CAAD Futures '93
U. Flemming and S. Van Wyk (Editors)

KNODES: Knowledge-Based Design Decision Support

James H. Rutherford

ABACUS Department of
Architecture University of
Strathclyde Glasgow, G4
ONG Scotland

This paper describes the work in progress to develop a knowledge-based design support (KNODES) environment.[1] The KNODES environment is intended to improve a designer's decision making potential, and to assist designers in a broad range of disciplines by making design knowledge and information more accessible. The framework exists as a multi-layered knowledge-based system reflecting both generic and domain specific aspects of the design activity and is designed to: enable the rapid dissemination of new design information, thus responding to shifts in design standards; take advantage of existing empirical design tools in unstructured areas of the design process; accommodate changes in user expectations and technological innovations at both the system and domain levels thus ensuring long term continuity and support; offer multiple view points of the same design data by means of interchangeable interpreters. A prototype building design framework configured using the KNODES development tools is used to illustrate some of the salient features of the framework Finally the paper will conclude with some insights into how such a design framework may be used as a knowledge acquisition tool in order to derive and formalise models of the design process.

Keywords: computer-aided design, design frameworks, multi-level knowledgebased systems, blackboard architecture.

1 Introduction

Technological advances in materials and manufacturing methods have necessitated the specification of more rigorous regulatory constraints and guidelines to which a designer must adhere. Over the past decade, new and sophisticated design tools, in the form of CAD systems, have emerged to assist the designer in solving a variety of domain specific problems, enabling informed decisions to be made. Advances in workstation technology and networking provide an almost limitless resource for performing computationally intensive tasks (Vidovic). While such technology exists, providing support for distributed problem solving within a multi-disciplinary design organization, the problem now faced

[1]This research is based on work undertaken in the domain of Intelligent Design Assistance (Rutherford, 1990) and is funded by the Science and Engineering Research Council (Rutherford and Mayer, 1991).

by members of a design team is one of how to manage and integrate an intimidating array of seemingly disparate design tools with manual techniques and procedures.

Many attempts have been made to combat the proliferation of esoteric tools and inconsistent user-interfaces. In order to alleviate the corresponding cognitive burden on the user, design frameworks, consisting of advanced user-interfaces, CAD tool and data management systems, are beginning to emerge, particularly in domains such as the VLSI industry, where design optimization is of paramount importance. This development, however, has concentrated largely on the mechanistic aspects of CAD tool management, with little regard for the user, thus adding to the bewildering array of tools and environments available. In addition to providing a means of integrating a large base of design tools and exploiting the potential of distributed problem solving, KNODES, a knowledge-based design support environment, in recognition of multi-disciplinary design activity, attempts to assist users by offering consistent views of design knowledge resources, data, and procedures. To ensure a high degree of generality in the KNODES framework a top down approach has been taken; ensuring that both generic and domain specific aspects of the design activity are accommodated.

In the context of decision support, design assistance may be defined as; the utilization of specialist, expert consultants to supplement a designer's own experience and expertise. Each participant, whether human or machine, in a multi-disciplinary organization, by definition, has a specific focus. Any design framework must, therefore, accommodate different views of the same design data space for each of the different disciplines involved in the design process. In this context, regardless of the domain, the key to successful multi-party interaction and computer supported co-operative work is the effective communication of design information.

Given that communication, defined as the exchange of meanings between individuals through a common syntax of symbolism (Encyclopaedia), is perhaps the most general aspect to multi-disciplinary design activity, a simple model of communication between two cognitive systems, agents A and B (Figure 1), is used as the basis for the design of the KNODES environment. Notation from mental models research (Gentner, 1983) is used to simplify the description.

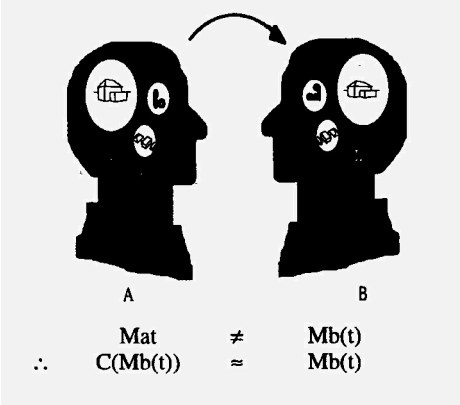

Figure 1. Communication between two cognitive systems.

In order to communicate information between the two systems, both participants in the dialogue must already have an understanding of the concept being conveyed. However, each agent, based on individual experience of external events, will have preconceived mental models (Ma(t) and Mb(t)) (Gentner, 1983) of the target system 't'. As no two individuals experiences are likely to be identical, their individual mental models will differ. Therefore, in order to convey meaning effectively, it becomes necessary to decompose the description of the target system into elemental components common to both participants. This takes the form of a semantic preserving translation or conceptualization, "C" of a mental model. Therefore, in this simple model, effective communication is achieved if the conceptualization, C(Ma(t)), tends towards the mental model Mb(t). In any dialogue the initial conceptualization 'C' is formulated on the basis of stereotypes and refined according to certain conversational cues (feedback) from the participants involved.

1.1 Communicating with Computers

Dialogues with computer-based tools may be considered in much the same manner as human communication. The major difference between human and computer interaction is that there is an opportunity to explicitly make available the extent of the system's conceptual vocabulary. This enables the user to formulate and structure a conceptualization appropriate to the domain of discourse. This is often essential as very few stereotypes are available for the user to initiate a dialogue. Consistency between applications is, therefore, of paramount importance. In instances where the extent of the system's vocabulary is limited, inappropriate or not immediately apparent, the range of knowledge a user must utilise in order to formulate a conceptualization widens. This places an increasing cognitive burden upon the user and ultimately detracts from the design problem at hand.

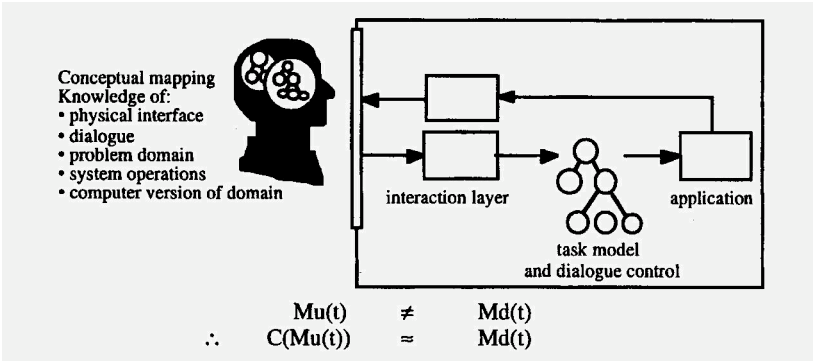

Figure 2. The end-user's task, where u = the user, and d = the developer.

The situation is worsened when several CAD tools, each with specific data requirements and esoteric user-interfaces, are deployed. User-interface management systems aim to assist the developer in building applications that are visually and behaviorally consistent. Regardless of the software development tools used, however, a software developer will have a preconceived mental model of the problem domain, which becomes encoded within the application (Figure 2). The software is therefore inherently biased towards a particular class of user or problem solving activity. In such instances the resulting system may either be too complex for all users to employ or too inflexible by making general assumptions about the user's level of expertise. Such an approach may also force a user to

think and structure their work in a manner appropriate to the software tool being utilised; resulting in what is referred to as induced system restrictiveness (Chu, 1990).

While many design tools are effective in representing expert knowledge in broad domains, as Weis (1992) indicates, there are many problems for which additional knowledge, not represented by these systems, is required. The supplementary knowledge required to perform the conceptual mapping, C(Mu(t)), between the user's task model and the computer version of the domain, Md(t), identified by Morton (1979), consists of knowledge of the dialogue and interface, together with an awareness of system operations and the computer version of the domain (Wilson, 1987). (See Figure 2.)

In order to alleviate the cognitive burden on the designer, and improve design decision making, a computer-based design framework must utilize knowledge of: the dialogue, the user (in the form of task models and belief structures), the problem domain (in the form of design methods and procedures), and system operations. Such a system must also accommodate a range of users by taking account of differing levels of experience and individual task models.

The KNODES environment achieves this by representing the additional knowledge the user must bring to solve a problem within the computing environment itself, as illustrated in Figure 3. At the front-end, the environment allows the user to deal in domain concepts rather than the esoteric data requirements of individual application programs. This conceptual model is *then translated* by a domain handler into a product description, which is made accessible to a range of CAD tools by means of data translators. The end-user is therefore presented with a dialogue that closely matches the problem domain and given access to a range of CAD packages in a consistent manner, while being isolated from the operational aspects and data requirements of the computer version of the domain.

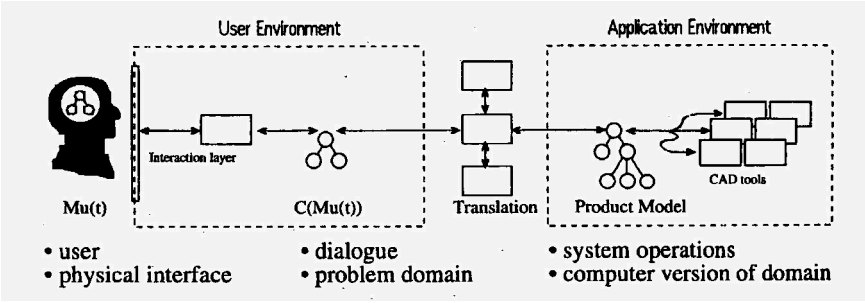

Figure 3. Application architecture necessary to support the user.

This knowledge has been organized, in an attempt to improve system maintainability, into levels of abstraction (Figure 4), each layer intended to manage subordinate layers of knowledge.

The resulting framework exists as a multi-level knowledge-based system (after Hart, 1982) composed of surface level models, i.e., those derived from human laws and consisting of naturalistic human knowledge, centred around naive description of physical systems (Hayes, 1979). These are then used to orchestrate the acquisition of information from deep models of reasoning encapsulated within knowledge bases, design tools, and other supporting systems such as expert systems and product databases.

The benefits of such an approach, as Lansdown (1986) points out, are threefold. The system provides:

1. More force to explanations arising from the knowledge-based system;
2. A sharper distinction between objective and subjective knowledge; between guesswork and more fundamental reasoning;
3. More knowledge and power.

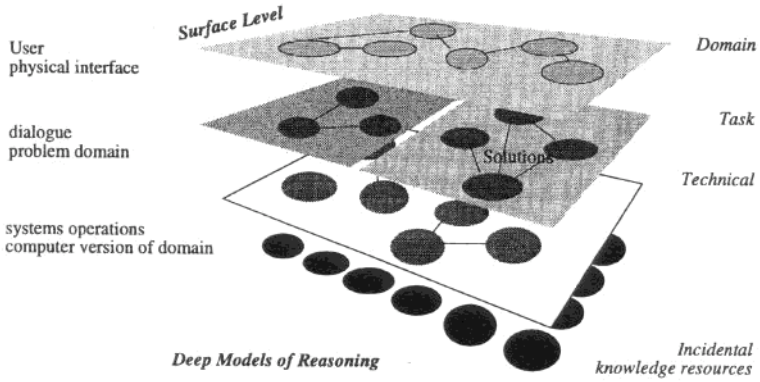

Figure 4. Multi-level knowledge-based system consisting of surface level knowledge derived from human laws and used to orchestrate deep models of reasoning.

2 Implementation

In order to manage a knowledge-based system of this form, a number of tools and knowledge handlers have been developed. These are categorized in terms of their generality and are described as being either:

- Incidental knowledge resources - comprising expert systems, appraisal packages and databases, that are activated by surface level knowledge sources to resolve domain specific tasks; or
- Resident knowledge resources - required to be active throughout a problem solving session. These elements, supporting user-interaction and decision making activities, consist of: a series of dialogue handlers, a user model, a domain specialist, a resource manager, and a design database management system.

Central to the operation of the KNODES environment and the management of information between these agents is a blackboard.

2.1 Blackboard

The KNODES blackboard, (Figure 5), is a skeletal blackboard system (Englemore) developed to support a large number of both resident and incidental knowledge sources. It consists of a blackboard panel, which is dynamically partitioned into areas. Each area represents one of the levels of the problem domain, in this case design assistance (Figure 5), and is managed by one or more knowledge sources. Information is propagated though the blackboard by means of a system of "post and subscribe." Messages are posted, by

means of the blackboard interface, into named areas of the blackboard and passed to knowledge handlers expressing an interest in that region.

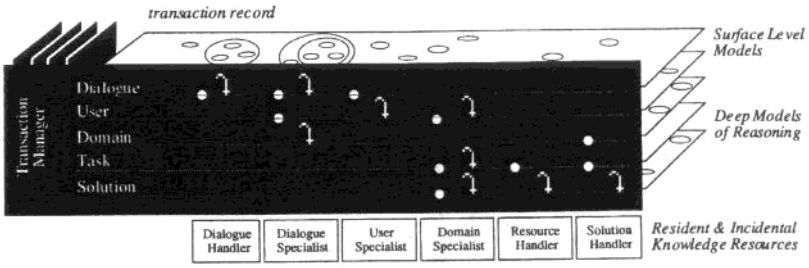

Figure 5. The KNODES blackboard and resident knowledge resources necessary to support the user and design task. Knowledge levels are represented as horizontal lines and are managed by an array of knowledge handlers shown below the blackboard. A knowledge handler's interest in a particular level is represented by a spot, its contribution by an arc (after Hearsay (Reddy et al., *1973)).*

2.2 Blackboard Interface

The KNODES Blackboard Interface consists of a set of C functions, which may be bound into any development environment, such as Lisp or Prolog, that accommodates foreign functions. This enables knowledge resources to be developed in a language and methodology appropriate to the task. The interface (Figure 6), contains an event notifier and responder, together providing a totally transparent communications link to the blackboard from anywhere on a network, thus enabling distributed applications to be configured with relative ease.

The example below, written in C, illustrates the ease with which a dialogue with the blackboard may be established

```
main()
{
        resource_name("resource_handler");
        establish_dialogue("Blackboard," 'host_server");
        resource_interests("Tasks");
        post("User Dialogue," feedback_area,"
                "KNODES resource Handler Version 2.3 Active");
        listen();
        close_dialogue();
 };
```

In the above example, the program first sets the resource name and then establishes a dialogue with the named blackboard on a specified host. Once connected the resource declares its interest in an area of the blackboard, in this case the *Task* level. If this area does not exist already, it is created. A message is then posted into the user dialogue area of the blackboard. This will be interpreted by knowledge resources which monitor that area, and are ultimately passed on to the user. Once the dialogue and application have been initialized, the interface listens for events and calls the user definable "responder" function,

passing the type of event, the area in which it occurred, and the message itself. It is then the responsibility of the application to interpret the message and respond in an appropriate manner.

A major feature of the blackboard and blackboard interface is the ability to establish and close a dialogue at any time. This facilitates suspend time editing of resources enabling rapid contextual proto-typing, a facility found in some user-interface management systems (Prime, 1988).

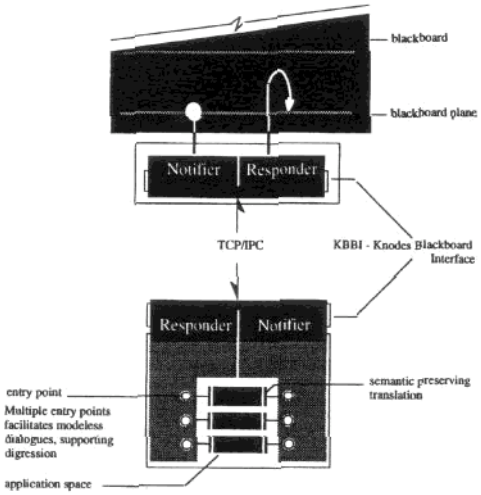

Figure 6. KNODES environment blackboard interface showing the relationship between a knowledge resource, the interface communication functions, and the blackboard.

2.3 Inter-client Communication

It is important to note that blackboard clients do not communicate directly with each other, but indirectly by means of messages posted onto appropriate areas of the blackboard. Any other client expressing an interested in that area of the blackboard is notified of the event and therefore must respond opportunistically to any message posted. Applications must, therefore, be structured around condition-action rules ensuring that concepts may be addressed out of context. The added benefit of such an approach is that the resulting system will have an open dialogue and, therefore, enable an end-user to digress; a necessary activity for divergent or creative problem solving.

2.4 Message Structure

The blackboard message structure contains several types of information including; the originator of the message, where and when it was posted and the message identifier, together with the data itself. The message data may contain any series of text tokens including binary data. It is therefore the onus of the responding resource to anticipate the type of data being broadcast. In the prototype building design framework, described in section 3, message data consists of EXPRESS schema (Mead, 1990 and STEP, 1988).

2.5 *Consistency Enforcement*

The opportunistic nature of the KNODES blackboard system facilitates digression, a mechanism enabling designers to gather information in an unstructured manner, allowing prototypes to be generated and alternatives explored out of context of the final solution.

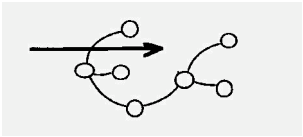

Figure 7. Consistency enforcement.

While opportunism supports divergent browsing, it introduces a major difficulty encountered in any large-scale knowledge-based system, namely; that of maintaining the integrity and consistency of the conceptual schema. This is resolved in the KNODES environment in two ways. First, by limiting the entry points into a knowledge base to the head of a transition network at the dialogue level, and second, by propagating changes made to a concept throughout the conceptual schema by means of a consistency enforcement routine.

Each time a concept is modified the knowledge handler attempts to enforce consistency throughout the conceptual model by exhaustively forward chaining through a dependency network (Figure 7). Dependencies are represented as lists. Although complex networks may be managed using this approach it is suggested that the scope of any network be limited to avoid unforeseen recursion. In the example:

```
dependent(floor_level,[room_functions]).
dependent(building_function,[room_function,
        spatial_primitives,
        constructional_methods]).
dependent(geometry,[performance_analysis]).
```

spatial functions and primitives, together with constructional methods, are declared as being dependent upon the building function, while performance analyses are dependent upon, amongst other things, the building geometry. Therefore, if the default building function is changed, from domestic to say educational, the range of spatial primitives, functions, and constructional methods on offer are adjusted accordingly. This simple representation enables dependencies to be easily defined and asserted or retracted dynamically depending upon the domain or context. The domain handler can, for instance, on behalf of the user, define a dependency between the building geometry and a building performance analysis. This is done to ensure that all views of the design data space are consistent. If, for example, changes are made to the building geometry, any appraisal package, performing an analysis of the building, is interrupted and updated with the current product description.

2.6 *Resident Knowledge Resources*

The resident knowledge resources, necessary to implement the multi-level knowledge-based system, comprise: a series of dialogue handlers, a user model, a domain specialist, a resource manager, and a solutions handler. A brief description of each of these resources is given below. The transient relationships between these modules is illustrated on the main blackboard panel (Figure 5).

2.7 Dialogue Handler

Dialogue handlers are responsible for managing human-computer interaction through the "User Dialogue' area of the blackboard. Each dialogue handler is tailored toward a specific form of interaction. Currently, two dialogue handlers are used:

- Forms (Rutherford, 1987 and 1990), developed in C++, is a dynamic X-based graphical user-interface utilizing a lexi-visual form filling metaphor. This interface tool kit verbalizes human activity by means, of high-level natural language utterances. Forms is specifically intended to interpret conceptual schema facilitating the redescription of conceptual models to users of differing levels of expertise. This is achieved by a two-level adaptation architecture (Totterdell, 1990).

- A graphical editor, configured around AutoCAD (Autodesk), is tailored to meet the data entry requirements of the current domain; in this case building design. Functional and spatial primitives have been added in an attempt to match the designers task model and preserve as much of the original design intention as possible.

An important and unique feature of the CAD tool integration methods, within the KNODES environment, is the ability to tie an external resource (the graphical editor for instance) to the main dialogue, ensuring that the user is able to relate the tool to a specific context. (See figures 10-14).

2.8 User Model

The user model, written in Prolog, by monitoring interaction cues, generated by the dialogue handlers, and posted into the User Dialogue area of the blackboard, categorizes the user into one of several user levels, adjusting the dialogue and users task models accordingly. The current user model promotes and demotes the user to one of four levels: expert, intermediate, novice and beginner (Chin, 1989). This categorization is used to determine the style of dialogue and the level and type of assistance received by the user in the form of design tools, on-line help, and default values.

2.9 Domain Specialist

The domain specialist contains expert knowledge in a particular domain and guides the designer through a series of processes using any one of a number of design methods. This resource is also responsible for assigning abstract task specifications, on behalf of the designer, to the *Tasks* level of the blackboard, which is monitored by a resource manager.

2.10 Resource Manager

The resource manager is used to solve domain specific problems presented as abstract task descriptions. Using a process of subgoaling a solution plan is configured from a list of process agents by matching the input requirements of one process to the output of another. This effectively decomposes a given task into sub-tasks. The overall task is resolved by backward chaining through the solution plan, acquiring information at each stage by deploying the appropriate application program.

Figure 8 illustrates a solution plan for the task "calculate energy consumption of building.' The agent providing the desired output is used as the basis for the solution plan. The resource manager, taking each input of the agent in turn, then selects secondary agents that satisfy these input requirements. This is repeated until all information can be obtained. Values at unconnected process inputs are obtained by placing a request onto the blackboard. The information is then obtained either from a product database, or, if not found, elicited directly from the user, in a manner appropriate to that user. Each agent then

spawns a drone process on a suitable processing node, which in turn activates the design tool. In the above example, agents, representing the product database, a geometry analyzer, materials, and climate databases, are used to provide the inputs to a simulation model.

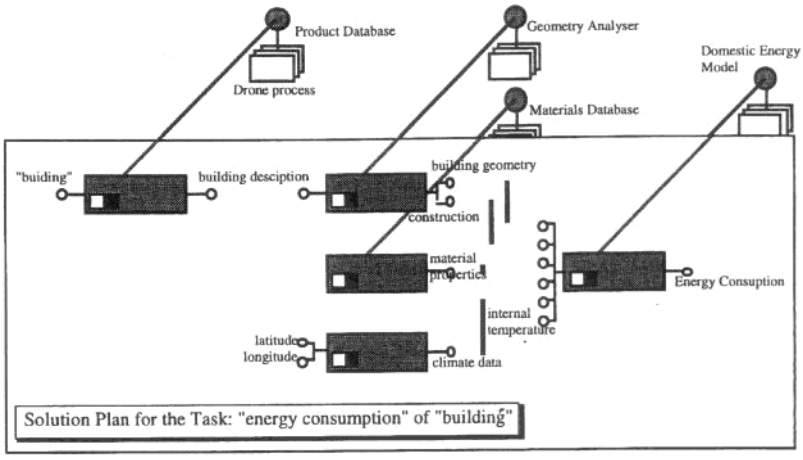

Figure 8. A solution plan for the task 'calculate energy consumption of building.'

Agents are described in a high-level manner in terms of a process category, identifying the agent's domain of operation, and a series of properties; including input requirements and output descriptions (after Kaemmerer, 1986). This isolates the resource manager from the operational aspects of the application it is utilizing. The major benefit of such an approach is that while the functional description of an agent remains unchanged for the given domain, the application, used to process the information, may be substituted without disturbing domain knowledge. This accommodates shifts in design standards that may be embedded within an application program, and enables the functionality of the framework to be extended by introducing new and enabling technologies.

Several solution plans may be formulated for a given task. The optimum solution plan, used as the final executable plan, may be selected on the basis of either:

- the least number of unknown inputs, and/or
- the relationship between the design stage and the granularity of the goal output; i.e., a simple wireline perspective would be generated at the early sketch design stage in preference to a fully rendered ray-traced image.

2.11 Solutions Handler

The solutions handler is an interface to a design database management system, which records all design information posted through the blackboard. The current implementation is an object oriented database developed using Ontos (Ontological, 1990).

3 A Prototype Building Design Framework

Using the KNODES development tools, a prototype building design framework has been configured. It is intended to teach a student the complex relationships between spatial configuration and building performance.

In order to achieve this, a three-stage design method (Figure 9), is used to orchestrate the dialogue with the student and produce, on behalf of the student, a series of building appraisals, collectively referred to as a design fingerprint, which the student uses to evaluate the design solution.

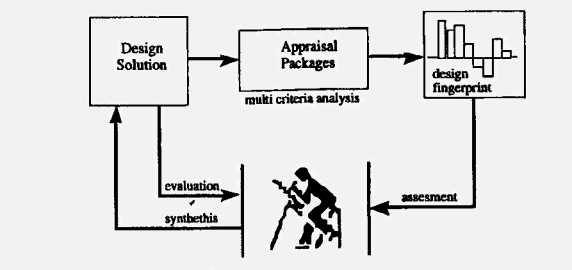

Figure 9. Three phase design method (Markus et al., *1972).*

By making changes to the design solution the student is able to observe the changes in the buildings performance characteristics. By making such information immediately accessible, the student is quickly able to relate desired performance characteristics to certain spatial and constructional configurations, and rapidly converge on a design solution. The current prototype building design framework deploys the following incidental knowledge resource in order to produce design fingerprints:

- a natural lighting package, which calculates the distribution of natural light within a space;
- an energy analysis module based on the Building Research Establishments Domestic Energy Model, BREDEM (Anderson et a!., 1985), which predicts the annual energy consumption for the proposal;
- an energy design tool for non-domestic buildings, based on the lighting and thermal value of glazing (Baker);
- a spatial analyzer is being developed for plan optimization;
- visualization facilities which allow the evolving design to be viewed in varying degrees of detail, from bound box, to fully coloured, textured, and shaded images;
- a structural analysis package, soon to be added to the framework, will optimize both the structural configuration and the plan formation, offering alternative solutions to the designer; and
- a costing package which calculates capital costs, running costs, and life cycle costs.

Data translations between each of the packages is managed by a series of filters acting as pre-processors; the most significant being a geometry analyzer which performs

Boolean operation on groups of geometric bodies and provides appropriate quantities necessary for other analysis systems.

The following series of illustrations show a typical dialogue with the user.

3.1 Building Description

In the example shown in Figure 10, the user enters a geometrical description of the building using the spatial editor. The tool being used is AutoCAD, but rather than allowing the user to interact directly with AutoCAD and describe the building in terms of lines and polygons, a framework has been added on top of the package in an attempt to preserve as much of the original design intention as possible.

It is also worth noting that AutoCAD is not floating in a window of its own, but is tied into the user dialogue within the context of building geometry. The user is, therefore, immediately able to relate the tool to the task at hand.

In the example (Figure 10), a space is created by selecting an appropriate spatial primitive and function from two menus of the geometry framework. Using the graphical editor, the user then sketches in the space. The editor posts onto the User Dialogue area of the blackboard a record of the activity, and objects are created in the design database. Similarly, any changes made to the geometry at the user-dialogue end are immediately reflected in the design database. Each object is described in complete detail using a series of defaults; such as the percentage of glazing for each function type and a general description of the building's fabric.

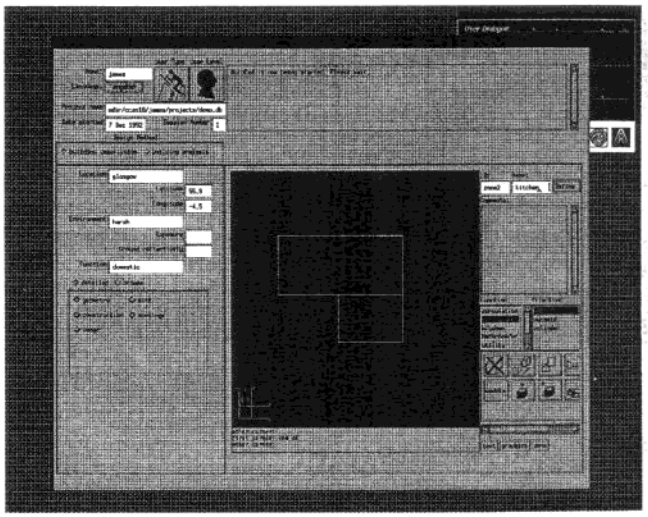

Figure 10. Building design framework-geometrical description.

Default values, assigned by the domain handler, may be overwritten by the designer using appropriate sub-frameworks such as those illustrated in Figures 11 and 12 dealing with multi-layered construction and openings, respectively.

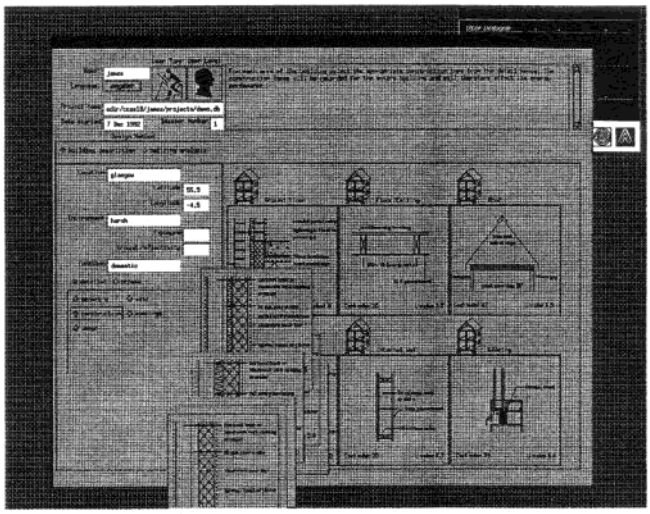

Figure 11. Building design framework-Materials and Construction template.

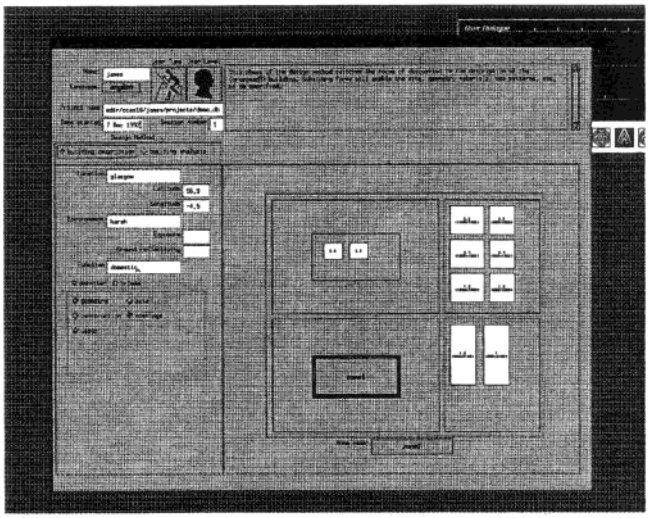

Figure 12. Building design framework Openings editor.

Figure 11 illustrates the dialogue template used to enable a designer to attribute construction types to elements of a building. The information is presented in a manner tailored to the users level of expertise. In this instance, the designer is considered to have little experience with building construction and therefore a range of solutions are made

available by means of cascading menus, enabling the designer to make informed decisions regarding the construction of the design solution. This enables the designer to concentrate more specifically on the production of alternative plan formations.

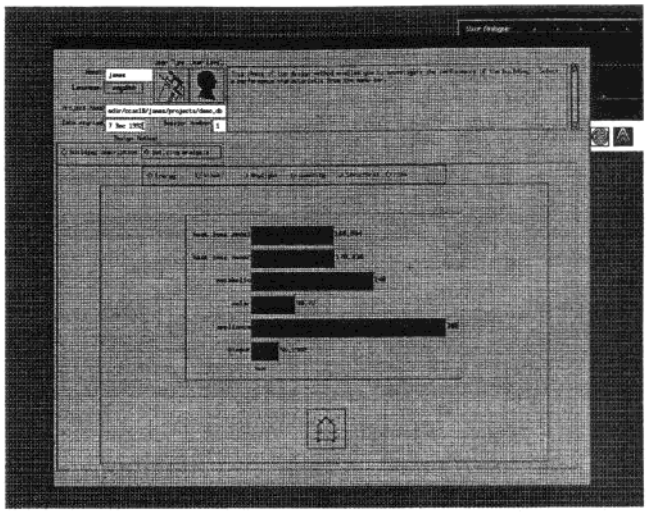

Figure 13. Building design framework-Thermal performance prediction.

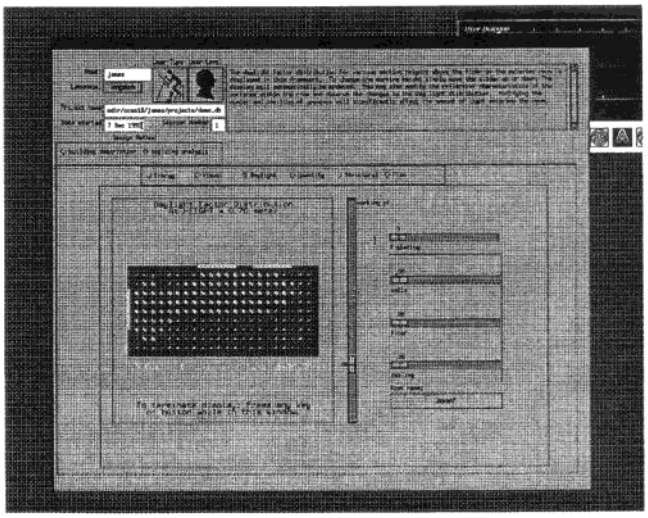

Figure 14. Building design framework -Daylight distribution factors.

Although the description of the current building is sufficiently detailed to perform a computation analysis, the user may override default values by, in this instance, employing an incidental knowledge resource or consultant, specializing in openings (Figure 12).

3.2 Building Analysis

Once a complete description has been provided, the designer may begin to appraise the building in a number of ways. The current building design framework offers several types of analysis, including energy (Figure 13), daylight (Figure 14), cost, and structural and functional analyses, together with visualizations.

4 Conclusions

The overall openness and modularity of the KNODES environment (Figure 15), makes it possible to respond quickly to shifts in design standards and introduce new enabling technologies, thereby increasing a designer's potential to make informed decisions and in turn improve design conceptualization.

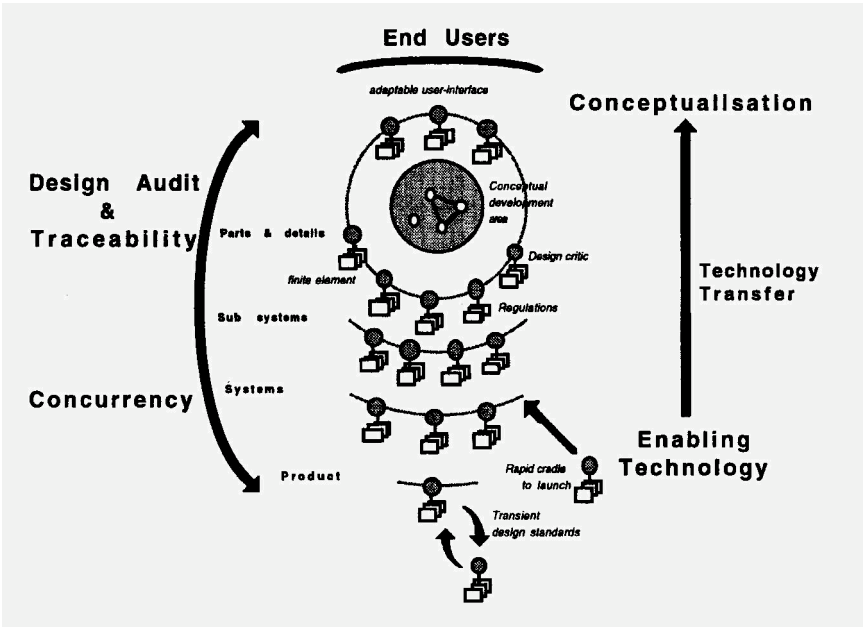

Figure 15. Conceptual overview of the KNODES environment-layering after Björk (1990).

By representing and managing conceptual models within the framework, users of varying levels of experience are able to participate in the design activity and make useful contributions to emerging design solutions. Students using the building design framework were able to make design decisions, formulate and explore alternative scenarios, and con-

verge on a design solution earlier than would have been possible had they had to overcome the operational obtuseness of the design tools at a systems level.

The majority of the tools being used at the back-end of the framework would normally be used once a complete description of a building had been resolved. The use of sophisticated design tools of this form would therefore normally be used in a post hoc checking mode. The framework, adding contextually sensitive defaults, makes these applications accessible at an earlier stage of the design process, providing an opportunity to identify and resolve potential problems before the production drawing stage, thus saving valuable time and resources.

4.1 Future Research

The KNODES environment blackboard, by monitoring and recording transactions between knowledge resources, provides a mechanism for recording and tracing design decisions.

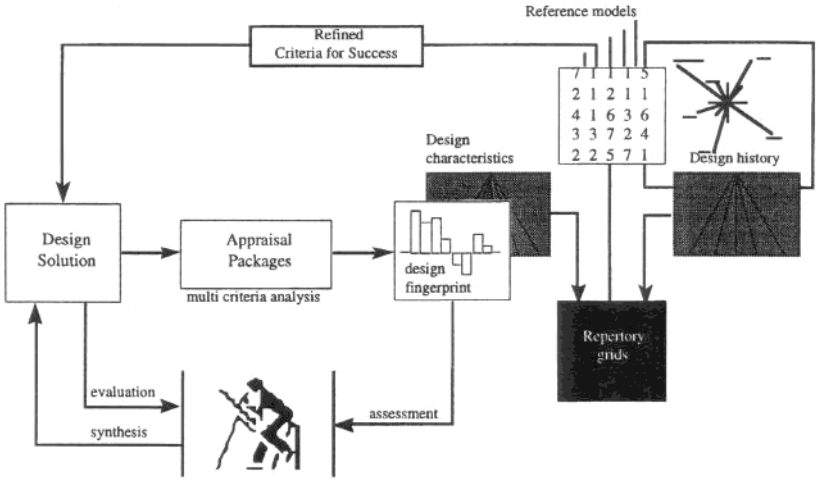

Figure 16. Automated elicitation of a designers belief structures and the derivation of models of the design process.

This facility is to be used to provide on-line protocols of design activity. It is believed that where it is possible to enumerate designs by means of a classification or a diagnostic system (Figure 16), design fingerprints, measured against known metrics, can be used to elicit a designer's belief structures and therefore provide some insights into the nature of design decision making.

References

Anderson, BR., Clark, A.J., Baldwin, R., and Milbank, *NO., 1985. BREDEM - BRE Domestic Energy Model: Background, Philosophy and Description*. Building Research Establishment, Department of the Environment

Autodesk, AutoCAD R11 *Reference Manual,* Autodesk, Guildford Surrey.

Baker, N., Hoch, D., and Steemers, K., "The LT Method Version 1.2- Energy Design Tool for Non Domestic Buildings," Commission of the European Communities Directorate-General XII for Science, Research and Development and Directorate-General XIII for Telecommunications, Information Technology and Innovation.

Björk, B.C., 1989. "Basic Structure of a Proposed Building Product Model," *Computer Aided Design* 21(2), March 1989.

Chin, D.N. 1989. "Modelling What the User Knows in UC," in *User Models in Dialogue Systems, Symbolic Computation,* Springer-Verlag, pp. 74-107.

Chu, P., and Elam, J.J, 1990. "Induced System Restrictiveness: An Experimental Demonstration," *IEEE Transactions on Systems, Man and Cybernetics* 20(1), pp. 195-201.

Encyclopaedia Britannica Inc., 1974. *Micro Media* (15th edition),Vol. 3, p. 45.

Englemore, R.S. and Morgan, A.J. (eds.), 1988. *Blackboard Systems,* Insight Series in Artificial Intelligence, Reading: Addison Wesley.

Gentner D. and Stevens, A.L., 1983. *Mental Models.* London: Lawrence Erlbaum Assoc.

Hart, P. 1982. "Directions for Al in the Eighties," *SlGARTNewsletter, Vol.* 79, pp. 11-16.

Hayes, P.J., 1979. *Naive Physics 1 - Ontology for Liquids.* Memo Centre pour les etudes Semantiques et Cognitives, Geneva.

Kaemmerer, W., and Larson, J. 1986. "A Graph Oriented Knowledge Representation and Unification Technique for Automatically Selecting and Invoking Software Functions," *AAAI-86,* Philadelphia, PA.

Lansdown, J., 1986. "Requirements for Knowledge Based Systems in Design," in A. Pipes (ed.), *CAAD Futures* '85, International Conference on CAAD, September 1985, London: Butterworths.

Markus, T.A., Mayer, T.W., Whyman, P., Morgan, J., Whitton, D., Canter, D., and Flemming, J. 1972. *Building Performance.* New York: Halsted Press.

Mead, M. 1990. "Data Modelling Language 'Express," Informatics Department, RAL.

Morton, J., Barnard, P.J., Hammond, N. and Long, J., 1979. "Interacting with the Computer: A Framework," in E.J. Boutmy and A. Denthe (eds.), *Teleinformatics 79.* The Netherlands.

Ontos, 1990. ONTOS *Object Database - Development and User Guide,* Burlington, MA: Ontological Inc.

Prime, Martin, 1988. "User interface Management Systems (UIMS) - A Current Product Review," Informatics Division, Rutherford Appleton Laboratory.

Reddy, D.R., Earman, L.D. and Neely, R.B. 1973. "A model and a Aystem for Machine Recognition of Speech," *IEEE Transactions on Audio and Electro-acoustics AU-*21, pp. 229-38.

Rutherford, J.H. 1987. "Forms - A Multi-faceted User Interface," M.Sc. Dissertation, Department of Architecture, University of Strathclyde, Glasgow, UK.

Rutherford, J.H. 1990. *An Intelligent Design Support Environment.* Ph.D. Dissertation, Department of Architecture, University of Strathclyde, Glasgow, UK.

Rutherford, J.H. and Mayer, T.W., 1991. "A Multi-Level Knowledge-Based Framework to Support Design Decision Making," grant proposal to the EDRC and SERC.

STEP, 1988. "General AEC Reference Model (GARM)," *Standard for the Exchange of Product Data ISO* TC184/SC4/NGI (Draft) Document 3.2.2.1, October 12, 1988.

Totterdell, P., and Rautenbach, P., 1990. "Adaptation as a Problem of Design," in D. Browne, P. Totterdell, and M. Norman (eds.), *Adaptive User Interfaces,* Chapter 3. Boston: Harcourt Brace Jovanovch.

Vidovic, N., 1990. "Design Framework Tool Kits for Concurrent Engineering Environments," Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA.

Weis, S., Kulikowski, C., Apte, C., and Uschold, M., 1992. "Building Expert Systems for Controlling Complex Programs," in *Proceedings,* AAAI 1982, pp. 322-326

Wilson, M.D., 1987. 'Task Analysis for Knowledge Acquisition," in *Proceedings,* SERC Workshop: Knowledge Acquisition for Engineering Applications, Abingdon, UK.