THE APPLICATION OF INTELLIGENT FRONT ENDS IN BUILDING DESIGN

Damian Mac Randal

Rutherford Appleton Laboratory, Chilton, Didcot, Oxon. OX11 0QX U.K.
e-mail: ARPA: @ucl-cs.arpa:damian@vd.rl.ac.uk JANET: damian@uk.ac.rl.vd

ABSTRACT

Over the years there have been many attempts to introduce sophisticated computer-based appraisal techniques to the building design profession. Most of these foundered due to the high level of rather specialist expertise required from the user before the packages can be effectively applied. The research described here is tackling this expertise problem directly by developing an *intelligent front-end* for a large and complex building appraisal package in the field of dynamic energy modelling. The *ife* will contain an encapsulation of both the energy modelling domain knowledge and the knowledge required to use the package in a design environment. The objective is to recast the perceived user interface in the terminology appropriate to the designer, but still provide the sort of power and sophistication that the package can deliver, To do this requires translating from (natural language?) descriptions of the problem in the users terminology to the formalised language of the problem solver. In order to perform this translation or mapping, it is necessary to handle at least two independent, but highly interlinked, conceptual views of the domain. This is currently being implemented using Sowa's formalism of conceptual graphs.

INTRODUCTION

Over the years there have been many attempts to introduce sophisticated computer-based appraisal techniques to the building design profession. There are two main incentives for providing a designer with such appraisal packages. Firstly, buildings are typically very complex mechanisms involving many highly technical fields - for example, movement, energy and cost appraisal, etc. In these fields, the manual methods or rules of thumb currently used cannot cope with the complexity of modern buildings and only computers can provide the accuracy and flexibility required. Secondly, particularly at the earlier stages of the design process, there is a need for rapid feedback as to the consequences of alternative design decisions. The present system of specialist consultants, while adequate for the final specification and detail design of the appropriate subsystems, fails to provide this immediate 'ad hoc' advice.

Most of the currently available packages are failing to be adopted as standard design tools, mainly because of shortcomings in the user interface. These shortcomings derive predominately from the conflict between the necessity for the packages to be powerful, comprehensive and rigorous to deal adequately with the problems, while, at the same

time, to be simple, straightforward and intuitive to facilitate user interaction. This situation is exacerbated by the divergence of the conceptual framework of the design oriented user and the technically oriented developer of the appraisal package. To complete this confusion, there is the subtly different terminology of the scientific, engineering and design professions.

The current, and laudable, trend towards 'user friendly' interfaces carries the risk of negating the power and flexibility of the actual package by restricting the interaction to the 'lowest common denominator' user level. Worse, the provision of more and more defaults and pre-packaged analysis will only erode the users intuitive 'feel' for what the package is actually doing. This leads to the situation where the package, though theoretically sound, either has been simplified to the extent that it can't deal with the complexity of the real world, or else it makes so many implicit assumptions that the problem actually solved bears little relation to the user's real world problem.

The difficulties mentioned above apply to all but the most trivial appraisal systems. The next section examines these in more detail before the current work on building intelligent front end is described. The appraisal system chosen as the target for this research is ESP, a dynamic energy simulation system developed by the ABACUS CAD unit at the University of Strathclyde. ESP is a large, state-of-the-art package which predicts the environmental performance of any proposed or existing building from its basic thermophysical properties. As such, it represents one extreme of the energy appraisal software, requiring high levels of expertise and a familiarity with building thermodynamics (Clark[2]). Thus it is an ideal candidate with which to explore the potential, and problems with an *ife*

LIMITATIONS OF EXISTING USER INTERFACES

If sophisticated appraisal systems, such as ESP, are to be accepted and used by designers, they will have to be (1) correct, and seen to be correct (2) appropriate for the design/problem solving context in which they are needed. (3) easy to use (4) operating at a technical level commensurate with the designers knowledge. The first two points are technical issues currently being addressed by the system developers. The final two points, however, fall into the broad category of user interfacing, and are the long term objectives of the research described here.

Categories of User

Before dealing with the details of the user interface, the various classes of users, and the facilities they require, have to be identified. Experience to date with the ESP model tends to classify users into two categories, the 'expert' and the 'novice'.

expert: generally computer and energy literate, spending a lot of time using the system either as a research tool into energy modelling or as a design tool investigating building behaviour.

novice: generally using the computer and/or package infrequently either for appraisal of his building's performance or to isolate the effect of a particular design decision. Note that the novice is usually an expert in his own 'design' field , the term 'novice' relating only to energy modelling.

The requirements of these two classes of user differ substantially. The expert, as usual, is concerned with speed and flexibility of input, direct control of the operation of each module of the system and access to all the resultant output in a structured but flexible manner. The novice, also as usual, wants a clear and coherent interface where the system provides guidance as to the current options and their implications, error

trapping and easy error recovery, and a concise summary of the results in an easily understood manner.

Appraisal aspects

As well as distinct classes of user, there are 3 distinct facets of the appraisal process, each raising its own set of problems for the user. These are 'data input', 'appraisal control' and 'output analysis'.

input: One of the basic difficulties facing current designers is the sheer quantity of data required to describe/manipulate current buildings. Not only is gathering this data a time consuming task, but frequently the data has not yet been specified, as is the case at the early design stages. Also, due to the complex interrelationships, ensuring the integrity of the data can demand very high levels of understanding of the underlying thermodynamics and the simulation techniques.

This creates two problems for the user. Firstly, if the data requested is not available, no help is provided to generate a sensible default. Secondly, without a good knowledge of the simulation mechanism, the importance, and hence the required accuracy, of an individual piece of data is very difficult to judge.

As well as the question of 'what', there is also the problem of 'how' to input such a large quantity of highly inter-related data. The various factors associated with data acquisition, together with the users' often idiosyncratic conceptualization of its inter-relationships, tend to conflict with the rigid question/answer style of input common to many of todays programs.

control: Generally, control of the appraisal does not require much sophisticated user interaction. At this stage the major difficulty faced by the user is the selection of the simulation parameters to produce a sufficient quality and quantity of output to allow a worthwhile appraisal of the building. For the novice, a lack of understanding of the implications of the selections being made can lead to confusion, or even erroneous deductions, due to the inadequacy of the output data.

output: It is here that the requirements of the novice and expert differ most. The expert will be trying to detect patterns in and relationships between different building parameters, in order to build up a picture of the dominant energy flowpaths. To do this, all the data generated by the simulation has to be available and capable of being displayed in juxtaposition with any other data. The novice, on the other hand, merely wishes for a concise summary of the building performance, preferably in terms of those variables most meaningful to himself and his client. Unfortunately, due to the primitive nature of the system's output, the novice may experience difficulty relating poor performance to the design decisions that caused the problem, or, indeed, even to the possible design modifications that could improve this performance.

SCOPE FOR IMPROVEMENT

There are basically two areas whose improvements would greatly enhance the usefulness of appraisal packages, such as ESP, as a design tool.

The first is the level of expertise required to drive the system. In order to select only the necessary and sufficient input data and to interpret the output data, requires a lot of expertise in the appraisal field. Some means of reducing this level of knowledge to that of the normal designer would have enormous impact on the uptake of building appraisal

by the design profession, and hence on the quality of the built environment. This could be achieved by an Intelligent Front End, to reduce both the quantity of data required from the user, and the number of errors that get through to the simulation stage. The *ife* will act as a consultant to the user, taking what data the user offers, conducting a dialogue to gather further information and identify the user's objectives, and then performing the simulation. The system will contain at least a basic model of the user, that is information about his level of competence, modus operandi, working terminology, etc. This, updated during the course of the session, will be used to restrict any querying of the user to the appropriate level. To supplement the data gained from the user, the system will use a knowledge base of energy modelling data and relationships supplied by an expert in the field. This can be tailored for the practice in which it will be used, but it is hoped that ultimately it can be personalised, perhaps automatically, for the individual user.

The second, related, potential area for improvement is in the manner of communication with the user. The current generation of small but powerful workstations makes it possible to apply the latest MMI techniques to provide users with the powerful interaction tools that current appraisal packages demand. One such technique is multi-windowing, where multiple outputs are displayed on the screen in a manner analogous to sheets of paper on a desk. This also allows the user to use the machine for other, possibly unrelated, tasks without interfering with the dialogue from the appraisal package, allowing files/databases to be consulted, subsidiary analysis to be carried out, side issues to be investigated, etc. Other techniques include the use of spreadsheet-like input incorporating dynamic (and pseudo-intelligent) defaults, graphical feedback of the current state of the building definition and animation of the energy flows around the building. Although the current research focusses on the intelligent front end, the opportunity will be taken to investigate the impact of these enhancements.

INTELLIGENT FRONT ENDS

In the above section, the idea of an *intelligent front end* was introduced. It is probably worth examining this idea more closely, and identifying the objectives and required functionality of an ideal system. *Ife*'s are an intricate synthesis of user modelling, contextual and domain knowledge and the usual mmi interfacing techniques. and the back-end interface.

Contextual knowledge

The context in which the system is being used and the underlying motivation of the user must be taken into account. Evidently, the type of help proffered by the *ife* will be qualitatively different if the user is a student in an School of Engineering, a technician in an architectural practice or a researcher in energy simulation. At the extremes are the distinction between the roles of *tutor*, trying to place specific concepts into the users mind, *consultant*, interrogating the user and telling him what to do, and *colleague*, intelligently checking and carrying out the users instructions. Although these roles are probably sufficiently distinct to require different, specialized *ife*s, since a users motivations are rarely singular each should be able to handle some aspects of the other two roles, for example, a consultant role *ife* should be able to discuss and negotiate with the user as a colleague would, or teach the user at least those concepts necessary to understand the consequences of his actions.

User modelling

To be of any assistance, the *ife* must interact at the users level, and in order to do this it requires a user model (Ross[4]). It's main function is to provide a model of the user's knowledge and problem solving strategies in order to extract all the implicit information and unstated assumptions from the users statements. Considering the misunderstandings

that can arise in man-man communication, the difficulty of this task will be appreciated. However, another, possibly more important, function of the user model is to track the users mental model of the system and react accordingly. This, of course, must adapt, or be adapted, as the user progresses from novice to expert. It must, however, not hinder the building in the users mind of a coherent model of the back-end by changing the interface characteristics too rapidly or arbitrarily. Instead, it should hasten this process firstly by hiding itself, the hardware and any other situational aspect not directly relevant to the users task, and secondly, by presenting the underlying concepts of the back-end in an easily assimilable form. Furthermore, where the users mental model is deficient or wrong, it should take appropriate steps to counteract or remedy his misconceptions.

Back-end interfacing

The distinguishing feature of a *front-end* is the need to interface to a back-end. This requires mapping from the users model of the back-end to the back-ends model of the user. It therefore must have a detailed understanding of the back-ends function and operation, as well as knowing almost as much about the methods of solving the users problems as the back-end. Abstracting this knowledge can be as difficult as deriving an expert system to replace the back-end.

Conceptual mapping

An *ife* can ultimately be considered as a machine translation system, converting from, ideally, natural language descriptions of the problem in the users terminology to the formalised language of the problem solver (Bundy[1]). Although this could be achieved by mapping user input onto an intermediate knowledge representation which is in turn mapped to the back-end commands, this loses the explicit representation of both the user's and the back-end's conceptual context. Therefore, it is preferable to have two independent conceptual structures, one corresponding to the user's viewpoint, and with which the user interacts, and a second one corresponding to the back-end's viewpoint, in which the actual problem solving is carried out. The mapping then occurs between two well defined conceptual structures. The major advantage of this scheme is that the user is interacting with a system that 'thinks' along the same lines as he does.

SYSTEM OVERVIEW

The previous section indicated in general terms the attributes of an *ife*. In more specific terms, and in the context of encouraging uptake of energy modelling by the building design profession, the functions required by the interface system consists of the following:

- User modelling: to track the users mental design process and ensure the system responds appropriately.
- User dialogue handling: to converse with the user in a suitable (eventually natural) language, making maximum use of current graphical MMI techniques.
- Plan recognition & Planning: to identify the users objectives, ie plans, and decide on the most appropriate simulation scheme.
- Building modelling: to organize the data describing the building under consideration.
- Back-end handling: to drive the package, in this case ESP, and feed it the necessary data.
- Knowledge base: to contain the necessary strategy knowledge about designing and about using ESP as well as the energy modelling domain knowledge.

The system currently being developed, and which is described below, see Figure 1., is intended to address these various functions in a modular fashion. The main objective is to provide a clearer understanding of the design and implementation of a complete *ife*, so emphasis is placed on the interaction and requirements of the various tasks, rather than on tackling any particular aspects in depth. It will be appreciated that the implementation of even one of these tasks to a state-of-the-art level would be a major undertaking. Accordingly, the research is initially concentrating on the creation of a flexible and powerful overall structure and the implementation of an adequate knowledge representation, with the supporting modules being fleshed out later.

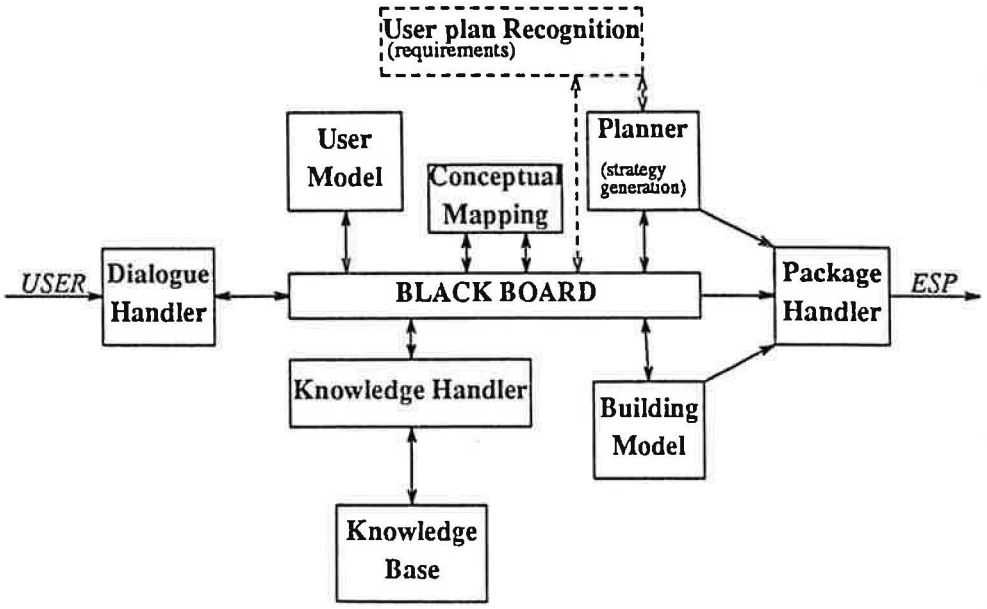

Figure 1. Outline of System Architecture

SYSTEM DESCRIPTION

This functional division leads naturally to the notion of implementing the system as a suite of independent modules communicating via a central *working memory*, ie. a sort of 'blackboard' model (Nii[3]). The major advantage of this approach is that, after specification of the blackboard, the development of the other modules can proceed independently. It is hoped that the highly modular nature of the system, together with the isolation of design and domain knowledge into the knowledge base, will open up the possibility of using the *ife* with other appraisal packages and design methodologies. Before dealing with the heart of the system, the blackboard and the knowledge base, the other modules will be described. It should be pointed out that, as the project is still at an early stage, these modules have not been designed in detail, so the following descriptions will necessarily be rather sketchy.

Dialogue Handler:
Initially this will deal with a restricted command language, designed to give the user the opportunity to volunteer information, abort or redirect the system's line of inquiry and give "I don't know" replies. Although classified as dialogue, this module will handle non-textual input as well to allow the latest MMI techniques to be employed. Although it

is not proposed to do so in the current project, this could eventually be extended to handle a subset of English by borrowing techniques from research into natural language.

User Model:

The user model is perhaps the most interesting of the proposed modules. Idealy it would be the computers internal representation of the users's mental processes, ie. the computers understanding of what the user is thinking, based on knowledge of the user and on his past and present actions/statements. Unfortunately, modelling users conceptualizations is very difficult, mainly because the mental processes concerned are not well understood, but also because identifying internal concepts from external behaviour is error-prone even for humans - for example, in deciding whether someone views a building as an envelope subdivided into rooms or an aggregation of rooms. A further problem arises when the user switches viewpoints, or operates with two different, but complementary conceptualizations simultaneously. Given these difficulties with 'real' user models, it was decided to settle for the more simple *stereotype* version, in which all that is necessary is to decide which stereotype, or stereotypes, most closely approximate the user. It is proposed to restrict this module initially to classifying the user in one of a small number of categories, and controlling the user dialogue accordingly. For example, an 'architect' at an 'early design' stage would not be asked to provide information about the plant control strategy, whereas an 'energy modeller' would be expected to provide information about the desired timestep control mechanism. The decision will be based on a user database, initialised by querying the user, and modified should the dialogue appear to be breaking down, eg. the user is giving either too many 'don't know' answers or a lot of unsolicited information at an inappropriate level. Later in the project this will be expanded to allow for varying levels of user competence in different aspects of the appraisal process.

Plan Recognition:

Since a prerequisite for this module is a comprehensive user model, no attempt will be made to address this issue in this project. The user will be required to state his objectives in the predefined terms that the planner requires, eg overheating analysis, solar gain analysis, etc. However, when the user model and dialogue handler are extended sufficiently, then some form of plan recognition will become essential, and therefore the system, and in particular the planner, will be constructed such that an appropriate module can be added.

Planning:

Given that the user and ultimately the Plan Recognition module, will identify a goal to be achieved by the back-end, and the Knowledge Base will contain the knowledge of what the back_end can actually do, the Planner just has to select the correct strategy, or at most select a series of such strategies. Planning will initially be restricted to selection of one out of a number of predefined processes . For example, if the user indicates a desire to investigate the possibility of overheating, the planner will initiate the relevant simulations and display the necessary output results. Later in the project the amount of predefinition will be reduced, until the planner is actually using simulation primitives and modelling knowledge from the Knowledge Base.

Building Model:

This module will depend rather critically on the design of the Knowledge Base module, since it will have to build up the data the domain knowledge specifies as necessary for achieving the users objectives. At present, it is envisaged as a straightforward database, containing the building's geometry, construction, occupancy and system data as input by the user or as supplied, in the form of defaults, by the knowledge base. This will correspond closely to the current input formats as required by the appraisal package,

ESP.

Back-end Handling:
The emphasis here is to minimize the changes in the back-end, ESP. Initially this will simply extract the data from the building model and planner, creating the data and control files to drive ESP in a 'batch' mode. This module will be the only one specifically tailored to ESP, so that the lessons learned from constructing the system would be more easily generalized for wider engineering design applications.

Knowledge Base:
There are two types of knowledge to be stored and manipulated by the *ife*, corresponding approximately to human *working memory* and *long-term memory*. It is the second sort that is contained in the Knowledge Base. Actually, there are several separate knowledge bases, containing the built-in knowledge about the application (design); the domain (energy modelling); the tools (ESP); and the user (stereotypes), all stored explicitly and as independently as possible to facilitate modification or replacement. Obviously they cannot be completely independent since each on will want to refer to entities that are defined in another.

These knowledge bases will be implemented using a knowledge representation language based on the conceptual graphs formalism put forward by Sowa. The basic structure is a semantic network of *concepts* and *conceptual relations*. A concept may be an undefined primitive, taken from the domain, or it may be defined in terms of other concepts and conceptual relations structured into a mini-network called a *conceptual graph*. Each conceptual graph is thus a representation of a particular piece of knowledge. Both abstract concepts and real world objects can be handled by the notation, and, since a conceptual graph may refer to other conceptual graphs, representing meta-knowledge is possible. By adding/deleting concepts and conceptual relations to/from a conceptual graph is a form of specialization/generalization of the defined concept. As well as defining concepts in terms of other concepts, the formalism provides a separate *type hierarchy* in which concepts can be identified as being a super-type or sub-type of another. This type hierarchy, or more accurately a type lattice, together with generalization and specialization of conceptual graphs, gives the basis for a knowledge manipulation or inferencing mechanism. Finally, the notation provides the ability to tag particular conceptual graphs as having a particular type. The types supported include definitions, ie. the definition of a concept in terms of other concepts, schema, ie. typical contexts for use of the concept and prototypes, ie. typical instantiation of the concept, as well as straightforward propositions, ie. statements about the concept. This knowledge representation scheme is more fully explained by Sowa (Sowa[5,6]).

Blackboard:
As well as the *long-term memory* type of knowledge, there is a lot of temporary information about the current session, that is, about the user, his objectives, the building under discussion, etc. This is not built-in to the system and is gradually accreted as the session progresses, principally from interaction with the user but also from the other modules. This transitory knowledge is organised as a blackboard, a sort of *working memory*, which other modules peruse for information they can use, and post their results back on it. The scheme facilitates multiple use of information, eg one user statement posted by the dialogue handler may have implications for the user model, knowledge base, planner and building model. The blackboard is thus the communications center of the system, upon which everything else hangs. In this system, however, it has to be more than a simple organised repository for information and should be considered as a sort of intelligent blackboard, since it has knowledge manipulation facilities closely associated with it. Again, the knowledge representation language being used is based on conceptual

graphs. The design/implementation of the blackboard constitutes the major part of the project.

MULTIPLE VIEWPOINTS

It was pointed out earlier that an *ife* would have to support at least two independent conceptual views of the problem, that of the user and that of the back-end. Thus the blackboard is divided into two major partitions, each partition containing a conceptual graph based knowledge structure. The form of these conceptual structures is dictated by the corresponding knowledge base, and the content is obtained from the other modules. For example, the User Dialog Handler will place a piece of information in a scratch area of the blackboard and the Knowledge Base Handler will, on the basis of the relevant Knowledge Bases as indicated by the User Model, integrate this information into the conceptual structure. Should the information be inappropriate, it will be passed back, via another scratch area, to the User Model and/or the User Dialog Handler for referral to the user. In this fashion, a conceptual structure corresponding to the user's viewpoint is built up.

In parallel with this, the Blackboard Handler will map the information in the user's conceptual structure into a second conceptual structure, whose form reflects the conceptual viewpoint of the back-end. This, of course, will be a much simpler structure, as the back-end is more limited in scope and more formally defined that most users. Thus, the user's input is ultimately mapped into a form suitable for the back-end. This communication works in both directions, so that any requirements for extra information discovered by the Planner or Back-end Handler will be mapped into the users conceptual structure before being passed to the User Model and/or the User Dialog Handler for referral to the user. If ambiguities, or difficulties, arise during this mapping, and they cannot be resolved by using the appropriate Knowledge Bases, the matter will again be passed back to the user as a sort of meta-level question.

In keeping with the philosophy of walking before you run, only a simplified version of the above will initially implemented. The main victim of this will be the adaptiveness of the conceptual structures 'on-the-fly', and meta-level problems will be referred straight to the user for resolution, perhaps with a ''consult a guru'' advisory message. The main problems identified so far, apart from the obvious ones of scheduling the modules, defining the protocols and capturing the requisite knowledge, is the creation of ''correct'' links between the two conceptual structures, so that the mapping does not affect the semantics of the interaction.

CONCLUSIONS

It has been found that, for all but the most knowledgeable and experienced designers, assistance with current appraisal packages is essential. One of the advantages of an *ife* in the field of energy modelling is that, for typical designers, any help at all in dealing with the varying scientific, engineering and design vocabularies would be very valuable. It would substantially increase the uptake of sophisticated appraisal packages. This will help improve the quality of the built environment by allowing the designer to eliminate poor performance features at the design stage.

It should be clear from the system description above that in order to produce a working system, a number of research-level problems have to be overcome. However, given the scale of these problems, it will be necessary to severely restrict the scope of most of the associated modules in order to achieve a working prototype within the timescale of the project. It is hoped that this prototype system would be sufficiently

complete and robust to allow evaluation its potential impact on the design profession, from the user interface point of view. As well as giving useful feedback about the system design, it would provide a testbed for further research into user modelling and design methodology. The experience of *ife* design gained should allow similar interfaces to other existing packages to be constructed more quickly and in the long run opens up the possibility of truly integrated multi-criteria design appraisal. This would allow the designer to more easily handle the various trade-offs necessary in modern buildings.

References

1. Bundy, A (1984) ''An Architecture for Intelligent Front Ends'' in *Alvey IKBS Research Theme Workshop: Intelligent Front Ends 2, University of Sussex, UK, 10-11 July 1984*, ed. A Bundy, IEE, Hitchin, Herts. UK.
2. Clarke, J (1986) *Energy Simulation in Building Design.* Adam Hilger, Bristol & Boston, 1985
3. Nii, P (1986) ''Blackboard Systems: Part I''. *The AI Magizine*, 7, 2:38-53, Summer 1986.
4. Ross, P (1984) ''The Virtues and Problems of User Modelling'' in *Colloquium on Intelligent Knowledge Based Systems: The Path to User friendly Computers, Institute of Electrical Engineers, London, Dec 1984*, IEE, Hitchin, Herts. UK.
5. Sowa, J (1984) *Conceptual Structures.* Addison-Wesley, Mass. USA, 1984.
6. Sowa, J (1986) ''Implementing a semantic interpreter using conceptual graphs''. *IBM J Res Develop*, 30 1:57-69, Jan 1986