Knowledge Conceptualization Tool

Hiroko Fujihara, *Member*, *IEEE Computer Society*,
Dick B. Simmons, *Senior Member*, *IEEE*, Newton C. Ellis, and Robert E. Shannon

Abstract—Knowledge acquisition is one of the most important and problematic aspects of developing knowledge-based systems. Many automated tools have been introduced in the past, however, manual techniques are still heavily used. Interviewing is one of the most commonly used manual techniques for a knowledge acquisition process, and few automated support tools exist to help knowledge engineers enhance their performance. This paper presents a knowledge conceptualization tool (KCT) in which the knowledge engineer can effectively retrieve, structure, and formalize knowledge components, so that the resulting knowledge base is accurate and complete. The KCT uses information retrieval technique to facilitate conceptualization, which is one of the human intensive activities of knowledge acquisition. Two information retrieval techniques employing best-match strategies are used: vector space model and probabilistic ranking principle model. A prototype of the KCT was implemented to demonstrate the concept. The results from KCT are compared with the outputs from a manual knowledge acquisition process in terms of amount of information retrieved and the process time spent. An analysis of the results shows that the process time to retrieve knowledge components (e.g., facts, rules, protocols, and uncertainty) of KCT is about half that of the manual process, and the number of knowledge components retrieved from knowledge acquisition activities is four times more than that retrieved through a manual process. Furthermore, KCT captured every knowledge component that the knowledge engineer manually captured. KCT demonstrates the effectiveness of the knowledge acquisition process model proposed in this paper.

Index Terms—Knowledge acquisition, information retrieval, intelligent system, knowledge-based system, knowledge engineering, conceptualization.

1 Introduction

KNOWLEDGE-BASED systems are useful in business and industrial environments. However, their implementation has been obstructed by lacking reliable methods for acquiring expertise and constructing knowledge bases. Automated efforts have been under development [1], [2], [3], but these systems are still primarily research prototypes and not meant to handle all the complexity and resourcefulness of knowledge found in a knowledge acquisition process. Knowledge acquisition is one of the most important aspects of knowledge-based system development and possibly the most problematic process [1], [3], [4], [5], [6]. Many knowledge acquisition techniques and methods [1], [2], [3] have been studied, and various knowledge acquisition tools have been introduced to resolve the problems and difficulties associated with knowledge acquisition. In most cases, however, structuring concepts that involve understanding of semantics and contexts still must be handled by a knowledge engineer. Interviewing continues to be the primary method of acquiring expert knowledge, and it is useful to have a computerized knowledge conceptualization tool that helps knowledge engineers extract and structure knowledge from interview data.

This paper presents a Knowledge Conceptualization Tool (KCT) in which the knowledge engineer can filter, retrieve, categorize, and hierarchically structure knowledge contained in an interview. Accomplishing these tasks require the knowledge engineer to become fully familiar with the domain. However, not only are knowledge engineers scarce but it is difficult for any knowledge engineer to be familiar with multiple domains. The KCT supplements the knowledge engineer's lack of domain experience and it helps the knowledge engineer build up domain knowledge and structure knowledge components. The KCT was applied to the domain of a distributed network advisor, however, the application does not restrict to that particular domain; in most domains, KCT should be helpful when the knowledge structure is difficult to conceptualize due to the complexity of knowledge, involving multiple domain experts, and / or a large acquisition project.

KCT's input is an interview transcript, which contains several question and answer pairs consisting of unstructured conversational sentences. KCT parses the input and filters information irrelevant to the domain. The filtered interview transcript is then processed for clustering based on the semantic similarity and structuring based on contextual significance. KCT uses two information retrieval models [7], [8], [9] to provide a knowledge engineer with the capability of retrieving embedded knowledge from interview data and refining the knowledge through clustering and ranking processes.

KCT also offers varying degrees of automation in the conceptualization activity thereby reducing some of the tedious manual tasks associated with the process. The goal of KCT is to acquire complete and correct knowledge from domain experts where emphasis is placed on acquiring quality knowledge effectively and efficiently. A prototype

- *H. Fujihara is with the R&D Laboratory, Hewlett-Packard, Boise, ID 83714. E-mail: hiroko@boi.hp.com.*
- *D.B. Simmons, N.C. Ellis, and R.E. Shannon are with the Cognitive Systems Laboratory, Texas A&M University, College Station, TX 77843. E-mail: {simmons, ellis}@cs.tamu.edu.*

Manuscript received July 6, 1994; revised May 23, 1995.
For information on obtaining reprints of this article, please send e-mail to: transkde@computer.org, and reference IEEECS Log Number K97012.

of KCT is developed and tested using interview data obtained from industry experts. The result shows that KCT effectively assists the knowledge engineer obtain more detailed knowledge with higher accuracy in a shorter time than manually processing the same tasks, thus it increases the knowledge engineer's productivity.

This research was motivated through actual knowledge acquisition activities. While conducting knowledge acquisition sessions, the knowledge engineer realized that many of the tasks not only can be automated, but more symbol-level of knowledge (e.g., rules, facts, protocols, and uncertainty) should be extracted through a well-designed computer assisted processes. A hybrid approach of automated information retrieval and minimum human intervention presented in this paper helps the knowledge engineer structure and refine knowledge effectively.

2 Background

This section describes the problems found in knowledge acquisition activities and knowledge-based system development that KCT addresses and the importance of providing computer-aided assistance to knowledge engineers.

2.1 Knowledge-Based Systems and Expertise

Numerous knowledge-based systems have been developed since the 1980s [10], [11]. Development of knowledge-based systems is a major focus of research in artificial intelligence [3]. Although underlying principles of designing and building useful knowledge-based systems have been well-defined and widely known, knowledge-based systems are still difficult to construct and the building process is time-consuming [1]. The major difficulty in developing knowledge-based systems is in eliciting expertise that the domain expert uses to analyze and solve problems but that has never been codified or formulated into a knowledge base.

The concepts knowing how and knowing what are significantly different [12]. Fodor states [13] that knowing how generally involves tacit knowledge, whereas knowing what requires knowledge accessible to consciousness [3]. Tacit knowledge is neither easily separated into its components nor easily modified. On the other hand, conscious knowledge can be identified without too much difficulty. The difficulty in knowledge elicitation activity centers on the difficulty the knowledge engineer has in identifying and representing the tacit knowledge that an expert subconsciously processes.

2.2 Knowledge Acquisition

Achieving high-quality knowledge acquisition is not an isolated problem in knowledge-based system development. Knowledge acquisition must be considered in conjunction with knowledge refinement and maintenance. Automated learning from experience is useful when a system refines and updates knowledge continually [14]. Refining and updating knowledge occurs in all tasks in building a knowledge-based system, and such iterative refinement of knowledge should lead to building a valid operational knowledge-based system.

Although both the expertise-transfer tools and the empirical inductive tools are popular in knowledge acquisition research, manual knowledge acquisition approaches are more often used in the real world [2]. Interviewing by the knowledge engineer is one of the most commonly used manual knowledge acquisition approaches to extract knowledge from a domain expert. The extracted knowledge is later formalized into a knowledge base. Interviewing requires little equipment and can yield a considerable amount of knowledge if the knowledge engineer is skilled. Success of interviewing also highly depends upon the domain expert's ability to structure expertise and express it.

2.3 Knowledge Acquisition Life Cycle

Methodologies for building knowledge-based systems typically attempt to provide a life cycle model of software development [15]. The most widely supported view of a knowledge acquisition life cycle model (see Fig. 1) contains the five discrete stages described by Buchanan et al. [1]. Although it is quite difficult to develop actual knowledge-based systems in such a well-structured manner, the model captures the important aspects of a knowledge acquisition process.

The emphasis of the work presented in this paper is on the first two phases: identification and conceptualization. In the Identification phase, important problem aspects, including "participants, characteristics, resources, and goals" [1], are defined. The nature of the problem that needs to be solved, the data needed for problem solving, and the goals for the completed system are characterized and established by the domain expert working with the knowledge engineer.

In the Conceptualization phase, the primary concepts and relationships in the domain are explicitly specified. The knowledge engineer establishes the linguistic terms used by domain experts and attempts to define the concept represented by those terms. This phase produces a conceptual model of the problem area, which is characterized by tasks, the phenomenology of a domain, and the expert's problem-solving behavior. Later the conceptualized knowledge is formally implemented in a computer language and tested as Fig. 1 indicates.

2.4 Knowledge Acquisition Bottleneck

Knowledge acquisition is the bottleneck of a knowledge-based system development process [1], [4], [5], [6], [16], [17]. Some studies [18] put the blame on the lack of a formal theory for knowledge engineering. Others attribute the problem to the limited ability of experts to introspect on their problem-solving behavior and to encode their behavior within a computer representation framework [19]; the problem may come from the cognitive and linguistic barriers.

McGraw and Harbison-Briggs [2] argue that a weakness found in manual knowledge acquisition methods is due to poorly trained knowledge engineers. Besides the time and cost required for a knowledge engineer to become familiar with the domain to elicit knowledge efficiently, the knowledge engineer is required to use a variety of skills during the knowledge acquisition process. It is difficult to standardize the ability of knowledge engineers since various factors are involved.

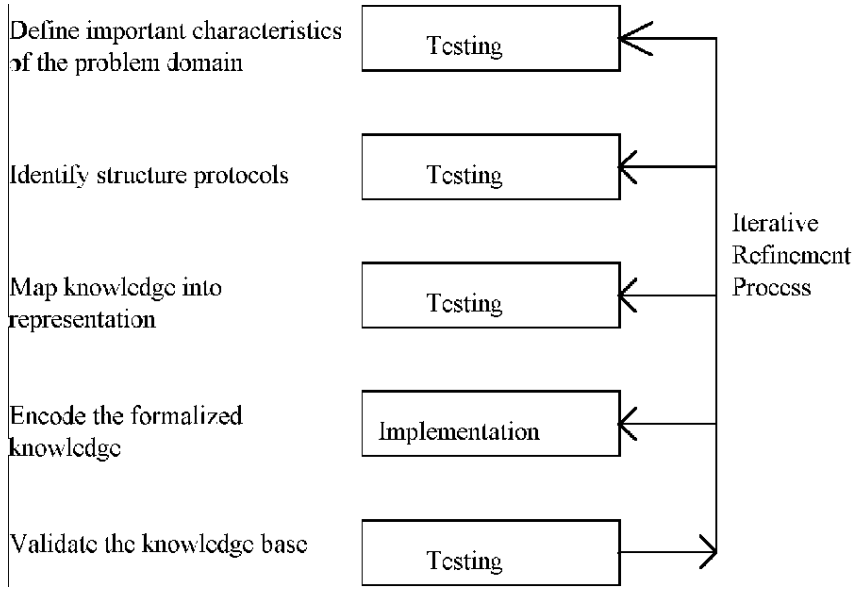

Fig. 1. Knowledge acquisition stages.[1]

2.5 KCT and Knowledge Acquisition Tool

Often knowledge acquisition research emphasizes semiautomatic tools that keep a human expert in the acquisition loop unlike machine learning research [20], [21], [22] that emphasizes automatic concept learning. Knowledge acquisition tools available in the field can be classified based on their features, level of automation, efficiency of use, life cycle support, the target users, and some other dimensions as shown in Boose's work [23].

KCT is a semiautomated tool of which the target user is a knowledge engineer. Before KCT is applied, formal knowledge elicitation sessions in a form of interviews between the domain expert and the knowledge engineer must be conducted during which the knowledge engineer records the domain expert's answers to questions. Each recorded session must be transcribed into a machine readable transcript form which nonrelevant conversation is filtered out. A domain specific key term list generated by the knowledge engineer is used to filter the transcript.

KCT is a domain independent tool, which requires very little training for its effective use. The focus of the tool is on the conceptualization phase of knowledge acquisition process. Many of domain-independent knowledge acquisition tools are used for diagnostic tasks [23], but very few, if any, for conceptualization tasks. In general, it is easier to automate tools that are domain dependent at the expense of built-in domain-specific problem-solving power [23].

The significance of KCT is its application and access to the most common manual knowledge acquisition techniques, whereas almost all knowledge acquisition tools establish a separate knowledge acquisition method suited for computer-based techniques. For example, well-known knowledge acquisition tools such as AQUINAS, ASK, and MORE use the interviewing knowledge acquisition technique, however, they interview the expert; in other words, the expert has to use the tool, which is not necessarily practical.

1. Adapted from *Building Expert Systems* by B. Buchanan et al., Addison-Wesley, Reading, Massachusetts, Copyright 1983.

3 Overview of KCT

The environment that KCT provides involves human input and judgment in its iterative refinement processes. KCT uses human judgment when it is preferred, while many processes are automated. KCT aims to facilitate the knowledge acquisition process with optimal use of a computer-aided environment. Optimal use of automated processes refers to that KCT helps generate a high quality knowledge base in terms of completeness, correctness, and clearness with minimal human intervention. The information retrieval techniques used in KCT help the knowledge engineer filter massive, unstructured information, and organize, understand, and structure it efficiently.

3.1 Using Information Retrieval Techniques in KCT

A number of information retrieval techniques have been introduced since the mid-1960s. Among these, the vector space model [24] and the probabilistic ranking principle model [8] are used in KCT. These two models have demonstrated their effectiveness to retrieve information from unstructured data [24], [25], [26], [27], [28], in which interviewing results. KCT deals with unstructured data as its input, which is a transcribed interview script. In an interview, a knowledge engineer asks questions and the domain expert answers, and a collection of question and answer pairs becomes the data file for KCT.

KCT is a tool that allows the knowledge engineer to structure and conceptualize knowledge without becoming intensely familiar with the domain. KCT provides the knowledge engineer with an effective computer-aided knowledge acquisition environment; it guides the knowledge engineer to categorize knowledge and spot the missing information by using divide-and-conquer strategy. It facilitates retrieving the fine granularity of knowledge, which can be easily formalized.

KCT takes some of underlying assumptions in the use of the two information retrieval techniques into account by

keeping a human in the analysis and refinement process. For example, the knowledge engineer removes any question and answer pairs that are obviously correlated to the other question and answer pairs during the transcript filtering process, as the terms being structured by the clustering technique are assumed uncorrelated. KCT learns from what it produces, and refines the outputs by applying the information retrieval techniques repeatedly. The refinement process is also used to infer relevance and semantic similarity of question and answer pairs. Iterative application of KCT along with manual analysis and validation helps the knowledge engineer to obtain and structure refined and thorough knowledge effectively.

3.2 Knowledge and Concept in KCT

KCT processes knowledge that has been interpreted, categorized, modified, and applied by the domain expert over many years to analyze and solve problems. Exemplified knowledge is concepts, constraints, heuristics methods for using probabilistic data, and procedures that govern domain-specific operations [11]. The total network of concepts and relations of different knowledge dimensions is represented as rules and facts, properties, uncertainties, and protocols. More specifically concepts generate rules and facts; constraints yields properties; heuristic methods determine how to manage uncertainty; and task analysis generates protocols. These become important components of a knowledge base that the knowledge engineer needs to retrieve.

3.3 Terminology and Notations Used

To make our later discussion clear to the readers, some of the terminology frequently used in this paper are clarified.

Question and Answer (qa) Pairs. Generally a document can be an article, a book, or a segment of text. When Salton [24] explains his clustering algorithm, he refers to a document as part of a file that contains information of interest. Robertson [8] similarly uses the word, document, for a chunk of information that can be ranked based on contextual relevance and statistical similarity. The chunk of information clustered and ranked in this work will be question and answer pairs. The knowledge engineer asks the domain expert a question(Q) and the domain expert responds with an answer(A) resulting in a question and answer(qa) pair.

Interview Transcript(IT). An interview of the domain expert by the knowledge engineer results in an interview transcript that is made up of *qa*s plus irrelevant conversation that must be filtered out before the IT is processed.

Key Terms **(k_1 ... k_N)**. During the Identification and Conceptualization phases of the knowledge acquisition life cycle, the knowledge engineer continuously creates a list of key terms that can be used later during the filtering, clustering, and ranking processes. The key terms are generally technical terms selected from the document to be processed.

Clustering Query. A clustering query is a collection of key terms that represent concept. The knowledge engineer composes a clustering query, which is denoted as $Q_{clustering}$.

Clustering. The works of Salton et al. define clustering [24], [29] as grouping information based on their similarity coefficient. Given a query, $Q_{clustering}$, which consists of a number of key terms clustering is performed. Using key terms, *qa*s in a document are categorized when associations between *qa*s show the joint relevance to a given $Q_{clustering}$.

Ranking. Ranking uses probability and statistical techniques to quantify relevance of *qa*s. The definition of ranking is based on Probabilistic Ranking Principle introduced by Robertson [8]. Given a ranking query, $Q_{ranking}$, that contains a collection of key terms, the ranking process determines significant *qa*s for a particular $Q_{ranking}$. The raking process produces a list of *qa*s with a quantified indication of the degree of relevance to the particular $Q_{ranking}$.

Concept of Relevance. A *qa* pair is said to be relevant if the *qa* covers a particular concept that is represented by a ranking query. Relevance is a binary variable that exists outside the system. The knowledge engineer uses subjective perception to determine relevance.

Concept Hierarchy. A concept hierarchy is a structural taxonomy that represents a common characteristic or relationship that is shared by objects, elements, or events that are otherwise different [30]. Concept hierarchies are usually formed as trees with more common associations at the higher level of the tree and less common ones at the leaves.

Weighting. A heuristic measure for ranking the *qa*s in response to each ranking query is chosen based on the concepts of relevance, *qa*s, and ranking query. Term weighting introduced by Robertson et al. [31] is an exploration of relevance information to weight terms that compose ranking queries. Term weighting is an established method that is justified by several statistical techniques [31]. A series of relevance weighting functions are derived within a theoretical framework [31]. KCT uses the Presence/Absence (P/A) scheme described in [31]. The P/A scheme assumes that

1) the distribution of key terms in relevant *qa*s are both independent, and
2) both the presence of the key terms and their absence from *qa*s determine the probable relevance.

3.4 Divide-and-Conquer Strategy

A two hour interview can easily exceed 100 *qa* pairs with an interview transcript of more than 50,000 bytes. Each *qa* pair contains many incomplete sentences as well as scattered information, as it is not uncommon for people to get sidetracked off of the original subject. Consequently, the resulting interview transcript contains domain irrelevant information along with information that is significant.

The knowledge engineer extracts necessary information from the transcript, and if any information is inconsistent, incomplete, or ambiguous, the knowledge engineer clarifies the information with the expert in the next interview session. One way to accomplish the task is to divide the bulk of information into categories, and structure information within each category. The activity allows the knowledge engineer to easily spot missing, inconsistent, or ambiguous information.

KCT has three phases. First, it filters the massive, unstructured *qa* data and extracts only relevant *qa* pairs. Second, it divides a large interview transcript of *qa* pairs into groups based on contextual similarity. Lastly, it ranks the *qa*s based on relative semantic significance to organize information as if it creates a semantic net. The resulting structures from the KCT process become trees composed of several levels.

3.4.1 Clustering Knowledge Using Vector Space Model

Clustering information is a common technique to facilitate a search mechanism. Salton et al. [24], [29], [31] demonstrate clustering as a search strategy since a clustered file provides efficient file access by limiting the searches to those clusters that are likely to be most similar to the corresponding queries. Clustering is said to provide effective retrieval of the wanted items whenever the so-called clustering hypothesis [24] holds. The hypothesis holds when "associations between documents based on their similarity coefficients indicate the joint relevance of documents to queries in a collection" [33], [34], [35].

Instead of categorizing an interview transcript manually, clustering is used in KCT. The knowledge engineer may not know exactly how to divide the interview transcript, IT, due to unfamiliarity of the domain, the large IT size, and/or unstructured information. Only if the knowledge engineer can identify key terms that represent the domain regardless of the degree of importance of each term, the clustering algorithm [24] groups *qa*s of the interview transcript into appropriate clusters based on the similarity factors.

3.4.2 Ranking Knowledge Using the Probability Ranking Principle

The objective of the Probability Ranking Principle model [8] is to rank items in the order of their probability of relevance to the ranking query, or of usefulness to the user, or of satisfying the user, given all the evidence available. The principle is based on the assumption that representation of both query and ranked items is uncertain, and the relevant relationship between them is also uncertain. The probabilistic information retrieval models take into account a variety of sources of evidence that could be used to estimate the probability of relevance of an item to a ranking query [25]. The typical source of such evidence is the statistical distribution of key terms in an interview transcript, and in relevant and nonrelevant *qa*s.

In KCT, ranking process is performed for each ranking query that is suggested from the previous clustering process, and corresponds to the concept hierarchy. Each ranking query, Q_{ranking}, contains key terms to express the concept. Ranking process determines if a *qa* is significant to a particular concept. If a *qa* is determined to be significant, then the degree of significance is determined relative to other *qa*s. The knowledge engineer would be able to use this information to construct a knowledge base and formulate rules, facts, relations, and properties.

4 KCT Architecture

KCT has three phases (see Fig. 2); the parser and filter phase, the clustering phase, and the ranking phase. Thorough discussion on each phase including required input and human intervention involved is given in this section.

4.1 Parser and Filter

When an interview is transcribed, the transcript tends to have irrelevant information. For example, social conversation is a irrelevant piece of information to KCT operation. Before the clustering and ranking phases, the KCT parses the original interview transcript and filters out irrelevant information. The resulting filtered and parsed interview transcript, IT, is composed of N question and answer pairs, qa_1 to qa_N. The knowledge engineer then develops a set of domain key terms, $k_1, \ldots, k_p$. The key terms are used to check if a piece of conversation contains any domain related issues. If not, it is filtered out of the original interview transcript. The key terms are used to create N question and answer pairs $qa_1 \ldots qa_n$, of length p. The p elements of the qa_i, vectors correspond to the coefficients representing the presence or absence of key terms $k_1, \ldots, k_p$. The array of *qa*s represent the interview transcript, IT. A subset of the key terms is used to query the KCT Cluster and Ranking Units.

4.2 Clustering Unit

The Clustering Unit processes an interview transcript IT made up of vectors qa_1 to qa_N using the vector-space model [24]. The vector space model assumes that an available term set is used to identify stored records. The qa_1 can then be represented as term vectors of the form

$$Q_i = (a_{i1}\, a_{i2} \ldots a_{ik})$$

and

$$A_i = (b_{i1}\, b_{i2} \ldots b_{ik})$$

where the coefficients a_{ik} and b_{ik} represent the values of term k in question Q_i and answer A_i. Typically a_{ik} (or b_{ik}) is set equal to 1 when term k appears in question Q_i (or answer), and to 0 when term is absent from the vector.

Consider now a situation in which t distinct terms are available to characterize record content. Each of the t terms can then be identified with a term vector T, and a vector space is defined whenever the T vectors are linearly independent. In such a space, any vector can be represented as a linear combination of the t term vectors. Hence the rth question Q_y and answer A_y can be written as

$$Q_r = \sum_{i=1}^{t} a_{ri} T_i$$

and

$$A_r = \sum_{i=1}^{t} b_{ri} T_i$$

where a_{ri}s and b_{ri} s are interpreted as the components of Q_r and A_r along the vector T_i.

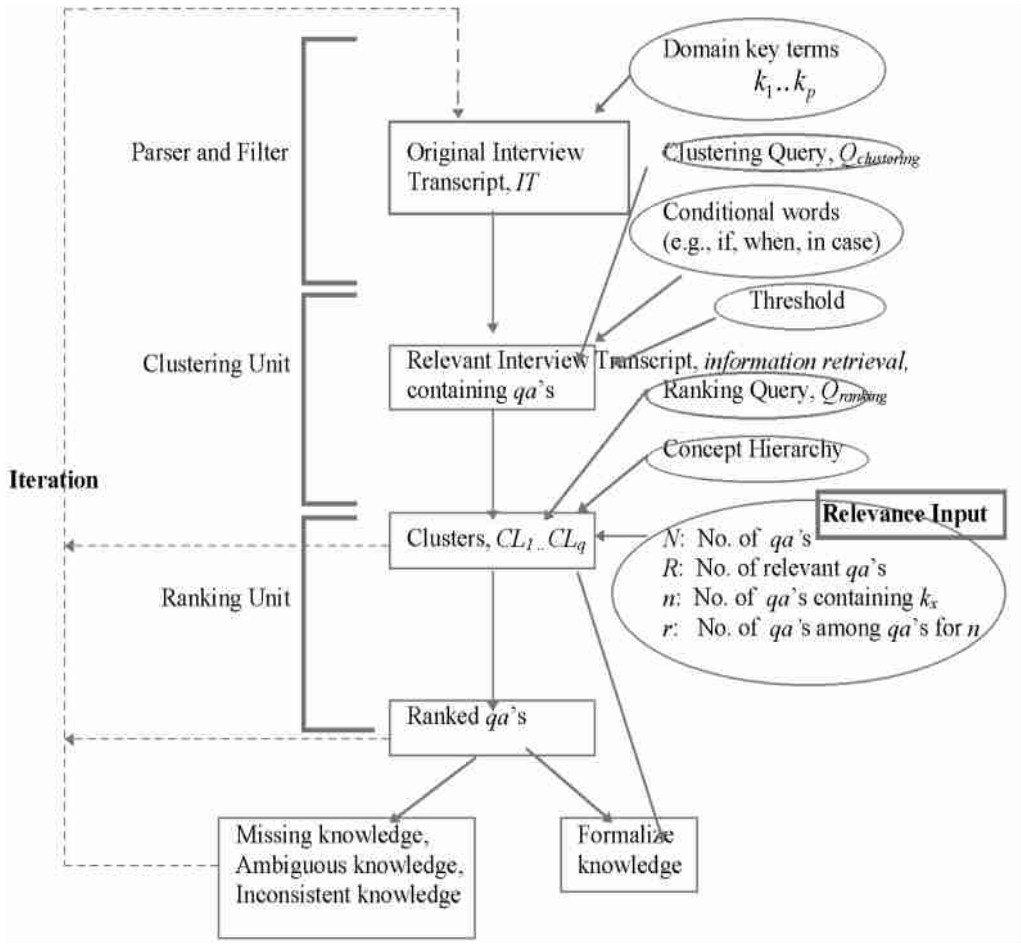

Fig. 2. KCT architecture.

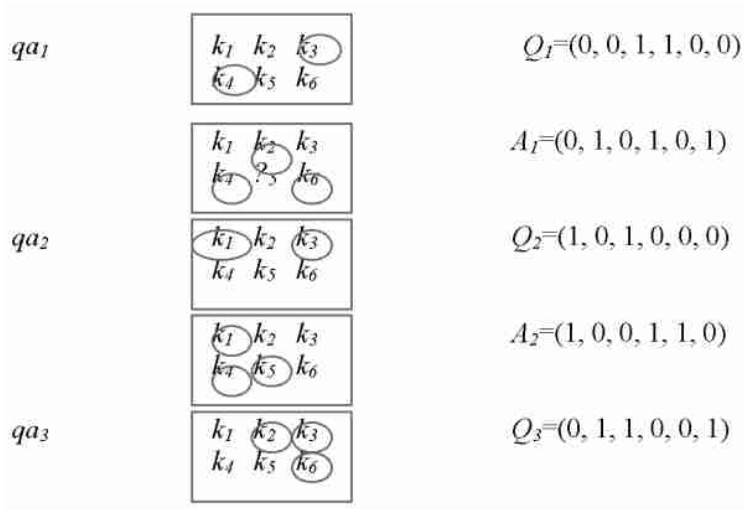

Fig. 3. Example of interview transcript similarity computations.

Fig. 3 illustrates equations above. Each of the five transcript questions and/or answers (Q_1, A_1, Q_2, A_2, and Q_3) contains one or more of the terms $k_1, \cdots, k_t$ where $t = 6$. The terms that the qa_i contains are circled in Fig. 3, and each Q_i, A_i becomes as shown in Fig. 3. By using binary term vectors instead of weighted term vectors, the equations for Q_r and A_r become

$$Q_r = |Q_r|$$

and

$$A_r = |A_r|$$

where $|X|$ =number of terms in X.

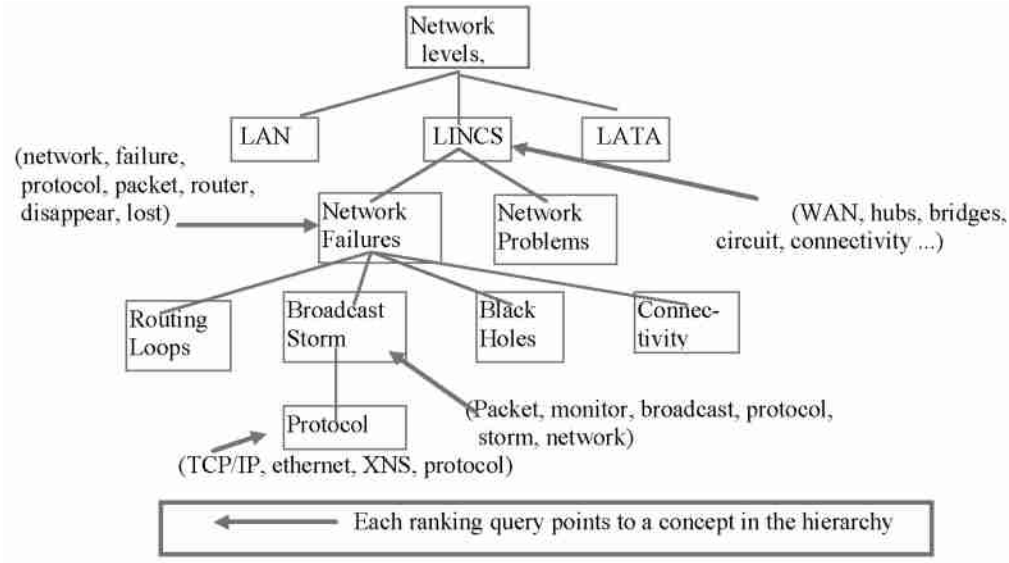

Fig. 4. An example of concept hierarchy and ranking query generation.

The interview transcript *IT* can be represented as the following matrix

$$IT = \begin{array}{c} \\ Q_1 \\ A_1 \\ Q_2 \\ A_2 \\ \vdots \\ \vdots \\ Q_N \\ A_N \end{array} \begin{array}{c} \begin{array}{cccc} T_1 & T_2 & \cdots & T_N \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ b_{11} & b_{12} & \cdots & b_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ b_{21} & b_{22} & & b_{2N} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \\ b_{N1} & b_{N2} & \cdots & b_{NN} \end{bmatrix} \end{array}$$

To avoid the term-correlation problem described by Salton [24], the terms are assumed to be uncorrelated, in which case the term vectors are orthogonal (that is, $T_i \cdot T_j = 0$, except when $i = j$ and $T_i = T_j$). When the t term vectors are orthogonal, linear independence follows automatically, and the t term vectors form a proper basis for the vector space. The similarity $sim(X, Y)$ between any question or answer X and any other question or answer Y can be measured by the cosine coefficient similarity measure [24] as follows

$$sim(X, Y) = \frac{|X \cap Y|}{|X|^{\frac{1}{2}} \cdot |Y|^{\frac{1}{2}}}$$

where $|X \cap Y|$ = number of terms appearing jointly in X and Y.

Calculate similarity values for all *qa*s compared to each of the other *qa*s that have key terms included in CL_i. A threshold, N, is chosen. Create the first cluster CL_1 from qa_1 and compute similarity between each of the other *qa*s based on similarity to the first segment. Use N to determine if similarity is large enough for *qa*s to be associated with the cluster. If the similarity is above or equal to the N, then the *qa* is associated with the current cluster. If the similarity is below N, then a new cluster is created and the *qa* and its vector is associated with the new cluster. Refine clusters by merging all isolated clusters with clusters which are sufficiently similar.

The Clustering Unit also contains a list of key words (e.g., if, then, when, in case, why), that help identify rules and the resulting facts that can be combined similarity clusters to create concept hierarchies.

4.3 Ranking Unit

The KCT Ranking Unit accepts the interview transcript IT that contains N *qa*s as input. The knowledge engineer manually scans the IT to select R *qa*s that he/she subjectively considers to be relevant. Next, concept hierarchies are built.

4.3.1 Concept Hierarchy for KCT

The domain expert and the knowledge engineer work together to build concept hierarchies. Building a concept hierarchy is a much easier task than formalizing knowledge as it is a higher level of task. Fig. 4 is a sample concept hierarchy based on an actual interview with a computer network administrator.

After the concept hierarchies are built, the knowledge engineer builds a ranking query, $Q_{ranking}$, that contains one or more key terms, $k_1 \ldots k_t$, representing the concept as shown in Fig. 4. Relevant *qa*s for a ranking query are selected as the knowledge engineer reads the interview transcript. Relevant *qa*s for every ranking query are stored under their *qa* number along with the corresponding concept. A ranking query is generated for every concept in the concept hierarchies. The list of queries is stored in a file for ranking processing by KCT. The concept broadcast storm, for example, is represented by the query, which would become (packet, monitor, broadcast, protocol, storm, network). Generating ranking queries are highly leveraged from the clustering process, because the clusters generated through clustering algorithm often represent a sibling of a concept tree.

4.3.2 Ranking Process

A query is generated for every concept in the concept hierarchies. Once the concept hierarchy key terms for $Q_{ranking}$

Cluster No. 9

Q8 Knowledge Engineer: You say sites. Would each floor of a building be a site?
A8 Domain Expert: I am talking about a physical building is a site.
Q10 Knowledge Engineer: But, is this building one of the 24 sites?
A10 Domain Expert: Yes.
A18 Domain Expert: The farthest site is typically using 90% of their bandwidth average.
A71 Domain Expert: A site that is typically using 90% of their bandwidth average of 56KB. If they are moved to a T1 that usually drops to above 7%.

Fig. 5. Cluster no. 9 (site).

have been chosen, then KCT determines the number, *n*, of *qa*s that have at least one of the key terms and the number, *r*, of *qa*s that are among the *R* relevant *qa*s. The KCT ranking heuristics are then derived using the probability ranking principle [8], [26] and empirically derived weighting coefficients. The output of the ranking process is a list of *qa*s with a quantified indication of the degree of relevance, which indicates the relevance of *qa*s to each ranking query. The knowledge engineer would be able to use this information to construct a knowledge base and formulate rules, facts, properties, etc. relating to the concept tree provided by the domain expert.

5 Discussion

KCT is used in a real environment, where knowledge acquisition activities with domain experts are carried out and a knowledge-based system was built. Real outputs from both clustering process and ranking process are used to explain how KCT is used to conceptualize knowledge from an unstructured interview transcript obtained during real knowledge acquisition activities. The results are discussed in this section.

5.1 Knowledge Domain Processed by KCT

The Cognitive Systems Laboratory at Texas A&M University is developing a Distributed Network Advisor that helps computer users manage and maintain large, interconnected local-area networks (LANs). The Distributed Network Advisor monitors the LAN, looking for errors and indications of performance problems such as excessive packet retransmissions. When a problem is found, it uses its knowledge of the network topology and LAN troubleshooting techniques to locate the problem.

The problem domain involved three domain experts; one for networks inside buildings and between buildings, one for wide area networks inside a state, and the other for wide area networks across five states. A knowledge engineer was assigned to each domain expert. The problem domain was complex and required each knowledge engineer to learn not only basic network terminology and concepts, but inter-relationships between subdomains. Having three domain experts increased difficulty of the knowledge acquisition process. Each expert had his own style of explaining problems; three knowledge engineers must combine knowledge from each expert into a seamless knowledge-based system. In order to accomplish the knowledge acquisition activities, the knowledge engineers in the Distributed Network Advisor project had several interview sessions with domain experts. The knowledge engineers recorded and then transcribed each interview.

The most difficult task during the knowledge elicitation activity was to identify completeness, consistency, and correctness of knowledge obtained from domain experts. Earlier interview sessions were rather unstructured interviews, and the knowledge engineers asked general questions and record the contents as much as possible. After a rough concept was grasped, focused and structured interviews were used more often to cover a list of topics and/or a list of specific questions relating features of systems experts were using.

5.2 Clustering Output from KCT

Fig. 5 shows one of the clusters that is generated using the vector space model described previously. The cluster no. 9 provides information about the network site that can be used to formalize rules and facts. The following 1) through 4) are facts about sites and the 5) indicates a rule that can be used to diagnose "slow response" on the network:

1) Each floor is a site.
2) There are 24 sites.
3) The farthest site is about 80 miles away.
4) Site usage is 90 percent of its bandwidth.
5) IF T1 is used, THEN site usage drops to 7 percent of the average T1 bandwidth.

The result shows the knowledge engineer how much knowledge has been acquired about the concept. Furthermore, the knowledge engineer would find it much easier with clustering knowledge than without it to formalize the knowledge into rules and facts.

5.3 Ranking Output from KCT

KCT produces meaningful interview transcript ranking lists. For example, Fig. 6 indicates that qa_{40} discusses network protocol extensively and qa_{109} and qa_{35} discuss the same topic to a lesser extent. The qa_{103}, qa_1, and qa_2 address the same topic, but to a much lesser degree than the other

Ranking Query No. 11: (TCP/IP, Ethernet, XNS, protocol)

qa number	Value used for ranking
40	6.693540
109	4.770155
35	3.846770
103	1.923385
1	0.923385
2	0.923385
...	...
71	0.923385

Fig. 6. An example of ranking result with query no. 11.

qa_{35}

Knowledge Engineer: What is in the database?
Domain Expert: List of all sites and list of all equipment. Another one which shows if I am mainly working on a TCP/IP or XNS basis, simply by a yes or no ...

qa_{40}

Knowledge Engineer: You are concerned with XSN, TCP/IP, LAT, and probably some others?
Domain Expert: Well, those are all protocols running on the ethernet. There is also Appletalk. That would be main one.

qa_{109}

Knowledge Engineer: What is this trouble ticket about?
Domain Expert: This is something we just run into with TCP/IP. Apparently it takes a lot more memory to manage a session with a TCP/IP protocol. They are supposed to be able to handle up to 40 sessions on a server. So we started to look into how to reconfigure the memory allocation in the server itself.

Fig. 7. *qa* pairs ranked high by query no. 11.

*qa*s ranked above them in Fig. 6. It may be necessary to emphasize that this ranking is not based on a simple keyword pattern matching algorithm. The relevance of *qa*s to the concept throughout the interview transcript and the relative relevance of a concept to other concepts are taken into account. The underlying idea is that a rare term is valued more due to its specificity, hence it receives a higher weight. Fig. 7 shows actual interview transcript of qa_{40}, qa_{109}, and qa_{35}.

qa_{40} enables the knowledge engineer to conceptualize the organization of protocol as shown in Fig. 8. Protocols that run on Ethernet are XNS, TCP/IP, LATA, and Appletalk. Also the fact formalized is that the main protocol is Appletalk. Further knowledge about protocols can be acquired from qa_{109} and qa_{35} similarly.

Fig. 9 reveals that the knowledge engineer needs to look over the query and determine if the terms used for the query properly represent the concept. KCT found that a ranking query represents two concepts rather than a concept. Thus, two different rankings were produced and flagged the knowledge engineer that the terms in the first set do not relate to the second set in any *qa*s. KCT ranks *qa*s as if it were given two sets of queries when terms representing a concept do not intersect. The result also indicates that qa_{14} and qa_{53}, both previously selected by the knowledge engineer as relevant *qa*s appear to be nonrelevant or much less relevant to the concept. This is a useful feature the ranking process provides for the knowledge engineer who is new to a particular domain.

A total of 13 queries is given by the knowledge engineer, and the results obtained show the well-constructed structure of knowledge for each concept. The result is useful since KCT has captured more complete knowledge and structured it in a helpful way. The knowledge obtained from KCT is compared later with the result without using KCT.

5.4 Relevance Prediction

Predicting relevance of information is one of the goals of KCT. In the past, many researchers [8], [32], [33] attempted to estimate relevance of information, however, the results were not successful.

Relevance weighting is used to derive an optimal ranking rule for *qa*s. One assumption to support the optimal ranking rule is that the available information consists of terms distributional information; information concerning the frequencies of occurrences of terms in relevant and other *qa*s as dichotomous criterion variable. That is, a term is observed if it occurred or did not occur in both groups of *qa*s, namely relevant *qa*s and nonrelevant *qa*s. It is also assumed that no term co-occurrence information is given. Thus, occurrence of a term is independently observed throughout *qa*s. The ordering principle that is the basis for the formal analysis of the optimal ranking rule is that both the presence of the terms and their absence from *qa*s determine the probable relevance.

The results indicate that prediction of relevance may be possible using the weighting function [32], however, dependable prediction could not be obtained from our experiment. Only two out of 13 cases had lower relevance values than retrospective performance; that is to generate ranking using a human judgment. The two cases that had lower relevance values were those that had disjoint sets

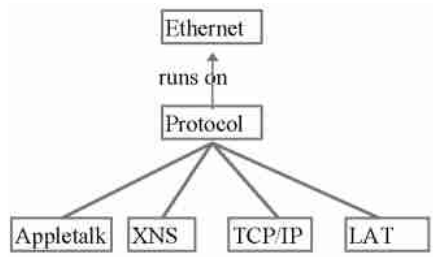

Fig. 8. Protocol organization retrieved from ranking process.

Ranking Query No. 7-1 (DEC, DEC server)
Ranking Query No. 7-2 (CS-200)

Ranking with respect to ranking query 7-1

qa Number	Value used for ranking
13	2.000
17	1.983245
14	0.0000
53	0.0000

Ranking with respect to ranking query 7-2

13	0.00000
14	–1.193820
53	–1.193820

Fig. 9. An example of ranking result with query no. 7.

explained in this section. However, predicting relevance could not be formalized in the experiment, and more research is needed.

6 Analysis of the Results

The effectiveness of KCT is evaluated by

1) measuring its performance with process time, and
2) the number of rules and facts generated.

This information is compared to a similar evaluation of the manual process. As stated, the actual knowledge acquisition sessions were conducted with three knowledge engineers at the Cognitive Systems Laboratory at Texas A&M University and with the domain experts of a telecommunication company. Table 1 summarizes the data from one of the knowledge acquisition sessions by

1) comparing the formalized knowledge obtained by each method in terms of the number of details captured, and
2) comparing the process time of each method from the identification phase of the knowledge acquisition life cycle until the formalization phase.

6.1 Performance Time

A knowledge engineer has to read the interview transcript repeatedly to capture rules and facts if conceptualization is processed manually. The transcript was about 49K byte in size and it contained 125 *qa* pairs. It took approximately 6 hours for a knowledge engineer to go through the transcript to capture 24 rules and 33 facts. The process time of KCT in the first phase (parse and filter) was processed in approximately 2 minutes; the second phase (clustering phase) was in approximately 3 minutes with 0.6 threshold; and the third phase (ranking phase) was processed in less than 2 minutes.

6.2 Actual Number of Rules and Facts Captured

Clustering process captured 47 rules and 44 facts; the clustering query continued close to 247 domain and conditional terms. Thirteen ranking queries consisting of 72 terms were fed into the ranking process, and the ranking results were similar to those shown in Figs. 6 and 9. The output clearly shows that the ranking process not only found relevant *qa* pairs to the ranking query but also gave significantly useful information.

Table 1 shows that the total performance time of the manual process was 360 minutes versus 172 minutes for KCT. The performance time of KCT was only 47.7 percent of manual process. The number of rules obtained through the manual process was 24 versus 47 from the KCT clustering process and 53 from the ranking process. The manual process only captured 24 percent of the rules that KCT captured. Similarly, 33 facts were captured by the manual process, whereas the clustering process captured 44 facts and the ranking process captured 87 facts. The total number of facts captured by KCT (131) became close to four times more than what the manual process captured.

All the facts and rules manually captured were included in KCT. Although the quality of knowledge is difficult to measure, the total number of rules and facts that KCT retrieved would seem to indicate that KCT captured much deeper and thorough knowledge. It is important to note that KCT did not miss any facts or rules captured by the knowledge engineers manually. KCT also gives the knowledge engineer a clear view of how much knowledge has been obtained and what still needs to be acquired. Giving an overview of a knowledge acquisition process status was found helpful for the knowledge engineer to continue a knowledge acquisition process.

7 Conclusion and Future Research Directions

This research describes a knowledge acquisition conceptualization tool, KCT, that helps the knowledge engineer conceptualize knowledge in multiple domains. KCT allows the knowledge engineer to filter, classify by topics, and structure knowledge segments contained in an unstructured interview transcript. Although many knowledge acquisition tools have been introduced, most of them are still research prototypes and limit their uses to a particular domain. KCT has proven to be very useful in helping identify knowledge structure protocols during the conceptualization stage of a distributed network advisor project where multiple domain experts and knowledge engineers were involved. It is our opinion that KCT should help conceptualize knowledge structures for other large complex knowledge acquisition projects.

KCT is a domain-independent semiautomated tool, which uses two crucial information retrieval techniques. Vector Space model [24] is effective not only to retrieve wanted information but to group information based on relevance of knowledge and its similarity. Probability Ranking Principle [8] helps organize information based on

TABLE 1
Comparison between KCT and Manual Performance

knowledge acquisition Process Model	Manual Process	KCT	
		Clustering	Ranking
Preprocessing time (min.)	0	2	165
Processing time (min.)	360	3	2
No. of rules captured	24	47	53
No. of facts captured	33	44	87

estimated probability of relevance of information not simply by pattern matching or frequency of appearance of terms, but all the evidence available. Employment of these two information retrieval techniques is significant when problem domain is complex and large, and initial elicitation procedure is unstructured and domain is not familiar to the knowledge engineer or the domain expert is not familiar with knowledge acquisition activities. Proper analysis and review by the knowledge engineer enhances the validity and accuracy of the structured knowledge from the automated process of KCT using information retrieval techniques. KCT is made to learn relevance of knowledge and eventually refine knowledge retroactively. It also requires very little human validation and analysis. However, KCT has not shown significant result in its prospective.

A knowledge acquisition process is normally human intensive process, and sensible applications of automation should improve productivity and efficiency. That is, activities in a knowledge acquisition process can be accomplished in a shorter time. The degree of computerized processes used should be optimal. Human intervention should be used when it improves quality, validity, and/or productivity. KCT requires a machine readable transcript as its input. Thus, the recorded session must be transcribed. Transcribing knowledge acquisition interview sessions is fairly human intensive work. As a future direction, automated transcription methods such as voice recognition need to be explored.

A computer-aided environment must be easy to use and a natural extension to human thinking. It should require as little training as possible. The tool presented here is a tool that represents a first step to build such an environment.

REFERENCES

[1] B.G. Buchanan, D. Barstow, R. Bechtal, J. Bennett, W. Clancey, C. Kulikowski, T. Mitchell, and D.A. Waterman, "Constructing an Expert System," *Building Expert Systems*, F. Hayes-Roth, D.A. Waterman, and D.B. Lenat, eds., Addison-Wesley, Reading, Mass., 1983.

[2] K.L. McGraw and K. Harbison-Briggs, *Knowledge Acquisition: Principles and Guidelines.* Englewood Cliffs, N.J.: Prentice Hall, 1989.

[3] M.A. Musen, *Automated Generation of Model-Based Knowledge-Acquisition Tools*. San Mateo. Calif.: Morgan Kaufmann, 1989.

[4] F. Allard, "A User's View of Current Practice and Possibilities," *Proc. EKAW '92: Sixth European Knowledge Acquisition Workshop—Current Developments in Knowledge Acquisition*, pp. 1-6, Springer-Verlag, New York, May 1992.

[5] G. Johannsen and J.L. Alty, "Knowledge Engineering for Industrial Expert Systems," *Atomatica*, vol. 27, pp. 97-114, 1989.

[6] M. Cookson, J. Holman, and D. Thompson, "Knowledge Acquisition for Medical Expert Systems: A System for Eliciting Diagnostic Decision Making Histories," *Research and Development in Expert Systems*, M.A. Bramer, ed., Cambridge Univ. Press, London, 1985.

[7] G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*. New York: McGraw-Hill, 1983.

[8] S.E. Robertson, "The Probability Ranking Principle in Information Retrieval," *J. Documentation*, vol. 33, pp.294-304, Dec. 1977.

[9] T. Pearl, *Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference*, San Mateo. Calif.: Morgan Kaufmann, 1988.

[10] E. Feigenbaum, P. McCorduck, and H. Nii, *The Rise of the Expert Company*. New York: Times Books, 1988.

[11] D. Waterman, *A Guide to Expert Systems.* Reading, Mass.: Addison-Wesley, 1988.

[12] G. Ryle, *The Concept of Mind.* New York: Barnes and Noble, 1949.

[13] J.A. Fodor, "The Appeal of Tacit Knowledge in Psychological Explanation," *J. Philosophy*, vol. 65, pp. 627-640, 1968.

[14] A. Admodt, "A Computational Model of Knowledge-Intensive Learning and Problem Solving," *Current Trends in Knowledge Acquisition*, B. Wielinga, J. Boose, B. Gaines, G. Schereber, and M. Someren, eds., pp. 1-20, IOS Press, Washington, D.C., 1990.

[15] B. Gaines, "Knowledge Acquisition: The Continuum Linking Machine Learning and Expertise Transfer," *Proc. Third European Workshop on Knowledge Acquisition for Knowledge-Based Systems*, J. Boose, B. Gaines, and J.G. Ganascia, eds., Paris, 1989.

[16] M.A. Musen, "Conceptual Models of Interactive Knowledge Acquisition Tools," *Knowledge Acquisition*, vol. 1, no. 1, pp. 73-88, 1989.

[17] F. Hayes-Roth, D. Waterman, and D. Lenat, *Building Expert Systems*. Reading, Mass.: Addison-Wesley, 1983.

[18] J. Breuker, B. Wielinga, M. Someren, R. De Hoog, G. Schreiber, P. De Greef, B. Bredewe.g., J. Wielemaker, and J.P. Billault, "Model-Driven Knowledge Acquisition: Interpretation Models," Technical Report P1098, Univ. of Amsterdam, the Netherlands, 1987.

[19] H.L. Dreyfus, "From Micro-Worlds to Knowledge Representation: AI at an Impasse," *Mind Design*, J. Haugeland, ed., pp.161-204, MIT Press, Cambridge, Mass., 1981.

[20] T.M. Mitchell, R M. Keller, and S.T. Kedar-Cabelli, "Explanation-Based Generalization: A Unifying View," *Machine Learning*, vol. 1, pp. 47-80, 1986.

[21] D.H. Fisher, "Knowledge Acquisition via Incremental Conceptual Clustering," *Machine Learning*, vol. 2, pp. 139-172, 1987.

[22] J.R. Quinlan, "Induction of Decision Tree," *Machine Learning*, vol. 1, pp. 81-106, 1986.

[23] J. Boose, "A Survey of Knowledge Acquisition Techniques and Tools," *Knowledge Acquisition*, vol. 1, no. 1, pp. 3-37, Nov. 1988.

[24] G. Salton, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Reading, Mass.: Addison-Wesley, 1989.

[25] S.E. Robertson, M.E. Maron, and W.S. Cooper, "Probability of Relevance: A Unification of Two Competing Models for Document Retrieval," *Information Technology: Research and Development*, vol. 1, no. 1, pp. 1-21, 1982.

[26] H.R. Turtle and B. Croft, "A Comparison of Text Retrieval Models," *The Computer J.*, vol. 35, no. 3, pp. 279-290, 1992.

[27] G. Salton, "Global Text Matching for Information Retrieval," *Science*, pp. 1,012-1,015, Aug. 1991.

[28] G. Salton, "Development in Automatic Text Retrieval," *Science*, vol. 253, pp. 974-980, 1991.

[29] G. Salton, A. Wong, and C.S. Yang, "A Vector Space Model for Automatic Indexing," *Comm. ACM*, vol. 18, no. 11, pp. 613-620, 1975.

[30] J. Kagan, *Psychology: An Introduction*, second edition. New York: Harcourt Brace Jovanovich, 1972.

[31] G. Salton, "Automatic Text Analysis Science," *Science*, no. 168, pp. 335-343, 1970.

[32] S.E. Robertson and K.S. Jones, "Relevance Weighting of Search Terms," *J. Am. Soc. for Information Science*, vol. 27, pp. 129-146, 1976.

[33] C.J.v. Rijsbergen, *Information Retrieval*, second edition. London: Butterworths, 1979.

[34] E.M. Voorhees, "The Cluster Hypothesis Revisited," *Proc. Eighth Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, pp. 188-196, ACM, New York, June 1985.

[35] N. Jardine and C.J.v. Rijsbergen, "The Use of Hierarchic Clustering in Information Retrieval," *Information Storage and Retrieval*, vol. 27, pp. 129-146, 1976.

Hiroko Fujihara received the BS and MS degrees in computer science from the University of Texas, Tyler, in 1988 and 1989, respectively; and the PhD from Texas A&M University in 1993. She has been a software design engineer at Hewlett-Packard, Boise, Idaho, since 1993. Her research interests include software engineering, software model and metrics, artificial intelligence, document recognition, and process improvement. Dr. Fujihara is a member of the ACM and the IEEE Computer Society.

Dick B. Simmons (S'59-M'62-SM'78) received the BS degree in electrical engineering from Texas A&M University, College Station; and the MS and the PhD degrees in electrical engineering and computer and information sciences, respectively, from the Moore School of Electrical Engineering at the University of Pennsylvania, Philadelphia. He served in the U.S. Army Signal Corps and has held positions with RCA and Bell Telephone Laboratories. In 1970, he moved to Texas A&M University, where he has served in several teaching, research, and administrative positions. For 10 years, he directed the growth of the Texas A&M University System computer network from a relatively small computer center to a centralized computer network serving 1,700 interactive terminals and remote entry stations. In addition to directing the computer network, he has served as principal investigator on research projects sponsored by the American Association of Railroads, Boeing Computer Services, CDC, Floating Point Systems, General Dynamics, IBM, NASA, Texaco, the State of Texas, Hewlett-Packard, Rockwell International, Objective Systems Integrators, Southwestern Bell Telephone, and the U.S. Air Force. His current research interests are software engineering, software models and metrics, and knowledge-based systems. Dr. Simmons is a senior member of the IEEE and a member of the ACM, AAAI, and American Society for Engineering Education. He has served on the IEEE Board of Directors and as president of the IEEE Computer Society, Amdahl Users Group, and Data General Users Group.

Newton C. Ellis received his PhD in experimental psychology from Texas Christian University in 1964. Currently, he serves in two roles at Texas A&M University: professor of industrial engineering and co-director of the Cognitive Systems Laboratory. Dr. Ellis has 14 years of industrial experience. At various companies, he served in such capacities as human factors engineer, senior engineer, group supervisor, senior scientist, and executive vice president. Dr. Ellis joined Texas A&M with a dual research and teaching appointment. He teaches courses in human factors, HCI, and expert systems. The current focus of his research is intelligent systems and human computer interaction. He is a fellow in both the Human Factors and Ergonomics Society and the Institute of Industrial Engineers. He received the IIE David F. Baker Award for Distinguished Research in 1995. Dr. Ellis is a registered professional engineer, a licensed psychologist, and a certified safety professional.

Robert E. Shannon is a professor of industrial engineering at Texas A&M University. His current research is in expert- and knowledge-based simulation systems and telecommunication systems. Dr. Shannon is the author or co-author of three books, *Systems Simulation: The Art and Science*, which won the H.B. Maynard Technical Book of the Year award and has been translated into Russian, Japanese, and Spanish (Prentice Hall), *Engineering Management* (John Wiley and Sons), and *Introduction to Simulation Using Siman* (McGraw-Hill).