Number	304
#Sentence	3
Intention_Word	是否(If)、支援(Support)
Intention_Type	ANA_CAN_SUPPORT
Question_Type	是否(If)
Operation_Type	支援(Support)
Query_Patterns	[S3]<S1 是否 支援 S2> [S2]<是否 支援 S1>
Focus	S1

To build the query templates, we have collected in total 1215 FAQs from the FAQ website of six famous motherboard factories in Taiwan and used them as the reference materials for query template construction. Currently, we only take care of the user query with one intention word and at most three sentences. These FAQs were analyzed and categorized into six types of questions. For each type of question, we further identified several intention types according to its operations. Finally, we define a query pattern for each intention type. Table 1 illustrates the defined query patterns for the intention types. Now all information for constructing a query template is ready, and we can formally define a query template [3,6]. Table 2 illustrates an example query template for the ANA_CAN_SUPPORT intention type. Note here that we collect similar query patterns in the field of "Query patterns," which are used in detailed analysis of a given query.

According to the generalization relationships among intention types, we can form a hierarchy of intention types to organize all FAQs. Currently, the hierarchy contains two levels as shown in Fig. 4. Now, the system can employ the intention type hierarchy to reduce the search scope during the retrieval of FAQs after the intention of a user query is identified.

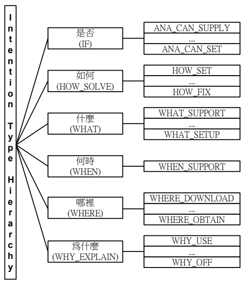

Fig. 4 Intention type hierarchy

3 System Architecture

3.1 User Modeling

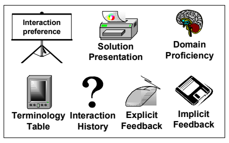

Fig. 5 Our user model

A user model [11] contains interaction preference, solution presentation, domain proficiency, terminology table, query history, selection history, and user feedback, as shown in Fig. 5. The interaction preference is responsible for recording user's preferred interface, e.g., favorite query mode, favorite recommendation mode, etc. The solution presentation is responsible for recording solution ranking preferences of the user. We provide two types of ranking, either according to the degree of similarity between the proposed solutions and the user query, or according to user's proficiency about the solutions. The domain proficiency factor describes how familiar the user is with the domain. By associating a proficiency degree with each ontology concept, we can construct a table, which contains a set of <concept proficiency-degree> pairs, as his domain proficiency. To solve the problem of different terminologies to be used by different users, we include a terminology table to record this terminology difference. Finally, we record the user's query history as well as FAQ selection history and corresponding user feedback in each query session in the Interaction history, in order to support collaborative recommendation.

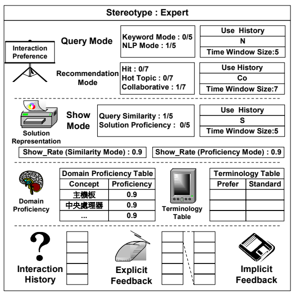

Fig. 6 Example of expert stereotype

In order to quickly build an initial user model for a new user, we pre-defined five stereotypes [5], namely, expert, senior, junior, novice, and amateur [9], to represent different user group's characteristics. This approach is based on the idea that the same group of user tends to exhibit the same behavior and requires the same information. Fig. 6 illustrates an example user stereotype. When a new user enters the system, he is asked to complete a questionnaire, which is used by the system to determine his domain proficiency, and accordingly select a user stereotype to generate an initial user model to him. However, the initial user model constructed from the stereotype may be too generic or imprecise. It will be refined to reflect the specific user's real intent after the system has experiences with his query history, FAQ-selection history and feedback, and implicit feedback [2].

3.2 System Description

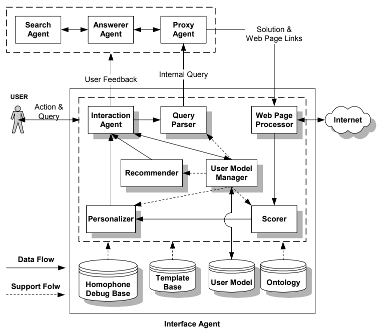

Fig. 7 Interface agent architecture

Fig. 7 illustrates the architecture of the Interface Agent and how it interacts with the backend agents. This section gives detailed description of each component inside the Agent.

3.2.1 Interaction Agent

The Interaction Agent consists of the following three components: Adapter, Observer and Assistant. First, the Adapter constructs best interaction interfaces according to user's favorite query and recommendation modes. It is also responsible for presenting to the user the list of FAQ solutions (from the Personalizer) or recommendation information (from the Recommender). During solution representation, it arranges the solutions in terms of the user's preferred style (query similarity or solution proficiency) and displays the solutions according to the "Show_Rate." Second, the Observer passes the user query to the Query Parser, and simultaneously collects the interaction information and related feedback from the user. The interaction information contains user preferred query mode, recommendation mode, solution presentation mode, and FAQ clicking behavior, while the related feedback contains user satisfaction degree and comprehension degree about each FAQ solution. The User Model Manager needs both interaction information and related feedback to properly update user models and stereotypes. The satisfaction degree in related feedback can also be passed to the Proxy Agent for tuning the solution search mechanism [12].

Finally, the Assistant provides proper assistance and guidance to help the user query process. First, the ontology concepts are structured and presented as a tree so that the users who are not familiar with the domain can check on the tree and learn proper terms to enter their queries. We also rank all ontology concepts by their probabilities and display them in a keyword list. When the user enters a query at the input area, the Assistant will automatically "scroll" the content of the keyword list to those terms related to the input keywords. Fig. 8 illustrates an example of this automatic keyword scrolling mechanism. If the displayed terms of the list contain a concept that the user wants to enter, he can double-click the terms into the input area, e.g., "華碩" (ASUS) at step 2 of Fig. 8. In addition to the keyword-oriented query mode, the Assistant also provides lists of question types and operation types to help question type-oriented or operation type-oriented search. The user can use one, two, or all of these three mechanisms to help form his query in order to convey his intention to the system.

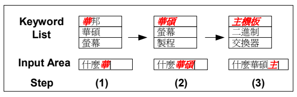

Fig. 8 Examples of automatic keyword scrolling mechanism

3.2.2 Query Parser

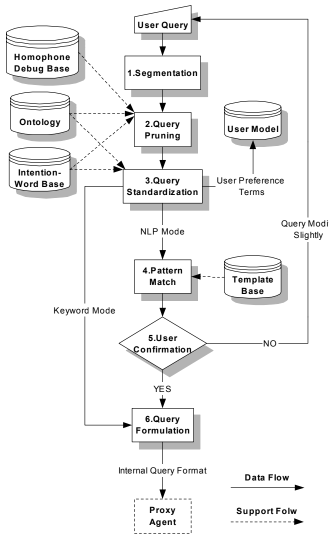

Fig. 9 Flow chart of the Query Parser

The Query Parser pre-processes the user query by performing Chinese word segmentation, correction on word segmentation, fault correction on homophonous or multiple words, and term standardization. It then employs template-based pattern matching to analyze the user query and extract the user intention and focus. Finally, it transforms the user query into the internal query format and then passes the query to the Proxy Agent for retrieving proper solutions [12]. Fig. 9 shows the flow chart of the Query Parser. Detailed explanation follows.

Given a user query in Chinese, we segment the query using MMSEG [7]. The results of segmentation were not good, for the predefined MMSEG word corpus contains insufficient terms of the PC domain. For example, it does not contain the keywords "華碩" or "AGP4X", and returns wrong word segmentation like "華", "碩", "AGP", and "4X". The step of query pruning can easily fix this by using the ontology as a second word corpus to bring those mis-segmented words back. It also performs fault correction on homophonous or multiple words using the ontology and homophone debug base [2].

The step of query standardization is responsible for replacing the terms used in the user query with the canonical terms in the ontology and intention word base. The original terms and the corresponding canonical terms will then be stored in the terminology table for solution presentation personalization. Finally, we label those recognized keywords by symbol "K" and intention words by symbol "I." The rest are regarded as stop words and removed from the query. Now, if the user is using the keyword mode, we directly jump to the step of query formulation. Otherwise, we use template-based pattern matching to analyze the natural language input.

The step of pattern match is responsible for identifying the semantic pattern associated with the user query. Using the pre-constructed query templates in the template base, we can compare the user query with the query templates and select the best-matched one to identify user intention and focus. Fig. 10 shows the algorithm of fast selecting possibly matched templates, Fig. 11 describes the algorithm which finds out all patterns matched with the user query, and Fig. 12 removes those matched patterns that are generalization of some other matched patterns.

```
Template Selection :
Q : User Query.
Q.Intention_Word = {$I_1, I_2, ..., I_N$}, Intention words in Q.
Q.Sentence : Number of sentences in Q.
Template Base = {$T_1, T_2, ..., T_M$}, M : Number of templates.
For each template $T_j$ in Template Base
{
    If $T_j$ conforms to the follow rules, then select $T_j$ into C.
        1. $T_j$.Sentence = Q.Sentence.
        2. $T_j$.Intention_Word $\subseteq$ Q.Intention_Word.
}
return C : Candidate Templates.
```

Fig. 10 Query template selection algorithm

```
Pattern Match :
For each template $T_j$ in C, candidate templates
{
    $T_j$.Pattern = {$P_1, P_2, ...$}, $P_k$ : pattern k in template j.
    For each $P_K$ in $T_j$.Pattern
    {
        If $P_k$ match Q, the user query, then
            $P_k$.Intention_Word = $T_j$.Intention_Word,
            $P_k$.Intention_Type = $T_j$.Intention_Type,
            $P_k$.Quertion_Type = $T_j$.Question_Type,
            $P_k$.Operation = $T_j$.Operation,
            $P_k$.Focus = $T_j$.Focus,
            and put $P_k$ in M and break this inner loop.
    }
}
return M : Patterns matching Q.
```

Fig. 11 Pattern match algorithm

```
Pattern Removal :
For each pattern $P_k$ in M, matched patterns
{
    If $P_k$ conforms to follow rule, then remove $P_k$ from M.
    {
        $\exists P_i \in M$, $P_k$.Intention_Type = $P_i$.Intention_Type and
                    $P_k$.Intention_Word $\subset$ $P_i$.Intention_Word
    }
}
```

Fig. 12 Query pattern removal algorithm

Table 3 Internal form

User_Level	…
Query_Mode	…
Intention_Type	…
Question_Type	…
Operation	…
Keyword	…
Focus	…

Take the following query as an example: "華碩 K7V 主機板是否支援 1GHz 以上的中央處理器呢？", which means "Could Asus K7V motherboard support a CPU over 1GHz?" Table 2 illustrates a query template example, which contains two patterns, namely, <S1 是否 支援 S2> and <是否 支援 S1>. We find the second pattern matches the user query and can be selected to transform the query into an internal form by query formulation (step 6) as shown in Table 3. Note that there may be more than two patterns to become candidates for a given query. In this case, the Query Parser will prompt the user to confirm his intent (step 5). If the user says "No", which means the pattern matching results is not the true intention of the user, he is allowed to modify the matched result or change to the keyword mode for placing query.

3.2.3 Web Page Processor

The Web Page Processor receives a list of retrieved solutions, which contains one or more FAQs matched with the user query from the Proxy Agent, each represented as Table 4, and retrieves and caches the solution webpages according to the FAQ_URLs. It follows to pre-process those webpages for subsequent customization process, including URL transformation, keyword standardization, and keyword marking. The URL transformation changes all hyperlinks to point toward the cached server. The keyword standardization transforms all terms in the webpage content into ontology vocabularies. The keyword labeling marks the keywords appearing in the webpages by boldfaced <B>Keyword<\B> to facilitate subsequent keywords processing webpage readability.

Table 4 Format of retrieved FAQ

Field	Description
FAQ_No.	FAQ's identification
FAQ_Question	Question part of FAQ
FAQ_Answer	Answer part of FAQ
FAQ_Similarity	Similarity degree of the FAQ met with the user query
FAQ_URL	Source or related URL of the FAQ

3.2.4 Scorer

```
A : Answer FAQs List, $A = \{F_1, F_2, ..., F_n\}$, $A \subset C$, FAQ Collection.
A : FAQs found by the system that best match user query.
$F_i$.Q : The Query part of FAQ $F_i$.
For each $F_i \in A$, we calculate each $F_i$'s Proficiency Score for user k.
$Fi = \frac{1}{\text{Number of Concept appering in Fi.Q}} * \sum_j \text{Appearance(Concept j)} * \text{Proficiency(Concept j)}$,
where Concept $j \in$ Ontology,
Appearance(Concept j) : $\begin{cases} 1, \text{Concept j appears in } F_i.Q \\ 0, \text{Concept j doesn't appear in } F_i.Q, \end{cases}$
Proficiency(Concept j) : The drgree of user k's proficiency in Concept j.
```

Fig. 13 Proficiency degree calculation algorithm

Each FAQ is a short document; concepts involved in FAQs are in general more focused. In other words, the topic (or concept) is much clearer and professional. The question part of an FAQ is even more pointed about what concepts are involved. Knowing this property, we can use the keywords appearing in the question part of an FAQ to represent its topic. Basically, we use the table of domain proficiency to calculate a proficiency degree for each FAQ by calculating the proficient concepts appearing in the question part of the FAQ, detailed as shown in Fig. 13.

3.2.5 Personalizer

The Personalizer replaces the terms used in the solution FAQs with the terms in the user's terminology table, collected by the Query Parser, for improving the solution readability.

3.2.6 User Model Manager

The first task of the User Model Manager is to create an initial user model for a new user. To do this, we pre-defined several questions for each concept in the domain ontology, for example, "Do you know a CPU contains a floating co-processor?", "Do you know the concept of 1GB=1000MB in specifying the capacity of a hard disk?", etc. The difficulty degrees of the questions are proportional to the hierarchy depth of the concepts in the ontology. When a new user logs on the system, the Manager randomly selects questions from the ontology. The user either answers an YES or NO to each question. The answers are collected and weighted according to the respective degrees and passed to the Manager, which then calculates a proficiency score for the user according to the percentage of correctness of his responses to the questions and accordingly instantiates a proper user stereotype as the user model for the user.

The second task is to update user models. Here we use the interaction information and user feedback collected by the Interaction Agent in each interaction session or query session. An interaction session is defined as the time period from the time point the user logs in up to when he logs out, while a query session is defined as the time period from when the user gives a query up to when he gets the answers and completes the feedback. An interaction session may contain several query sessions. After a query session is completed, we immediately update the interaction preference and solution presentation of the user model. Specifically, the user's query mode and solution presentation mode in this query session are remembered in both time windows, and the statistics of the preference change for each mode is calculated accordingly, which will be used to adapt the Interaction Agent on the next query session. Fig. 14 illustrates the algorithm to update the Show_Rate of the similarity mode. The algorithm uses the ratio of the number of user selected FAQs and that of the displayed FAQs to update the show rate; the algorithm to update the Show_Rate of the proficiency mode is similar.

$\mathrm{N_S}$: Number of FAQ in Solution FAQ List.

$\mathrm{N_{(Similarity\ Mode)}}$: Number of FAQ shown to user in Similarity Mode.

$$\mathrm{N_{(Similarity\ Mode)}} = \left\lceil \mathrm{N_S} * \mathrm{Show\_Rate}_{\mathrm{(Similarity\ Mode)}_{old}} \right\rceil$$

$\mathrm{N_{Hide}} = \mathrm{N_S} - \mathrm{N_{(Similarity\ Mode)}}$, Number of hidden FAQ.

$\mathrm{N_{Select}}$: Number of FAQ selected by user in the query session.

$$\mathrm{Show\_Rate}_{\mathrm{(Similarity\ Mode)}} = \mathrm{Show\_Rate}_{\mathrm{(Similarity\ Mode)}_{old}} + Variation$$

$$Variation = \begin{cases} \left(\dfrac{\mathrm{N_{Select}}}{\mathrm{N_S}} - 0.7\right) * \left(1 - \exp\left(-\dfrac{\mathrm{N_{(Similarity\ Mode)}}}{\alpha}\right)\right), \text{ if } \dfrac{\mathrm{N_{Select}}}{\mathrm{N_S}} \geq 0.7 \\ 0, \text{ if } 0.3 < \dfrac{\mathrm{N_{Select}}}{\mathrm{N_S}} < 0.7 \\ \left(\dfrac{\mathrm{N_{Select}}}{\mathrm{N_S}} - 0.3\right) * \left(1 - \exp\left(-\dfrac{\mathrm{N_{(Similarity\ Mode)}}}{\alpha}\right)\right), \text{ if } \dfrac{\mathrm{N_{Select}}}{\mathrm{N_S}} \leq 0.3, \end{cases}$$

where α : weight change rate,

$$\mathrm{Show\_Rate}_{\mathrm{(Similarity\ Mode)}_{new}} = \mathrm{Max}(\mathrm{Min}(\mathrm{Show\_Rate}_{\mathrm{(Similarity\ Mode)}}, 1), 0.01)$$

Fig. 14 Algorithm to update show rate in similarity mode

$F = \{F_1, F_2, \ldots\} \subseteq C$, FAQ Collection.

$F_i \in F$: FAQs selected and rated by user.

For each F_i, update user proficiency of each Concept j.

$$\text{Proficiency (Concept j)} = \text{Proficiency}_{old}\text{(Concept j)} + \alpha * \left(\frac{\text{T(Concept j)}}{\text{Number of Concept in } F_i} * \text{Understanding Level} \right)$$

$$\text{Proficiency}_{new}\text{(Concept j)} = \mathrm{Max}(\mathrm{Min}(\text{Proficiency(Concept j)}, 1), 0),$$

where α : Learning rate,

Concept $j \in$ Ontology,

T(Concept j) : The times Concept j appears in F_i,

$$\text{Understanding Level} : \begin{cases} +2, \text{ user rates "very familiar" for } F_i. \\ +1, \text{ user rates "familiar" for } F_i. \\ 0, \text{ user rates "average" for } F_i. \\ -1, \text{ user rates "not familiar" for } F_i. \\ -2, \text{ user rates "very not familiar" for } F_i. \end{cases}$$

Fig. 15 Algorithm to update the domain proficiency table

In addition, each user will be asked to evaluate each solution FAQ in terms of the following five levels of understanding, namely, very familiar, familiar, average, not familiar, very not familiar. This provides an explicit feedback and we can use it to update his domain proficiency table. Fig. 15 shows the updating algorithm. Finally, after each interaction session, we can update the user's recommendation mode in this session in the respective time window. At the same time, we add the query and FAQ-selection records of the user into the query history and selection history of his user model.

```
For each user_i in each user proficiency group
{
    $\text{Proficiency}_{avg}(\text{user}_i) = \dfrac{\sum_j \text{Proficiency(Concept j)}}{\text{Number of concepts in Domain Proficiency Table}},$
    where Concept j : jth Concept in user_i's Domain Proficiency Table,
    if $(0.8 \leq \text{Proficiency}_{avg}(\text{user}_i) \leq 1.0)$, user i reassigned to Expert group.
    if $(0.6 \leq \text{Proficiency}_{avg}(\text{user}_i) < 0.8)$, user i reassigned to Senior group.
    if $(0.4 \leq \text{Proficiency}_{avg}(\text{user}_i) < 0.6)$, user i reassigned to Junior group.
    if $(0.2 \leq \text{Proficiency}_{avg}(\text{user}_i) < 0.4)$, user i reassigned to Novice group.
    if $(0.0 \leq \text{Proficiency}_{avg}(\text{user}_i) < 0.2)$, user i reassigned to Amateur group.
}
```

Fig. 16 Algorithm to re-cluster all user groups

```
$S_{Expert}$ : Stereotype of Expert.
$U = \{U_1, U_2, \ldots, U_N\}$, Users in Expert group.
$S_{Expert}.\text{Show\_Rate}_{\text{(Similarity Mode)}} = \dfrac{\sum_{i=1}^{N} U_i.\text{Show\_Rate}_{\text{(Similarity Mode)}}}{N}$
$S_{Expert}.\text{Show\_Rate}_{\text{(Proficiency Mode)}} = \dfrac{\sum_{i=1}^{N} U_i.\text{Show\_Rate}_{\text{(Proficiency Mode)}}}{N}$
For each Concept j in Domain Proficiency Table (DPT)
{
    $S_{Expert}.\text{DPT.Proficiency(Concept j)} = \dfrac{\sum_{i=1}^{N} U_i.\text{DPT.Proficiency(Concept j)}}{N}$
}
```

Fig. 17 Example to update the stereotype of Expert

The third task of the User Model Manager is to update user stereotypes. This happens when a sufficient number of user models in a stereotype has undergone changes. First, we need to reflect these changes to stereotypes by re-clustering all affected user models, as shown in Fig. 16, and then re-calculates all parameters in each stereotype, an example as shown in Fig. 17.

3.2.7 Recommender

The Recommender uses the following three policies to recommend information. 1) High hit FAQs. It recommends the first N solution FAQs according to their selection counts from all users in the same group within a time window. 2) Hot topic FAQs. It recommends the first N solution FAQs according to their popularity, calculated as statistics on keywords appearing in the query histories of the same group users within a time window. The algorithm does the hot degree calculation as shown in Fig. 18. 3) Collaborative recommendation. It refers to the user's selection histories of the same group to provide solution recommendation. The basic idea is this. If user A and user B are in the same group and the first n interaction sessions of user A are the same as those of user B, then we can recommend the highest-rated FAQs in the (n+1)th session of user A for user B, detailed algorithm as shown in Fig. 19.

For each faq $i \in C$, FAQ Collection.

HOT Score faq i =

$$\frac{1}{\text{Number of Concepts appearing in faq i}} * \sum_j \text{Appearance(Concept j)} * \text{Weight(Concept j)},$$

where Concept $j \in$ Ontology,

$$\text{Appearance(Concept j)} : \begin{cases} 1, \text{ Concept j appears in faq i} \\ 0, \text{ Concept j doesn't appear in faq i,} \end{cases}$$

Weight(Concept j) : Within a time window, the times that the Concept j appears in user queries of the same user group,

Fig. 18 Algorithm to calculate the hot degree

$G = \{U_1, U_2, \ldots, U_m\}$, Users in the same group G.

$Y = \{U_j \in G \mid U_j \neq U_i\}$.

FAQ Collection $= \{F_1, F_2, \ldots, F_n\}$.

$U_i = \{S_{i,1}, S_{i,2}, \ldots, S_{i,x}\}$, Query sessions of user i.

$S_{i,x} = \{R_{i,x,1}, R_{i,x,2}, \ldots, R_{i,x,n}\}$, FAQs rated and/or selected by User i in session x.

$$R_{i,x,y} : \begin{cases} 1, \text{ If User i select } F_y \text{ in session x, and rate more than satisfying.} \\ 0.8, \text{ If User i select } F_y \text{ in session x, and rate satisfying.} \\ 0.6, \text{ If User i select } F_y \text{ in session x, and rate average.} \\ 0.6, \text{ If User i select } F_y \text{ in session x, but no rating.} \\ 0.4, \text{ If User i select } F_y \text{ in session x, and rate dissatisfying.} \\ 0.2, \text{ If User i select } F_y \text{ in session x, and rate less than dissatisfying.} \\ 0, \text{ If User i doesn't select } F_y \text{ in session x.} \end{cases}$$

$$\text{Similarity}(S_{i,x}, S_{j,k}) = \arg\max_{S_{j,k}} \left(\sum_{U_j \in Y} \sum_{S_{j,k} \in U_j} (1 - \text{Distance}(S_{i,x}, S_{j,k})) \right),$$

$$\text{where Distance}(S_{i,x}, S_{j,k}) = \frac{\sqrt{\sum_{z=1}^{n} (R_{i,x,z} - R_{j,k,z})^2}}{n}.$$

Let $S_{j,k}$ be the most similar to $S_{i,x}$, and $S_{i,x+1}$ be user i in current query session.

Recommend following FAQs for $S_{i,x+1}$ =

$$\{F_y \mid R_{j,k,y} > 0.5 \text{ and } R_{i,x,y} = 0\} \cup \{F_y \mid R_{j,k+1,y} > 0.5 \text{ and } R_{i,x,y} = 0\}$$

Fig. 19 Algorithm to do the collaborative recommendation

4 System Demonstrations and Evaluations

Our Interface Agent was developed using the Web-Based client-server architecture. On the client site, we use JSP (Java Server Page) and Java Applet for easy interacting with users, as well as observing and recording user's behavior. On the server site, we use Java and Java Servlet, under Apache Tomcat 4.0 Web Server and MS SQL2000 Server running Microsoft Windows XP. Fig. 20 illustrates the main tableau of our system, which consists of the following three major tab-frames, namely, query interface, solution presentation, and logout. The query interface tab is comprised of the following four frames: user interaction interface, automatic keyword scrolling list, FAQ recommendation list, and PC ontology tree. The user interaction interface contains both keywords and NLP query modes. The keyword query mode provides the lists of question types and operation types, which allow the users to express their precise intentions. The automatic keyword scrolling list provides ranked-keyword guidance for user query. A user can browse the PC ontology tree to learn domain knowledge. The FAQ recommendation list provides personalized information recommendations from the system, which contains three modes: hit, hot topic, and collaboration. When the user clicked a mode, the corresponding popup window is produced by the system.

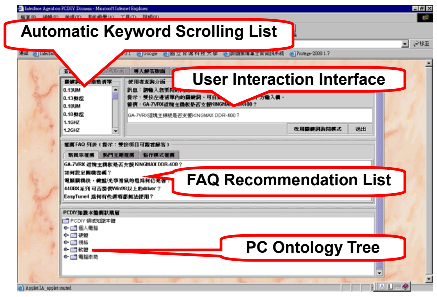

Fig. 20 Main tableau of our system

The evaluation of the overall performance of our system involves lots of manpower and is time-consuming. Here, we focus on the performance evaluation of the most important module, i.e., the Query Parser. Our philosophy is that if it can precisely parse user queries and extract both true query intention and focus from them, then we can effectively improve the quality of the retrieved. Recall that the Query Parser employs the technique of template-based pattern matching mechanism to understand user queries and the templates were manually constructed from 1215 FAQs. In the first experiment, we use this same FAQs for testing queries, in order to verify whether any conflicts exist within the query. Table 5 illustrates the experimental results, where only 33 queries match with more than one query patterns and result in confusion of query intention, called “error” in the table. These errors may be corrected by the user. The experiment shows the effectiveness rate of the constructed query templates reaches 97.28%, which implies the template base can be used as an effective knowledge base to do natural language query processing.

Table 5 Effectiveness of constructed query patterns

#Testing	#Correct	#Error	Precision Rate (%)
1215	1182	33	97.28 %

Our second experiment is to learn how well the Parser understands new queries. First, we collected in total 143 new FAQs, different from the FAQs collected for constructing the query templates, from four famous motherboard factories in Taiwan, including ASUS, GIGABYTE, MSI, and SIS. We then used the question parts of those FAQs for testing queries, which test how well the Parser performs. Our experiments show that we can precisely extract true query intentions and focuses from 112 FAQs. The rest of 31 FAQs contain up to three or more sentences in queries, which explain why we failed to understand them. In summary, 78.3% (112/143) of the new queries can be successfully understood.

5 Discussions and Future work

We have developed an Interface Agent to work as an assistant between the users and systems, which is different from system architecture and implementation over our previous work [10]. It is used to retrieve FAQs on the domain of PC. We integrated several interesting techniques including user modeling, domain ontology, and template-based linguistic processing to effectively tackle the problems associated with traditional FAQ retrieval systems. In short, our work features an ontology-supported, template-based user modeling technique for developing interface agents; a nature language query mode, along with an improved keyword-based query mode; and an assistance and guidance for human-machine interaction. Our preliminary experimentation demonstrates that user intention and focus of up to eighty percent of the user queries can be correctly understood by the system. In the future, we are planning to employ the techniques of machine learning and data mining to automate the construction of the template base. As to the allover system evaluation, we are planning to employ the concept of usability evaluation on the domain of human factor engineering to evaluate the performance of the user interface.

Acknowledgements

The author would like to thank Yai-Hui Chang and Ying-Hao Chiu for their assistance in system implementation. This work was supported by the National Science Council, R.O.C., under Grant NSC-95-2221-E-129-019.

References:

[1] Chandrasekaran, B., Josephson, J.R., and Benjamins, V.R., What Are Ontologies, and Why Do We Need Them? *IEEE Intelligent Systems*, Vol. 14, No. 1, 1999, pp. 20-26.

[2] Chiu, Y.H., *An Interface Agent with Ontology-Supported User Models*, Master Thesis, Department of Electronic Engineering, National Taiwan University of Science and Technology, Taiwan, R.O.C., 2003.

[3] Hovy, E., Hermjakob, U., and Ravichandran, D., A Question/Answer Typology with Surface Text Patterns, *Proc. of the DARPA Human Language Technology conference*, San Diego, CA, USA, 2002, pp. 247-250.

[4] Noy, N.F. and McGuinness, D.L., Ontology Development 101: A Guide to Creating Your First Ontology, Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Tech. Rep. SMI-2001-0880, 2001.

[5] Rich, E., User Modeling via Stereotypes, *Cognitive Science*, Vol. 3, 1979, pp. 329-354.

[6] Soubbotin, M.M. and Soubbotin, S.M., Patterns of Potential Answer Expressions as Clues to the Right Answer, *Proc. of the TREC-10 Conference*, NIST, Gaithersburg, MD, USA, 2001, pp. 293-302.

[7] Tsai, C. H. MMSEG: A word identification system for Mandarin Chinese text based on two variants of the maximum matching algorithm. Available at http://technology.chtsai.org/mmseg/, 2000.

[8] Winiwarter, W., Adaptive Natural Language Interface to FAQ Knowledge Bases, *International Journal on Data and Knowledge Engineering*, Vol. 35, 2000, pp. 181-199.

[9] Yang, S.Y. and Ho, C.S., Ontology-Supported User Models for Interface Agents, *Proc. of the 4th Conference on Artificial Intelligence and Applications*, Chang-Hwa, Taiwan, 1999, pp. 248-253.

[10] Yang, S.Y., Chiu, Y.H., and Ho, C.S., Ontology-Supported and Query Template-Based User Modeling Techniques for Interface Agents, *2004 The 12th National Conference on Fuzzy Theory and Its Applications*, I-Lan, Taiwan, 2004, pp. 181-186.

[11] Yang, S.Y., An Ontology-Supported and Query Template-Based User Modeling Technique for Interface Agents, *2006 Symposium on Application and Development of Management Information System*, Taipei, Taiwan, 2006, pp. 168-173.

[12] Yang, S.Y., How Does Ontology Help Information Management Processing, *WSEAS Transactions on Computers*, Vol. 5, No. 9, 2006, pp. 1843-1850.