DESIGN TOOLS INTEGRATION IN AN INTEGRATED DESIGN ENVIRONMENT

by

Inhan Kim

Object and Knowledge Based Systems Group
Department of Computer Science
University of Wales at Cardiff
Wales, U.K.

INHAN KIM

is a Research Fellow with the Object and Knowledge Based Systems Group in the Department of Computer Science at the University of Wales in Cardiff, U.K. He holds a Ph.D. in Architecture from the University of Strathclyde, U.K., a M.S in Architecture from Carnegie-Mellon University, USA and a B.S. in Architecture from Seoul National University in Korea.

ABSTRACT

The design problem has a multi-disciplinary nature and the design itself evolves as solutions are attempted by the designer. To support inter-disciplinary communication of design concepts and decisions, the integration of relevant CAAD tools is essential. Based upon a large set of design criteria and all corresponding knowledge, with the help of computer aided design tools, the result could be highly effective and novel. The integration of CAAD tools should be performed on the basis of generating better solutions by enabling designers to manipulate and appraise various solutions quickly and with a minimum of effort.

The proposed system provides the foundations for a seamless and continuous working environment for architects and building engineers through a *data modelling module*, an *integrated data management framework* and *various design tools*. In the environment, stand-alone design tools can be plugged-in in order to access information stored in central databases. The suggested *data modelling module* helps integrated CAAD systems represent and exchange domain dependent design informa-

tion at a semantic level, such as exchanging components and features of a building rather than graphical primitives. The suggested *data management framework* supports the straight-forward mechanisms for controlling the data representation through the inter-connected modules and design tools.

Design Processes and Detailed Design Knowledge

Traditionally, the internal mechanisms that generate design solutions are considered to be mystical forces within a black box. This view has been a major deterrent to scientific studies of design. People think that design creativity is something mystical and inexplicable. The study of the psychology of designers gives a picture of the initial stages of design process. Akin [Akin, 1986] illuminated architects' design exploration processes by studying their behaviour and designing information-processing models that account for their behaviour. The creative idea is a complex process of the interactions between many mental operations. By clarifying the scope of design knowledge, at least parts of the process can be formalised and represented on the computer.

The popular view of problem solving processes assumes that progress occurs through methodical collection of data and careful inferences from those observations [1]. Another approach is to carry out a heuristic search of the problem space, using *rules of thumb* to suggest likely states to expand and likely operators to select. Human problem solvers appear to rely heavily on heuristic search methods. Simon [Simon, 1982] has proposed a theory that combines models of human memory with information-processing models to explain human problem solving mechanism. He argues that the *chunks* which are related to a higher level-structure contain detailed domain specific knowledge and the *chunks* are necessary for the creative problem solving activity and may allow one to move directly to the goal.

Recent research on problem solving emphasizes the importance of *detailed knowledge*. Detailed past experience is crucial in determining how efficiently an individual solves a novel problem. Developing expertise involves acquiring much knowledge about specific situations, so that one can deal with a new situation on the basis of how it resembles situations faced before. Greeno [Greeno, 1980] notes that much of real problem solving is due to an inability to identify the knowledge underlying the problem solver's performance. Adriaan deGroot [deGroot, 1966] studied the skill of chess masters. The expertise, which is one of the prerequisites for master chess players, comes from years of study and detailed visual memory of chess positions. On the basis of the above argument, it is reasonable to assume that past experience is the basis for the initial approach to a problem and creative action is slow and incremental, and comes from a sound basis of background knowledge. Provided with a set of supporting facts and all corresponding knowledge, with the help of computer aided design tools, the result could be highly effective and novel.

Design Decision Support Systems

Design is an exploratory activity. Usually, the design of an artifact does not follow a fixed process, but involves evolutionary change. An ideal CAAD system should allow a variety of design concepts and design process mechanisms. In applying computers appropriately to systematic domains, it is necessary to develop effective tools. A system equipped with sophisticated tools would be able to manipulate a design database which can model high level design concepts. Design can be simulated by means of heuristic search procedures and a network of inter-related objects. Such simulation will facilitate the transition between successive design states, maintain their internal consistency and guide the process until a recognizable solution is reached. With the right representation and structure, computers can provide multiple abstractions and decompositions for managing and exploring partial design descriptions. Compiled design histories from previous design experience could also be used to evaluate and select relevant design decisions.

Future CAAD system should concentrate on a system which does not confine designers' creativity. The potential for creative design occurring is increased when designers are provided with the capability to explore an expanded formulation space or more of a given solution space for a task. Although computers cannot perform anything that a human cannot think of, a computer can help explore pre-constrained sets of alternatives. This has been the motivation for the development of *decision support systems*. A *decision support system* can be realized by involving the computer in the decision making process. Decision support implies the use of computers to help designers in their decision processes in semi-structured tasks and support, rather than replace human design decisions [Keen, 1978]. One of main aims of the *tools integrated system* is to enhance human decision making by suggesting alternatives, predicting consequences, and pulling together the information that goes into making decisions.

A Product Modelling Approach for an Integrated Design Environment

Traditional CAAD environments are not much concerned with providing easy access to design data through unified data description and standardised data exchange formats. They have inherent shortcomings which diminish possible achievements for architectural practices. Almost all conventional CAD systems rely on a pure geometric data model. All non-geometric information about objects of architectural interest have to be attached to these geometric entities. This restricts the ability to describe semantically

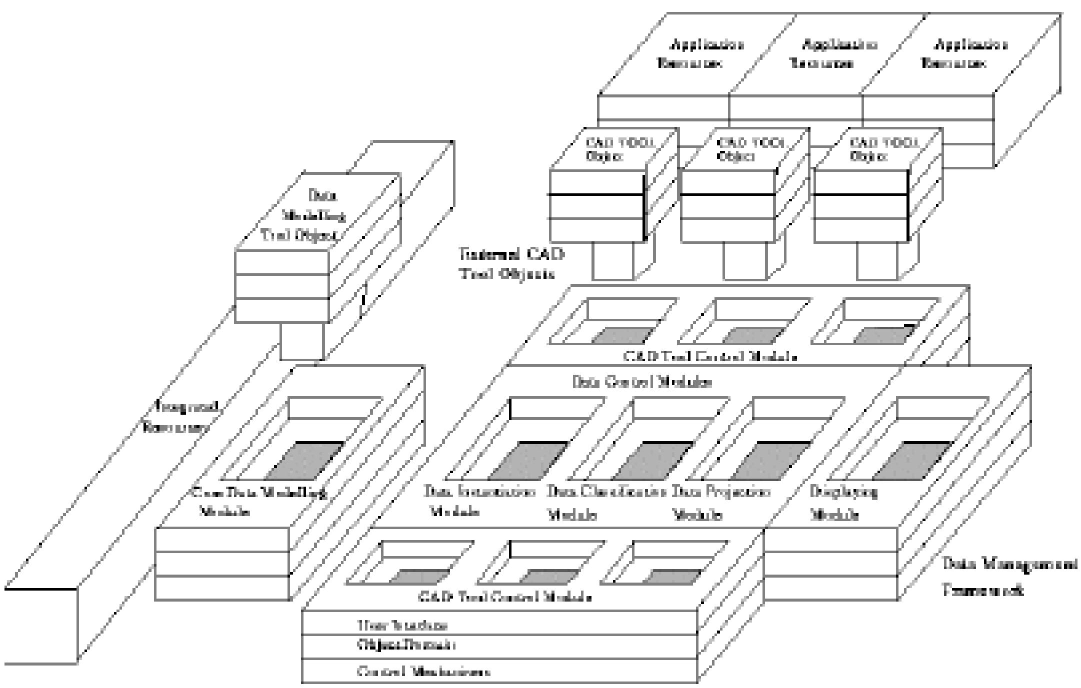

Figure 1.

dependent relationships. Also, the data exchange remains restricted, since it is based on a fairly low semantic level of a document-based exchange of information, such as geometric representation in DXF or IGES, rather than on a high semantic level of a model-based exchange. In consequence, the integration of different design tools for building and construction is still very limited. An appropriate environment, on the other hand, has to ensure effective data communication among the various design stages and design actors, which requires a way to accommodate the multiple views of the disciplines [Fenves, 1990]. Thus, an integrated design environment enables various building design tools to create complex project data over the life-cycle of a design artifact, and to appraise its performance according to several criteria, using the same basic set of project data. The design team would then be able to efficiently cooperate and to predict the performance of buildings in order to improve the quality of the design.

Product modelling is a process leading to an information model which provides an abstract description of facts, concepts, or instructions about a product or a set of products [ISO, 1992a]. This kind of data model, specifying the categories of information about an artifact during its complete life-cycle, is commonly known as a Product Model [PM]. The basic concept used in a PM to describe objects of interest is the entity within a hierarchy of sub/super relationships. The entity is specified in terms of its attributes, relationships and constraints. Thus, a PM incorporates some characteristics of the well-known object-oriented techniques, such as data encapsulation and inheritance.

Furthermore, the product modelling methodology draws a clear distinction between the conceptual level and the instantiation level. a PM specifies the structure of information in schemata with no regard to any format in which the information is later stored. In order to unveil the inherent structure of object data, modelling tools, e.g., the graphical schema definition languages NIAM [Thomson, 1986] or EXPRESS-G and the textual data definition language EXPRESS [ISO, 1991], have been developed, and they are widely used as formal languages to construct uniform expressions about design objects.

ID'EST [2] consists of a *data management framework* and a set of *it computer-based* *design and assessment tools* (See Figure 1); The *data management framework* organises the structure of the design data to keep the design consistent, and forms the framework in which various tools can be built. Accordingly, it has to depend on a data model. The *computer based design tools* assist designers in creating and evaluating the design artifact, and in validating its correctness. These tools are treated as task-related editors of the data model. They receive all relevant data from the central database, map these data into their own separate data structure, and send the modified data back to the central database, in order to keep consistency among themselves [Kim, 1994a]. Only very private data, which are unlikely to be used by other modules, are permanently stored within the individual databases of the design tools. ID'EST is partitioned into smaller, more manageable pieces in order to be maintainable. Therefore, the modular structure is an essential requirement for the suggested design environment. This structure is based on the decomposition criterion known as information hiding which allows system details that are likely to change independently to remain to separate modules. A complete description of ID'EST is beyond the scope of this paper. This paper focuses on the issue of design tool integration, as one of the main aims of the development of ID'EST.

Design Tools Integration

Today a variety of automated tools exist, starting with ones which are for the initial conceptual designs process through to ones for the construction process. The aim of various CAAD tools is to improve the designer's understanding of his/her design by providing computerised evaluations of that design in various aspects. The materials being shared in a design project may include CAD drawings, pictures and video of design components, documents, spreadsheets, case records and so forth. Object-oriented database technology provides the means to maintain this rich set of design information.

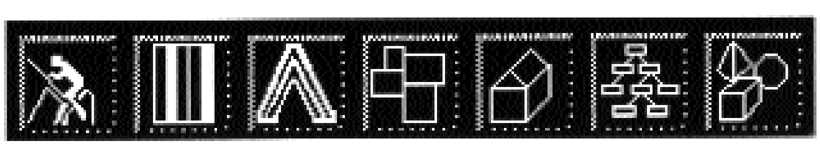

Figure 2

The *rich data model* is an essential element for the integration of multi-media design information. ID'EST aims to support a designer's navigation through the database of design artifacts; co-operative working between design team members; and interdisciplinary sharing of data and knowledge among project participants. Ideally, all internal and external CAAD tools which are integrated in ID'EST should have the semantic data structure which does not violate the common data model. In reality, however, external CAD tools, such as a *geometric analyzer* and *building performance evaluation tools*, require specific data according to their own data format. In ID'EST, stand-alone external CAAD tools can be ``plugged-in" in order to access the required information with the help of the *tool control module* which controls data exchange between the CAAD tools and the central database. When the tool *object* sends a message to ask for appropriate data defined by its methods, the *tool control module* responds to this message and sends the appropriate data to the tool object. There have been similar approaches which define design tools as objects [Vidovic and Siewiorek, 1990] [Daniell and Director, 1989]. The design tools should have a uniform external interface and common data structure to be communicated through the *tool control module*. A tool's external interface is described on the basis of the following classification:

- *Tool Class*: describes the attribute values of each tool class which contains the tool specific values.

- *Required Input:* describes the required input for each tool. Each tool requires specific data format according to their own data structure.

- *Definitions and Control Mechanism:* describes the definitions of data type and data structure and the control algorithm. Necessary data type conversions will be performed accordingly.

- *Output:* describes the result which can be either displayed on the screen and stored in an external file, as a part of a distributed database, for later access by the application. The result of the analysis could be formatted in a number of different ways according to the requirements of the intended application.

```
Geometric_Representation_Item
Group_Of_Spaces
Horizontal_Enclosure
Inner_Space_Boundary
Line
Manifold_Solid_Brep
Material
Measure
Object
Opening
Outer_Space_Boundary
Physical_Component
Physical_Space_Boundary
Physical_Space_Connection
Placement
Poly_Loop
Representation_Item
Roof
Site
Site_Space
Site_System
Solid_Of_Linear_Extrusion
Space
Space_Behaviour
Space_Boundary
Space_Component
Space_Connection
Spatial_System
Surface
Swept_Area_Solid
System
Topological_Representation_Item
Vector
Vertex
Vertex_Point
Vertical_Enclosure
Virtual_Space_Boundary
Wall
Window
```

Figure 3.

By this means, an independent CAAD tool can be transferred into a system with high level object-oriented interface structure that can communicate with the *data management framework* and other CAAD tools relatively easily. The following CAAD tools have been integrated within ID'EST to support basic functionalities of the integrated design environment and test the capability of interactive data communication of the environment:

Geometry Analyzing Tool: In an integrated design environment, it is necessary to have a geometry analyzing tool which has the basic function of analysing a physical description of building design which is held in the project database. The geometry analyser plays a critical role in the environment as a means for improving the designer's understanding of his/her design by providing computerised evaluation of that design geometrical and topological aspect. This tool also automates the extraction of geometry related data for certain evaluations. *Space Layout Tool*: This tool helps a user to explore alternative layouts of a certain building with various constraints and criteria. Although this tool has been initially developed to be used as a stand-alone tool, it is easily integrated within the environment. The integration of this tool shows a great potential of the integration capability of the environment. *Graphic*

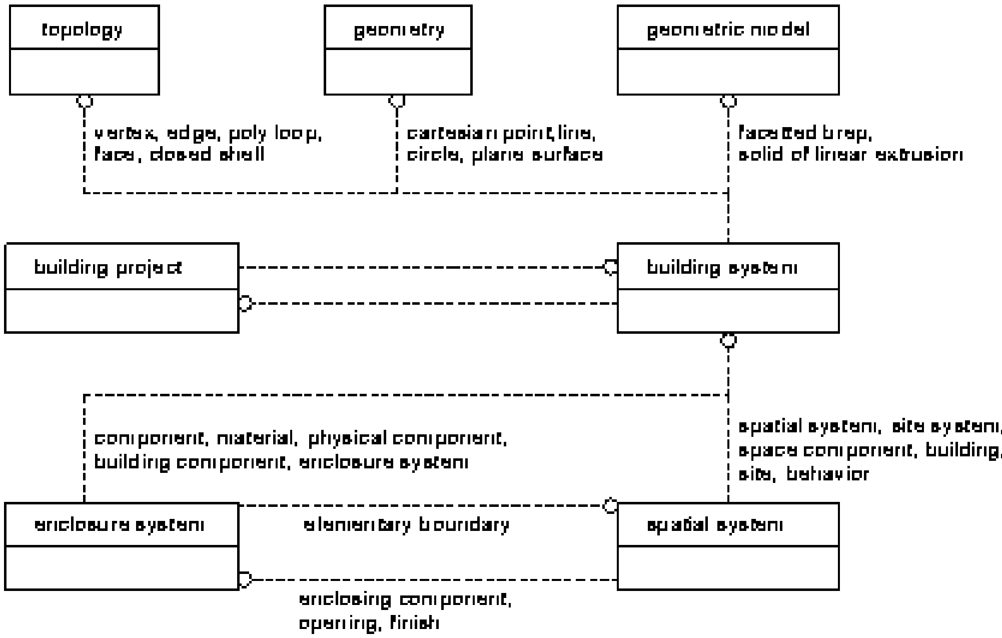

Figure 4.

Manipulation Tool: This tool helps the user to create, view, or change the required object graphically. The graphic viewing operations are performed interactively. Data entry and data modification can be accomplished using various types of input methods, e.g., *keyboard* or *mouse*. *Other Tools*: Other tools such as a *building thermal performance eval - uation tools* and a simple *database querying tool* have also been implemented within the context of intelligent design assistance. Figure 2 shows the CAAD tool icons in the ID'EST interface window.

Data Modelling Processes

ID'EST uses a PM developed according to the STEP methodology. The PM defines the structure of the data instantiation tool, as well as the syntax of the exchange format. The design tools which are connected to ID'EST, communicate on the basis of a STEP physical file, the common exchange format of the environment. The data modelling module specifies entity types that represent objects within the model (See Figure 3).

Based on this specification, the data instantiation module allows the definition and the exchange of unique entity instances. In consequence, every time the data model is changed, the instantiated data model has to be recompiled. Product modelling can be referred to as a developing process including the three phases of analysis, design, and implementation as follows:

Analysis

During this phase, a conceptual model on a high level of abstraction is created (See Figure 4). All schemata make use of integrated resources following the STEP methodology. Therefore, a subset of the generic resources for *topology*, *geometry*, and *geometric-model* [ISO, 1992b] has been defined as application integrated resources. The development of the PM has been carried out using EXPRESS-G.

Design

During this phase, a conceptual model, i.e., a derivation of a neutral data specification form, is completed. The graphical notation has to be mapped into a textual form. In addition, more detailed specifications, such as constraints, functions and rules, are added to the model.

Implementation

During this phase, an implementation code which integrates and controls the design data across multiple representations of design in different design tools, is generated. The EXPRESS definition is mapped into any application form (e.g., C++ classes, frame-structures, or DBMS). The processing tools, developed at NIST [Clark, 1993] were used to facilitate the generation of an instantiation model during the development process of ID'EST.

Schema Evolution during the Design Processes

There are mainly two approaches in designing a building data model. The first is making a *global data model* to describe everything in a single model and then obtain views from it. The second is making an *aspect data model* to include what is likely to be needed as a specific view in a local model while including commonly required information in a *core model*. It is obviously impossible to develop a global schema which meets all of the information requirements of all buildings throughout their life-cycle. An *aspect data model* assumes that it is possible that from one single common model the different actors, i.e. architects, structural engineers, HVAC engineers etc., can derive their building models as views. There will be different models for different purposes created by different actors.

A representation of objects could be common to several disciplines. Most models, especially architectural and structural, have close links to the physical shape. Therefore, the dimensions and physical attributes of objects could be used as the framework in which the representations of other models can be based. In some cases, however, the domains of entities in the different models will not necessarily be the same. They represent only a small part of the information and capabilities needed for the design and life cycle maintenance. These problems, however, can be solved by using the control mechanisms provided by the object oriented paradigms, e.g., attribute overloading, multiple/partial inheritance and data encapsulations.

There are basically two types of users of building modelling systems. Architects use the systems mainly to generate the final shape of building, whereas environmental engineers or construction engineers use the systems mainly to evaluate and construct the design.

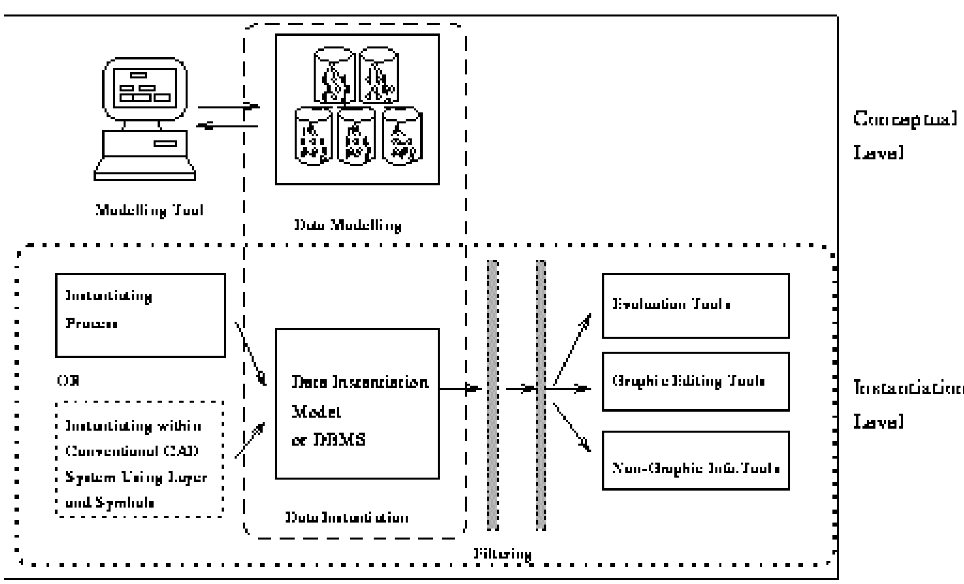

Figure 5.

Currently, ID'EST mainly support the latter view as it is normally applicable to the building situation where the shape has been completely defined. However, even in the middle of building shape evolution, a form of "freeze frame" snapshot of the instantiated data model in ID'EST can be used for evaluation, as it is possible to change schemata and to re-compile the data model at any stage. The structure of relations needed to model a complete building can vary during design or even during construction planning. Therefore, it would be of great benefit to have building design systems which deal properly with the early stages of building design in which shape and configuration are not yet stable. There have been several data modelling approaches which take into account the open and dynamic view of design process [Eastman, 1992] [Luiten et al, 1993]. To allow STEP to deal with the schema evolution, EXPRESS and corresponding STEP parts should be revised [3].

Data Management Framework

The data management framework controls and interprets the collection of data instances within an application form of the data model. The framework is responsible for creating, maintaining and viewing a consistent database of the design description. Consistency checking and constraint propagation are further task of this framework. The data management framework consists of several modules. The interplay between these modules is given in the environment's architecture (See Figure 5). The data management framework has a *tool control module* which pre-processes the data for design tools, which can be either realised as a file-based exchange or provided as a direct access interface [4]. An *integrated design database* is the core part of the data management framework. The *integrated design database* is a prerequisite for creating an integrated design environment by which design data can be shared among the different design tools of the design environment. With the help of the *integrated design database*, the design data management framework organises the design description within each representation, correlates equivalent descriptions across the representations, and attempts to maintain these correspondences as the design evolves incrementally.

The Creation of Building Design Instances The main objective of the data instantiation module is to have an instance manipulation environment for the semantically structured entities defined in the suggested data model. This module creates, edits or views information stored in the instantiation model. An external CAD tool is used to convert the external CAD tool data format to the STEP physical file format, in order to import data to the project database. In the ID'EST system, DataProbe [5] has been used to create, edit, or view data corresponding to the information model for which it was created. DataProbe is also used to read, merge and write STEP physical files.

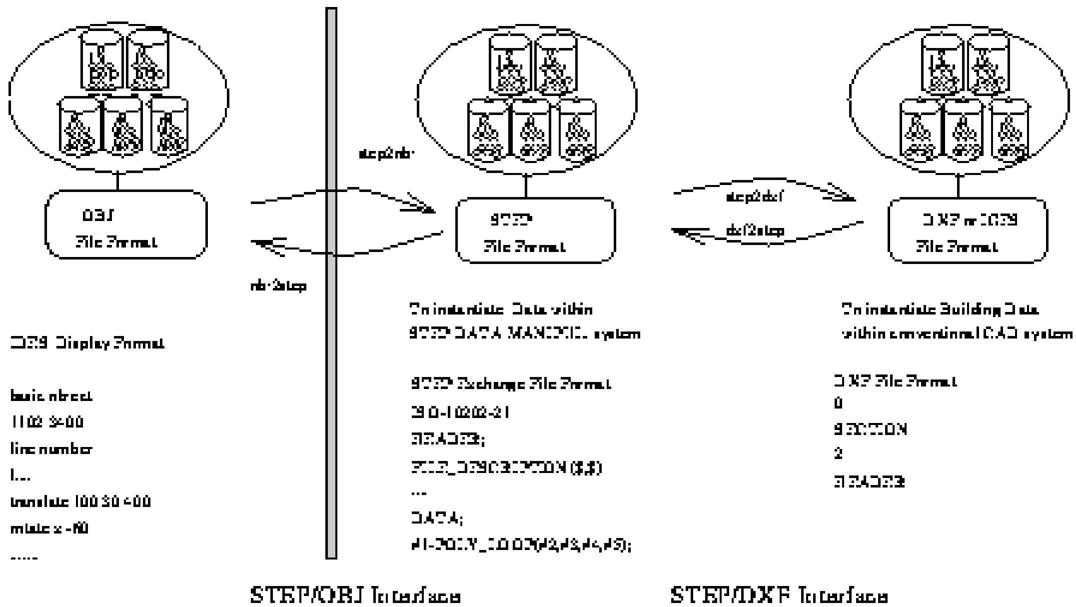

Figure 6.

The creation of building design instances is normally performed using conventional CAD systems, especially for the input of geometry data. Beside the geometrical representation, more semantic information is needed as input into the instantiation model.

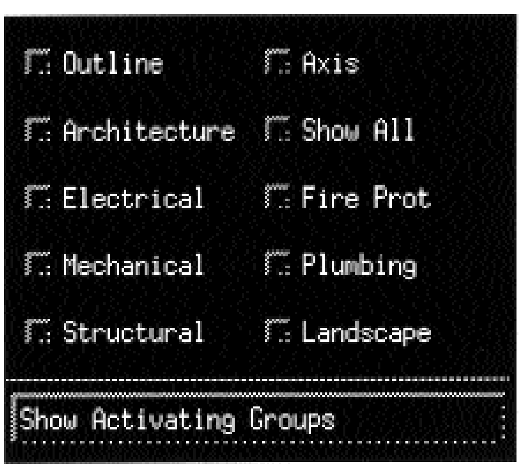

Figure 7.

Therefore, the inherent structure of CAD systems, provided by layers, macros, and attached attributes, plays a key role in enabling the extraction of appropriate data. Naming conventions, as suggested by the AIA in the U.S. [Schley, 1990], which has several recommendations to classify graphic data [6], thus giving the opportunity to map this implicit information into the object-oriented mode of data representation, are used in ID'EST.

The usage of file-based exchange between traditional CAD systems and an object-oriented design environment also shows a path to preserve the usability of existing tools for a certain period after the introduction of a new software generation. The STEP/DXF interface, which was developed as a part of the proposed environment, reads a DXF file and translates it into a STEP physical file, according to the given PM. The interface makes use of a user-defined mapping table allowing other *layer conventions* to be easily adopted. The mapping table defines the relation between layer names and entity types as well as between attached CAD attributes and attributes in the STEP file. Semantic information, however, which can not be created using conventional CAD systems, has to be added within the instantiation model. In this case, the instantiation model is further refined within DataProbe, after the STEP file has been read. Besides the STEP/DXF interface, ID'EST communicates through other interfaces, such as the STEP/OBJ interface, as well. All interfaces have in common that they map the data model as contained in the instantiated data model and in the STEP physical file, to the specific formats, used by other modules (See Figure 6).

Data Filtering Process

The data filtering process is designed to acquire object descriptions, both textually and graphically, and to reflect the user's intentions and descriptions. This module has the capability of performing selected inheritance for a required discipline. This module classifies the *project database* according to the design discipline and elicits necessary infor-

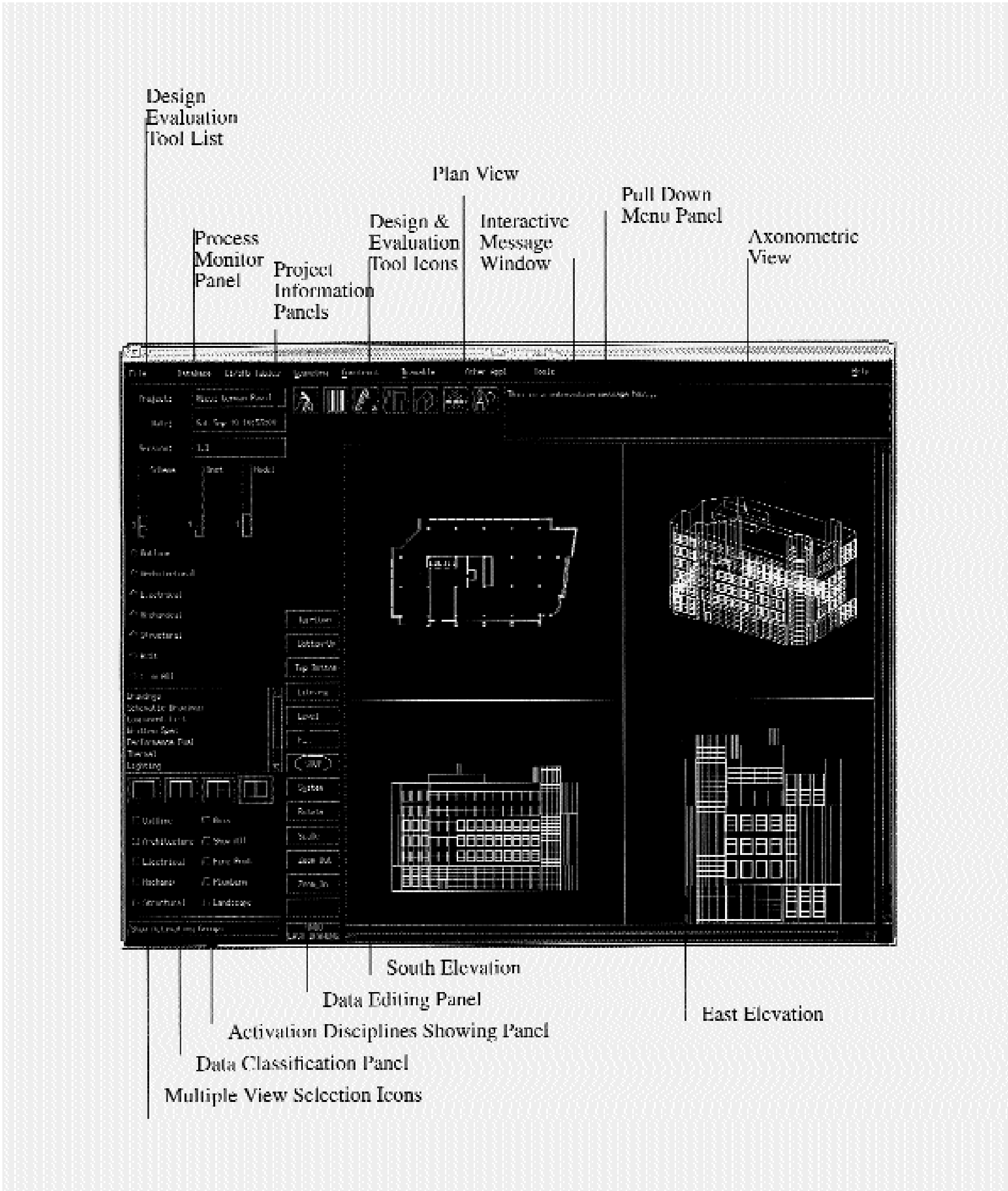

Figure 8.

mation for the required discipline from the database (See Figure 7). To elicit appropriate information from the *project database*, the data classification process checks the desired entity types, filters necessary information from the database, and sets special marks on the information to identify the necessary information. By checking entity types and attributes, the data projection process is executed with possible partial model projection and possible re-execution of the data classification process. For example, to generate a

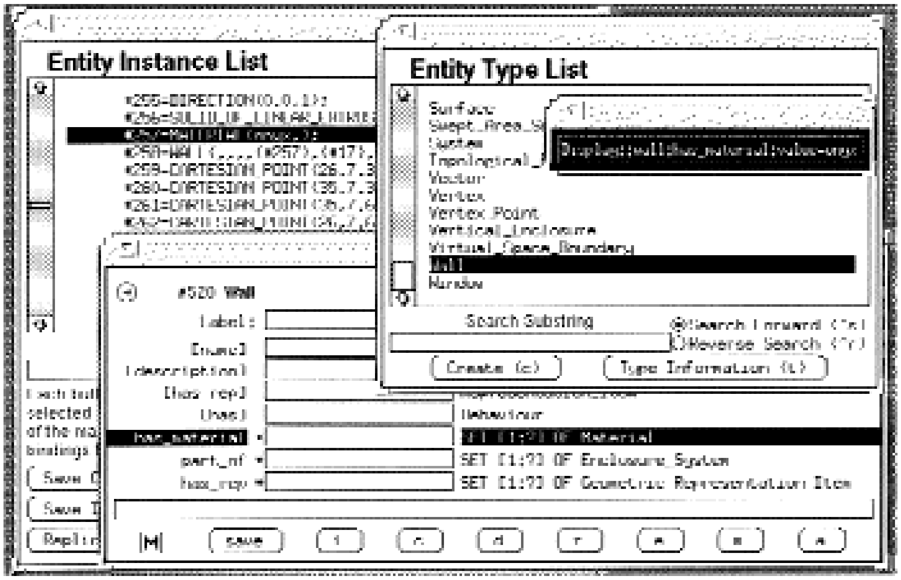

Figure 9.

floor plan of a building, only data of the architectural discipline have been selected to search at the first stage. With proper projections including selected inheritance, data segregation, and geometric transformation, partial information for the required plan view can be generated.

Test Case

A medium-rise office building located alongside the river Clyde in Glasgow has been selected as an example to test the capability of the data management framework and the tools integration. Most of the building elements have been defined in PM, and have been instantiated and processed to DataProbe and ID'EST. The geometric attributes, including additional information such as material, was created in AutoCAD and structured according to the given layer naming convention. The AutoCAD file was later converted to a STEP physical file by the STEP/DXF interface and included into the data instantiation module. After more semantic information had been entered, relevant data were sent to the data filtering modules. As a result of the data filtering process, an OBJ data file, i.e., a projected file of the STEP physical file, can be generated. The OBJ file [7] can store the semantic as well as syntactic information since it is allowed to have structured entities.

Figure 8 shows several different representations of the building in ID'EST. Information about the building can be accessed and changed from any of the representation drawings. The representations of the building are just projected views of the non-redundant *project database* resulting from the *data filtering* processes. If users update one of the representations windows, all the other representations of the building will be propagated and updated accordingly. To keep the representations of the design object consistent, a data propagation facility has been developed in ID'EST. To propagate changes across representations, the design propagation facility will require the detailed semantics of the design representations to be interpreted by the system. When the data are changed, all relevant data are propagated accordingly.

The display setting can be controlled by a layered control module, as the interface system allows selection of desired entities from the instantiated data model. Data control facilities have also been devised to manage the different presentations of an object. To extract a series of flat views from a project database, layers, macros and attached attributes are used to differentiate information between different types of information. For example, on a building plan, only the outline of a room is shown; the details of the room plan are suppressed. In an elevation drawing, details beyond the plane of the elevation are suppressed. Top left view of figure 8 shows the *ground floor plan* of the building. In the *ground floor plan* drawing, with the help of a display depth control mechanism, several different levels of semantic structures have been included. The outlines of the plan drawing are derived from ``outline" layers, section views of column objects are derived from column layers in the structural layering set, and section views of windows and doors in the level of the floor are derived from a library of ``Reusable Object".

To view or edit the desired information from the project database, a high level command based interface is provided. The commands can be used to query entities in the project database, as an entity is described as a collection of features associated with the object. For example, to check all walls which consist of a certain material, a simple command like

Display::wall:has-material:value="onyx"

can recursively check all walls in the project database and highlight them on the monitor (See Figure 9).

CONCLUSIONS

Although the use of the computer brought a fundamental reconsideration of architectural design practices, effective computational design environments have developed relatively slowly. One of the major obstacles originated in the lack of *evaluation* and *selection* abilities of the computer. Without these key abilities, it is very hard for computers to employ strategies akin to the mechanisms that a human employs in designing. To have an ideal design system which helps the designer to do creative design as complementary abilities of computer, integration of design tools is essential. By providing appropriate information and calculating large sets of design criteria using various design tools at each design stage, the outcome is more likely to bring novel and surprising results.

The suggested design environment facilitates cooperation between the different disciplines of building design and construction. It offers control and management to keep the design description consistent. Several design and evaluation tools can access data from the ID'EST, thus making the behaviour of the design artifact more predictable. These data can also be further used in later stages, such as facility management or post evaluation. The design environment shows the possibility of a seamless and continuous working environment for architects from the initial data modelling process to the final design solution by providing a *data model*, integrated design data control modules and various design tools.

Acknowledgement

The author wishes to thank the ABACUS unit in the University of Strathclyde, and the Object and Knowledge Based Systems Group at University of Wales Cardiff for their support of this project. The development of ID'EST, especially the data modelling module, is the result of co-work with Thomas Leibich, CAB, Germany.

Endnotes

[1] Hadamard [Hadamard, 1949] suggests four distinct stages that seem to occur in every case of scientific insight: *preparation*, *incubation*, *illumination*, and *verification*.

[2] ID'EST is an acronym for Integrated Design Environment using STep methodology.

[3] Currently, revised EXPRESS is under development by ISO STEP working groups.

[4] Using STEP, there are two standardised definitions for data interchange. Part 21 [ISO, 1992c] describes the implementation method to physically exchange data (often referred to as STEP physical file). Part 22 [ISO, 1992d] describes the common access interface.

[5] This software is developed at NIST [National Institute of Standards and Technology], USA. DataProbe can have an entity type list which contains the list of entity types defined in the information model.[Sauder, 1993].

[6] A layer name (according to AIA layer guide) using the short format, has the principle form xyyzz-00, where x is the major group, representing the different disciplines (e.g., A = architecture, S = structural engineering), yy is the minor group, designating construction systems, zz is an user-defined minor group, and 00 reflects the level.

[7] The STEP/OBJ interface reads a STEP physical file and translates it into OBJ file format [Kim, 1994b].

References

Akin, O., “Psychology of Architectural Design,’’ Pion Ltd., London, 1986.

Clark, S.N., “An introduction to the nist pdes tool kit,” Technical report, NISTIR 4336, National Institute of Standards and Technology, USA, 1993.

Daniell, J. and Director, S.W., “An Object Oriented Approach to CAD Tool Control Within a Design Framework,’’ Engineering Design Research Center, CMU, Pittsburgh, PA, USA, 1989

deGroot, A., “Perception and memory versus thought: some old ideas and recent findings,’’ In Kleinmuntz, B., editor, Problem Solving: Research, method, and theory, pages 19—50, New York, 1966. John Wiley.

Eastman, C.M., “A Data Model Analysis of Modularity and Extensibility in Building Databases,’’ Building and Environment, pp135-148, Vol 27, No: 2, 1992

Fenves, S.J., and Flemming, U., Hendrickson, C., Maher, M.L. and Schmitt, G., “Integrated software environment for building design and construction,’’ Computer-aided design, 22(1):27—36, jan 1990.

Greeno., J.G. “Trends on the theory of knowledge for problem solving,’’ In Tuma, D.T. and Reif, R., editors, Problem Solving and education: Issues in teaching and learning, Hillsdale, NJ, 1980. Erlbaum.

Hadamard, J., “The Psychology of Invention in the Mathematical Field,’’ Princeton University Press, Princeton, NJ, 1949.

ISO TC184/SC4/WG3/P6., “Industrial automation systems- product data representation and exchange - part 11: Descriptive methods: The express language reference manual,” Technical Report April, N14, 1991.

ISO TC184/SC4., ``Industrial automation systems- product data representation and exchange - part 1: Overview and fundamental principles,” Technical Report September, 1992a.

ISO TC184/SC4/WG3., “Industrial automation systems- product data representation and exchange - part 42: Integrated generic resources: Geometric and topological representation,” Technical report, 1992b.

ISO TC184/SC4/WG3/P6., “Industrial automation systems- product data representation and exchange - part 21: Clear text encoding of the exchange structure,” Technical report, 1992c.

ISO TC184/SC4/WG7., “Industrial automation systems- product data representation and exchange - part 22: Standard data access interface,” Technical Report, September, 1992d.

Keen, P. and Scott-Morton, M.S., “Decision Support Systems: An Organizational Perspective,’’ Addison-Wesley, Reading, MA, USA, 1978.

Kim, I., “Data Representations in an Integrated Architectural Design Environment: Unified Data Organisation and Management,’’ PhD thesis, Dept. of Architecture and Building Science, University of Strathclyde, Glasgow, United Kingdom, September 1994a.

Kim, I., “Data organization and management: In an integrated design environment,’’ In Maver, T.W. and Patric, J. editors, Virtual Studio, ECAADE ‘94, Glasgow, U.K., 1994b.

Luiten, G., Froese, T., Bjork, B.C., Cooper, G., Junge, R., Karstila, K. and Oxman, R., “An Information Reference Model for Architecture, Engineering and Construction,” Engineering and Construction, edited by K.S. Mathur et al, pp391-406,1993

Sauder, D.A., “Data probe user’s guide,’’ Technical Report, National PDES Testbed Report Series, NISTIR 5141, National Institute of Standards and Technology, USA, 1993.

Schley, M.K., “Computer aided design layer guideline: Recommended designations for architecture, engineering, and facility management computer-aided design,” Technical report, The American Institute of Architects Press, Washington, D.C. USA, 1990.

Simon, H., “The Science of the Artificial,’’ MIT Press, 2nd edition, 1982.

Thomson, D., “Nyssen information analysis method (niam),” Technical report, Control Data, 1986.

Vidovic, N. and Siewiorek, N.P., “Integration Infrastructure: A Software Model for Integrated Environment for Concurrent Engineering,’’ Engineering Design Research Center, CMU Pittsburgh, PA, USA 1990