6-2006

Cognitive mapping techniques for user-database interaction

Keng SIAU
Singapore Management University, klsiau@smu.edu.sg

X. TAN

Citation

SIAU, Keng and TAN, X.. Cognitive mapping techniques for user-database interaction. (2006). *IEEE Transactions on Professional Communication*. 49, (2), 96-108.
Available at: https://ink.library.smu.edu.sg/sis_research/9382

Cognitive Mapping Techniques for User–Database Interaction

—KENG SIAU AND XIN TAN

Abstract— *In this paper, we first develop a framework of user–database interaction. Based on this framework, we then provide a discussion on how notable human factors influence various dimensions of user–database interaction. Following that, we propose using cognitive mapping techniques to overcome some cognitive and behavioral biases during user–database interaction. Three popular cognitive mapping techniques—causal mapping, semantic mapping, and concept mapping—are introduced as techniques to elicit an individual's belief systems regarding a problem domain. Through an example database application, we demonstrate how to use these cognitive mapping techniques to improve user–database interaction. Finally, we discuss the implications of this research for technical communicators during user–database interaction analysis and design.*

Index Terms—*Cognitive mapping, technical communication, user–database interaction.*

In the Information Age, databases are vital for utilizing information resources. Individuals make use of databases to maintain valuable personal data or to obtain information that concerns them one way or another. Organizations use databases as a type of critical resource in order to function properly and to compete effectively. The pervasive adoption of Enterprise Resource Planning (ERP) systems by organizations worldwide reflects a trend to integrate critical organizational information into a central database, which can provide various stakeholders with unified data. With the increasing availability of the internet and the emergence of wireless computing, the ability to access databases anywhere anytime is expected and demanded.

The utilization and usability of databases depend on user–database interface. A user–database interface is where the interaction between users and the database system takes place. In this paper, we are interested in user–database interaction (i.e., database query). Efficient and effective use of databases requires users to be able to formulate queries accurately and quickly [1]. However, user–database interaction, especially for novice users, remains an area that is poorly explored and understood [2]–[6]. This has become a critical issue as the spread of information technology has given more end-users access to sophisticated database systems [7]. It is, therefore, imperative to extend and expand the study of user–database interface so as to provide novice-friendly and intuitive database interfaces that enable end-users to perform reliable database retrieval and to update their own databases.

Manuscript received November 9, 2004; revised June 1, 2005.
The authors are with the Department of Management, University of Nebraska – Lincoln, Lincoln, NE 68588 USA (email: ksiau@unl.edu; xtan3@unl.edu).

IEEE DOI 10.1109/TPC.2006.875074

Many communication issues exist in user–database interaction. When we analyze and design database interfaces, we should consider not only technical issues but also human factors, such as cognitive constraints and individual knowledge. Without a comprehensive understanding of human factors in user–database interaction, it is difficult to design an effective and efficient user–database interface. By developing a framework of user–database interaction, this paper contributes to identifying and understanding these human factors.

Cognitive mapping techniques have been widely used in strategic management and political science to depict and explore the cognitive structures of members of organizations [8]. Some researchers have hinted at the usefulness of cognitive mapping in systems development (e.g., [9]–[12]) and in technical communication (e.g., [13], [14]). This paper discusses how cognitive mapping can be applied to improve user–database interaction.

The rest of the paper is organized as follows: First, we present a framework for user–database interaction. Second, we build on the framework to assess influential human factors in user–database interaction. Third, we introduce three cognitive mapping techniques. This is followed by the proposed use of these techniques in supporting user–database interaction. Fourth, we discuss the implications for user–database interface analysis and design from technical communicators' perspective. Finally, we highlight the contributions of the paper.

USER–DATABASE INTERACTION

A database system aims to provide users with an abstract view of data by hiding certain details of how data is stored and manipulated. The architecture of most commercial database systems (relational or otherwise) is based on the proposal by the ANSI/SPARC Study Group on Data Base Management Systems [15], [16].

Database Schemas Some database researchers have elaborated on the different views associated with the database architecture [17], [18]. A schema is the overall definition of a specific database view [17]. Each individual user views the database from his or her own perspective. Known as an external schema, such an individual view is composed of definitions of each of the various things about which data is needed and the characteristics of those things that are important to know. A conceptual schema defines a community or shared view of the data. A logical schema is used by programmers and designers to describe data in detail, while a physical schema is used by database administrators to define how the data should be physically stored in the computer.

When an end-user interacts with a database system, he or she will develop an individual user view about the database, which is defined and represented by an external database schema.

User–Database Interaction User–database interaction involves three major tasks: database design, database update, and database retrieval [17]. In this paper, we focus on the interaction between ordinary users and the database system. In such a context, the major user–database interaction activity is data retrieval, which is usually conducted by writing and executing queries on database systems.

To help understand how database queries are generated, Ogden proposed a three-stage cognitive model of database query [19]:

(1) QUERY FORMULATION. Users decide what data they need in the context of an application domain. An example of the output of this stage is: "What are the names of the students born before 1985?"
(2) QUERY TRANSLATION. Using the output of the previous stage as input, users decide what elements of the data model (perceived database schema) are relevant, as well as the necessary operations. An example of the output of this stage is: "The relations Student is needed, the column StudentName is to be printed, and a restriction of before 1985 must be specified on the column DateofBirth."
(3) QUERY WRITING. Users arrange the output from the previous stage into the format required by the query language provided by the user–database interface. An example following Structured Query Language (SQL) notations is: "select StudentName from Student where DateofBirth < #1985-01-01#."

A Framework of User–Database Interaction A distinction can be made between the style, structure, and content of human–computer dialog [20]. In view of database query being a special case of user–computer dialogue, we assert that these dimensions are also applicable to user–database interaction. Based on both the cognitive model of database query [19] and the human–computer dialogue dimensions [20], a framework for user–database interaction is developed and presented in Fig. 1.

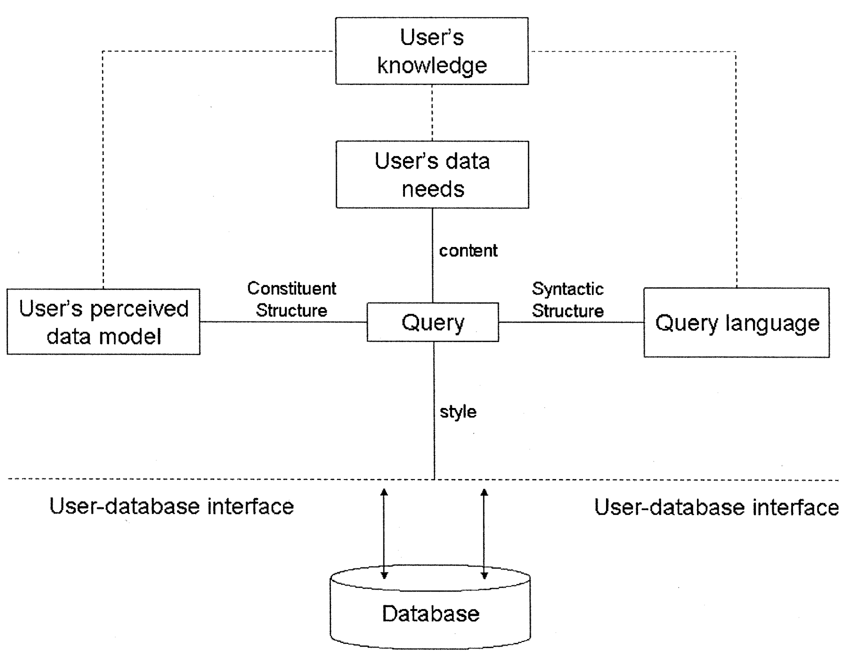

Fig. 1. Framework for user–database interaction.

According to this framework, user–database interaction has three dimensions—content, structure, and style. The content (i.e., semantics) dimension is determined by user's data needs. The structure dimension is determined by both user's perceived data model (constituent structure) and query language (syntactic structure). The style dimension is dictated by user–database interface. The query example that was used when we introduced Ogden's cognitive model of database query [19] will be revisited here to demonstrate the applicability of this framework. The content of this query is determined by the question "What are the names of students born before 1985?" In other words, our data needs will consist of the names of those students who were born before 1985. Deciding the content of database query, therefore, corresponds to the first stage in Ogden's cognitive model of database query [19].

When we try to translate the data needs (content) into a database query, we first need to decide what elements and operations of the database schema (constituent structure) are relevant. Supposing that the database schema is perceived as a single table regarding students' information (a relational database schema at the logical level), this schema forms the external database view for this task. Then, we can decide that the table Student is needed, the column StudentName is to be printed, and a restriction of earlier than 1985 must be specified on the column DateofBirth. The identification of these database elements and operations is the second stage in Ogden's cognitive model of database query and is dependent on the perceived database schema [19].

After we have identified the database elements and operations, the query is written using the syntax structure as dictated by a query language. Following SQL notations, the query is written in the form of "select StudentName from Student where DateofBirth < #1985-01-01#." This corresponds to the third stage in Ogden's cognitive model of database query [19].

It should be noted that the user's knowledge is critical to understand the data needs, form a correct and appropriate database schema, and properly apply the query language's syntax. In the framework, we include "user's knowledge" to reflect its influential effects on database querying (denoted as dashed lines in Fig. 1). Finally, the query is entered into the system in a style offered by the user–database interface. The possible styles include command languages, menu selection, and form fill-in.

HUMAN FACTORS IN USER–DATABASE INTERACTION

Even though user–database interface is used for the communication of semantics (queries and results) between users and database systems [5], the above framework indicates that many human factors may affect the effectiveness of user–database interaction. A good understanding of these factors is crucial for improving user–database interaction. In this section, we discuss how human factors affect all four dimensions of user–database interaction—content, constituent structure, syntactic structure, and style (see Fig. 1).

Human Factors and Content Query formulation deals with the user's data needs (i.e., the semantics dimension of the query). Constraints on humans as an information processor may affect users' ability to come up with a complete and accurate list of data needs [21]. The notable constraints include:

- COGNITIVE STRAIN. Limited capacity of working memory and its serial information processing plague humans as an information processor and a problem solver [22], [23]. This limitation, to a great extent, determines how fast an ordinary person can process a certain amount of information. When cognitive resources, especially the working memory, are exhausted, people become cognitively overloaded.
- HEURISTIC-DRIVEN BIASES. People tend to use rules of thumb (heuristics) to make sense of the environment and make decisions accordingly [24]. As these rules are generally imperfect, people have biased beliefs on the problems they are facing. Tversky and Kahneman identified several judgment heuristics that may cause heuristic-driven biases, including representativeness, the availability heuristic, as well as anchoring and adjustment [25].
- SATISFICING. Simon has noted that people tend to satisfy minimal constraints and are unable to attempt to optimize the solution [26].
- AUTOMATICITY. With automatic processing, people perform routine actions effortlessly, unconsciously, and involuntarily [27]. As a result, people may not be able to describe their routine tasks accurately and completely.

These constraints on human's cognition and behavior have influential impacts on the accuracy and completeness of data needs.

Human Factors and Constituent Structure When we try to translate the data needs (content) into a database query, we first need to decide what elements and operations of the database schema (constituent structure) are relevant. Therefore, a correct external view of the database, as defined by an individual user's database schema, is critical for this step.

A user can either develop his or her database schema by making sense of the database structure, or by making sense of a pre-specified database schema given by the system. All the constraints on human's cognitive capability, as we identified in the previous subsection, are applicable either way. Many empirical studies indicate that ordinary users can work easily with a database schema at the conceptual level (e.g., [6], [28], [29]). Such database schemas are often in the form of an entity-relationship model (ERM) [30] or an object-role model (ORM) [31]. This is because the models at the conceptual level often use real-world terms as names for the entities, relationships, objects, and properties as compared to the database terms (e.g., relations, join, etc.) in the logical database schema (e.g., relationship data model and object-oriented data model) [32]. In other words, users need not make extra cognitive efforts to map their real-world concepts to database concepts when making sense of database schema at the conceptual level.

Human Factors and Syntactic Structure With constituent structures available, the user then needs to write the query in the syntax as dictated by a query language. When it comes to syntax, it is of significant importance to provide users with easy-to-learn and easy-to-use languages. Based on the related ease of learning and using, query languages may be classified into textual (also known as linear keyword or computer language), visual (also known as graphical), and natural languages, in the increasing order of syntactic ease [4]. The major database query language is still SQL, which is textual and was developed almost 30 years ago [33]. SQL was found to be very difficult to use, even for trained users [34], [35].

Chan and Lim conducted a meta-analysis on experimental studies that compared the effects of textual query language and natural language on user query performance [3]. They found that users of the natural language were more accurate and took less time to formulate queries than users of textual languages. Chan and Lim asserted that less cognitive efforts are needed when users write queries using natural languages [3].

Other studies also support the notion that visual query language (e.g., QBE) is easier for users to write queries than traditional textual query languages (e.g., [36], [37]).

The impacts of human factors on the constituent and syntactic structure of database query can also be understood through the notion of Gulf of Execution. The Gulf of Execution, as Norman describes [38], is where the user knows what needs to be achieved but does not know which physical variables to adjust, or in what way to adjust them. For constituent structure, the difficulty lies in the fact that a user must map his/her psychological variables (information needs) to appropriate database variables. The Gulf of Execution, therefore, is where the user knows what information is needed, but does not know how to map the psychological variables to database terms. For syntactic structure, the problem is that the user must map database elements and operations to appropriate query language syntax. In other words, the Gulf of Execution exists when the user knows what database elements and operations are needed, but does not know how to map them to a specific query language.

Human Factors and Style When the user tries to input the query through the user–database interface, the choice of interface style can have a profound effect on the nature of this interaction. There are a number of common interface styles, including (a) command line, (b) menus, (c) form fill-in, and (d) direct manipulation [39], [40]. The notion of control is very relevant when we discuss human factors related to the interface style.

Command line is one of the oldest interface styles. The user has to type in textual commands to the database system. An alternative to command line is to use menus, which avoids the problem of remembering commands. Both styles (command line and menus) leave the user in control of the interaction [41].

One style that leaves the user with very little control is form fill-in (also known as “fill the blanks”). In form fill-in, the user is presented with a form in which various portions must be filled-in, leaving the user with few alternatives. Although the user has little control over the interaction, the user is relieved from remembering commands or other syntax [41].

TABLE I
IMPLICATIONS OF HUMAN FACTORS IN USER-DATABASE INTERACTION

Stages in User-Database Interaction	Dimension	Impacts
Query Formulation	Content	(1) Cognitive capability: understand the information needs accurately and completely
Query Translation	Constituent structure	(1) Cognitive capability: make sense of the database and form an appropriate database schema (2) Gulf of execution: map psychology variables to database elements and operations
Query Writing	Syntactic structure	(1) Cognitive capability: difficulty in remembering commands and language syntax (2) Gulf of execution: map database elements and operations to a specific query language
	Style	(1) Control of the system: trade-off between control of the system and cognitive efforts

In direct manipulation, the users can point at visual representations of objects and actions, thus carrying out tasks rapidly and observing the results immediately. WIMP (which stands for windows, icons, menus, and pointers) is the default interface style for the majority of interactive computer systems in use today, such as Microsoft Windows and Apple MacOS [40]. It is appealing to novices, is easy to remember for intermittent users, and, with careful design, can be rapid for frequent users [40]. Direct manipulation leaves the user in control of the dialogue and does not require him or her to remember commands and syntax.

Table I summarizes the impacts of human factors on user–database interaction.

COGNITIVE MAPPING TECHNIQUES

As discussed in the previous section, constraints on humans as information processors have a profound effect on user–database interaction. Many techniques have been suggested to overcome an individual's cognitive and behavioral biases [42]. Cognitive mapping, as a technique to elicit an individual's belief systems regarding a problem domain, has great potential in overcoming some of the problems. Before we propose the specific ways of using cognitive mapping techniques to improve user–database interaction, we briefly review some widely used cognitive mapping techniques.

Nature of Cognitive Mapping COGNITIVE MAPPING refers to a set of techniques used to identify subjective beliefs and to portray these beliefs externally [43]. The general approach is to extract subjective statements from individuals, within specific problem domains, about meaningful concepts and relations among these concepts and then to describe these concepts and relations in some kind of diagrammatical layout [44]. The outcome of a cognitive mapping technique is usually referred to as a cognitive map. Eden clarified the nature of cognitive map by defining a cognitive map as an artifact rather than the conceptual device in psychology developed by Tolman [45]. To avoid confusion, we use the term "cognitive maps" to exclusively refer to the outcomes of cognitive mapping techniques in this paper.

Three Cognitive Mapping Techniques Some cognitive mapping techniques have been widely used in the study of sociology, political science, organizational behavior, and strategic management (e.g., [46]–[48]). A few published works have offered comprehensive reviews of three popular cognitive mapping techniques in the IS related settings [12], [14]. In this section, we provide an example, in the context of the present paper, of each cognitive mapping technique.

Causal Mapping: Causal mapping is the most commonly used cognitive mapping technique by researchers when investigating the cognition of decision-makers in organizations [44]. As revealed by its name, a causal map represents a set of causal relationships among constructs within a system (i.e., one construct is linked to others through cause–effect relationships) [47]. By capturing the chains of argument, insights into the reasoning of a particular person are acquired. Fig. 2 illustrates the reasoning in the subsection "Human Factors and Content" where we discuss the impacts of human factors on content dimension of user–database interaction.

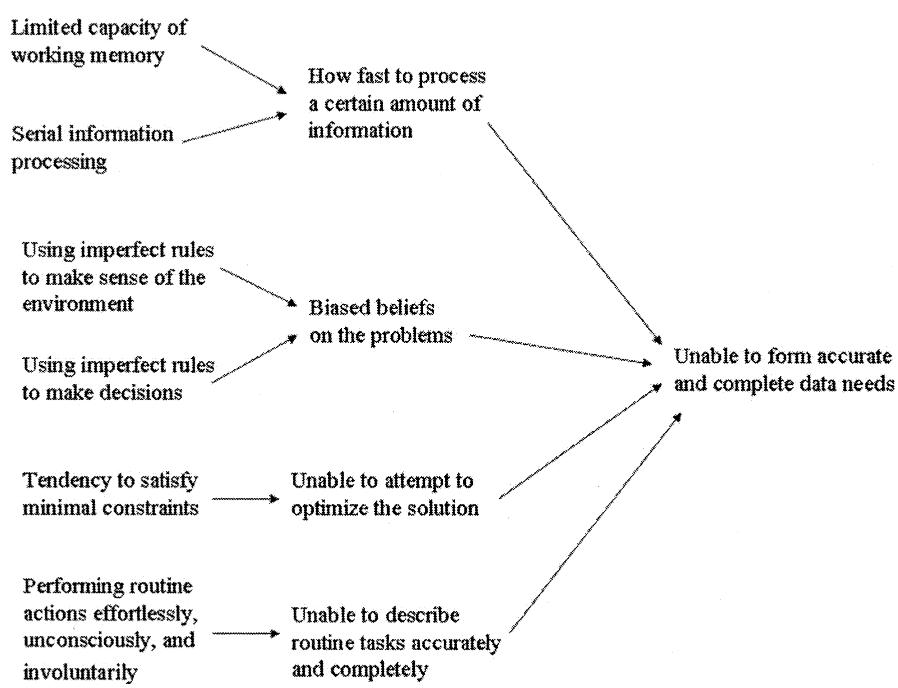

Fig. 2. Causal map example.

Semantic Mapping: Semantic mapping, also known as idea mapping, is used to explore an idea without the constraints of a superimposed structure [49]. A semantic map has one main or central concept with tree-like branches. Fig. 3 is an example of a semantic map that depicts related words around the main idea "User–database interaction." The semantic map is created based on the text in the section, "Human Factors in User–Database Interaction."

Concept Mapping: Another technique is called concept mapping [50]. A concept map is a graphical representation where nodes represent concepts, and links represent the relationships between concepts. The links, with labels to represent the type of relationship between concepts, can be one-way, two-way, or nondirectional. Concept mapping is useful in generating ideas, designing a complex structure, communicating complex ideas, aiding learning by explicitly integrating new and old knowledge, and assessing understanding or diagnosing misunderstanding [51]. Fig. 4 is an example of a concept map, which represents the major concepts in the section where we introduced the framework for user–database interaction.

In general, these cognitive mapping techniques are commonly used to reveal the cognitive structures (i.e., belief systems that individuals or groups use to interpret the problem domain and take actions).

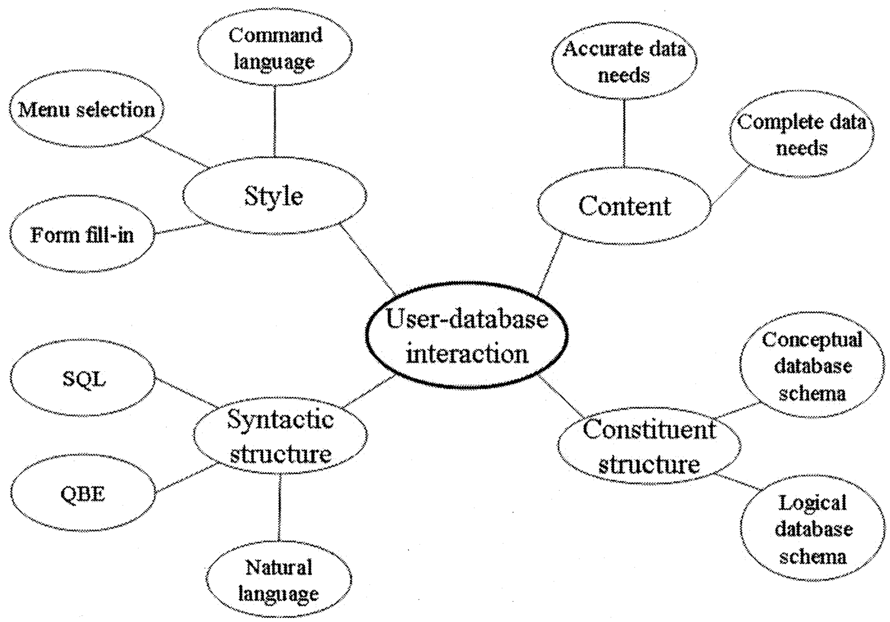

Fig. 3. Semantic map example.

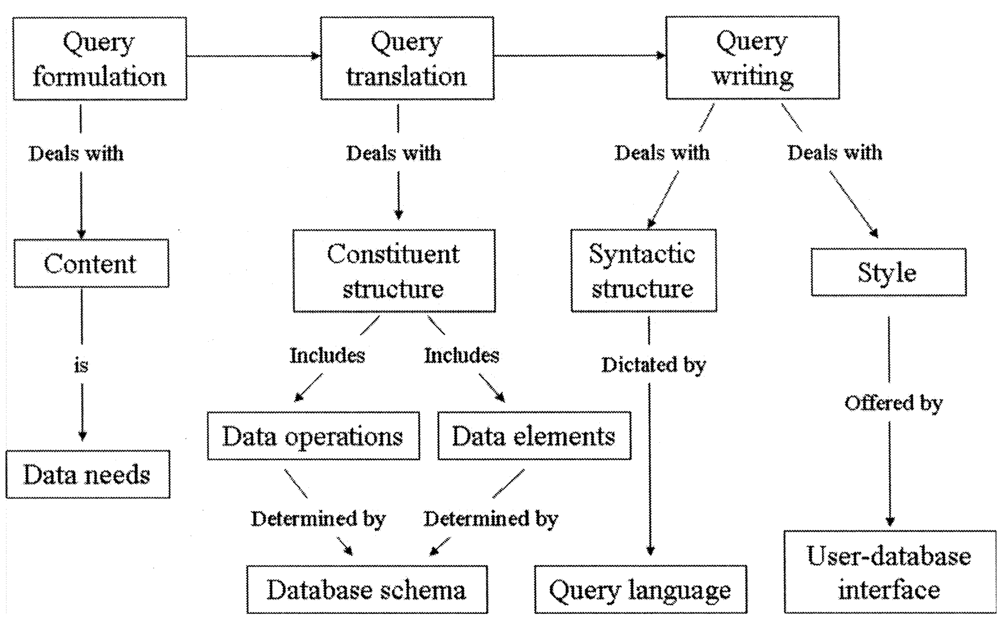

Fig. 4. Concept map example.

Table II summarizes the main characteristics of the three cognitive mapping techniques.

USING COGNITIVE MAPPING

According to Fiol and Huff [43], cognitive mapping techniques as well as the resulting cognitive maps have the following advantages:

- able to focus attention and trigger memory;
- help highlight priorities and key factors;
- may supply missing information;
- reveal gaps in information or reasoning that need more direct attention.

The diagnostic qualities of cognitive mapping make it an ideal analytical and communication tool. In this section, we propose ways of using cognitive mapping techniques to improve user–database interaction.

Help in Eliciting Knowledge As we discussed above and illustrated in Fig. 1, user's knowledge about the data needs, database schema, and query language plays a critical role in generating an accurate database query. Many human factors may influence the processes of forming complete data needs, developing correct database schema, and remembering query syntax. By helping in eliciting knowledge, cognitive mapping techniques are able to improve user–database interaction in two ways.

First, cognitive mapping techniques can be used to elicit user's knowledge about data needs, database schema, query language, as well as the process of developing queries. The cognitive maps are then examined by experts to identify faulty reasoning and inappropriate representations, which are traditionally difficult to identify. By eliciting user's knowledge, cognitive mapping can play an important role in user training.

Second, cognitive mapping techniques can be used to elicit expert's knowledge about database schema, query language, and query formulation process. The resulting cognitive maps are suitable materials for user's guides and online help. The expert's knowledge, as represented by cognitive maps, is easier to make sense of for ordinary users than is plain text. For instance, a database expert can make his/her understanding of the database schema explicit by developing a cognitive map. This map is then used to train users to form an accurate individual view about the database. In addition, the expert can demonstrate his/her processes in developing a query through a cognitive map, which can serve as a blueprint for users to develop similar queries.

Help in Communication Cognitive mapping techniques can be used in the communication between user and IT professionals. The communication is often plagued by the constraints on human cognition, such as heuristic-driven biases and automaticity. For example, a user may not be able to describe his/her data needs clearly to a systems analyst because some of the user's assumptions are unknown to the analyst. By using cognitive maps, the analyst can probe the links within the user's belief structures. Some IS studies have indicated such diagnostic uses of cognitive mapping in identifying users' information requirements [10], [11], [52]. In addition, by focusing attention and triggering memory, cognitive mapping can help alleviate the effects of automaticity for both ordinary users and IT professionals.

An Illustration We use an example to demonstrate some potential uses of cognitive mapping techniques in user–database interface analysis and design. The database for this demonstration is a student registration database (Appendix A), which is adapted and is easy for potential readers of this paper to make sense of [53].

Eliciting Knowledge: An expert's knowledge about this database application may be depicted using a semantic map (Fig. 5). This map offers ordinary users a holistic view of the database application.

An effective user guide or training material often provides examples for users to imitate the process of writing database queries. However, most examples on database query writing only present the question and the resulting database query. As an example, for a question such as "What are the names of the students born before 1985?" the corresponding database query would be "SELECT lastname, firstname FROM Student WHERE dob < #1985-01-01#".

TABLE II
CHARACTERISTICS OF THREE COGNITIVE MAPPING TECHNIQUES

	Causal mapping	Semantic mapping	Concept mapping
Basic Elements	Concepts expressed in single-polar or bi-polar phrase, causal relationships among concepts	A main idea and branches of sub-ideas	Concepts and labeled links
Theoretical Foundations	Personal Constructs Theory [47]	Mind map [49]	Knowledge assimilation [50]
Structure	Complex network	Tree-like structure	Complex network
Focus	Cause-effect structure of concepts	Organization of sub-ideas around the main idea	Semantic-rich links among various concepts

The way to translate the physical variables to database elements and operations (query translation, according to the cognitive model of database query) is not presented and missing. Even though some users may infer from the above example to write similarly simple queries for similarly simple questions, many ordinary users may not be able to learn effectively from examples like the one above.

Cognitive mapping techniques can elicit knowledge about query formulation from database experts. The resulting maps are easier than plain text for ordinary users to understand the underlying logic. The following concept map (Fig. 6) demonstrates the concepts (steps) needed to write a query to answer the question: "What are the names of the students who have registered for more than 15 credit hours in Fall 2004?"

According to the concept map, the concepts at the top of Fig. 6 are related to finding the appropriate columns, while the concepts at the bottom of Fig. 6 focus on confining the results with specified constraints. The words in the shaded blocks can be used to form the following formal SQL query.

SELECT lastname, firstname, semyear, sum(credit)
FROM Student, Course, Term, Enrollment, Cousesection
WHERE Student.student_id = Enrollment.student_id
AND

Enrollment.coursesection_id
= Coursesection.coursesection_id

AND Coursesection.term_id = Term.term_id
AND Coursesection.course_id = Course.course_id
AND semyear = "Fall 2004"
GROUP BY lastname, firstname
HAVING sum(Credit) > 15

By eliciting knowledge on developing a query, the cognitive mapping techniques can reduce the cognitive burden on ordinary users in learning. The explicit knowledge shown in the resulting cognitive maps can provide appropriate mapping logic, and narrow the Gulf of Execution.

Help in Communication: In training users to write database queries, it is critical to let them form a correct external view of the database, as defined by an individual user's database schema. Traditional database schema is often represented by an entity-relationship (ER) model. The ER model for the example database is shown in Fig. 7.

Ordinary users, however, may find the ER model too complicated to understand. In this case, we propose using cognitive maps to help communicate the database schema to users. Fig. 8 is a concept map depicting the database schema. There are two distinct differences between the ER model and the concept map. First, the multiplicity (crow's feet) in the ER model is a kind of formal notation that is not critical for ordinary users to understand the database schema. Second, the concept map has no associative entity (like the entity "Enrollment" in the ER model) that may cause confusion for ordinary users. Using the concept map, users can easily grasp the essential semantics of the application domain. To some extent, the concept map is similar to a high-level ER model that includes roles between entities. However, the concept map is not as formal as the ER model; thus, it is easier for ordinary users who are not familiar with formal ER notations to understand.

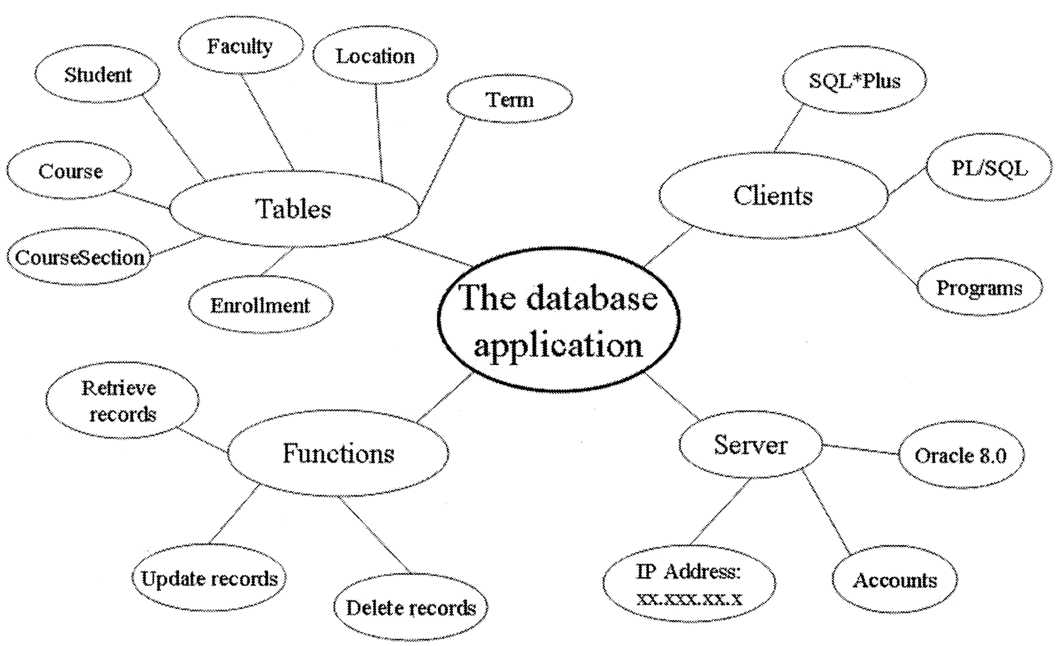

Fig. 5. Semantic map of the database application.

The illustration in this section is by no means exhaustive. For example, causal maps can be used extensively in helping users form accurate and complete data needs. Semantic maps can describe the structure of a user manual to provide a holistic view of the embodied knowledge. To sum up, we hope to stimulate the creative use of cognitive mapping techniques in user–database interface analysis and design.

IMPLICATIONS FOR USER–DATABASE INTERFACE ANALYSIS AND DESIGN

We have presented a framework on user–database interaction and discussed the impacts of human factors. We then introduced cognitive mapping techniques to overcome human factor issues in user–database interaction and to improve user–database interaction. In this section, we discuss the implications for user–database interface analysis and design. The discussion focuses on the roles technical communicators play during user–database interface analysis and design.

Many researchers and practitioners have pointed out that technical communicators can and should play an important role in information systems development. For example, Fisher claimed that the role of technical communicators in the development of information systems relates particularly to the human factors of systems development [54]. Siau and Tan examined the extensive roles of technical communicators in systems development and discussed how human factors affect the tasks performed by technical communicators [14]. Following the same logic, we contend that technical communicators, with a user-centered orientation and a system-usability emphasis, can and should contribute much to user–database interface analysis and design.

Technical communicators, acting as user advocates, can talk with end-users about their data needs and represent users in discussions on the usability of databases. Many end-users are not familiar with database architecture and are not fluent in query languages. Technical communicators can create easy-to-navigate online help to facilitate end-users when they interact with the database systems. Empirical studies have demonstrated that a good feedback system, which can come in the form of system messages, can help the user's query performance (e.g., [55]). Technical communicators with expertise in communication can help develop a better feedback system for user–database interface. Finally, technical communicators possess the expertise in graphic design, layout, and illustration, and thus can provide good advice on user–database physical interface design.

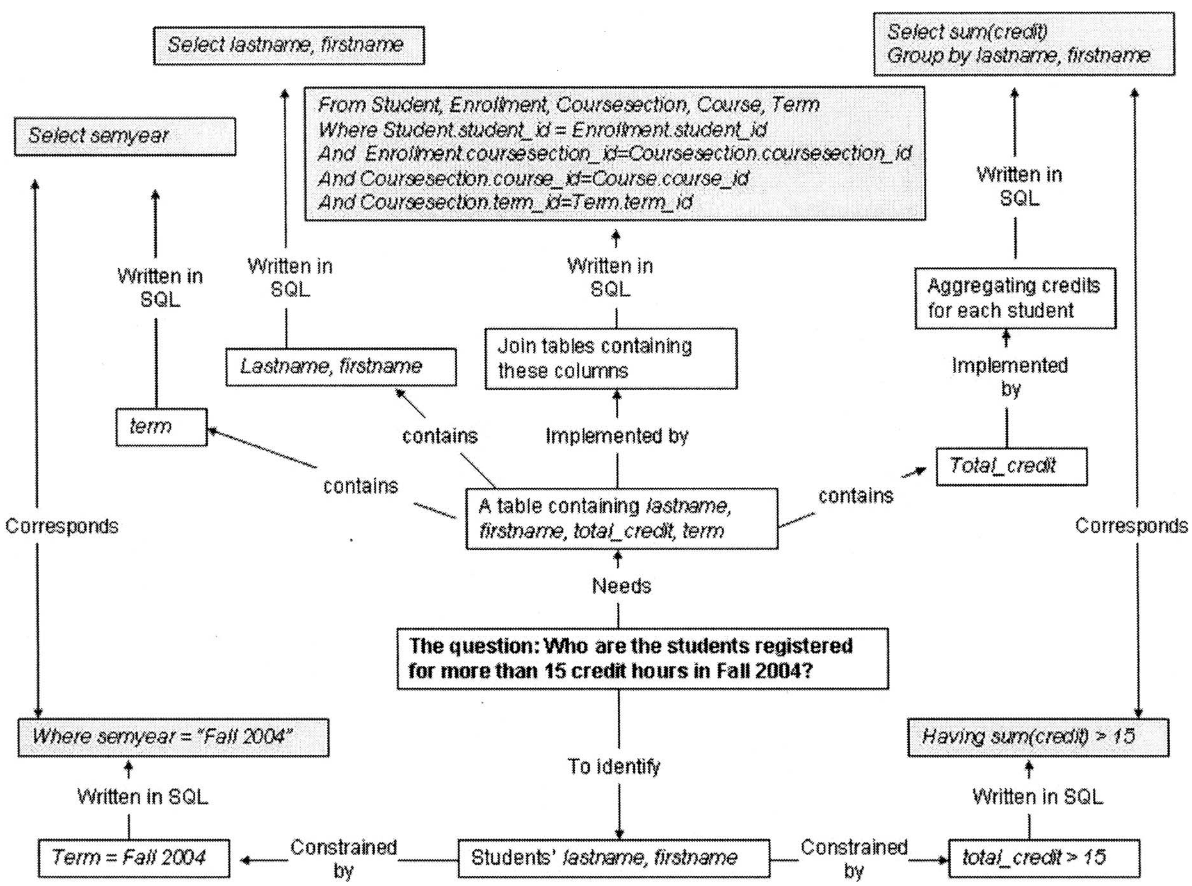

Fig. 6. Concept map of query formulation.

Human factors have significant implications for technical communicators during user–database interface analysis and design. For instance, the problems related to an individual's cognitive capabilities, such as limited memory, cognitive biases, automaticity, and faulty reasoning, have a profound impact on the technical communicators' efforts to understand the needs of end users. In particular,

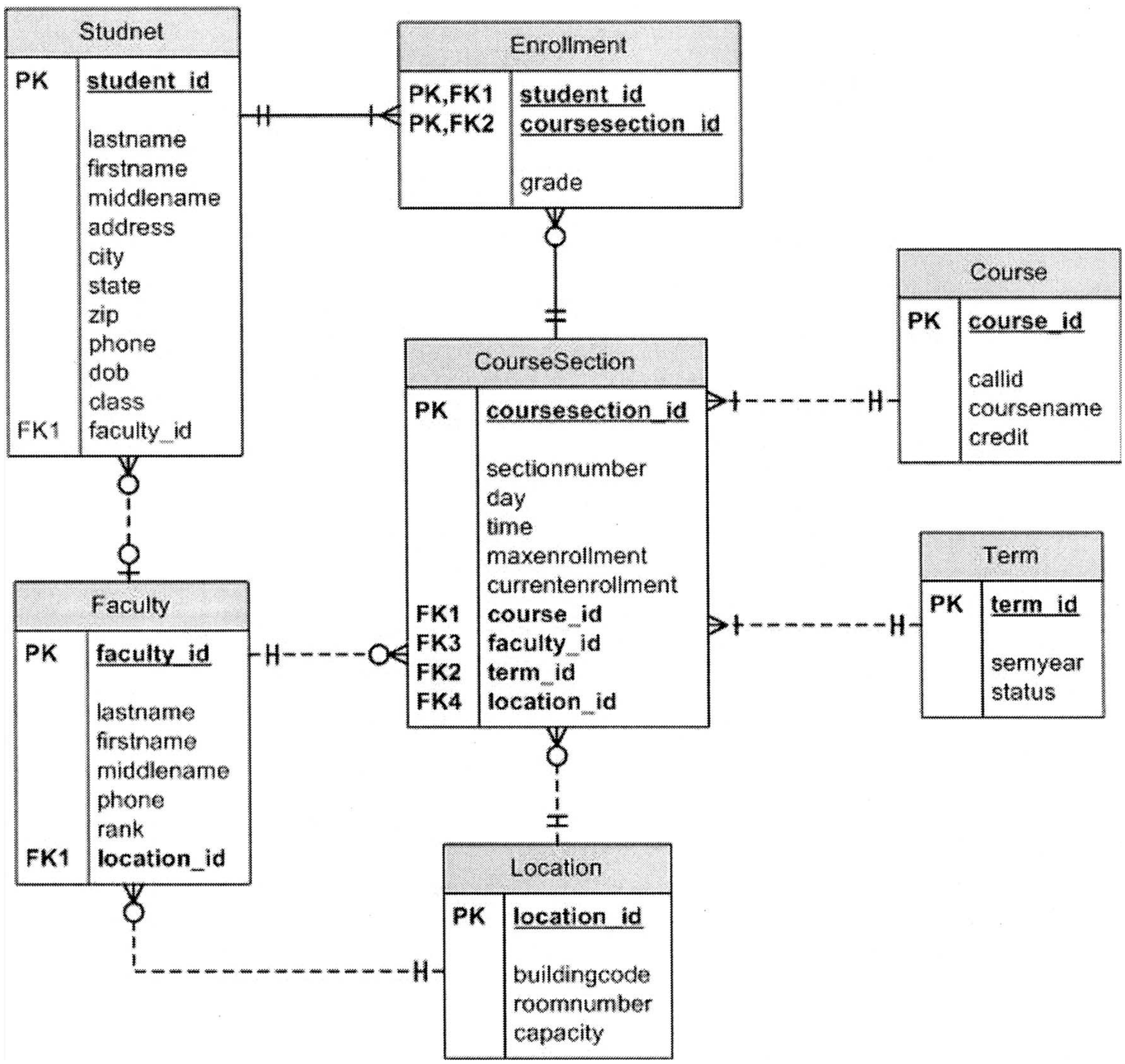

Fig. 7. ER diagram of the database schema.

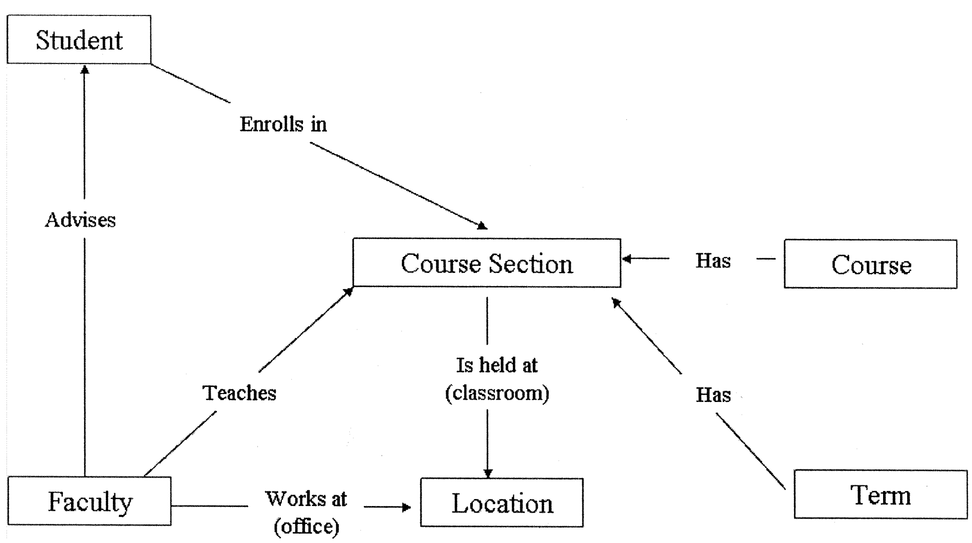

Fig. 8. Concept map of the database schema.

these problems pose great difficulties for technical communicators to serve as user advocates. How to clearly understand the user's information needs and sufficiently represent them in the system usability discussion poses a challenging question for technical communicators.

How to help ordinary end-users make sense of the database, and thus form an appropriate database schema, is a key topic for technical communicators when designing effective user guides and training material for user–database interface. Technical communicators should understand what level of data models (conceptual or logical) are preferred by ordinary end-users.

Facing difficulties and heavy burdens in remembering syntax and commands of query language, end-users have strong needs for effective online help and system message in the user–database interface. Technical communicators should be able to identify some common questions and update the online help during system maintenance. The online help must be structured for ease of navigation and ease of search for needed information. The system messages and error messages must be designed to give users immediate feedback on the syntax quality of the query.

With regard to the user–database interface style, technical communicators can utilize their expertise in graphic design, layout, and illustration to help find a fit between the end-user's expectations and the interface style. The system-usability principle can guide technical communicators to suggest an appropriate control level for end-users.

The diagnostic qualities of the cognitive mapping techniques can be used for practical purposes by technical communicators. Particularly, cognitive mapping is suitable to overcome some cognitive problems and biases in the process of determining user's requirements, therefore improving the skills of technical communicators to act as user advocates.

Technical communicators may apply various cognitive mapping techniques to capture the database developer's knowledge in order to design correct user guides, system and error messages, and online help. When providing advice on interface designs, technical communicators can also take advantage of cognitive mapping techniques. Besides using them to capture and understand the user's requirements on interface, technical communicators may apply cognitive mapping techniques to develop conceptual models for user interfaces. Rubin claimed that the conceptual model of a user interface contributes greatly to the ease of learning and ease of use [56]. When that model is unclear, is absent, or does not match the user's expectations, it may hinder the usability of the interface.

CONCLUSION

Human factors have a profound impact on the effectiveness of user–database interaction. In this paper, we first developed a framework of user–database interaction, based on the cognitive model of database query and the human–computer dialogue dimensions [19], [20]. We then discussed how human factors influence various dimensions of user–database interaction. Following that, we proposed using cognitive mapping techniques to overcome some cognitive and behavioral biases during user–database interface analysis and design. Three popular cognitive techniques were introduced as approaches to elicit an individual's belief systems regarding a problem domain. Through an example database application, we demonstrated how to use these cognitive mapping techniques. In the end, we discussed the implications of this study for technical communicators.

This paper provides both a theoretical model and a practical tool for user–database interface analysis and design. Future studies can build on our research to better understand the communication problems between users and database systems, and to design effective user–database interfaces.

APPENDIX

The relational database schema:

Student	(<u>student_id</u>, lastname, firstname, middlename, address, city, state, zip, phone, dob, class, faculty_id);
Faculty	(<u>faculty_id</u>, lastname, firstname, middlename, location_id, phone, rank);
Location	(<u>location_id</u>, buildingcode, room, capacity);
Term	(<u>term_id</u>, semyear, status);
Course	(<u>course_id</u>, callid, coursename, credit);
Course_section	(<u>coursesection_id</u>, course_id, term_id, section_number, faculty_id, location_id, day, time, maxenrollment, currentenrollment);
Enrollment	(<u>student_id</u>, <u>coursesection_id</u>, grade).

REFERENCES

[1] R. Ramakrishnan, *Database Management Systems*. Boston, MA: McGraw Hill, 1998.
[2] K. Siau, "A visual object-relationship query language for user-database interaction," *Telematics and Informatics: Int. J. Telecommun. Inf. Technol.*, vol. 15, no. 1-2, pp. 103–119, 1998.
[3] H. C. Chan and J. Lim, "Database interface: A conceptual framework and a meta-analysis on natural language studies," *J. Database Manage.*, vol. 9, no. 3, pp. 25–32, 1998.
[4] ——, "A review of experiments on natural language interfaces," in *Advanced Topics in Database Research*, K. Siau, Ed. Hershey, PA: Idea Group Pub., 2003, vol. 2.
[5] K. L. Siau, H. C. Chan, and K. K. Wei, "Effects of query complexity and learning on novice user query performance with conceptual and logical database interfaces," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 34, no. 2, pp. 276–281, Mar. 2004.
[6] H. C. Chan, K. K. Wei, and K. L. Siau, "User-database interface: The effect of abstraction levels on query performance," *MIS Quarterly*, vol. 17, no. 4, pp. 441–464, 1993.
[7] J. MacGregor, "A comparison of the effects of icons and descriptors in videotex menu retrieval," *Int. J. Man-Machines Studies*, vol. 37, pp. 767–777, 1992.
[8] A. S. Huff, *Mapping Strategic Thought*. New York: Wiley, 1990.
[9] D. E. Avison and G. Fitzgerald, *Information Systems Development: Methodologies, Techniques and Tools*, 3rd ed. New York: McGraw-Hill Education, 2003.
[10] T. A. Byrd, K. L. Cossick, and R. W. Zmud, "A synthesis of research on requirements analysis and knowledge acquisition techniques," *MIS Quarterly*, vol. 16, no. 1, pp. 117–138, 1992.
[11] A. R. Montazemi and D. W. Conrath, "The use of cognitive mapping for information requirements analysis," *MIS Quarterly*, vol. 10, no. 1, pp. 45–56, 1986.
[12] K. Siau and X. Tan, "Improving the quality of conceptual modeling using cognitive mapping techniques," *Data Knowledge Eng.*, vol. 55, no. 3, pp. 343–365, 2005.
[13] C. Soderston, N. Kleid, and T. Crandell, "Concept mapping: A job-performance aid for hypertext developers," in *Proc. 14th Annu. Int. Conf. Systems Documentation*, 1996, pp. 179–186.
[14] K. Siau and X. Tan, "Technical communication in information systems development: The use of cognitive mapping," *IEEE Trans. Prof. Commun.*, vol. 48, no. 3, pp. 269–284, Sep. 2005.
[15] "ANSI/X3/SPARC study group on data base management systems, 'Interim report'," *FDT–Bull. ACM SIGMOD*, vol. 7, no. 2, pp. 1–140, 1975.
[16] D. Tsichritzis and A. C. Klug, "The ANSI/X3/SPARC DBMS framework report of the study group on database management systems," *Inf. Syst.*, vol. 3, no. 3, pp. 176–191, 1978.
[17] C. J. Date, *An Introduction to Database Systems*, 8th ed. Reading, MA: Addison-Wesley, 2003.
[18] D. C. Hay, *Requirements Analysis: From Business Views to Architecture*. Upper Saddle River, NJ: Prentice Hall, 2003.
[19] W. C. Ogden, "Implications of a cognitive model of database query: Comparison of a natural language, a formal language, and direct manipulation interface," *ACM SIGCHI Bull.*, vol. 18, no. 2, pp. 51–54, 1985.
[20] P. J. Barnard and N. V. Hammond, *Cognitive Contexts and Interactive Communication*: IBM Hursley Human Factors Laboratory, 1983.
[21] G. B. Davis, "Strategies for information requirements determination," *IBM Syst. J.*, vol. 21, no. 1, pp. 4–30, 1982.
[22] A. D. Baddeley and G. J. Hitch, "Working memory," in *The Psychology of Learning and Motivation*, G. Bower, Ed. New York: Academic, 1974, pp. 47–89.
[23] A. D. Baddeley, *Working Memory*. Oxford, UK: Oxford Univ. Press, 1986.
[24] D. Kahneman and A. Tversky, "Prospect theory: An analysis of decision under risk," *Econometrica*, vol. 47, no. 2, pp. 263–291, 1979.
[25] A. Tversky and D. Kahneman, "Judgment under uncertainty: Heuristics and biases," in *Judgment Under Uncertainty: Heuristics and Biases*, D. Kahneman, P. Slovic, and A. Tversky, Eds. New York: Cambridge Univ. Press, 1982.
[26] H. A. Simon, "A behavioral model of rational choice," *Quart. J. Economics*, vol. 69, no. 1, pp. 99–118, 1955.
[27] L. Hasher and R. T. Zacks, "Automatic and effortful processes in memory," *J. Experimental Psychology: General*, vol. 108, pp. 356–388, 1979.
[28] D. Bartra, J. A. Hoffer, and R. P. Bostrom, "Comparing representations with relational and ER models," *Commun. ACM*, vol. 33, no. 2, pp. 126–139, 1990.
[29] T. A. Halpin, *Conceptual Schema and Relational Database Design*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1995.
[30] P. P. Chen, "The entity-relationship model: Toward a unified view of data," *ACM Trans. Database Syst.*, vol. 1, no. 1, pp. 9–36, 1976.
[31] S. Khoshafian, *Object-Oriented Databases*. New York: Wiley, 1993.
[32] K. Siau, "Information modeling and method engineering: A psychological perspective," *J. Database Manage.*, vol. 10, no. 4, pp. 44–50, 1999.
[33] D. D. Chamberlain and R. F. Boyce, "SEQUEL: A structured english query language," in *Proc. 1974 ACM SIGMOD Workshop on Data Description, Access, and Control*, Ann Arbor, MI, 1974, pp. 249–264.
[34] C. Welty, "Correcting user errors in SQL," *Int. J. Man-Machines Studies*, vol. 22, no. 4, pp. 463–477, 1985.
[35] C. J. Date, "What's wrong with SQL?," in *Relational Database Writings, 1985–1989*, C. J. Date, Ed. Reading, MA: Addison-Wesley, 1990.

[36] H. Chan, K. Siau, and K. K. Wei, "The effect of data model, system and task characteristics on user query performance—An empirical study," *The Data Base for Advances in Information Systems*, vol. 29, no. 1, pp. 31–49, 1998.
[37] Y. M. Yen and R. W. Scamell, "A human factors experimental comparison of SQL and QBE," *IEEE Trans. Software Eng.*, vol. 19, no. 4, pp. 390–409, Apr. 1993.
[38] D. A. Norman, "Cognitive engineering," in *User-Centered System Eesign: New Perspectives on Human-Computer Interaction*, D. A. Norman and S. Draper, Eds. Hillsdale, NJ: Lawrence Erlbaum, 1986.
[39] A. J. Dix, J. E. Finlay, G. D. Abowd, and R. Beale, *Human-Computer Interaction*. Glasgow, UK: Pearson Education, 1998.
[40] B. Shneiderman, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Reading, MA: Addison-Wesley, 1998.
[41] P. A. Booth, *An Introduction to Human-Computer Interaction*. Hove, UK: Lawrence Erlbaum, 1989.
[42] G. J. Browne and V. Ramesh, "Improving information requirements determination: A cognitive perspective," *Inf. Manage.*, vol. 39, no. 8, pp. 625–645, 2002.
[43] C. M. Fiol and A. S. Huff, "Maps for managers: Where are we? where do we go from here?," *J. Manage. Studies*, vol. 29, no. 3, pp. 267–285, 1992.
[44] J. Swan, "Using cognitive mapping in management research: Decisions about technical innovation," *Brit. J. Manage.*, vol. 8, no. 2, pp. 183–198, 1997.
[45] E. Tolman, "Cognitive maps in rats and men," *Psychol. Rev.*, vol. 55, pp. 189–208, 1948.
[46] Axelrod, *Structure of Decision*. Princeton, NJ: Princeton Univ. Press, 1976.
[47] C. Eden, "Cognitive mapping," *Eur. J. Operational Res.*, vol. 36, pp. 1–13, 1988.
[48] K. M. Carley and M. Palmquist, "Extracting, representing and analyzing mental models," *Social Forces*, vol. 70, no. 3, pp. 601–636, 1992.
[49] T. Buzan, *The Mind Map Book*. London, UK: BBC Books, 1993.
[50] J. D. Novak, "How do we learn our lesson? Taking students through the process," *The Science Teacher*, vol. 60, no. 3, pp. 50–55, 1993.
[51] D. H. Jonassen, K. Beissner, and M. A. Yacci, *Structural Knowledge: Techniques for Conveying, Assessing, and Acquiring Structural Knowledge*. Hillsdale, NJ: Lawrence Erlbaum, 1993.
[52] S. Sheetz and D. Tegarden, "Distributed capture of system requirements as individual and group cognitive maps," in *Proc. 4th Americas Conf. Information Systems*, 1998, pp. 983–984.
[53] J. Morrison and M. Morrison, *A Guide to Oracle 8*. Cambridge, MA: Course Technology, 2000.
[54] J. Fisher, "Defining the role of a technical communicator in the development of information systems," *IEEE Trans. Prof. Commun.*, vol. 41, no. 3, pp. 186–199, Sep. 1998.
[55] H. C. Chan, K. K. Wei, and K. L. Siau, "The effect of a database feedback system on user performance," *Behavior Inf. Technol.*, vol. 14, no. 3, pp. 152–162, 1995.
[56] J. Rubin, "Conceptual design: Cornerstone of usability," *Tech. Commun.*, vol. 43, no. 2, pp. 130–138, 1996.

Keng Siau is a Professor of Management Information Systems (MIS) at the University of Nebraska, Lincoln (UNL). He is currently serving as the Editor-in-Chief of the *Journal of Database Management* and as the Book Series Editor for *Advances in Database Research*. Dr. Siau has published over 200 academic publications. He has published more than 80 refereed journal articles and more than 90 refereed conference papers. In addition, he has edited/co-edited more than ten scholarly and research-oriented books, edited/co-edited nine proceedings, and written more than 15 scholarly book chapters.

Xin Tan received the B.E. degree from Shanghai Jiao Tong University and the M.B.A. degree from Miami University. He is working toward the Ph.D. degree in the Department of Management at the University of Nebraska–Lincoln. His current research interests include conceptual and data modeling, user acceptance of advanced information technology and enterprise systems, and information systems development methodologies.