KOMBASE - A Knowledge Representation System With Frames For An Object-Oriented Knowledge Base

by
Phua Chiu Kiang

Faculty of Computer Science and Information System
University of Technology Malaysia,
Jalan Semarak, Kuala Lumpur.

Abstract

Knowledge representation is an important area of research which is currently being done in the field of Artificial Intelligence (AI). In order to manipulate the wealth of information available in a typical AI application, mechanisms must be provided to represent and to reason with knowledge at a high level of abstraction. Knowledge representation with frames is a structured and object-oriented approach to this problem. KOMBASE is a prototype to a frame-based system containing organizational information of companies and other corporate bodies. This paper describes the approach adopted in the development of KOMBASE and discusses its implementation, particularly from a knowledge representational perspective.

Keywords: Knowledge representation, frames, object-oriented, knowledge base.

Abstrak

Perlambangan pengetahuan merupakan satu bidang yang penting di dalam penyelidikan kepintaran buatan (AI) masa kini. Untuk mengendalikan jumlah maklumat yang begitu banyak yang terdapat di dalam suatu aplikasi AI yang biasa, mekanisma-mekanisma mesti diwujudkan untuk melambang dan menggunakan pengetahuan pada satu tahap pengabstraktan yang tinggi. Perlambangan pengetahuan menggunakan kerangka adalah satu pendekatan yang berstruktur dan berorientasikan-objek kepada masalah ini. KOMBASE merupakan satu prototaip kepada satu sistem berdasarkan kerangka yang mengandungi maklumat organisasi bagi syarikat-syarikat dan badan-badan korporat yang lain. Kertas kerja ini menerangkan pendekatan yang digunakan di dalam pembangunan KOMBASE serta membincangkan perlaksanaannya, khususnya dari segi perlambangan pengetahuannya.

1.0 Introduction

Although knowledge bases share some of the functions found in database management systems such as storing and retrieving information, knowledge bases are much more powerful. In most conventional database systems, the structure of data storage is usually flat, unlike knowledge bases which organize knowledge in hierachical structures. Theoretically, database management systems are not able to handle the hierachical information necessary to fully describe the relationships between various categories of information in the way that knowledge bases can.

Kertaskerja ini disunting oleh: Pn Faizah Ahmad.

In a knowledge based system, the hierachy and the data are all part of the same structure. Knowledge based systems have more powerful retrieval features compared to those found in database systems. Unlike database systems where only the data that is already stored in the database can be retrieved, knowledge based systems can infer information from the data already in the system.

Artificial Intelligence (AI) programs in general, and knowledge bases in particular, use structures called knowledge structures to represent objects, facts, rules, relationships and procedures. Functionally, these knowledge structures provide the needed expertise and information so that a program can operate in an intelligent manner. Knowledge structures are usually composed of both traditional data stuctures and other complex structures such as frames, scripts and semantic networks. The knowledge representational scheme of a knowledge base can be done using several techniques, which is usually central to the effectiveness and efficiency of the knowledge base towards its specific problem area (or *problem domain*).

Over the years, numerous knowledge representation schemes have been proposed and implemented. The object-oriented approach to data and knowledge representation is one of the most important areas of research in recent years. The ability to directly model real-world entities such as customers, cars, accounts, orders and so on, through object abstractions, offer software developers the modularity and functionality to design and construct complex systems quickly and efficiently. The frame-based approach to knowledge representation presents a closely related model to the object-oriented approach.

KOMBASE is a prototype to a database/knowledge base[1] which uses frames as its primary representational scheme. It was built as part of a study to investigate the effectiveness of using a frame-based approach to manage potentially vast amounts of information in a quick and efficient manner.

2.0 Knowledge Representation With Frames

Knowledge representation with frames was first introduced by Marvin Minsky. The concept of frames consists of separating knowledge or information into their respective categories and grouping them into tables or templates. These clusters of information contain knowledge specific to a certain subject, and is given a name for which the frame shall then be identified.

As all related information to a subject is grouped together, frames, as well as the frame-based system underlying it, have a few advantages compared to other rule-based systems. These are:

i. They are more structured and easier to manage.

ii. They represent information organisation of real-world systems in a more realistic manner as they model the way humans think and make decisions.

A frame consists of an indefinite number of named slots. These slots are almost similar to the fields in a database, but have a variable length or size and are not constrained to a certain type of information only. Each slot can store default values and may have subcomponents or *facets* into which one may place additional information concerning the slot value such as type specifications, constraints, or triggered procedures known as *demons*. Demons are useful for consistency checking and for simulating user initiated responses (for example, reacting to a push button or switch).

---

[1]*Although all the functions performed by KOMBASE are quite similar to that of a database management system, the approach towards its design is that of a knowledge base, using more complex structures to represent information. Hence, all future references to KOMBASE will consider it as a knowledge base.*

A slot can also contain a pointer to another frame, otherwise known as a child frame. This gives the frames system a hieracnical structure and properties of inheritance. Inheritance allows frames to inherit the properties and information of other frames. For software engineering purposes, this not only means a considerable saving in terms of time and cost, but eliminates redundancy and ensures the integrity of the information as well. It is this nature of frames which make it so consistent with the object-oriented approach to knowledge representation.

3.0 The KOMBASE System

KOMBASE was designed primarily as a prototype system to compare, in terms of design and implementation, the AI approach using a frame-based knowledge base against the traditional method using conventional databases. For this purpose, the programming language Prolog was chosen because of its declarative and procedural nature, which made it easier to describe the complex relationships between the frame structures. In reality, however, almost any language can be used, although the code needed to simulate the mechanisms in the system using pure procedural languages would almost certainly be much longer and more difficult to write. For this prototype version, KOMBASE was written in Turbo Prolog Version 2.0 for the IBM PC.

KOMBASE was designed as a knowledge based system which contains information to companies and other corporate bodies. It was built with these objectives in mind:

i. Provide the user with a quick and easy way to access information concerning any company he or she wishes.

ii. Determine the companies which supply a certain product or service which the user is interested in.

iii. Provide the user with a list of alternatives other than the company/supplier which the user already has.

iv. Provide all critical and up-to-date information concerning all the companies and corporate bodies in Malaysia.

v. Enable the user to assess the geographical locations and distribution of any company, product, or service provided within the country.

vi. Act as a guide to local selling and buying.

3.1 System Design

Kombase must be able to retrieve information needed by the user in the following ways:

i. The organisation chosen.

ii. The category of the company, product, or service needed.

iii. The product or service needed.

iv. The geographical area concerned.

v. Other factors. (These include information retrieval using any of the criteria used to categorise the information in the knowledge base.)

3.1.1 The KOMBASE System Hierachy

KOMBASE can be represented as a hierachical structure with the companies' information strucure as its base. Data pertinent to the company's organisational information is used to represent the basic data structures for the system. Based on this hierachy, a frame is then opened for each of the information blocks in the company structure. Everytime a new company is added into the system's knowledge base, all the frames will be opened and filled in sequence. Figure 1 shows the hierachy of the structure used for the system.

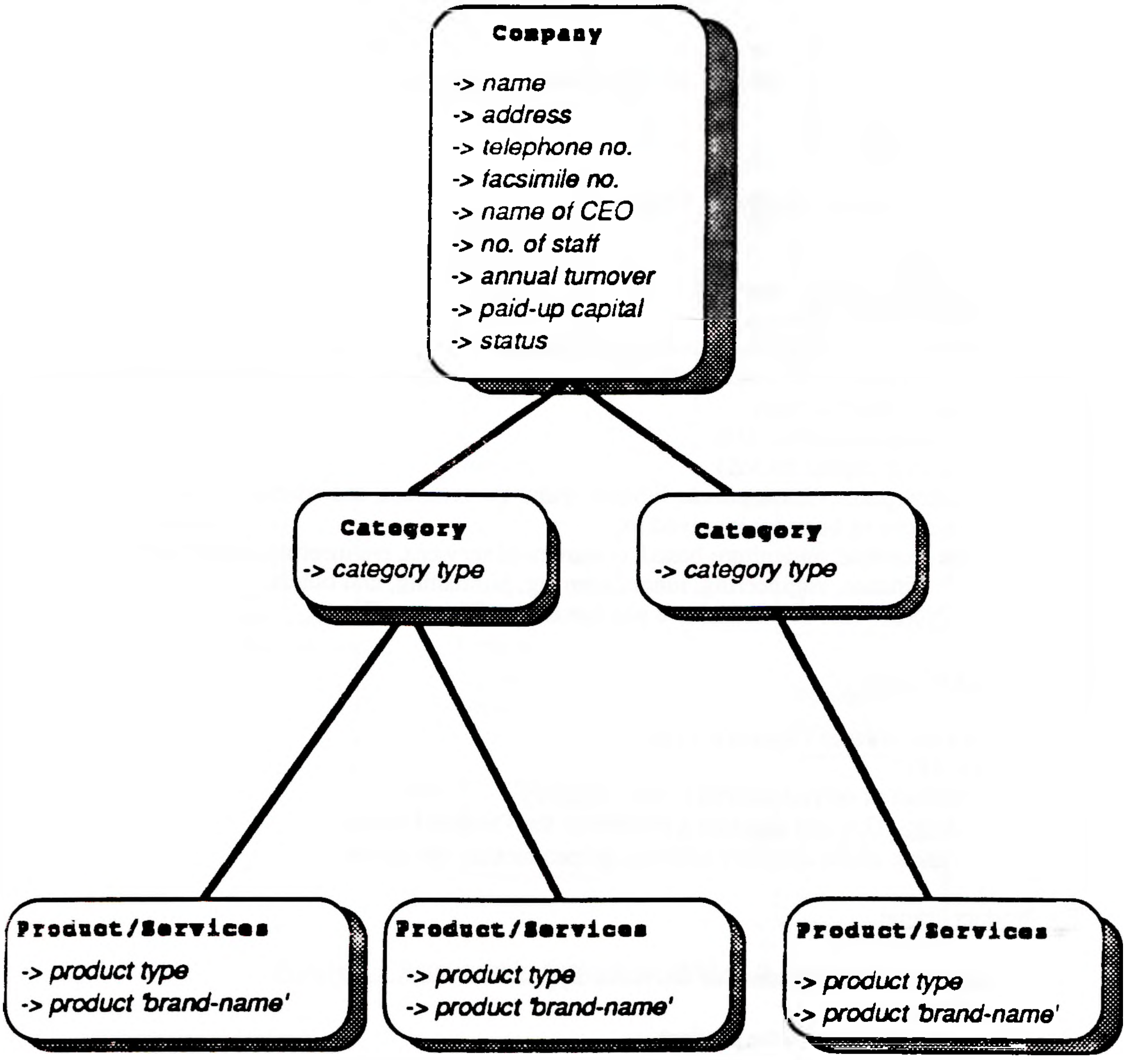

Figure 1 Hierachy of the KOMBASE frames structure

KOMBASE must have the capability to be retrieve its information by the name of the company, its category, or the products or services offered. The values used to identify the categories as well as the products/services, however, are not unique. Due to this, the category and the products/services frames, must include a slot to identify their parent frame, that is, the frame from which these frames are referred.

3.1.2 KOMBASE Data Structures

The KOMBASE knowledge base consists of a dynamic system of frames which can be expanded, updated, and deleted. These frames themselves consist of slots, each one responsible for a certain piece of information relevant to the frame. A slot can refer to another frame, and in KOMBASE, these slots are marked in the slot list as *'frame'*. All the other slots are marked as *'open'* frames and contain string-typed values.

One of the inherent qualities of frames, that is its ability to store default values and perform range-checking for data entered in its slots, has not been implemented in this system. This is because most of the slots in this system contain values which are unique and do not have a specific range, making it difficult to assign default values and perform range checks on them. Furthermore, this feature would have considerably increased the storage demands on space for the system. Default attributes and range-checking, however, remain a feasible solution for the system if more discrete data areas are included into its knowledge domain.

The following is the structure of the frames used in KOMBASE:

<u>The Company Frame</u>

Frame Name: Company Name
Slots:
- address
- telephone contact number
- facsimile number
- name of the Chief Executive Officer (CEO), or Head of the Organisation
- total number of staff
- annual turnover (in M$)
- paid-up Capital (in M$)
- status (public-listed, private limited, quasi-government, and others)
- category of business involved in, for example, agriculture-based, commercial services, engineering, entertainment, finance, engineering, manufacturing, publishing, and others.

  <u>Note:-</u> This slot contains a pointer to the Category Frame.

<u>The Category Frame</u>

Frame Name: Category Type
Slots:
- product or service offered in this Category

  <u>Note:-</u> This slot contains a pointer to the Product Frame.
- name of the company offering the product/service above

<u>The Product Frame</u>

Frame Name: Product or Services Type
Slots:
- 'brand-name' of the product
- name of the company offering the product/service above

  <u>Note:-</u> For simplicity, the services and product categories share the same frames and are regarded as one single type throughout the system.

Please note that there may be more than one Category type for each company frame and more than one Product or Service offered under each Category. As an example,

Company XYZ is a multinational company with business interests in various fields such as rubber plantations (agriculture-based category type), industrial products (manufacturing) and real-estate. In manufacturing itself, XYZ products range from simple cutter-tools to large industrial machines, many of them marketed under different names.

Figure 2 shows how the Company, Category, and Product frames are linked to one another.

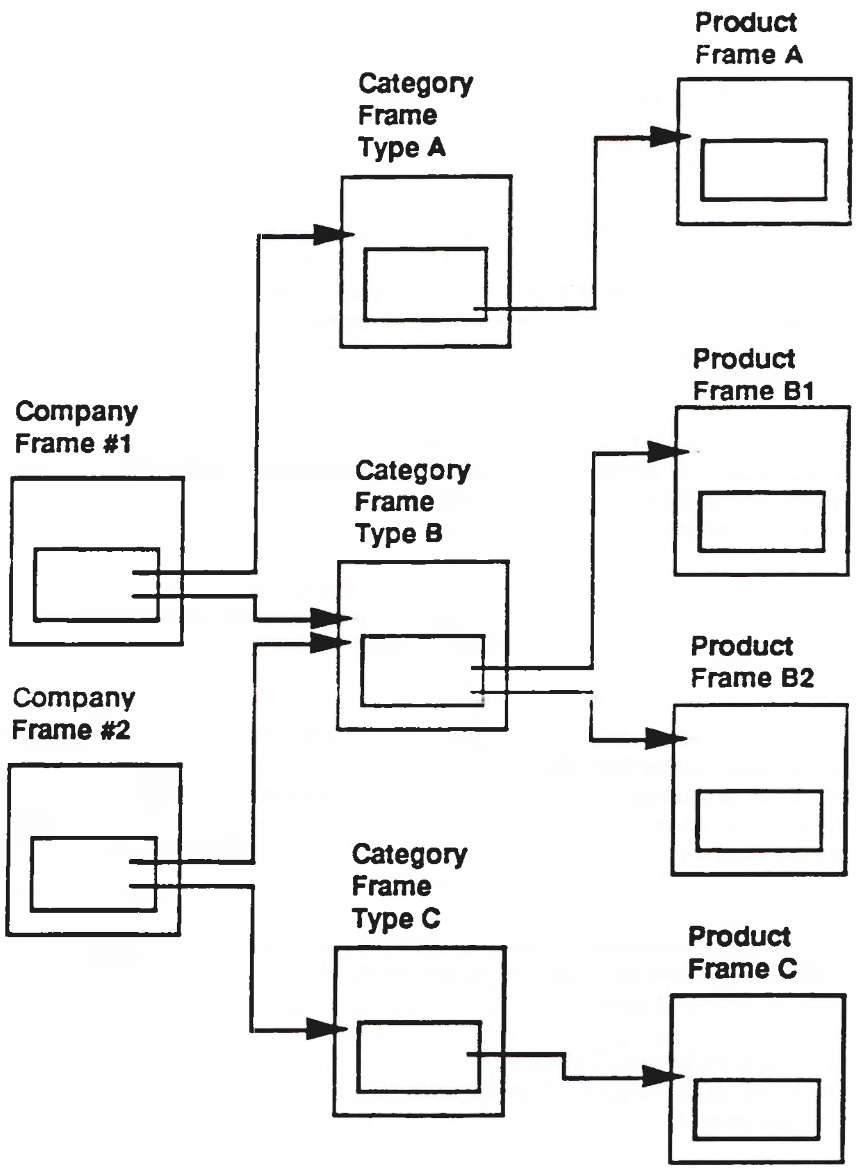

Figure 2 Relationship between KOMBASE frames

3.1.3 The Graphical User Interface

In order to give the user a faster and easier way to interact with the knowledge base, a simple menu-driven interface was built for KOMBASE. All consultation and inferences between the system and the knowledge base is done through the interface and should be transparent to the user. The user is guided through a session with KOMBASE by various menus and a series of prompts. Information retrieval and querying can be done through company, category, product type and name, and also geographical area identification.

The interface also provides facilities to add new frames into the knowledge base, edit frames, as well as removing them from the knowledge base. Users can choose another file for consultation and also save any new information into a diskette or hard disk using the interface.

4.0 Using KOMBASE

This version of KOMBASE can be used in two ways, either through the Turbo Prolog goal-driven interpreter or using the KOMBASE Graphical User Interface (GUI).

4.1 The Goal-Driven Approach

KOMBASE was designed in two separate modules, that is the main module which generates and manages all the frames in the knowledge base, and the menu-driven interface module. The main module can be executed on its own, without the GUI, and querying can then be made through Turbo Prolog's Goal window. Queries will be in tuples format, declared as goals for Turbo Prolog to interpret.

Two goals must be given prior to any other interaction with the knowledge base:- **init**, which removes any frame structure already present in the memory, and **consult("cdata.dba")**, which makes the initial consultation to load the knowledge base into primary memory, *cdata.dba* being the filename used to store the knowledge base.

The user can then use any of the goals which have been declared and defined in the main module. Among them are:- display_company("company_name"), add_company_frame, and save("filename.dba"). Users can also use compound goals by structuring several Prolog clauses to infer information from the knowledge base.

For example, in order to list out information about all the companies in Selangor under the category of manufacturing, the user will have to execute the program from Turbo Prolog which will respond with a prompt 'Goal?' in the Goal window. The user must then key in both the initialization goals mentioned earlier, init, and consult("cdata.dba"). Turbo Prolog will proceed to load the desired knowledge base and store it into primary memory. If it is successful in doing so, the system will flash a message to signal that all is well and another 'Goal?' prompt will be shown.

In order to retrieve the information on the companies desired, a compound goal, or a goal made up of several other goals, will have to be entered at this point. At the Goal Prompt, the user will have to type in the goals (on the same line):-

```
company_frame(X, state, "Selangor"),
company_frame(X, category, "manufacturing"),
display_company(X),
fail.
```

and pressing the Enter key at the end of the line. If any such company exists, the procedure to display the information for each company will be executed.

The user can thus continue to consult the knowledge base in this manner until he or she wishes to end the session by selecting Quit from the File Menu in Turbo Prolog.

4.2 The GUI Approach

The objective of the menu-driven interface in KOMBASE is to allow a casual user to browse, edit and query the information without making use of the query language. Upon startup, the user will be presented with a menu asking whether he or she wishes to query, add, modify, or delete entries from the knowledge base. Options are also provided for consultation to other files and saving the knowledge base to disk.

In query mode, the interface also provides several functions that can infer solutions from certain types of queries. It is capable of providing more than one solution to any given query given by the user although at present this is possible only at its most basic level.

For the case of the example in Section 4.1, the user will begin by selecting the option 'Consult Knowledge Base' from the Main Menu of KOMBASE. Hitting the Enter key will result in another menu being displayed. The user can then proceed to select the Category Type item from this menu and type in "manufacturing" in response to the question asked. Consequently, by selecting the item "location" from the next menu, the user will be asked if he or she wishes to locate the company by choice of state or town. In this case, 'state' would be the appropriate choice after which the system will ask the user to type in the name of the state which is required, in this case "Selangor". KOMBASE would then display a list of the companies in Selangor which come under the category of manufacturing in the knowledge base, or give a message to say that no such company exists if none is found. The user can then go back to the previous menu and choose to display information regarding the companies concerned.

The retrieval of information from the knowledge base using the GUI approach is basically by navigating through these few menus. More can be added to facilitate better and faster access for the user. Although it is slightly less flexible than the command-line like goal-driven approach, it is much easier to use and requires far less keystrokes for each activity.

5.0 System Evaluation

System evaluation for the KOMBASE prototype system was based on these criteria:- (i) the system's ability to achieve the goals set out in the objectives and, (ii) the system's performance as a gauge to the effectiveness of using frames as a representational method for an object-oriented, data-intensive knowledge base.

1. Meeting Its Objectives

Certainly, the objectives outlined for the creation of a practical KOMBASE system necessitates the implementation and maintenance of a very large and up-to-date information base. In this respect, the current system is severely handicapped as the number of items available in the prototype module represent only a very tiny fraction of the amount of information needed to make this a practical, working knowledge base. With this in mind, however, it is the potential of the system to provide the mechanisms that are able to handle such a situation in future that should be of more concern.

From its current working copy, KOMBASE has demonstrated that fast access and retrieval of information can be done, and this remains one of the system's main strengths. Once a consultation has been made with a specific information base, the system's response to a user's query on any specific company or product is instantaneous. The system is also able to provide alternatives for any one company or supplier which the user wishes to know about. At this point, the system is not able to give advice on these alternatives although with the mechanisms provided, this will remain a very viable option for the future. Additional rules can be added into the frames themselves which can then be accessed and evaluated by the knowledge base to provide a single, most optimistic solution.

The system is also able to provide solutions based on the geographical location and distribution of companies or their products and services. This feature has wide-ranging possibilities, such as identifying the actual distribution of all the companies and industries in Malaysia, the states in Malaysia and their category and range of products and services, and the location from where a particular supplier can be found anywhere in the country. The actual implementation of these functions will of course involve considerable systems development and maintenance.

Other methods of queries can be extended upon the goal-driven approach used within the main module of the system. This gives it the additional flexibility to infer knowledge from the information available in the information base.

2. KOMBASE as a frame-representational, object-oriented, knowledge base

From a development point of view, KOMBASE offers a few guidelines into the use of frames technology in the construction of a general knowledge base. Clearly, frames offer a few advantages in terms of providing the necessary structure for the knowledge representation of the knowledge base. Besides the advantages already mentioned in Section 2.0 of this article, KOMBASE and the frame-based system behind it has demonstrated these capabilities as well:

i. *Ease of development and implementation.*
The part of KOMBASE which does the actual creation, modification, and deletion of the frames is suprisingly small (about 150 lines of Prolog code), but it is able to perform all the functions necessary to manage every single frame and its slots. It is easily understood and can be quickly modified or expanded to enable it to handle different types of frames. Other procedures can be easily added to enhance the frame management system and the project can then proceed from that point.

ii. *Persistence.*
KOMBASE defines and manipulates its data structures as strings, which are then interpreted by the frame management system into hierachies of frames and slots. As there is no need for any mapping between persistent data stored in secondary storage and volatile data in the memory, a considerable amount of coding can be saved. Furthermore, there is no mismatch between the data or information typing facilities and the frame management system, thus providing data protection across the programming language and the filing system. Because of the way it is implemented, KOMBASE data may also exist beyond the lifetime of the program in which it is created.

In a strict sense, the object-oriented faculty of KOMBASE is incomplete. KOMBASE has some properties of an object-oriented approach, in the sense that its frame structures give it a hierachy of classes and inheritance. Although the company information structure may not provide a very good platform on which to model object types, the prototype has demonstrated that frames can be a very modular and structured way to do that. For objects with a high degree of classes, frames may prove to be very effective indeed. On the other hand, the object-oriented model is much larger and more complex than that. Although frame-based knowledge representation schemes provide ways of attaching procedural information to slots, they offer no specific facilities for declaratively describing the behaviour of the object represented by the frame structure [Hughes, 1991]. Frames are passive data structures which may contain procedural code, but which must still be retrieved and invoked by some agent other than the frame. This signifies an important difference between frames and the objects in true object-oriented systems, which encapsulate both state and behaviour and the ability to execute operations without the need for separate, external procedures.

The construction of a representation scheme which encompasses both knowledge and procedures is highly dependent on the type of programming language used. Knowledge representation schemes and languages would have to facilitate both the description of large knowledge based systems as well as the description of inference mechanisms for each subsystem. One method that has been proposed is called the *Distributed Knowledge Object Modeling (DKOM)* whereby a large knowledge based system is described as a collection of hierachical Knowledge Objects (KO) [Ishikawa and Tokoro, 1987]. A KO consists of the knowledge base part, the behaviour part (which procedurally describes metaknowledge of the KO), and the monitor part (which controls incoming messages, supervises the behaviour, and acts as a demon for the knowledge base part and the behaviour part). Although Prolog has some levels of procedural structures, it is still largely declarative and this makes it difficult to write the code necessary to perform the actions needed within the frame. In fact, this was one of the main problems which the KOMBASE designer faced when attempting to integrate the declarative and the procedural parts of an object into the frames. One idea that has surfaced is to write the code for the action part in a procedural language such as C. Its object code is then linked with the rule/fact-level of the frame object written in Prolog.

Possibly one of the main deterrents to the use of this system at the moment, however, is the large amount of storage needed for the knowledge base. This is mainly because KOMBASE stores its information in a series of tuples which are really made up of character strings. In addition to that, the rules and facts needed to represent the frames in the knowledge base have to be distinct so that they can be processed individually. This creates a high storage overhead for every assertion of a frame into the knowledge base. For example, the system took up a disk overhead of 19385 bytes for the storage of just 20 company frames. For the system to become practical it is estimated that at least 1000 companies will have to be included into the knowledge base, requiring a projected storage capacity of about 1Mbytes. More slots would probably be needed in order to widen the domain of the knowledge base and this would increase the demand on storage rather significantly. On the other hand, the high demands on both primary and secondary storage are in fact typical of present day knowledge based systems, and the frame based representational approach should not be dismissed on this count. Because of the way in which the knowledge base was created (lots of duplications of tuple names), data compression can be expected to significantly trim the original storage requirements for the knowledge base.

6.0 Suggestions for Improvements

Although KOMBASE is fundamentally a prototype for a study into the feasibility of using frames to meet the demands of a knowledge based system, there is no reason why such a system should not be implemented into a comprehensive and practical solution to the problems faced by buyers, suppliers, and consumers, in Malaysia daily. The main sources of company/supplier information which we have at the moment are from publications such as The Yellow Pages, Kompass, The FMM (Federation of Malaysian Manufacturers) Directory and others. These sources are not integrated and it may take a reader quite some time to find the information he or she needs. In addition to that, the reader may have to do quite a lot of back-referencing and rechecking in order to trim down the amount of choices which are available. It is with this in mind that KOMBASE was created. However, to make it a more practical system to use, the improvements suggested below must be made first:

i. *Increase the amount and scope of information into the knowledge base.*
The amount of information available in KOMBASE at the moment is clearly insufficient to provide any measure of practical use to a serious user. The addition of information into the knowledge base should not present any difficulties by itself, but the problem of memory management will inevitably arise. One advantage in using the frames system is that chunks of information can be easily organized into smaller parts. Data files can be sorted alphabetically according to the company frames or their category frames. The system can then select any one of these files at a time for consultation depending on the type of query made. The scope of the information can easily be expanded by inserting additional slots into the frames.

ii. *Provide a better and more flexible user interface.*
The present system incorporates a very rudimentary graphical user interface which is still lacking in terms of efficiency, ease-of-use, functionality, and aesthetic values (some menus may overlap onto the display screen and vice versa). Control among the various menus and the display options have not been fully established yet to cater for certain user errors and user initiated interruptions. Features such as *on-line-help* and a higher level of user error detection will be useful as well.

iii. *Provide an inference mechanism to advise the user on a list of selections.*
The inference mechanism will advice the user which company or supplier to transact with, given a choice of possible options. Ideally, the inference should be based on such factors as the location and distance of the companies, their business records, product ratings, and stock performance. This capability will undoubtedly be very difficult to implement effectively, and a module for a dedicated expert system to perform the inferences should

be developed separately. Similarly, KOMBASE can also provide the facilities needed to develop expert systems which utilize the former's knowledge base to give expert advice on business investments and stock market forecasts, among others.

7.0 Conclusion

Although the question remains whether or not KOMBASE is able to provide the services that was described with its objectives, the system has shown that the use of frames as a knowledge representation scheme for a knowledge base can be easily and efficiently managed. The frames system has proven to be a highly structured, very manageable, object-oriented approach to the methodology of representing complex relationships among objects within a knowledge base. Frames, however, do not provide the ability to activate procedures within the frames themselves, and this is one of the main limitations which the system faces in its approach to an object-oriented model. On the other hand, frames handle such properties as inheritance and classes very well.

The KOMBASE system has the mechanisms for the basic representation of complex objects and of their storage and retrieval. It has been successful with a very small working knowledge base and the possibilities for expansion is vast. The challenge remains to bring this prototype to a full-fledged system which is capable of providing practical and useful information to the user. ■

References

[Firebaugh, 1988] Firebaugh, Morris W., *Artificial Intellingence - A Knowledge Based Approach*, Boyd & Fraser Publishing Company, Boston, 1988.

[Hughes, 1991] Hughes, John G., *Object-Oriented Databases*, Prentice Hall International (UK) Ltd, 1991.

[Ishikawa and Tokoro, 1987] Ishikawa, Y., Tokoro, M., *Orient84/K: An Object-Oriented Concurrent Programming Language for Knowledge Representation*, in "Object-Oriented Concurrent Programming" edited by Yonezawa, A. and M. Tokoro, The MIT Press, 1987.

[Rauch-Hindin, 1987] Rauch-Hindin, Wendy B., *A Guide to Commercial Artificial Intelligence*, Prentice Hall Inc., Englewood Cliffs, NJ., 1987.

[Weiskamp, 1988] Weiskamp, K., T. Hengl, *Artificial Intelligence Programming with Turbo Prolog*, John Wiley & Sons, Inc., USA, 1988.