17 A Design System for Concurrent Reuse of Architectural Data

Inhan Kim

Dep. of Architectural Engineering, Kyung-Hee University, Yong-In Kun, 449-701, Kyung-Ki Do, Korea

This paper describes a design system which supports the concurrent re-use of existing design information by means of an object-oriented database system. The system manages component versioning within a flexible design environment which is to be used by a design team working on an evolving, complex design. A database of prototype designs has been built with a database system that supports versioning. The basic database operations are then extended with the routines that support inter-designer communication. The database system with these extensions produces a design environment in which designers using partitioned design databases holding multiple design component versions, may concurrently develop new designs. In addition, an expert system shell has been incorporated to deal with design evaluation processes. In this paper, the authors investigate the mechanisms by which existing design versions may be represented, combined and edited to provide new designs.

KEYWORDS: *Data Modelling, Object-Oriented Database, Versioning, Design*

1. INTRODUCTION

It is common practice in architecture to adapt previous designs to meet some new requirement by evolution of the designs. Currently, much of this process is manual although the drawings may be stored on a CAD system. In addition it is usual especially during the later stages of design for a team of people to be involved. If these people work on different components concurrently then there is always a problem of ensuring compatibility and concurrency consistency between versions of the components. Additionally, the design team all have to be aware of the effects that changes in a given component have on the overall design. Consequently, during the development of a design, different design versions will arise and their compatibility has to be managed. Such a management system has to control the versioning of the complete design and the versioning of its components to ensure concurrency consistency.

This study is investigating the necessary mechanisms for design re-use in a computer environment supporting the versioning of designs. Its main objectives are to build a system that supports versioning of engineering designs using an object oriented database, and to identify and implement the mechanisms needed to control inter-designer communications in this design support environment (Kim, 1995).

The prototype system has been implemented using the Eiffel and C programming languages, the O2 object-oriented database system, the CLIPS expert system shell language and X-Windows/Motif on a SUN-SPARC workstation.

2.0 CONCURRENT DESIGN ISSUES

The issues of collaborative concurrent design working are considered in this study. A partitioned design database environment is used to facilitate concurrent design work. With three levels of partition into private, project and public, this database, using check-out and check-in operations, permits concurrent working upon versioned design components by members of a design team (Carnduff and Gray, 1993). Information management and control within this design environment - with the possibility of designers being geographically remote - provides a formal structure which enhances the management of the design process, aids design re-use, and supports versioning of the design.

2.1 Design Data Representations

The representation of design information is a vital aspect of our work. Traditionally in a design office, information about a given design component has been stored in a variety of formats (e.g. text, drawings, computer models) and often in several places. Devising a data model that is able to communicate data effectively between various design stages is essential. By means of this data model, an integrated design system can represent physical features and components of an artefact rather than just graphical primitives.

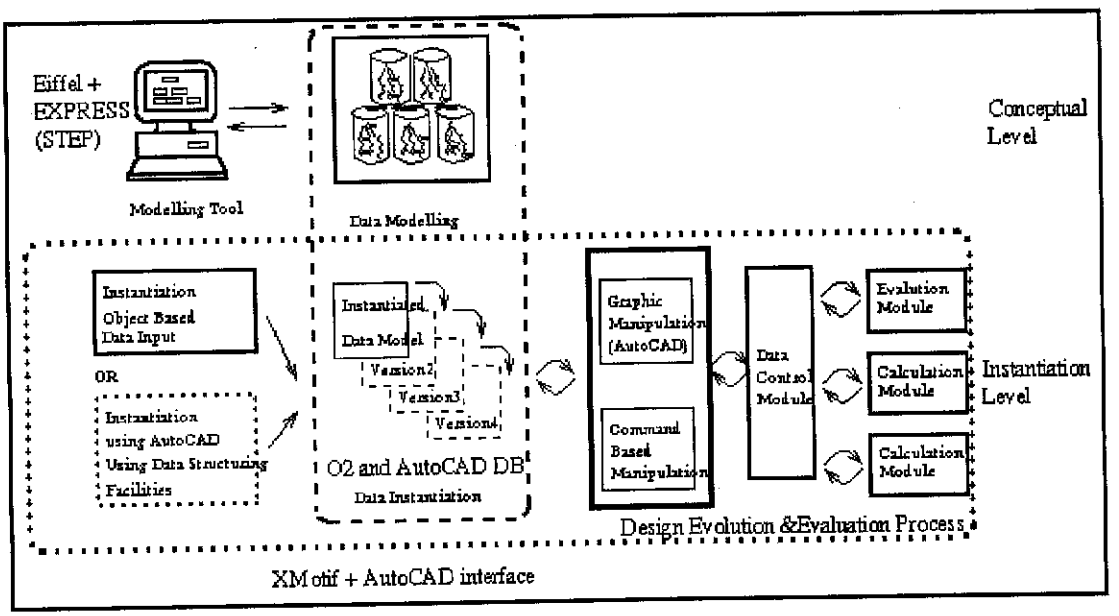

Fig. 1. System Architecture

The prototype system's architecture is given in figure 1. It is divided into two levels:

a) At the conceptual level, the schematic data model is constructed using the data specification languages, EXPRESS (ISO, 1991) and Eiffel (Meyer, 1992). This conceptual data model is a framework in which physical data can be accommodated. In this data model, the design systems can represent explicit and fairly complete definitions of building and engineering data semantics as well as permitting incomplete definitions. Determining the level of detail which can be stored in this way is an important part of the project. For example, if it is possible for such a system to cope with bar bending schedules, then when a component's size is altered, it should be possible to adjust these schedules automatically. This feature is a very attractive and time saving feature particularly in a system aiming to assist in managing versions of systems. There is also a cost estimating unit so that the re-design also includes an automatic update of the cost of the component.

b) At the instantiation level, the data model for holding the design data is constructed. This module helps to create, edit and view information which conforms to the data model. The main objective of this instantiation module is to have an instance manipulation environment for the semantically structured entities defined in the proposed conceptual data model. This level includes the physical database. The use of object-oriented database technology enables such data to be stored in a more consistent and logical manner which greatly facilitates its access for re-design over previous approaches which were based on relational systems (Belford and Santone, 1991). The object-oriented database system O2 (O2 Technology, 1994) has been used by DESCRIBE to produce a design environment in which designers use partitioned design databases holding multiple design component versions. Due to the conceptual data model, the data required for each calculation routine is structured and easily accessible. Using function materialisation, calculation results are readily available in an efficient manner during the design stages.

Most CAD systems are able to create the geometrical representation of building and engineering objects, and allow the addition of more semantic information into the data model (Kim, 1995). This means that their inherent structure to classify data is held in a standardised manner. AutoCAD has been used within the prototype system to deal with the geometrical entities of bridge data. To structure design data and to enable the extraction of appropriately structured design data from AutoCAD, we use three methods. Those methods are:

a) Layers: Layers are mainly used to control visibility. They offer the possibility of grouping sets of data, e.g., all load-bearing columns of a bridge. Many CAD systems still restrict the use of layers so that any entity is only allowed to be a member of one particular layer at a time. It is important our system is not restricted in this way.

b) Macros: Macros (e.g. blocks or parameterised macros) can be used to label and control all geometric entities, that are representations of an intended design object. Thus, they offer grouping mechanisms at the instance level, which is necessary to keep unique identifiers for the bi-directional exchange of meaningful descriptions of design elements. This is a useful feature when dealing with sub-components of the design.

c) Attached attributes: Attached attributes are mainly used to store non-geometrical information on entities or macros. Examples are the material, the building code and optional explanations.

Also the American Institute of Architects (AIA) naming conventions (Schley, 1990) make several recommendations about the classification of graphic data and give the opportunity to map this implicit information into the object-oriented form of data representation.

2.2 Re-use of Existing Design Data

The prototype system supports design versioning as described in (Carnduff and Gray, 1993) and (Carnduff and Gray, 1994), so that multiple designers can work concurrently on various aspects and components of the same design. Complex design objects are versioned at all levels of their composition. This gives rise to the need for configuration management. A configuration is a discrete, numbered collection of versioned objects within a composition hierarchy, which may be treated as a free-standing monolithic version. Any existing computer-based design artefact can be re-used in a new design by encapsulating it as an object with versioning capabilities. Once it has become an object version it can be adapted through a process of specialisation and version merging to the required form for the current design.

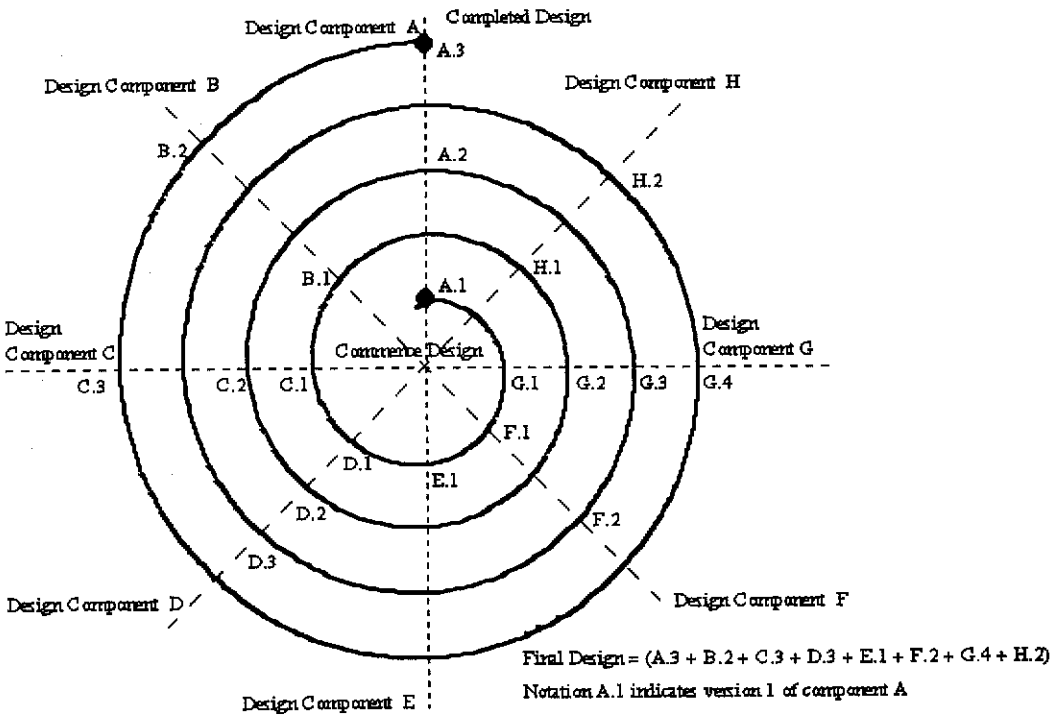

Fig. 2. Design Spiral showing Version Evolution, from (Carnduff and Gray, 1993)

The evolution of a particular design component may progress from preliminary to final design through a series of versions in a design spiral (See Figure 2), where approximate empirical methods in an early version may be replaced by full calculative procedures in a later version and by a directly materialised form near the end of the design. We have found this function materialisation technique to be particularly useful. Function materialisation allows evolving design objects to access the precalculated results of design calculations in materialised function objects (MFOs). This results in substantial time savings in returning functional values,

particularly for objects nearing design completion. The technique captures the dependencies between objects which contributed to the initial calculation of design functions. Should a dependent object change with a resulting invalidation of function results, the implemented prototype system will automatically recalculate the appropriate set of function results and store them in a new version of the MFO if they do not already exist in an MFO.

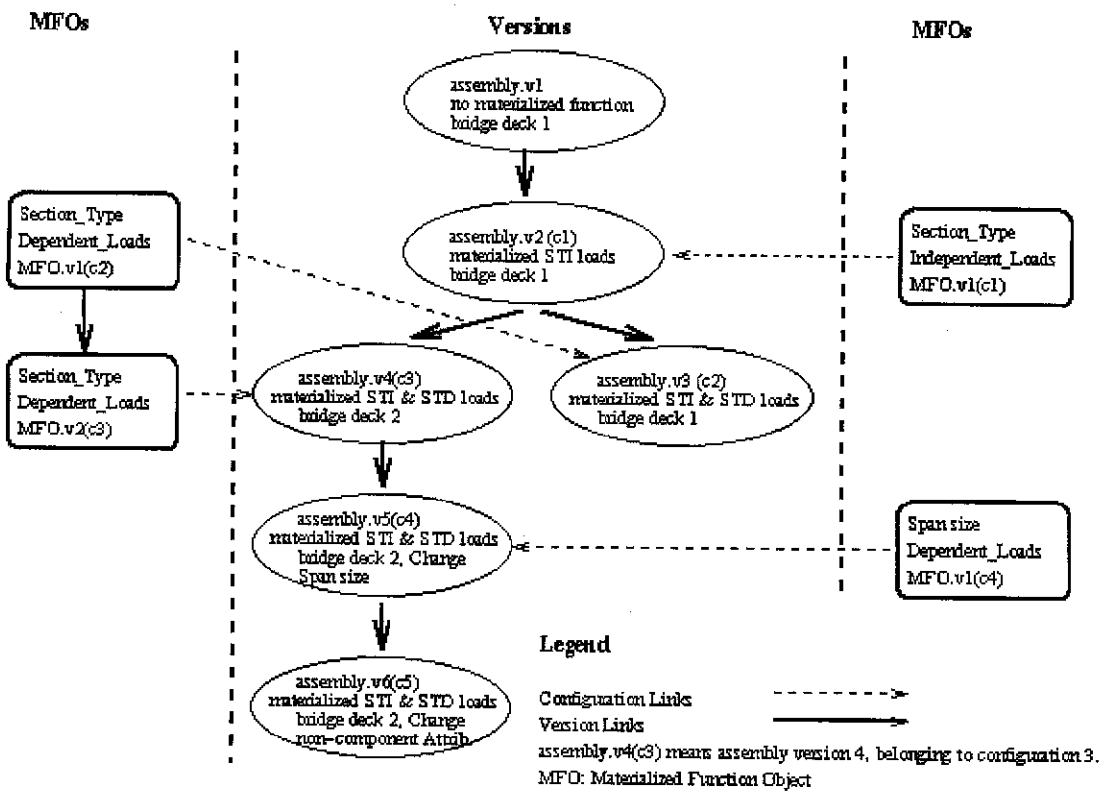

Fig. 3. Evolution of a complex objects with materialised functions

This is illustrated in Figure 3, where a bridge deck artefact and its MFOs can be seen to evolve through several versions. Each artefact version is bound to the relevant MFO versions through configuration management. This approach to versioning can support the use of approximate formulae to provide updated section sizes or areas of reinforced steel. The initial designs and approximations in the prototype system are based on the reuse of these prestored designs and costs. In our version model, the set of versions of an object is managed by a generic version. This generic version maintains the version evolution history and provides the means of access to each of the object versions in the version set, through a dynamic reference. Versioned objects may, however, also be referenced statically without reference to the generic version.

2.3 The Knowledge Based System Module

The growing demand for intelligent information systems requires closer coupling of rule-based reasoning engines with advanced database systems. In particular, active rule-based environments are needed to support operations such as data acquisition, validation, distribution, and management - both for raw data and derivative data.

The prototype system supports the notion of triggers that monitor database events and transactions. These triggers fire induced actions, which perform a variety of critical functions such as maintaining data integrity, monitoring access, and recording volatile information. The CLIPS expert shell has been used to provide these facilities, where the knowledge base rules refer to the design objects. The main task of the shell is to draw inferences from the rules during a consultation by searching for and evaluating them as soon as possible. For example, to check serviceability of a bridge deck, the stresses on each fibre are checked against the limits given in British Standard Part 5400.

Many expert system shells keep all the facts and rules in working memory without any external storage system connection. This can impose a severe limitation when a very large amount of information is to be represented in the knowledge base. It is also desirable to be able to reason with information stored in a database, perhaps even when the information is still evolving. The solution to this problem is to transfer new or updated data periodically from the database to the expert system. In the proposed system architecture, O2 and CLIPS will be integrated to support this type of operation. The interface between them represents a key feature of the design and will allow design data stored in O2 to be transferred to the CLIPS knowledge base.

3.0 TEST CASE

To give the work a focus and to ensure that it is relevant to the needs of practising designers, the system is being developed for the domain of bridge design. The bridge design data have been contributed by Ove Arup and Partners. Ove Arup's involvement also includes helping with the system design and prototype testing.

Beulah Bridge, a single 18 metre span bridge, located in mid-Wales has been chosen as a test case. Detailed design documents that include all of the calculation sheets used for the actual design of this bridge have been provided by Ove Arup. To date, only the design of the bridge deck has been built into the prototype system. A simple diagram of the design process used in this exercise is shown in Figure 4.

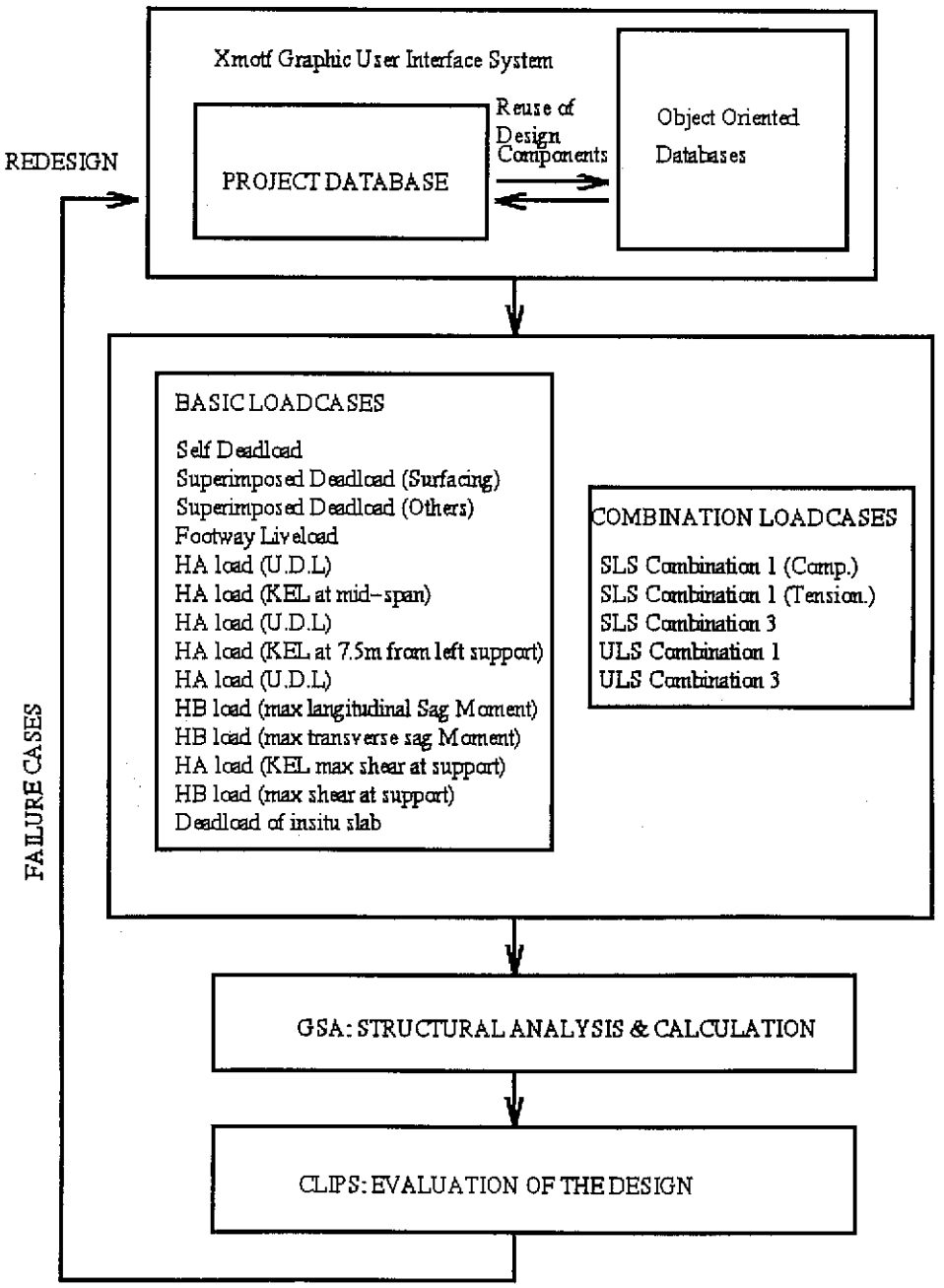

Fig. 4. Design Process in the prototype system

The database is a critical resource for bridge design information systems; all details for design configurations are contained in the database. The initial designs and approximations in the prototype system are based on re-use of these prestored designs. The types of information stored includes structural properties of materials and prefabricated components, which are necessary for the design of a bridge. In the case of the pre-cast, pre-stressed beams, information such as geometric dimensions, section area, I value and elastic modulus have been stored in the object-oriented database and can be easily accessed. Where component dimensions are changed or loadings vary from the original design then it becomes necessary to make modifications. The time and resources required for making such modifications can be

substantial if we use a conventional design approach. Existing CAD representations of engineering designs may be encapsulated within database objects which include both data and procedures, and hence, be given versioning capabilities, the database objects will allow them to contribute to version evolution graphs. A fully fledged system will need additional information to permit the proper analysis of the design, such as the general characteristics and costs of each component.

A program, BRILOAD, is under development to calculate basic loadcases such as dead load and several kinds of live load of a bridge deck. Dead load for each part of a deck can be calculated relatively easily. However, calculating live loads, such as HA load and HB load, involves rather complex calculation routines. In this test case, twelve different basic loadcases have been suggested and calculated. In addition, several combinations of these basic loadcases have to be considered in the design of the bridge to be used as input to a grillage program which analyses bridge decks. Beam-and-slab decks are most conveniently analysed with the aid of conventional grillage programs. This method is generally accepted as sufficiently accurate for design, although it sometimes ignores possible high horizontal shear forces in the slab. In the prototype system, a commercial grillage program, GSA provided by Ove Arup and Partners, has been incorporated to analyse bridge decks and pass the result to the rule-based evaluation module.

The rule-based design evaluation module partially described in preceding sections has a two-stage model that captures the overall design evaluation process, with primary emphasis being given to the characteristics and requirements of the task at hand. The proposed task-centred model is comprehensive in the sense that it covers the serviceability and ultimate limit states from the task identification to, possibly, final construction. The model is also general in the sense that it is possible to apply the same concept to a wider range of construction objects. This model incorporates in a systematic manner all of the critical decisions in any sound bridge deck design. The sequential order of these decisions, and the related phases, is important from a logical as well as an efficiency standpoint. In this design evaluation stage, the presence of the object-oriented database system, O2, is actually hidden from the end-user. The application built on top of CLIPS uses the meta-knowledge deduced from the stored database. The knowledge bases in this evaluation system could include knowledge about design criteria other than stress limits. For example, design costs and aesthetic aspects could be considered and used to rank selected designs.

4.0 CONCLUSION AND FUTURE WORK

The prototype design system will support flexible changes in engineering artefacts' design components that occur when a design team is working concurrently on parts of the design. It will also identify and implement the inter-designer communications protocols and the design version control mechanisms needed in such a computer design environment. Much of the proposed functionality exists in some form in the prototype.

The authors argue that this project produces useful insight into the computing mechanisms needed to control a concurrent design environment based on managing versions of the design, where the design versions are being created by changes in components of the design. The effects of changes in one component have to be communicated to the designer of other components affected by the change. Further research will concentrate on investigating the protocols and semantics of designer to

designer communications and change management, including consideration of the semantics and mechanics of merging alternate version abstractions of design components. The construction and engineering industry will benefit from this project as it makes possible the continuous interdisciplinary sharing of data and knowledge among project participants, from the initial design stage to the final design solution.

5.0 ACKNOWLEDGEMENT

The author wishes to thank the DESCRIBE team at University of Wales Cardiff for its support of this project.

REFERENCES

Carnduff, T. and Gray, A. Function materialisation through object versioning in object-oriented databases, In Proc. of British National Conference on Databases (BNCOD11), Springer -Verlag, pages 111--128, 1993.

Kim, I., Carnduff, T., Gray, A. and Miles, M., Object-oriented design system to support concurrent reuse of engineering design, In John Murphy, editor, The Second international Conference on Object-oriented information systems, IOOS'95, pages Dublin Ireland, 1995.

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.

Meyer, B. Eiffel: The Language. Prentice Hall, 1992.

Belford, G.G and Santone, A.L. Object-oriented databases for construction data. In EzNahouraii and Fred Petry, editors, Object-Oriented Databases, Computer Systems, pages 60-69, Los Alamitos, USA, 1991. IEEE Computer Society Press.

O2 Technology. O2C Reference Manual. O2 Technology Ltd., Horsham, West Sussex, U.K., 1994.

Kim, I. Data organisation and management in an integrated design environment. In T.W. Maver and Jelena Patric, editors, Virtual Studio, ECAADE '94, Glasgow, U.K., 1994.

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.

Carnduff, T. and Gray, A. unction materialisation in object-oriented databases. In S.Patel, Y.Sun, and D.Patel, editors, Proc. of International Conference on Object-Oriented Information Systems, Springer-Verlag, pages 292--305. ISBN 3540 1992 76, 1994.

Miles, J.C. and Moore, C.J. Practical Knowledge-Based Systems in Conceptual Design. Springer-Verlag, London, 230pp edition, 1994.

Philbey, B.T. and Miles, J.C. User interface design for a conceptual bridge design expert system. Computing Systems in Engineering, 4(2):235--241, 1993.