# Model for Lexical Knowledge Base

Michio Isoda, Hideo Aiso
Keio University, Faculty of Science and Technology

Noriyuki Kamibayashi and Yoshifumi Matsunaga
Fuji Xerox Co. Ltd., System Technology Laboratory

**Abstract**

This paper describes a model for a lexical knowledge base (LKB). An LKB is a knowledge base management system (KBMS) which stores various kinds of dictionary knowledge in a uniform framework and provides multiple viewpoints to the stored knowledge.

KBMSs for natural language knowledge will be fundamental components of knowledgeable environments where non-computer professionals can use various kinds of support tools for document preparation or translation. However, basic models for such KBMSs have not been established yet. Thus, we propose a model for an LKB focusing on dictionary knowledge such as that obtained from machine-readable dictionaries.

When an LKB is given a key from a user, it accesses the stored knowledge associated with that key. In addition to conventional direct retrieval, the LKB has a more intelligent access capability to retrieve related knowledge through relationships among knowledge units. To represent complex and irregular relationships, we employ the notion of implicit relationships. In contrast to conventional database models where relationships between data items are statically defined at data generation time, the LKB extracts relationships dynamically by interpreting the contents of stored knowledge at run time. This makes the LKB more flexible; users can add new functions or new knowledge incrementally at any time. The LKB also has the capability to define and construct new virtual dictionaries from existing dictionaries. Thus users can define their own customized dictionaries suitable for their specific purposes.

The proposed model provides a logical foundation for building flexible and intelligent LKBs.

## 1. Introduction

Computers have been powerful support tools for various kinds of human activities. In particular, high performance personal workstations provide convenient and friendly environments for office workers and engineers. Conventional systems, however, provide only basic support tools such as text editors, text formatters, spelling checkers, and mail handlers.

With the progress in natural language processing, database management, and user-machine interface techniques, more advanced support tools have emerged. They include machine translation systems, style analyzers, personal databases, an electronic book [Weyer82], and an electronic encyclopedia [Weyer85]. Currently, these systems are in the experimental stage and are being implemented and used individually. They will eventually be integrated to build knowledgeable environments in which non-computer professionals can perform their tasks more quickly and easily.

The fundamental components of these tools are large knowledge bases which store knowledge about natural languages and application areas. Since many application programs will share the same kind of knowledge, these knowledge bases must be application-independent components that can be accessed from application programs through some predefined interfaces.

Thus, it is important to establish methodologies to construct these knowledge bases [Amsler84, Calzorali84a]. The methodologies include basic models, architectures, knowledge representation schemes, and implementation techniques. Since knowledge representation schemes and their usage vary widely, it is difficult to build a general knowledge base capable of coping with all applications described above.

Thus, as a first step to building these general knowledge bases, we propose a model for an LKB focusing on lexical knowledge such as those obtained from machine-readable dictionaries [ICOT85]. Methodologies developed for the LKB may then be applied to other kinds of knowledge bases.

LKBs provide intelligent access as well as conventional keyword access to stored knowledge. Users can customize their own dictionaries, and this personalization includes marking and annotating existing dictionaries and defining new access paths through which the system looks up requested dictionaries. The addition of new knowledge and functions can be done incrementally; it does not require reorganization of the existing knowledge base or recompilation of the whole system.

## 2. Lexical Knowledge Bases

In this section we will give an overview of the proposed LKBs.

The most basic capability of the LKB is the conventional keyword search. Given a keyword from a user, the LKB retrieves a piece of stored knowledge whose headword matches the keyword. We call an access unit of stored dictionary knowledge a *lexical knowledge unit (LKU)*. Headwords in machine-readable dictionaries are usually standardized; i.e. without inflections or conjugations. Since users won't always give standardized headwords, it is necessary to adopt conversion techniques from non-standardized keywords to standardized ones.

Although the direct retrieval capability alone allows users quick and convenient access to stored lexical knowledge, it is possible to provide a more intelligent access. When an LKU is looked up by a given key, the LKB can interpret its contents and retrieve related LKUs through relationships among the LKUs. The collected LKUs are then shown to the user. When people write or translate documents, they often use more

than one dictionary, consulting one after another. With bulky hardcopy dictionaries, many look-ups can become bothersome. An LKB can combine and access multiple dictionaries at a time and thus reduce the users' effort to find desired dictionary descriptions.

Because it is preferable to allow users to use their own customized dictionaries, an LKB must have mechanisms to change the activation path which is specified by a combination of dictionaries. The combination is defined in terms of the relationships between dictionaries. For example, an LKU of a Japanese-English dictionary may contain such information as the English translation of Japanese headwords, synonyms, antonyms, idioms, related words, usages, or grammatical information. Some users may want to combine this Japanese-English dictionary with an English usage dictionary through English-Japanese relationships, and others may want to combine it with an English-English dictionary through synonym relationships.

Since the expected users of the LKB are non-computer professionals, the customization of a new dictionary should be easy and should not require users to write programs. In our model, a new object in the system (dictionaries and association interpreters described in 3.3) is constructed by combining a set of smaller, relatively independent, self-contained objects. The newly defined objects can be used recursively as parts of more complicated objects. Thus users can construct their customized dictionaries like building-blocks.

## 3. Model for Lexical Knowledge Base

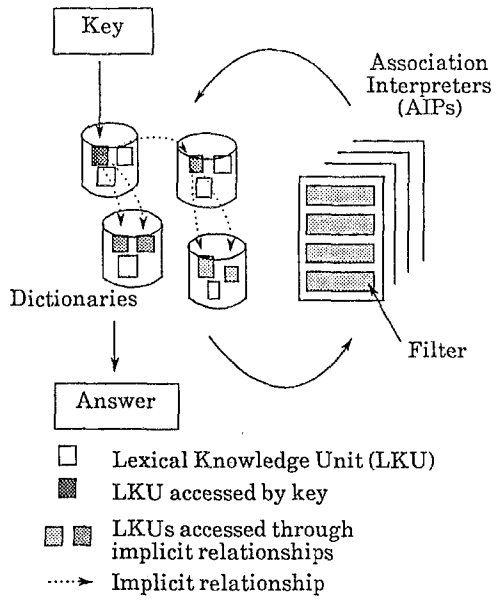


**Fig.1 Model for Lexical Knowledge Base**

### 3.1 Lexical Knowledge Unit

A unit of knowledge which is stored in a LKB is called a *lexical knowledge unit* (*LKU*). More precisely, a lexical knowledge unit is an independent description which is interpreted by a human or a computer to obtain lexical knowledge about a word. The format of an LKU is:

<Keys, Contents>

where *Keys* are words which are described in the LKU and are used to access this LKU. *Contents* are descriptions about the keys, and they are freely formatted.

For example an LKU of the word 'happy' is:

**happy** 1. glad; feeling contentment. ¶ I'm *happiest* when I'm playing baseball. ‖ 2....

In this example, "happy" is a key of this LKU, and the string "1. glad; feeling ..." is the content.

### 3.2 Implicit Relationships

There are complex and irregular relationships among words in the LKB. To represent these relationships, we employ the notion of implicit relationships which will be explained in this section.

In conventional database models like the relational model or the network model, relationships among data items are represented explicitly by means such as pointers or the equality of field values. We call these relationships *explicit relationships*. On the other hand, we define an *implicit relationship* to be a relationship that is represented in an LKU only implicitly; there are no notions like physical pointers or fixed fields in an LKU. The contents of an LKU are freely formatted; they are treated as a sequence of byte data. Implicit relationships in a LKU can be translated into explicit ones through an *interpretation* at run time. Procedures that interpret the contents of an LKU are called *association interpreters* ( *AIPs* ).

If all the relationships among the LKUs had to be represented explicitly, the resulting data schema would be highly application-dependent and inflexible. It is impractical and unnecessary to represent all possible relationships explicitly because some relationships are difficult to extract and unnecessary for immediate applications. Thus, when a user organizes data for a system that uses explicit relationships, he will extract and explicitly represent only those relationships that are useful for his applications. Time and labor consuming reorganization of data schema would then be required whenever a new application needed relationships which have not yet been extracted.

In the implicit relationship approach, however, relationships among LKUs are extracted at run time, not at the data generation time. When existing AIPs cannot extract relationships necessary for a new application, only additions of new AIPs are requested; reorganization of data schema is not necessary.

### 3.3 Association Interpreters

An association interpreter interprets the contents of a given LKU, extracts some implicit relationships in the LKU, and outputs a set of keys and auxiliary information as explicit representations of the implicit relationships. Different AIPs extract different relationships from the same LKU. Simple AIPs can be implemented using pattern matching techniques; complex AIPs may employ parsing techniques which have been adopted in many language processing systems [Calzorali84b, Nagao80].

As an example of a simple AIP, we will consider an AIP which returns synonyms such as 'glad' from an LKU of the word 'happy' described in

section 3.1. The following algorithm realizes this AIP:

(1)Divide the contents into a set of individual components (ex., definitions of the word, examples, idioms,...) using some special delimiter such as '¶' or' ‖ '.

(2)Eliminate everything except definitions.

(3)Further divide the definitions into a list of strings delimited by ';' and '.'.

(4)Eliminate strings which consist of more than one word (i.e., those which contain blanks).

We can implement this AIP by combining four procedures, each performing each step of the above algorithm. We call these procedures *filters*. An AIP is defined in terms of a combination of filters. One can define a new AIP having different functions by specifying the different combinations of filters. For example, if we use a filter which takes only the first definition of a word instead of one that performs step (2), we can make an AIP that returns synonyms from only the first definition of a word. If existing filters are not sufficient enough to make a new AIP, a user will have to write only those filter programs having functions that existing filters don't already have. Thus, this scheme requires anly a minimum programming effort for users.

In implementing AIPs in this scheme, it is important to standardize the input/output interface of filters, because different, nonstandardized interfaces restrict their combinations unnecessarily. To maximize the utility of each filter, the interface should be standardized as transparent data, independent of specific dictionaries or meanings.

### 3.4 Base Dictionary and Virtual Dictionary

A *dictionary* is defined as a set of LKUs of the same type. For example, a Japanese-English dictionary is a set of LKUs of the same type; the keys of each LKU are Japanese words and the contents correspond to English words, idioms, and examples. A query to a dictionary is a key with some auxiliary information, and the results of the query are corresponding LKUs. There are two types of dictionaries: *base dictionaries* and *virtual dictionaries*. A base dictionary has its LKUs actually stored in permanent storage while LKUs in a virtual dictionary are dynamically made from LKUs in base dictionaries. A virtual dictionary is composed of one or more base dictionaries connected by AIPs. The virtual dictionary mechanism allows users to define multiple views of the knowledge in the LKB.

When a virtual dictionary is given a query, it returns the answer through the following steps:

(1) Access base dictionaries with keys given in the query.

(2)Interpret the resulting LKUs to extract keys for accessing related LKUs using an AIP.

(3) Access other base dictionaries with the keys to obtain related LKUs

(4) Repeat steps (2) and (3) as necessary.

(5) Transform LKUs obtained so far into desired forms and return them as answers to the query.

In this way a virtual dictionary repeats a cycle of accessing base dictionaries and interpreting LKUs to respond to queries.

An example of a virtual dictionary is an English-Japanese synonym dictionary built from an English-English dictionary and an English-Japanese dictionary connected by an AIP described in 3.3. This virtual dictionary takes an English word as a query and returns Japanese translations of synonyms of the given English word.

## 4. Concluding Remarks

In this paper we have presented the basic model for a lexical knowledge base which stores various kinds of dictionary knowledge in a uniform framework and provides multiple viewpoints to the stored knowledge. The notion of implicit relationship is introduced to represent complex relationships among lexical knowledge units. By introducing the notion of the implicit relationships the electronic dictionary can interpret the lexical knowledge in various ways and thus allow the incremental development of electronic dictionaries. Virtual dictionaries and association interpreters can be built from smaller components (base/virtual dictionaries or filters), and this scheme minimizes the users' efforts to define their own customized dictionaries.

Currently we are implementing a prototype LKB system based on the proposed model. Our future plans are (1) to verify the utility of the proposed model and (2) to study friendly user-interface.

## References


[Amsler84] R.A.Amsler, *Lexical Knowledge Bases*, COLING 84, 1984

[Calzorali84a] N Calzorali, *Machine Readable Dictionaries, Lexical Data Bases, and the Lexical System*, COLING 84, 1984

[Calzorali84b] N. Calzorali, *Detecting Patterns in a Lexical Data Base*, COLING 84, 1984

[ICOT85] T. Ishiwata, H.Tanaka, H.Miyoshi, Y Tanaka, S.Amano, H.Uchida, T.Ogino, and T Yokoi, *Basic Specification of the Machine Readable Dictionary*, ICOT Technical Report TR-100, 1985

[Nagao80] M.Nagao, J.Tsujii, Y.Ueda, M. Takiyama, *An Attempt to Computerize Dictionary Data Base*, COLING 80, 1980

[Weyer82] S.Weyer, *Searching for Information in a Dynamic Book*, Xerox PARC SCG-82-1, feb. 1982

[Weyer85] S.Weyer and A.Borning, *A Prototype Electronic Encyclopedia*, ACM Trans. on Office Information Systems, Vol.3, No.1, Jan. 1985