Representations and Control of Design Information in an Integrated CAAD Environment

Inhan Kim

Object and Knowledge Based Systems Group
Department of Computer Science
University of Wales
Cardiff CF2 4YN
UK

Thomas Liebich

CAB
Osterwaldstrasse 10 D-8080
Muenchen
GERMANY

This paper investigates the mechanisms by which effective data communication between the various design stages and design actors may be facilitated in an Integrated Design Environment. The design team would then be able to cooperate efficiently and easily predict the performance of buildings, thus improving the quality of the design. Within the proposed prototype design environment, a core data model and a data management system have been implemented to connect all applications seamlessly. The core data model supports semantically meaningful descriptions of buildings. The data management system supports consistent and straightforward mechanisms for controlling the data representation through interconnected modules. An existing building is used to test the integration capability of the implemented system.

Keywords: product modelling, object-oriented database system, computer-aided architectural design [CAAD]

1 Introduction

A design process is a multi-disciplinary by nature and the design itself evolves as solutions are attempted by the designer. Traditional CAAD environments have inherent shortcomings which diminish possible achievements by architectural practices. The data exchange remains restricted, as the CAAD system is based on the fairly low semantic level of a document-based exchange of information rather than on the high semantic level of a model-based exchange. A mechanism to provide easy control of design data through uniform data descriptions and standardised data exchange formats would be of great benefit. The implemented system, IDEST [1], is an object-oriented design environment following the product modelling approach. IDEST creates semantically meaningful descriptions of project data within a single common conceptual foundation over the life-cycle of a design artifact, and appraises the performance of the design artifact employing various building design tools which use data models derived from a common conceptual foundation. A complete description of IDEST is beyond the scope of this paper which focuses on the data control mechanism as one of the main issues of the development of IDEST.

2 Structure of IDEST

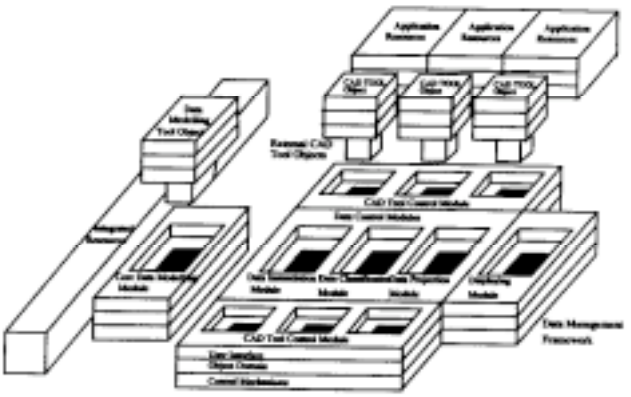

Figure 1 The Structure of ID`EST

The structure of an Integrated Design Environment is inherently complex because integration limits the amount of data to be processed, whereas the data structure should be flexible enough to satisfy the diverse foci of the views required in actual design [Fenves et al., 1990]. Therefore, the modular structure is an essential requirement for an Integrated Design Environment. IDEST is partitioned into smaller, more manageable pieces in order to be easily maintained. This structure is based on the decomposition criterion known as information hiding, which allows system details that are likely to change independently to be secrets of individual modules. IDEST comprises a core data model, a data management system and a set of computer-based design tools [2] (see Figure 1).

(a) The suggested core data model provides semantically meaningful descriptions of buildings. By means of the core data model, integrated CAAD systems could represent and exchange design information at a semantic level, i.e. the users way of thinking, such as exchanging components and features of a building rather than graphical primitives. In consequence, the data model reduces potential misunderstandings and communication problems among the multiple disciplines of architectural design.

(b) The data management system organises the structure of the design data to keep the design consistent throughout the design process, and forms the framework in which various building design tools can be seamlessly built. Accordingly, it has to depend on the core data model.

(c) 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 instantiated data model. They receive all relevant data from the instantiated data model, map them into their own separate data structure, and send the modified data back to the model, in order to maintain consistency [Augenbroe, 1993]. Only private data, which are unlikely to be used by other modules, are permanently stored within the individual databases of the design tools.

3 Data representation in IDEST

A data model provides a method of describing the data types, relationships and constraints of the information that is stored in a database. The database could be implemented in a proper object-oriented database system or could be implemented as a nonpersistent form, based on programming languages or data modelling tools, e.g., C++ or DataProbe [3]. The internal database schema in a proper database system should conform

to the schema in an application interpreted model. The data model has the potential to fully represent the real object through the information needed to analyse and simulate the object. ,thin a single common conceptual foundation, a variety of possible future data models can eventually be developed and communicated.

Two approaches in designing a building data model are predominant. The first involves constructing a global data model which describe everything in a single model and then draw projected views from it. The second involves constructing 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 data schema which meets all of the information requirements of all buildings throughout their life-cycle, as a building data model is a particularly complicated domain due to its complex and heterogeneous nature. An aspect data model, however, assumes it is possible that from single common conceptual model, the different actors, i.e. architects, structural engineers, HVAC engineers etc., can derive their building models as views. Thus, there will be different models for different purposes created by different actors.

A representation of objects could-be common to several disciplines. Most models, particularly 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 could be based.

In some cases, the domains of entities in the different models will not necessarily be the same. For instance, the shape definition of a particular wall could be different between architects and structural engineers. However, these incompatability problems represent only a small part of the information and capabilities needed for the design. These problems can be solved by using the control mechanisms provided by the object oriented paradigms, e.g., attribute overloading, multiple /partial inheritance and data encapsulations [Belford & Santone, 19911.

Figure 2 PADM in the Data Modelling Process

IDEST comprises the Prototype Architectural Data Model [PADMJ developed according to the methodology of STEP. PADM defines the structure of the core data model, as well as the syntax of the exchange format. The structure and contents of PADM are described in [Liebich and Kim, 1995] in more detail. The design tools, which are connected to IDEST, communicate on the basis of a STEP physical file, the common exchange format of STEP [ISO, 1992a}, and a database in 02 [02 Technology, 1994a] [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. Figure 2 shows a part of the data modelling process which specifies entity types that represent objects within the model. Based on this specification, the data instantiation module permits the definition and the exchange of

unique entity instances. In consequence, every time the generic data model is changed, the data instantiation module has also to be recompiled.

4 Control of design information in IDEST

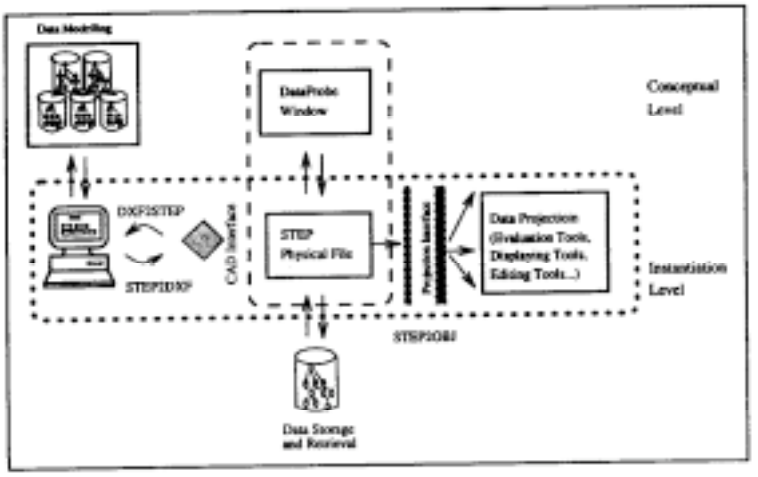

Figure 3 Data Processes in IDEST

To control and interpret the collection of data instances which is in an application form of the generic data model, a data management system has been devised. The data management system is responsible for creating, maintaining and viewing a consistent database of the design description. Consistency checking and constraint propagation are further tasks of this system which consists of several modules, i.e., data instantiation, data classification and data projection modules. The interplay between these largely independent modules is given in the environments architecture (see Figure 3). These modules assist in the processing of data for design tools, which can either be realised as a file-based exchange or provided as a direct access interface [5].

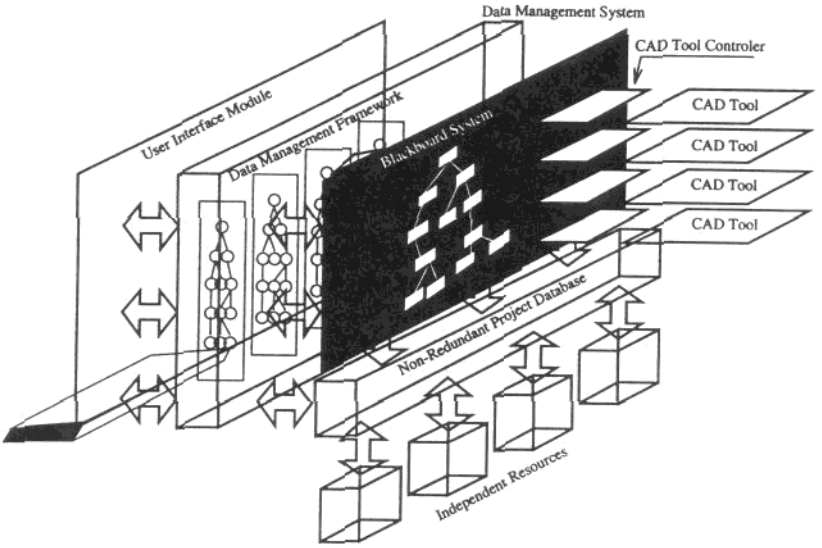

Figure 4 Data Management System in the Environment

Figure 4 shows a blackboard system and an integrated design database which provide an efficient means to store and manage design data in the data management system.

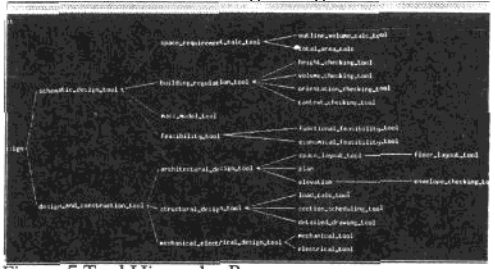

Figure 5 Tool Hierarchy Browser

(a) The blackboard system comprises the problem data and a hierarchy of hypotheses relevant to the problem. The blackboard system is similar to that of IFE [Clarke et al., 1989] and KNODES [Rutherford, 1993] as it serves as an inter-resource notifier and acts as a communication centre between the various modules. The system is also responsible for maintaining and browsing through the design tool hierarchies and rationale networks. Figure 5 shows the design tool hierarchies displayed in the form of an outline editor which allows the user to display the level of detail that he/she considers appropriate.

(b) The integrated design database is a prerequisite for creating an integrated environment by which design data can be shared among the different tools of the environment. In IDEST, to construct the integrated design database which comprises semantically rich design data, both 02 and a STEP physical file are used. These data storage mediums follow the same underlying data structure defined in the PADM definition for the environment. In the environment, 02 is used as the main medium to store massive library data, multi-media design information and non-graphical information of a design project, whereas the STEP physical file is mainly used for data exchange purposes.

With the help of these software support tools, the design data management system organises the design description within each representation, correlates equivalent descriptions across the representations, and attempts to maintain these correspondences as the design incrementally evolves.

4.1 Multi-media design information

The materials being shared in a design project may include CAD drawings, pictures and videos of design components, documents, spreadsheets, case records and so forth. To improve the designers understanding of his/her design, computerised evaluations of the design in various aspects are essential nowadays. The use of multi-media allows us to address an important aspect of collaborative design, as it helps a designer to understand the arguments of other members of the design team by providing examples or making reference to previous cases. In IDEST, multi-media information can be stored as a part of an integrated project database by virtue of the object-oriented database technology which provides the means to maintain this rich set of design information. To control the relationship between the underlying data structure of the instantiated data model and the multi-media information stored in 02, additional referencing methods are implemented in the data management system.

4.2 Design version management and concurrent design issues

One of the main data controlling requirements of IDEST is to cope with design versions in such a way that multiple designers can work on various aspects and components of the same design concurrently. There should be some form of management of the design versions of artifacts and their components, and the integration of component versions into versioned design configurations. The handling of the design versions should be realised by communicating changes to parts of the design to other members of the design team and drawing their attention to the effect of these changes on their own current design version of a component.

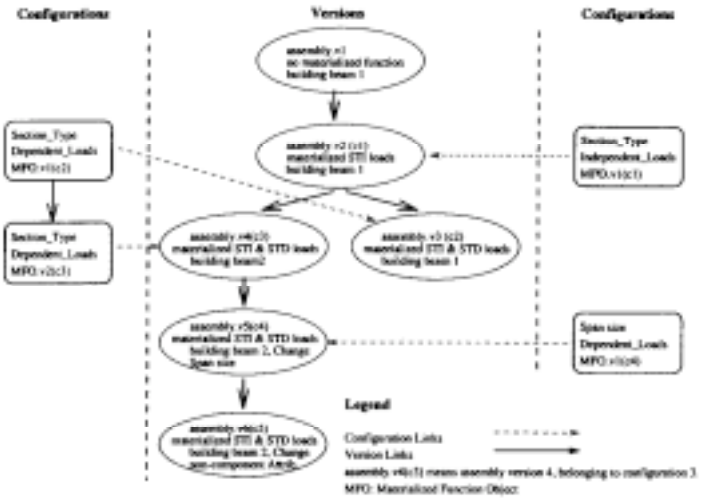

Figure 6: Evolution or Versioned Object from [Kim et al,, 1995]

Existing CAD representations of building and engineering designs may be encapsulated within database objects, and hence, be given versioning capabilities, which will allow them to contribute to version evolution graphs. In structural design this approach can include using approximate formulae to provide updated section sizes or areas of reinforced steel. In a version model, a set of versions of an object is managed by a generic version management approach. This generic version maintains a version evolution history and provides a means of access to each of the object versions in the version set, through a dynamic reference. Versioned objects may also be referenced statically without using the generic version. The design version management system is currently under development by the DESCRIBE team [Kim et al., 19951 using 02 (see Figure 6). Some important concepts of the versioning model are described as follows [02 Technology, 1994b]:

(a) Root: When a new version is created and initialized, it corresponds to the creation of a version unit and a first version of it. This first version is the root of the derivation graph.

(b) Default Version: Information of a default version is persistently stored and the system always chooses the default version when no other version is explicitly chosen.

(c) Class: An instance of the class is a particular version belonging to a version unit which manages the versions of object collection.

(d) Linked versions: Different versions of the same version unit are linked together by the derivation graph. All these versions comprise the version unit.

5 Data instantiation module

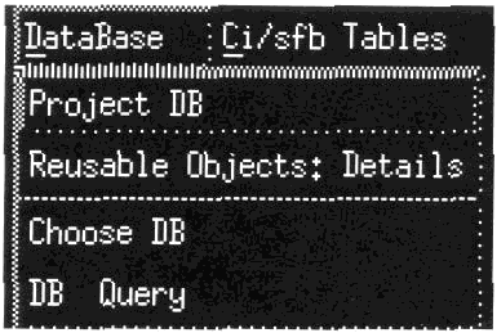

Figure 7 : A Pulldown Menu for Database Manipulations

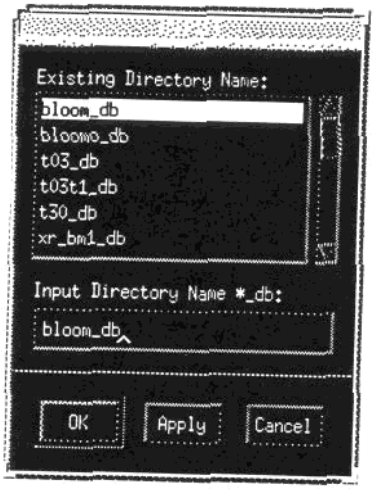

Figure 8 A Popup Window for Database Selections

The data instantiation module creates, edits or views information which is in the instantiation model. Existing CAD systems are able to create geometrical representations of building objects, and make it possible to add more semantics information to the instantiated data model. However, besides such geometrical and geometry related information, additional semantic information is needed for input into the instantiation model. Although it is initially proposed that a STEP physical file is used both for neutral file exchange and for implementing and sharing product databases and archiving, it is inefficient to include comprehensive project information in a STEP physical file. The use of an object-oriented

database system together with the STEP physical file provides the way to control and store massive library data, multi-media information and non-graphical information related to a design project. Figures 7 and 8 show the way a user can select a database to be used for a design project. The "Reusable Object" databases are stored in 02 and can be accessed by the data management system.

The main objective of this instantiation module is to provide an instance manipulation environment for the semantically structured entities defined in PADM. It is a tedious and time consuming task to actually instantiate values of each attribute of a data model, if every instance has to be entered manually. e.g. all cartesian coordinates for each object. Thus, an instantiation module should provide a mechanism to input data graphically to classify data when entering it into the database. A new generation of object-oriented 3D modelling systems which can represent information at a semantic level is necessary, but unless they are available, conventional CAD systems can be used as data instantiation tools with special conventions and certain restrictions. In IDEST, AutoCAD is used to instantiate entities. In this case, in order to import data to the project database, it is necessary to have an accessory tool to convert the external CAD tool data format to the STEP physical file format. Also, DataProbe [5] has been used to create, edit, or view data corresponding to the information model for which it was created. The software tool is also used to read, merge and write STEP physical files.

Although conventional CAD systems have the capability to create a geometrical representation of building objects, to supplement semantic information as needed for input into the data model, the inherent structure relating to the storing data in CAD systems should be classified so that appropriate data for each view can be extrapolated in a standardised manner [Kim, 1994]. To structure design data and to enable the extraction of appropriately structured design data from CAD systems, the methods of naming conventions [Schley, 1990], labelling layers and macros are used [6].

6 Data classification module and data projection module

The graphical representation, including additional information, such as material and cost of physical entities in the design project, was created in AutoCAD and structured according to the given naming conventions. The AutoCAD file was later converted by STEP/DXF Interface and included in the data instantiation module. After more semantic information had been entered, relevant data were sent to the data classification and data projection modules.

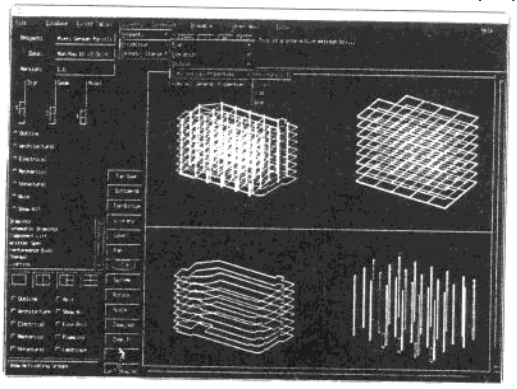

Figure 9 Dynamic Representations of the Structural Entities for the Prototype Building

The data classification module classifies the instances of the instantiated data model according to the selected design disciplines - architectural, structural, mechanical or any other related disciplines - and elicits information necessary for the required projections, such as a ground floor plan for the architectural view, an axonometric of columns for the structural view or an elevation of mechanical entities for the mechanical

view. Figure 9 shows four dynamically generated views of structural entities of the prototype building [7] comprising views of beams, slabs, columns and a combination of all three. The projected views are virtual entities which do not exist as data in their own right.

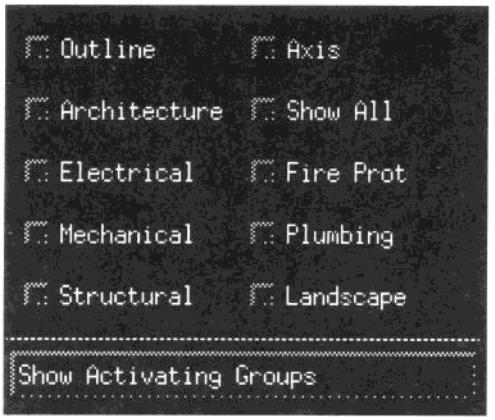

Figure 10 Discipline Selection Tool

Figure 11 Discipline Grouping Tool

In an ideal integrated design environment, a designer should be able to examine information relating to relevant disciplines which are stored in the project database. Figure 10 shows the Discipline Selection Tool. By selecting one or more buttons in the tool panel, the user can view or edit building data of one or more disciplines at the same time. Figure 11 shows the Data Grouping Tool which enables one to limit the views to the currently selected disciplines. To elicit appropriate information from the project database, the data classification module checks the desired entity types, filters necessary information from the database, and sets special marks on the information to identify that which is necessary. 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.

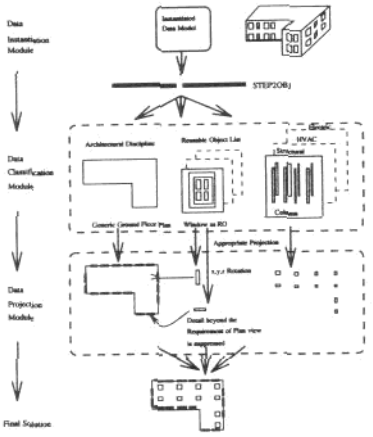

Figure 12 Data Filtering Process in IDEST

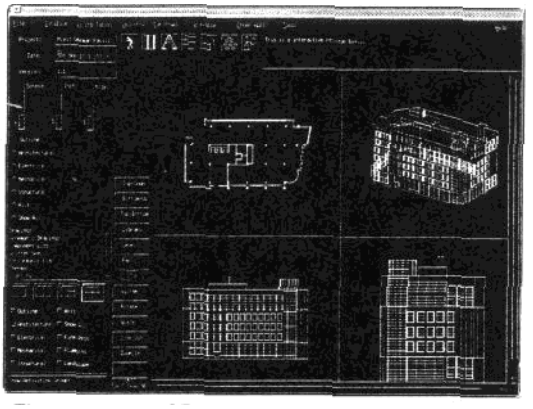

Figure 13 Several Dynamic Representations of the Prototype Building

Figure 12 shows a floor plan of a building in which, with the help of a display depth control mechanism, several different levels of semantic structures have been included. The top left view of figure 13 shows the ground floor plan of the prototype building. To generate this view dynamically, only data defined as the architectural entities will be searched during the first stage. During the next stage, with proper projections including selected inheritances, data segregations, and geometric transformations, a certain amount of information for the required plan view is generated. In this case, re-entering of higher level data abstraction is necessary to obtain a section view of necessary columns and their positions. The outline of the plan view is derived from "outline" layers; section views of column objects are derived from column layers in the structural layering set, and section views of windows or doors in the level of the floor are derived from the "Reusable Objects'.

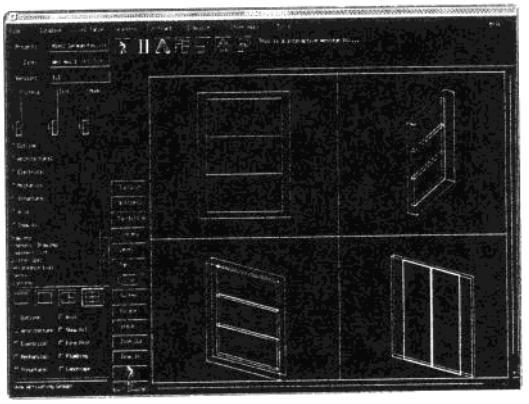

Figure 14 Four Dynamic Representations of Reusable Objects, a door and a window

To reduce the complexity of a design, it is helpful to use standard building elements, e.g., windows or doors, and a few rules relating to their composition. Standard building elements which are defined as "Reusable Objects' are stored in the object-oriented databases in ID'EST. "Reusable Objects" are derived from "template" objects. The "template" objects are defined as object oriented classes to be used as "templates" in creating copies of a generic object. The location of the insertion point of the object is stored in each instantiated data model and, with appropriate projections, the objects are inserted in the drawing. Figure 14 shows a screen display of four different representations of two "Reusable Objects", a door and a window, as a result of different projections. To use these "Reusable Objects" in the building, a user needs to specify a locational value and, if necessary, rotational and parametric values of the object. It is notable that the "Reusable Objects" are different from the library elements provided by conventional CAAD packages as these "Reusable Object" are defined as objects which follow the proposed PADM definitions. Therefore, it is possible to generate dynamic views from these objects according to the user's selection of disciplines in the same way that a building project database maybe manipulated.

As a result of the data classification and projection process, an OBJ data file can be generated. The OBJ file is a projected file of the instantiated data model and has the ability to store semantic as well as syntactic information since it is possible to have structured entities in the file. The display setting can be controlled by the several levels of control module through a well designed interface handler, as the interface system allows selection of desired entities from the instantiated data model.

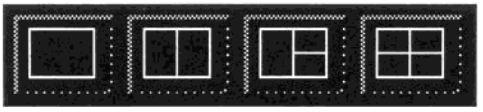

Figure 15 Control Buttons for the Number of Viewing Windows

Figure 13 shows several representations of the building, information relating to which can be accessed and changed from any of the representation windows. Alteration of information affects the non-redundant project database. Therefore, all other representations of the building will be propagated and updated accordingly as the representations of the building are the result of both the data classification and projection process which manipulate the project database. The viewing control buttons shown in Figure 15 provide an easy way to control the number of viewing windows in the screen. By select these buttons, the main drawing canvas will be divided into up to four small=.

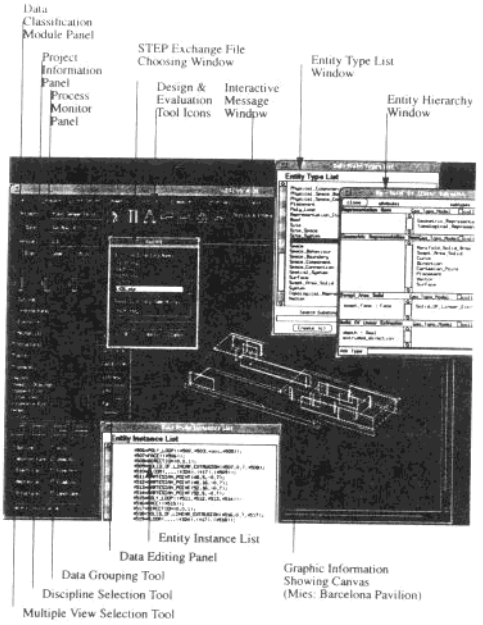

Figure 16 Screen Display of the Data Management System (Mies: Barcelona Pavilion)

The screen display of the entire data managing system with explanations is given in Figure 16.

7 Conclusions

The designer should receive all possible support for the crucial parts of the design work, in the form of collaboration, evaluation and consistency checks. IDEST shows the possibility of supporting a designers navigation through the database of design artifacts; co-operative working between the different disciplines of building design and construction; and interdisciplinary sharing of data and knowledge among project participants. It is also proved that ID'EST could generate dynamic views of representative entities from the nonredundant project database of the prototype building and that each entity in the generated views can be edited. When the edited entities are changed, all relevant data is propagated accordingly. Various design and evaluation tools can now access data from IDEST, thus the behaviour of the design artifact will be more predictable.

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. Currently, IDEST 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 IDEST can be used for evaluation, as it is possible to change schemata and to recompile the data model at any stage. IDEST provides the way to connect object oriented database sysems with conventional CAD systems under a common conceptual foundation which helps to control semantically rich design information more efficiently. To conclude, the authors argue that 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.

8 Acknowledgement

The authors wish to thank the ABACUS unit in the University of Strathclyde for its support of this project.

9 Endnotes

[1] ID'EST is an acronym for an Integrated Design Environment usingSTep methodology.

[21 In this paper, the details of the computer-based design tools are not described. See Kim [1995] for more detail about the integrated computer based design tools in IDEST.

[3] This software tool was developed by NIST [National Institute of Standards and Technology] in the U.S.A to provide a functionality for manipulating and maintaining the individual instances and the list of instances of an instantiated data model according to STEP definitions [Sauder, 19931.

[4] 02 is a distributed object-oriented database management system which can be interfaced to standard programming languages. In ID'EST, 02 has been used to produce partitioned design databases holding multiple design component versions.

[5] In NO 10303, there are two standard 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.

[6] In order to map the building element macro definitions into specific entity type instances, the STEP/DXF Interface makes use of user defined mapping tables. Besides the STEP/DXF Interface, ID'EST communicates through other converters, such as the STEP/OBJ Interface. Issues in relation to mappings between the core data model and an instantiated data model are described in Liebich and Kim [1995] in more detail.

[7] The authors have proved the interplay between the different modules of ID'EST, using an existing building. The Bloomielaw building, a medium-rise office building located alongside the River Clyde in Glasgow, has been used mainly to test the capability of the data management system and the user interface system.

10 References

Augenbroe, G.L.M., COMBINE Final Report JOUE-CT9O-0060', Delft University of Technology, Netherlands, 1993

Belford, G.G. & Santone, A.L., "Object-oriented databases for construction data, In Nahouraii, E. and Petry, F., editors, Object-Oriented Databases, Computer Systems, Los Alamitos, USA, IEEE Computer Society Press., 1991, pp 60-69

Clarke, J.A., & MacRandal, D., and Rutherford, J.H., 'The application of intelligent knowledge based systems in building design,', Technical report, SERC Grant GR/E/18018 Final Report, Strathclyde University, 1989.

Fenves, S.J., & Flemming, U., Hendrickson, C., Maher, M.L. and Schmitt, C., "Integrated software environment for building design and construction," Computer-aided design, 22(1): pp 27-36, Jan 1990.

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, 1992a.

Kim, I., Carnduff, T., Gray, A., & Miles, J., "Object-oriented design system to support concurrent reuse of building and engineering design," to be published In Murphy, J., editor, The Second International Conference on Object-oriented Information Systems, 001595, Dublin, Ireland, November, 1995.

Kim, I., "Design tools integration in an integrated design environment, to be published In B Kolarevic & L Kalisperis, editors, Computing in Design: Enabling, capturing, and sharing ideas, ACADIA 95 Conference, Seattle, USA, October, 1995.

Kim, I., "Data organization and management: In an integrated design environment,' In Mayer, T.W. & Petric, J..editors, Virtual Studio, ECAADE '94, Glasgow, U.K., 1994.

Liebich, T., & Kim, I., "ID'EST: An integrated design environment for management of architectural data," In Milton Tan, editor, Computer Aided Architectural Design Future '95, CAAD Future 95, Singapore, 1995.

02 Technology., '02C Reference Manual,", 02 Technology Ltd., Horsham, West Sussex, U.K., 1994a.

02 Technology., "02 Version Manager,', 02 Technology Ltd., Horsham, West Sussex, U.K., 1994b.

Rutherford, J., "Knowledge-based design decision support,", In Flemming, U., and Skip Van Wyk, editors, Computer-Aided Architectural Design Futures, Proc. of International Conf. on CAAD, Pittsburgh, USA, Butterworths Scientific Ltd., 1993, pp 357374.

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.