CUSTOMIZING ONE'S OWN INTERFACE USING ENGLISH AS PRIMARY LANGUAGE

Bozena Henisz Thompson
Frederick B. Thompson

Computer Science
California Institute of Technology

5165:TR:84

CUSTOMIZING ONE'S OWN INTERFACE USING ENGLISH AS PRIMARY LANGUAGE

Bozena Henisz Thompson
Frederick B. Thompson

September 1984

5165:TR:84

Computer Science Department
California Institute of Technology
Pasadena, California 91125

CUSTOMIZING ONE'S OWN INTERFACE USING ENGLISH AS PRIMARY LANGUAGE

Bozena Henisz Thompson
Senior Research Associate in Linguistics
and
Frederick B. Thompson
Professor of Computer Science
California Institute of Technology
Pasadena, California USA

ABSTRACT

ASK, A Simple Knowledgeable System, is a total system for the structuring manipulation and communication of information. Its primary user interface is a dialect of English. In this interface, the user may work with texts, pictures and images, all integrated within the same semantic net data base.

We concentrate here on the ASK System facilities for customizing the user's interface which are at a series of levels, from those immediately available to the user to those best exercised by the applications programmer.

The user can extend his vocabulary, not only by new terms -- names and attributes, nouns and verbs -- but also by definitions.

ASK includes two dialogue systems that are of particular importance for customizing the user's environment. The Dialogue Designing Dialogue is a complete natural language automatic programming system. The user can create his own mini-expert systems, so he can accomplish complex tasks by means of a simple dialogue. With some knowledge of programming, recursive calls, stacks, etc. provide powerful tools, all at the high level of a natural language dialogue. The Foreign Access Dialogue can be used to establish lines of communication to other, foreign, data bases, so that in answer to a single query, the system integrates data from the user's own data base and foreign data bases. No restriction is placed on these heterogeneous data bases other than their accessibility, possibly over phone lines, to the ASK system computer.

Corresponding to each natural language context for user interaction, there is a meta-context for the applications programmer. The central notion of meta-contexts is the rule of grammar and associated interpretive procedure. The applications programmer adds new capabilities by adding such rules, thus extending the range of resources at the fingertips of the user.

I. INTRODUCTION

ASK, A Simple Knowledgeable System, is a total system for the structuring, manipulation and communication of information. Its primary user interface is a limited dialect of English. In this single user interface, the user may work with texts, graphics and images, may communicate with others in the fashion of electronic mail, all integrated within the same semantic net data base. In this paper, we focus on the ASK System facilities for customizing the user's interface to his immediate needs. These facilities are at a series of levels, from those immediately available to the user to those best exercised by the applications programmer.

From the point of view of ASK, a complex of personal computers, together with its facilities for interaction and communication and its data base, constitutes the information context for a group of people who are working together on a single but complex task -- for corporate management, a research team or an administrative staff. Their network of computer work stations is the central nervous system of their organization. The data base constitutes their group memory, the tacit information which is the basis for their communication and coordination. As such, the structuring and maintenance of that data base, that information context, is a primary activity of the group itself.

The ASK System is designed to implement this concept, to make the computer a facile instrument of these dynamics. Thus it emphasises the processes of change, the structuring and extension and modification of this information context by the users themselves, the constant customizing of their own system to their immediate needs and current concepts.

It is instructive to examine this customizing process as it goes hand in hand with the development of a research team itself. When such a team is first brought together, they are disparate people who have just been reassigned from disparate tasks. If one overhears their first meetings, their sentences are long, with lots of descriptive material. It takes time to establish the ultimate, tacit context in which they will work. As this context develops, "social" English is slowly replaced by jargon, the meanings of words take on subtle distinctions and inferences can safely be drawn from more sparse cues [1]. The media of their interface also evolve; engineering drawings, standard report forms, shop flow charts, graphs of time series emerge as the visual embodiment of their common understanding and the instruments of their communication. Some aspects of their tasks are reduced to repetitive processing, initiated and directed by sparse inputs. The social English of their first exchanges becomes a background, primary language, available when sudden new circumstances require changes in jargon and established procedures, but usually replaced by the many forms of efficient communication that will have evolved.

These dynamics were the focus of our design of ASK. Thus ASK includes facilities for bringing graphics into the "picture." It provides tne means by which the users can build succinct dialogues to do the tedious, repetitive but often complex tasks in response to a few simple cues. It is these customizing aspects of ASK that are the subject of this paper. We will discuss these customizing aspects in four steps:

- o Customizing the Domain
- o Customizing for Specialized Objects and Procedures
- o Customizing the Interaction
- o Customizing the Source of Data

In each step, we will identify the requirement, characterize the particular customizing process and give a few examples of its application.

II. CUSTOMIZING THE DOMAIN

A. The Bulk Data Input Dialogue

Suppose an organization obtains the ASK System. How do they build into it the data and vocabulary that will make it into a useful tool? Such an organization today will already have one or more data bases. How can they apply their new system to their existing data? We assume that their data base is in machine readable form, and indeed available in formated text files. Figure 1 is an illustration of such a file. If the data is in another data base system, it is most likely that this system will have the facilities to produce such a text file which can then be used for input into ASK.

```
NN Alamo          Huston        3456 US
CC oil               New York
NN Tokyo Maru     Yokohama      2800 JA
CC elect. equip.     Hong Kong
CC vehicles          Bombay
NN British Star London          1688 UK
CC wheat             Liverpool
```

Figure 1: An Illustrative Data Base

ASK includes a dialogue, initiated by the request:

>Bulk data input

This dialogue elicits from the user information about the data file, namely: (a) the formats of its records, (b) declarative statements which specify how the data is to be stored in the ASK data base, and (c) a variety of related information such as translation and scaling instructions and how errors identified in the data should be processed. In the example given in Figure 1, this includes the information that each ship record consists of two kinds of physical records; first, one identified by "NN" that gives the ship's name, its home port, bale cube capacity (in meters cubed) and its flag; followed by one or more "CC" records, each giving an item of cargo and its destination. Within the ongoing dialogue, the user is prompted to enter a number of things about his data; for example, a translation table, "US" into "United States", "JA" to "Japan", etc. The last step of the bulk data input dialogue is to request the name of the file which includes the data. The system then proceeds to fetch the data, add the necessary words to the vocabulary, and add the data to the user's ASK context.

B. Building the Vocabulary and Adding Definitions

A data base, functioning in its role as group memory, is constantly being extended, changed and modified. One should be able to make such changes as easily and directly as asking questions.

The vocabulary of a natural language data base system is made up of three parts: (a) the so called "function words" such as prepositions and conjunctions, (b) those words which name specific entities in the data base and direct relationships between them which express primitive knowledge contained in the data; and (c) defined words and expressions which succintly express more complex concepts built up from the primitive ones. Definitions are a way to add knowledge of the domain to the system. How are new words to be added to the vocabulary? Words of type (a) are already available. We now illustrate how to add new primitive data words and definitions.

The ASK data base is of that family called "entity attribute" data bases. Thus we have four kinds of primitive data items: objects, classes, attributes and relations. Attributes and relations are binary; attributes differ from relations only in that they are single valued. Thus "father" and "home port" are attributes, whereas "child" and "destination" are relations and may be multiple valued. As we shall see in Section III below, objects may be of several types. Among these types are individuals, texts, pictures, numbers, etc. Classes, attributes and relations are strictly typed. The following protocol shows how to add new primitive vocabulary items [2].

>Create the individuals named Boston, Hong Kong and United States
The following new individuals have been added:

```
 Boston  Hong Kong  United States
>classes: city, male, female, human, section manager
 The following individual classes have been added:
 male  female  human  section manager
 The individual class city was already in the vocabulary.
>Create a city named New York
 The individual New York has been added as a member of the
 class city.
>Text: Budget Memo
 Enter the text for Budget Memo.
 There will be a budget meeting in my
 office tomorrow at 11:00 AM.

 The text Budget Memo has been added.
 File: Current Memo
 The text class Current Memo has been added.
>text/number attribute: page count
 The text/number attribute page count has been added.
```

Data interrelating these various items is added just as easily, as illustrated here:

```
>Boston and New York are cities.
 Boston has been added to the class city.
 New York was already a member of city.
>The page count of Budget Memo is 1.
 1 was added as page count of Budget Memo.
>File Budget Memo in Current Memos.
 Budget Memo has been added to Current Memos.
>Send Budget Memo to the section managers.
 Budget Memo has been added to the mail boxes of section
 Managers.
```

Definitions are particularly useful in cutting down on the amount a user must type. Some will be of general use, for example:

```
>definition:child:converse of parent
 Defined.
>definition:sibling:child of parent but not oneself
 Defined.
>definition:brother:male sibling
 Defined.
>definition:uncle:brother of parent
 Defined.
```

However, to realize these potential savings, they must express the immediate need of the user. Here is a group of more complex definitions that might be of use when completing an income tax return:

```
>definition:rate:.30 if income is less than 30000
 Defined.
>definition:rate:.45 if income is between 30000 and 39999
 Defined.
>definition:tax:(rate * (income - (600 * number of children))
 Defined.
>John's income is 35000.
 35000 has been added as John's income.
>John's tax?
 15210
```

C. Coordination and the Adding of Verbs:

Verbs, in natural language, usually express complex concepts highly idiosyncratic to the domain of the discourse. How are verbs to be customized in a system such as ASK [2]?

Verbs often interrelate a number of items around the same concept. This gives rise to two problems of coordination. To illustrate the first of these problems, refer to the data of Figure 1. Note that the ship Tokyo Maru is carrying electrical equipment to Hong Kong and vehicles to Bombay. One is tempted to paraphrase the verb "carry", in the context:

"Does the Tokyo Maru carry vehicles to Hong Kong?"

as:

"The cargo of Tokyo Maru is vehicles and its destination is Hong Kong?" but this paraphrase is true whereas the original question is false. As long as the data base is maintained in the form of binary attributes and relations, this distinction poses a problem.

The second problem of coordination can be illustrated using the following definition:

```
>definition:area:length times width
```

The use of this definition in the question:

```
>What is the area of the conference room?
```

poses no problems. However, suppose the hotel has three public rooms whose lengths and widths are given by the table:

	length	width
conference room	80	50
lobby	60	20
dining room	40	70

The answer to the question:

>What are the areas of the hotel's public rooms?

is not: 4000, 1600, 3200, 3000, 1200, 2400, 5600, 4200, and 2800; rather it is 4000, 1200 and 2800. (In mathematical terms, one wants the dot product, not the cross product.) This second coordination problem is also illustrated by consideration of the following question, noting that not only the missing ship but the implied shipment must be taken into consideration:

>What is carried to where?

In relational data base systems, the first problem of coordination is solved automatically as long as the query seeks interrelationships that are across the fields within a single relation. If the conceptual relation does not have a key field, an extra field or entity is added as part of the normalization procedure. Coordination across relations is not retained in the data base. Essentially, we use this same solution. Individual objects are created, often without assigning any name to them, which carry the task of coordination. In most instances, we have found, the class of these coordinating objects has a natural name.

Consider the data of Figure 1. We can easily think of each entry of type "CC" as associated with a shipment; we can even imagine that in a "more complete" data base, each of these "shipments" would have a legitimate name, say its invoice number. A solution is to introduce "shipment" as a relation. We illustrate by adding a shipment of vehicles to the cargo of the British Star, with destination Naples:

```
>Create an individual.
>It is a shipment of the British Star.
>Its cargo is vehicles.
>Its destination is Naples.
>List the cargo and destination of the shipments of the
 British Star.
 List the (cargo and destination) of (the shipments of the
 British Star).
 cargo     destination
 wheat     Liverpool
 vehicles  Naples
```

Initially, the only verbs in ASK are "to be" and "to have" and their other forms, possibly with comparators; e.g, "was", "exceeds". All other verbs are added by paraphrase. For example:

>verb:ships "carry" wheat to London:there is a shipment of ships whose cargo is wheat and whose destination is London

>What cargo is carried to Hong Kong by ships whose home port is Yokohama?

elect. equip.

ASK develops the deep case structure of the user's question, expanding all defined terms. Thus the above question becomes:

"there is a shipment of (ships whose home port is Yokohama) whose cargo is (what cargo) and whose destination is Hong Kong?"

Definitions play an important role in making this solution habitable.

For example, after the bulk data input dialogue one can ask:

```
>Is there a shipment whose cargo is vehicles?
 Yes
```

However one could not yet speak of the cargo of a ship, since only shipments have cargos. So we add:

```
>definition:cargo:cargo of shipments
>What is the Tokyo Maru's cargo?
 elect. equip.
 vehicles
```

Of course, now "cargo" has two meanings. "Inside" ASK, 'What is the Tokyo Maru's cargo?" is ambiguous. One analysis produces a diagnostic response: "There is no Tokyo Maru's cargo.", the other produces the correct answer. Since there is a non-diagnostic response, it is the only one output [3].

Thus ASK is able to handle correctly both of the two problems of coordination:

```
>What is the cargo of the Alamo?
 oil
>What is the cargo of shipments whose destination is Hong
 Kong?
 elect. equip.
>What is carried to where?
 oil             New York
 elect. equip.   Hong Kong
 vehicles        Bombay
 wheat           Liverpool
 vehicles        Naples
```

III. CUSTOMIZING FOR SPECIALIZED OBJECTS AND PROCEDURES

A. Object Types

In a great many applications of computers in professional environments, one finds specialized bodies of data with their own

data structures and specialized procedures for manipulating these structures. A good example are texts. A particular text, say Budget Memo, exists as an object in the data base, with its attributes and relations (e.g., "originator", "keyword") and with the classes (e.g., "Current Memo") that contain it. It also has a text body, with its own data structure. Text editing operations are good examples of the specialized procedures that are not concerned with data base aspects but work exclusively on this text body. ASK handles such objects that have both a data base life and a separate data entry life, and it does so in the same, common data base - natural language environment.

Suppose one wished to use a procedure 'word count" which, when given a text object, would return the number of words in that text. One would add the rule of grammar:

```
RULE
<noun_phrase:+attribute+number> => "word count"
POST word_count_proc
```

where the procedure 'word_count_proc", given a text as argument, would return the word count. ASK is designed so that such a rule would fit right into the rest of ASK English. Thus such requests as the following are immediately available:

```
>Display items of my mail whose word count is less than 50.
>List the author of each abstract whose word count exceeds
 300.
```

The data about an object in an ASK data base is held in a "record." Such a record mostly contains pointers to other records; these are the semantic net links that give the object meaning with respect to other objects, classes, attributes and relations in the data base. In addition, the header of an object record contains a field from which one can link other kinds of data structures. It is this extra field, for example, that holds the pointer to the text body for a text object. Simple utilities exist for getting at these data structures and for calling specialized procedures for manipulating them. Special facilities are also provided for declaring to ASK a new object type; these become types in the sense of strict typing of classes, attributes and relations.

ASK is implemented in Pascal. The specialized procedures which manipulate these specialized data structures may be implemented in other languages than ASK. Usually, in most current computer systems, the Pascal system provides for linking programs written in other languages with Pascal code. This is particularly true in a UNIX environment. In these cases it is particulrly easy to integrate the two systems, ASK and an existing body of data structures and procedures, thus embedding these special objects in a data base environment and providing a natural language interface.

B. Object Types Currently Implemented and Planned

The ASK System design calls for the initial support of the following types:

- o individuals
- o numbers
- o matrices
- o strings
- o schedules
- o texts
- o graphics
- o images

Since we are a small, academic research team, full implementation of these types is a rather large undertaking for us. Our objectives differ in each case. Here is what we are doing.

Individuals, having no special data structure or procedures, are fully implemented. So are numbers. The usual "pure" number operators, such as sine and cosine, absolute value, square root, are all there. As for "mixed" operators, e.g., average, variance, maximum, number of, they are well covered.

The simple matrix operations are there. We are now adding the more complex statistical operators. There is a system variable: "current matrix", which is set by an expression of the form: "Consider <noun_phrase:+matrix>." An important case is illustrated by the example:

>Consider the covariance matrix of height, weight and age of employees. Regression coefficients, factors etc. will be computed in terms of the current matrix.

Strings are only marginally there. We are waiting to experience some legitimate need for them. One use is in the response to dialogues; for example:

>Should the color be R(ed), B(lue) or G(reen)?

where such responses as "R", "gr" etc. should be acceptable. For this purpose we need the operator: "initial segment".

A series of dialogues for maintaining schedules has been designed and is partially implemented. It is awaiting the completion of those aspects of ASK English that handle tense and time. The latter, ASK time, is coming along well, however that will be a paper in itself. When time and the schedule dialogues are completed, one should be able to say:

>Schedule a meeting.
>When should the meeting be? .....
>Who should attend? .....
>The following rooms are available: .....
.....

with the obvious, desirable consequences of notification, reserving, bulletin board notice, alerting, etc.

ASK now includes text handling and electronic mail. We do not plan to implement our own text editor. It would be easy, though time consuming, to add one. In any commercial implementation of ASK, we expect that the commercial company will want to incorporate their own standard text editor.

We feel that graphics are an important addition. We have done extensive experimentation in this area. We plan to follow the standards for interactive graphic operators. However, we need to add to the underlying graphic data structures links with the data base. We had a fine Master's thesis [4] that supported the following protocol:

```
>Display the aft hold of the Alamo.
 (The outline if the cargo space is displayed together with
 all cargo items in place.)
>List the cargo type, destination and weight of <cursor>.
 ...
>Display a M-74 truck.
>Move it to <cursor>.
>Stow.
```

Much work remains to be done in this area.

Finally, consider images. We are participating in an application of ASK in which the domain of application concerns the images of the earth that are returned by the Space Shuttle, the so called LANDSAT data [5]. The data concerning each image forms an array; an entry in this array concerns an individual pixel, and contains nine spectral values from the electromagnetic spectrum emitted by the corresponding small area of the earth's surface. There exists a large family of procedures that manuipulate these image files. We are in the process of adding appropriate rules, incorporating these procedures, so that this very large data base can be approached more naturally by the many earth scientists who wish to use it. The following illustrates a typical query sequence:

```
>Consider the June 20, 1983 image of Singapore.
 The primary focus is June 20, 1983 image of Singapore.
>Enlarge it by 5.
 The primary focus is it [June 20, 1983 image of Singapore]
 enlarged by 5.
>Display it centered on longitude ... and latitude
 (displayed on separate monitor)
>What percent of it is water or vegetation?
 What percent of it [June 20, 1983 image of Singapore
 enlarged by 5]
 is water or vegetation?
              percent
 water            54
 vegetation       12
```

IV. CUSTOMIZING THE INTERACTION

A. Dialogues as Extended Definitions

The general notion of dialogue is that of a conversation between two or more conversants about a specific subject matter. Here we have in mind a much more limited notion, namely, where the computer prompts the user for needed inputs and then proceeds to respond to these inputs in a proscribed, perhaps complex, manner. Here is a simple dialogue for adding items to a bibliography:

```
>new item
>Title: Ivanhoe
>Author: Scott
>Keyword: romance
>Keyword: England
>Keyword:
 The book Ivanhoe has been added to your bibliography.
>_
```

Obviously, dialogues can be much more complex than this, involving complex decision structures which depend not only on the user's responses but also on the current state of the data base. A segment of a dialogue for loading cargo on a ship might look like this:

```
....
>On what deck: Upper Vehicle Stowage deck
>How many should be stowed there? 6
 There is only room for 4.
>Should 4 be stowed there? no
>On what deck: ...
```

Dialogues, in this restricted sense, can be considered an extended form of definition where the number of parameters involved can not be conveniently expressed in a simple syntax. Consider, in this regard, the notion of enlarging an image. Enlarging involves two parameters, the amount of enlargement and the point on the image around which the enlargement is to take place. If the second is not expressed in a request for an enlargement, it can be safely assumed to be the center. But the enlargement factor is an obligatory parameter. Here these usages are illustrated in equivalent ways:

```
>Enlarge San Francisco image #74 about the center.
>By what factor? 2

>Enlarge San Francisco image #74 by 2.
 Enlarging San Francisco image #74 about the center.

>Enlarge San Francisco image #74.
>By what factor? 2
>Centered at? center
```

Dialogues of this kind are a convenient way to use the computer in carrying out repetitive tasks. Customizing a system to one's needs should include providing such dialogues. There is a problem, however; in most cases the need for such a dialogue is transient, arising only in regard to an immediate task. What is needed is a means by which the user can create such a dialogue quickly and be able to use it immediately.

B. The Dialogue Designing Dialogue

ASK includes a dialogue, the Dialogue Designing Dialogue, by which one can create a dialogue of the above restricted form quickly and easily [6]. Simple dialogues, like the "new item" dialogue indicated above, can be implemented by the user, who presumably has no programming knowledge. However, this dialogue system includes more advanced capabilities, stacks and queues, and the ability for having one dialogue call another including both recursive calls and passing of arguments. In a formal sense, the Dialogue Designing Dialogue constitutes a complete natural language programming system. It thus becomes a powerful tool for the applications programmer, implementing the software engineering notion of rapid prototyping.

In designing a new dialogue, how is the the decision structure implemented? In the design, the designer sets up the prompts to which the ultimate user will respond; this anticipated response is assigned to a local variable. These local variables are numbered and indicated by the symbols <1>, <2>, etc. Suppose, for example, the response to the prompt: "Title: " is assigned to <1>, and the response to the prompt: 'Author:" is assigned to <2>. Then the designer would specify the action:

>Action: Author of <1> is <2>.

so that the data would be added appropriately to the data base when the dialogue is subsequently used. This example illustrates how action statements, and also condition statements, are similar in structure to normal ASK English statements. Actions which may be indicated this way include extending and modifying the data base, displaying information to the user, and producing, filing and sending texts. Using these latter actions, the Dialogue Designing Dialogue includes the full capabilities of a report generator.

V. CUSTOMIZING THE SOURCE OF DATA

A. Access to Data

Section II.A discussed how one uses the Bulk Data Input dialogue to build the data in a formated text file into an ASK data base. We consider three cases where that may not be a useful, or

even a possible thing to do. First, ASK may be put into a setting where large and important data bases already exist, together with a well trained staff who routinely maintain these data bases and supply standard reports, often in hardcopy, to key offices in the organization. It is unlikely that there will be willingness on the part of management to switch the whole operation into a totally contained ASK implementation during the first months ASK is being used. Such change usually takes a long time. Second, it may be that the data base is constantly and rather rapidly being updated and it would be too costly to rebuild the ASK data base to keep up with these changes. Third, it may be that access to the external data base is supplied by a commercial concern, is available only over the phone line, and would be too expensive to acquire each working day in order to maintain one's own copy.

In all three cases, it would be desirable to interface one's ASK System to the other, external data bases. What one really wants is the capability to incorporate access to the external data base as an augmentation of the data resources already available. When one asks a question, it may require some of the raw data to be drawn from one external data base, other of the raw data from a second external data base, possibly over a phone line obtaining the very latest information, and finally the remainder of the data from one's own data base. The integration of the data from all of these sources would then be carried out and the single, integrated result be presented to the user.

B. The Foreign Access Dialogue

ASK accomplishes this [7],[8]. Suppose we have acquired the capability to access an external data base and we wish to attach it to our ASK System in the manner described above. To do so one initiates the Foreign Access Dialogue by typing:

```
>foreign access
```

ASK proceeds to hold a dialogue with the user. This dialogue has two parts. In the first part, the System obtains the information of a "hardware" and system nature about the external data base system; for example, telephone number, baud rate, hand shake conventions, log on protocol, etc. In the second part of the dialogue, the user is asked for the access words he would like to use in refering to specific items of data he would like to access from the external data base, and for the translation of these individual words into the query language of the external data base system. When these two aspects of the Foreign Access Dialogue are completed, the access path has been established. If the user now includes one, or more, of these access words in a query, the ASK System automatically opens the channel to the external data base, obtains the necessary data and integrates it into the context of the query.

If the first part of the foreign access dialogue has already been accomplished for a given external data base, it does

not have to be repeated each time new access words need to be defined. One visualizes that, in an ongoing ASK installation, the technical library function will maintain a file of the access parameters for a wide variety of data bases of interest to its clientele. Thus, obtaining additional data from such a data base will be quite simple, taking only some knowledge of the external data base's query language. This latter could also be a function of the library.

Consider an illustration. Suppose one's firm subscribed to the Dow-Jones stock market service. One could maintain in one's own computer a record of investments in the market. The access word: "closing price" could be defined in the proper way for the Dow-Jones service, using the foreign access dialogue. Each day one could update the prices of one's stocks by the command:

```
>The current price of each of my stocks is its closing price.
```

Having added the definition:

```
>definition:percent change:
 100 * (current price - closing price)/current price
```

one could ask at any time:

```
>Display a bar chart of the percent change in the closing
 price of each stock in my high tech portfolio.
```

VI. SUMMARY

ASK has been designed so that it can constantly be re-customized to ongoing, changing user needs. We have discussed four principal ways this customizing can be done:

- adapting to a new or extended application domain
- incorporating specialized data structures and procedures
- adding user defined dialouges to obtain efficiency of communication
- attaching the wide range of available sources of data

By these means, ASK's natural language interface becomes only the primary language underlying a broad spectrum of efficient means for man - machine communication.

REFERENCES AND NOTES

[1] For a discussion of such phenomena from a psychological point of view see L. Vygotsky, Thought and Language, M.I.T. Press, Cambridge, Mass., 1962, ch. 7.

[2] B. H. Tompson and F. B. Thompson, "Introducing ASK: A Simple Knowledgeable System," in Proceedings, Conference on Applied Natural Language Processing, 1983, pp. 17-24.

[3] For details about diagnostic messages, see D. J. Trawick, "Robust Sentence Analysis and Habitability," Ph.D. thesis, California Institute of Technology, 1983.

[4] M. C. Chen, "REL Interface to Graphic Display," Ph.D. thesis, California Institute of Technology, 1979.

[5] This work is being done for the U.S. National Aeronautics and Space Administration in collaboration with the image Processing Laboratory, Jet Propulsion Laboratory.

[6] T. P. Ho, "The Dialogue Designing Dialogue," Ph.D. thesis, California Institute of Technology, 1984.

[7] A. Papachristidis, "A Unified Design for Distributed Data Base and Non-Homogeneous Data Base Access," Ph.D. thesis, California Institute of Technology, 1983.

[8] B. H. Thompson, F. B. Thompson, "ASK as Window to the World" in Proceedings, International Conference of Systems, Man and Cybernetics, Bombay and Delhi, 1984, pp. 1014-1018.