OSCONCAD: A MODEL-BASED CAD SYSTEM INTEGRATED WITH COMPUTER APPLICATIONS

SUBMITTED: July 1998
REVISED: December 1998
PUBLISHED: December 1998 at http://itcon.org/1998/3/
EDITOR: C. Eastman

Farhi Marir, Dr.
OD & KE Group, School of Computer Sc. & Electronic Sys., University of Kingston, UK.
email:F.Marir@Kingston.ac.uk

Ghassan Aouad, Dr.
Department of Surveying, University of Salford, UK.
email:G.Aouad@surveying.salford.ac.uk

Grahame Cooper, Dr.
Information Technology Institute, University of Salford, UK.
email:G.S.Cooper@iti.salford.ac.uk

***SUMMARY**: This paper presents OSCONCAD, an interactive system for integrating CAD and construction related applications to address the problems of design fragmentation and the gap that exists between construction and design processes. It provides a vehicle for storing architectural design information in an integrated construction object-oriented database that can be shared by a range of computer applications. The OSCONCAD model is characterised by several new features. It uses the object-oriented modelling approach to establish standard models for architectural design that comply with Industry Foundation Classes (IFC) for common interpretation of construction design objects and with CORBA (Common Object Request Broker Architecture) for distribution of the objects amongst the construction applications. It aims to achieve independence from the display environment by providing a set of Abstract Factory and Abstract Design Classes, which provide abstractions that the design model classes can use to draw and render themselves in any display environments. More importantly, graphical and textual information about the building design components is directly saved as instances in an object-oriented database without passing through the existing CAD databases. To demonstrate the independence from the display environment, two applications using OSCONCAD models are implemented. The first is an interactive AutoCAD application, which creates instances of the OSCONCAD design model and stores them directly in the distributed object database. The second A web-based application using VRML (Virtual Reality Modelling Language) for remotely interrogating information stored within the integrated database, visualising and manipulating the design components in 3D environment. Also, to demonstrate the feasibility and practicability of the OSCON (Open Systems for Construction) object-oriented product model, three OSCON construction applications that access and share the OSCONCAD building design instances are presented.*

***KEYWORDS**: Distributed Construction Object Database, Model-Based CAD System, CORBA, IFC, OSCONCAD, UML, and VRML.*

1. INTRODUCTION

The building industry consists of a vast range of people with different views, skills, and knowledge of the construction process that work together by custom and practice within a culture evolved over several centuries. Over time, the process of design and its associated support has become divorced from the manufacturing of the product (i.e. building) for which it exists. This creates problems for the organisation of both the design and construction processes caused by the large number of interfaces. Communication becomes difficult and this results in breakdown, misunderstanding, frequent litigation, time consumption and additional costs to the project. To solve these problems, many companies have selected their own CAD systems and applications and have developed specific ad-hoc interfaces between them. Many of these systems have been utilised effectively within object-oriented building application areas to form local "islands of automation". However, in the Architecture, Engineering and Construction (AEC) industry, it is usually more productive to integrate information along the paths through which work flows between project stages or specialists. This is supported by research in the construction domain, which has shown that there is a general agreement for the need for integrating design and construction (Adams, 1989, Ferguson, 1989). The 1994 Latham report (Latham, 1994) has also recommended the development of systems which aid a more harmonious relationship within the design/construction team.

However, the main difficulties of integration are exacerbated by the fragmentation of design information. The drawings are expected to provide the experts from different areas of interest the information they require and serve as the main medium for integration. Despite the fact that computer-aided design (CAD) systems are extremely powerful, they are not being utilised thoroughly in the AEC industry. Specialists in the AEC industry have used today's CAD systems as simply automated drafting tools, thereby automating their own narrow areas of specialisation. Each participant uses unique drafting conventions and their own CAD systems. Information is scattered about the project in an uncontrolled and uncoordinated way, on a variety of information systems and media, so that the design cannot be viewed as a complete entity. Such obstacles to the free flow of information between parties to the construction process lead to data re-entry and the consequent inaccuracies, which prejudice future design 'migration paths' and operational flexibility. These are responsible for many of the quality problems and for adding to the costs of construction projects.

This paper presents the development of OSCONCAD system, which addresses the design fragmentation problems and provide a vehicle for storing architectural design information in an integrated object-oriented database shared across a range of computer applications developed within the OSCON (Open Systems for CONstruction) project. One of the main aims of the OSCON project is the assertion of construction industry requirements for an 'integrated solution' comprising common definitions of objects and functions, supported by software applications written for dedicated use in the sector. OSCON achieves this integration by establishing standard object-oriented models that all applications can adopt and share.

From these models, the integrated object-oriented model used by OSCONCAD has been developed. The OSCONCAD models have been designed to comply with the Industry Foundation Classes (IFC), a major industry standard data model, which is developed to be adopted as the standard of real world construction objects (IFC, 1997). The use of IFCs allows a common interpretation of these objects and the construction-related applications using such objects do so in a consistent and interoperable manner. Moreover, to share objects (function and data sharing in an integrated way) distributed amongst applications, the OSCONCAD design model will be complying with the CORBA (Common Object Request Broker Architecture) standards.

The content of the paper is organised as follows. The second section reviews and analyses related research in the area of design-construction integration. The third section is devoted to the description of the features of the OSCONCAD system. The fourth and fifth sections present two front-end applications used to display instances of the OSCONCAD distributed object database. The first is interactive AutoCAD application that allows users to create and interact with instances of architectural components of a building. The second is a web-based application that uses VRML (Virtual Reality Modelling Language) for remotely interrogating information stored in the object database, and visualising and manipulating the design components in a 3D environment. The sixth section gives a brief description of three OSCON construction applications that have been developed to share and manipulate the building design instances of the OSCONCAD object database. This section includes a prototype for estimating and planning, a piece of wrapper software for CA-Superproject®, and an application to support project managers in managing communication and monitoring the progress of different phases of the construction project. The last section presents the conclusion of this project.

2. RESEARCH IN THE AREA OF DESIGN CONSTRUCTION INTEGRATION

A great deal of work has been carried out in the area of information modelling to support the sharing of information across participants in a construction project. Efforts in this area include projects such as ATLAS (Bohms et al., 1994) (for large scale engineering), COMBINE (Dubois et al., 1995) (for HVAC and building design), RATAS (Björk, 1994) and ICON (Aouad et al., 1994) (for building design and construction management). It is also the subject of standardisation efforts such as STEP (see: http://www.nist.gov/sc4/www /stepdocs.htm), and more recently the IAI (see: http://www.interoperability.com/). Currently, there exists no accepted way of sharing information between the different software applications used in a construction project.

An architect may use a CAD package to produce a set of drawings. A quantity surveyor (QS) will then use these drawings to produce costing information. A construction planner will also use the drawings to produce a set of construction activities in the form of a network or Gantt chart. Participants use their preferred software package, maintaining their own subset of the project's information. In some cases information is exchanged in electronic form. For example, QS may also use a CAD system to simplify take-off quantity from the architect's drawing. However, this type of data exchange takes place at a very low level (e.g. AutoCAD DXF files) with each participant having to reinterpret the data for his or her own purpose. This approach, although well established, does have a number of disadvantages:

- The existence of 'information lag' between participants,
- occurrence of errors when re-entering data,
- The difficulty in gathering an overall view of the project for management purposes, and
- The difficulty in integrating software packages due to the lack of a common conceptual information model.

The integration of software in the construction industry has occurred mostly through the development of links between stand alone commercial packages. For instance, integrated planning/estimating and integrated planning/CAD systems have been developed by individual researchers (Aouad, 1991, Tah et al., 1991, Howard, 1991). Another form of integration has been achieved through product modelling undertaken in Scandinavia, France, USA, The Netherlands, and the UK (Bjork and Wix, 1991). Integration through the establishment of standards for data exchange is also progressing under various groups: EDICON, AEC STEP Committee, etc. (Bjork and Wix, 1991; Brandon, 1993). Recently a group of software companies and users have come together to form the Industry Alliance for Interoperability. Their goal is to define specifications for a set of Industry Foundation Classes (IFC) and to deliver them promptly to end users, incorporated into commercial products (IFC, 1997).

In the last few years, researchers and practitioners have started to develop interfaces between CAD systems and relational and object-oriented databases that support the design process. Atkin and Gill at Reading University have configured an experimental integrated system to provide information for management purposes, besides producing drawings (Atkin and Gill, 1986). This system is based on the idea of integrating computer-aided design and relational database management systems. Details of specifications, costs, duration and resources are stored in the database system, which accepts graphical data from the CAD model and relates them to their associated non-graphical attributes. It is suggested that design practices should be modified to adopt a component-based approach and that the computer technology is available now to develop new approaches to the management of construction using CAD systems (Atkin and Gill, 1986).

An interface for an integrated plant model was developed (Zabilski and Hall, 1989). In this system, the capabilities of 3-D solid computer models are combined with the efficiency of a cost and schedule database to plan and schedule the construction process. BUILDER is a prototype system, which generates construction schedules from architectural drawings using KEE running on a LISP machine (Cherneff et al., 1991). BUILDER involves the development of an object-oriented CAD program that captures building component data as the graphics are created by making use of the KEE-Pictures utility embedded in the KEE system. Data generated by KEE-Pictures become an object-oriented model within the KEE environment. The object model serves as input to knowledge based planning and scheduling system implemented in KEE. While exploring the capabilities of KEE to create a CAD system within an artificial intelligence (AI) environment is innovative and interesting from the research point of view, utilising a non-standard CAD system running on a LISP machine is both uncommon and expensive in the AEC industry (Kartam, 1994). Timberline Corporation in the US, has developed a link, known as CAD integrator, between AutoCAD and their Precision Plus estimating system (Timberline, 1990). The CAD integrator performs take-off directly from AutoCAD drawings. It then moves the information into Precision Plus, an estimating spreadsheet-based program. Once estimating has been done, information can also be tied directly to Primavera and Microsoft Scheduling products to give more precise job planning. Another system, called CADlink, has been developed as an interface for translating a CIFECAD CAD model into an object-oriented database. CIFECAD is a component-based customised AutoCAD application from which the CAD link interface transfers blocks representing components (Ito et al., 1989). Kartam has developed an intelligent and customised CAD interface known as ISICAD (Intelligent Computer-Aided Design System) that is capable of generating an object oriented model during design

(Kartam, 1994). This model uses a component-based approach. The system has been implemented within the AutoCAD 12 environment. It integrates a CAD system with databases, to incorporate project data, and expert systems to evaluate the evolving design solution and resolve conflicts that may arise as a direct result of decisions made by the designer. The emphasis in this system is more on creating an intelligent design environment for the architect than on integrating structural design and construction. In the COMBINE project (Dubois et al, 1995) which relied on STEP-based data exchange integrate both Microstation and AutoCad systems. It developed its own data model of a building and integrated applications to it, without relying on CAD formats. Alshawi at Salford University developed SPACE (Simultaneous Prototyping in An integrated Construction Environment), an environment that allows the integration of industry standard software across a construction project’s lifecycle (Alshawi, 1996) using similar approach to that taken in the ISICAD system. A similar approach has been taken in the DICE (Distributed and Integrated environment for Computer-aided Engineering) project (Sriram and Logcher,1993, Sriram et al., 1995, Gorti et al.,1996). DICE has addressed the co-ordination and communication problem in engineering. It is designed as a network of agents or Knowledge Modules (KMs) which communicate through a shared workspace.

3. OSCONCAD SYSTEM

Most of the research projects mentioned above have developed a user interface in AutoCAD to capture building component data and then use different approaches to translate the CAD representations into an object-oriented building product model. This product model has been used as an input to other computer-based applications. However, several problems can be enumerated within most of these systems:

- Some of the systems are developed to be fully dependent on a specific CAD system and this limits the possibility for reuse and therefore requires the rewriting of most of the CAD functions of the system to suit a different display environment.
- Most of the systems keep translating drawing data from specific CAD databases into the object-oriented product model they provide. This process of generating the necessary input for other construction-related activities from AutoCAD is expensive and complicated, requiring several transformations and involving several computer languages (Ito, 1991).
- Some of them are designed around the concept of the AutoCAD "blocks" that is defined for each type of design element, e.g. wall, door, column etc., and the user of the CAD system builds up the design using instances of these blocks. This approach faces problems when it comes to representing the relationships between design elements, such as the connection of one wall to another, or the appearance of a window in a wall.
- To support data-exchange between different computer applications, most of the above systems are developed based on CAD standards like IGES, and DXF for graphical data. However, both of these standards impose restrictions on the data types and formats that they can exchange. This restriction is an obstacle to the integration process and encourages direct exchange of graphical data via customised one-to-one interfaces that requires tedious manual re-entry of data across diverse applications.
- Most of existing integrated systems concentrate mainly on the problem of integrating applications via data sharing and none provides distribution of objects among the different applications.

The ICON project adopted a top down approach based on a hybrid information and object-oriented approach to establish the principles for integration (Cooper et al., 1994, Aouad et al., 1995). This integration is based on applications independent, object-oriented database and architecture for providing access to that database through a variety of common and proprietary application software packages. The OSCON project builds on the ICON approach to address the limitations mentioned above. This is reflected in the OSCONCAD system, which adopts a model-based approach to capture graphical and textual information about building components and directly storing them into an object-oriented database, rather than the complex and ill-structured CAD databases. This approach avoids the expensive process of translating drawing data from the CAD database into an object-oriented database. Moreover this approach is enhanced with the concept of separation of the design data from the specific CAD package used to manipulate it. A consequence of this is that new CAD products will be much easier to incorporate and also other non-CAD applications can have direct access to the design data from such a database. In addition, OSCONCAD models were designed to comply with the IFC (IFC, 1997) to facilitate data exchanges between construction-related applications and to allow common interpretation of the design objects amongst the applications in a consistent and interoperable manner. Concerning project data interchange, the OSCONCAD system does not only concentrate on the problem of integrating applications via data sharing but goes further into sharing and inter-working between objects distributed amongst applications by complying with CORBA standards (CORBA, 1995). The following sections will show details of three important aspects addressed by the OSCONCAD model:

- An object-oriented database for CAD drawings,
- The independence of the architectural design object from the display environment, and
- The use of the IFC and CORBA standards to promote collaborative working environment.

3.1 Modelling Methodology

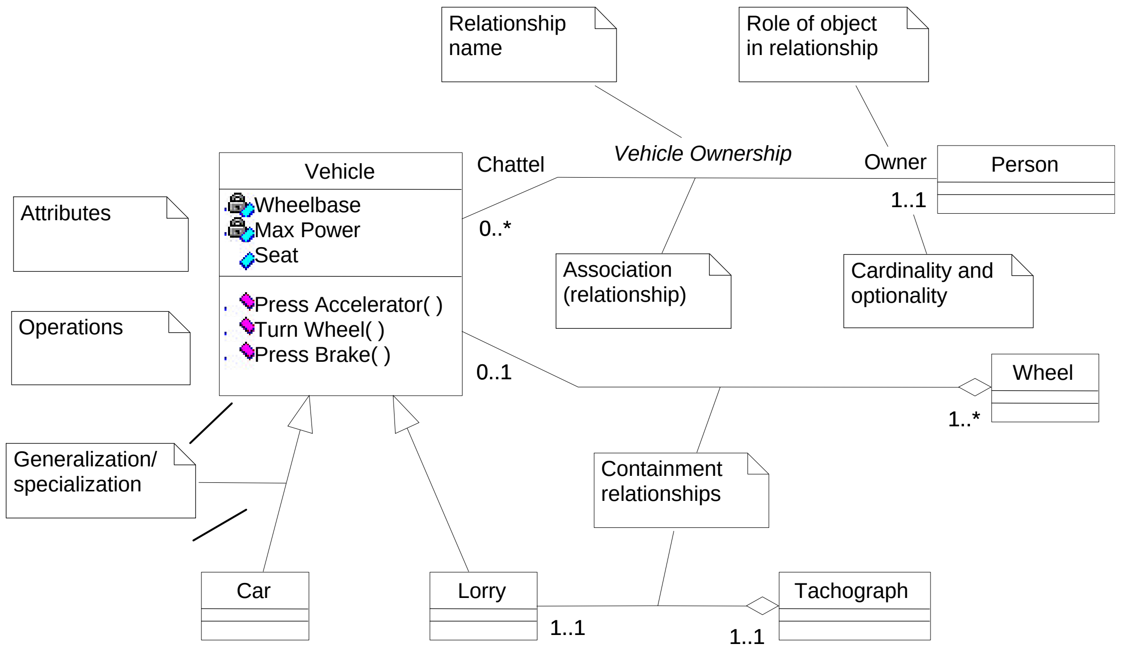


FIG. 1: Illustration showing the main elements of a UML class diagram. Labels show the main elements of the language. In the example, a class "Vehicle" is shown with attributes and methods. This class is specialised into subclasses "Car" and "Lorry". Two type of relationship are shown: association (between Vehicle and Person, and containment (or composition) between Vehicle and Wheel.

The first step in using common models for computer-based applications integration is to establish a common data model or modelling language-that is, a standard paradigm for representing and communicating information in general. The OSCONCAD project is committed to a fully object-oriented approach, in which an emphasis is placed on inter-working between software objects rather than on sharing of data using common formats. This approach provides a number of advantages in terms of the ability to use abstraction to handle complexity (Ahmed et al., 1991, Cooper, 1995, Booch, 1994):

- It provides superior facilities in terms of abstraction and rigour,
- It supports the creation of "intelligent" project databases, since project data can be stored as objects combined with code that responds to events and reacts to context. For example, a "column" object can inform other objects that its dimensions have changed or can recalculate its strength if alerted to a change in the concrete strength, and

- It provides a methodology for encouraging reuse of design elements and for capturing and reusing best practice in design and construction.

The current version of the OSCONCAD model is being re-built using UML (the Unified Modelling Language). UML is rapidly being established as a de facto standard for object-oriented modelling, and is well supported by the Rational Rose CASE Tool. UML is developed primarily from two of the most popular modelling formalisms for Object-oriented modelling: OMT (Rumbaugh et al., 1991) and Booch (Booch, 1994), and currently being adopted as an international standard within the Object Management Group (OMG). The models shown in this paper are expressed in the UML. Fig. 1.illustration showing the main elements of a UML class diagram that is used in this paper. Labels show the main elements of the language. In the example, a class “Vehicle” is shown with attributes and methods. This class is specialised into subclasses “Car” and “Lorry”. Two type of relationship are shown: association (between Vehicle and Person, and containment (or composition) between Vehicle and Wheel. Further information on UML can be found in (UML, 1997). The use of CASE tools like Rational Rose is helpful because the tool can automatically generate both C++ code and CORBA IDL (Interface Definition Language).

This allows the OSCONCAD implementation immediately to take advantage of the inter-working facilities provided by the CORBA.

3.2 OSCONCAD Architectural Design Model

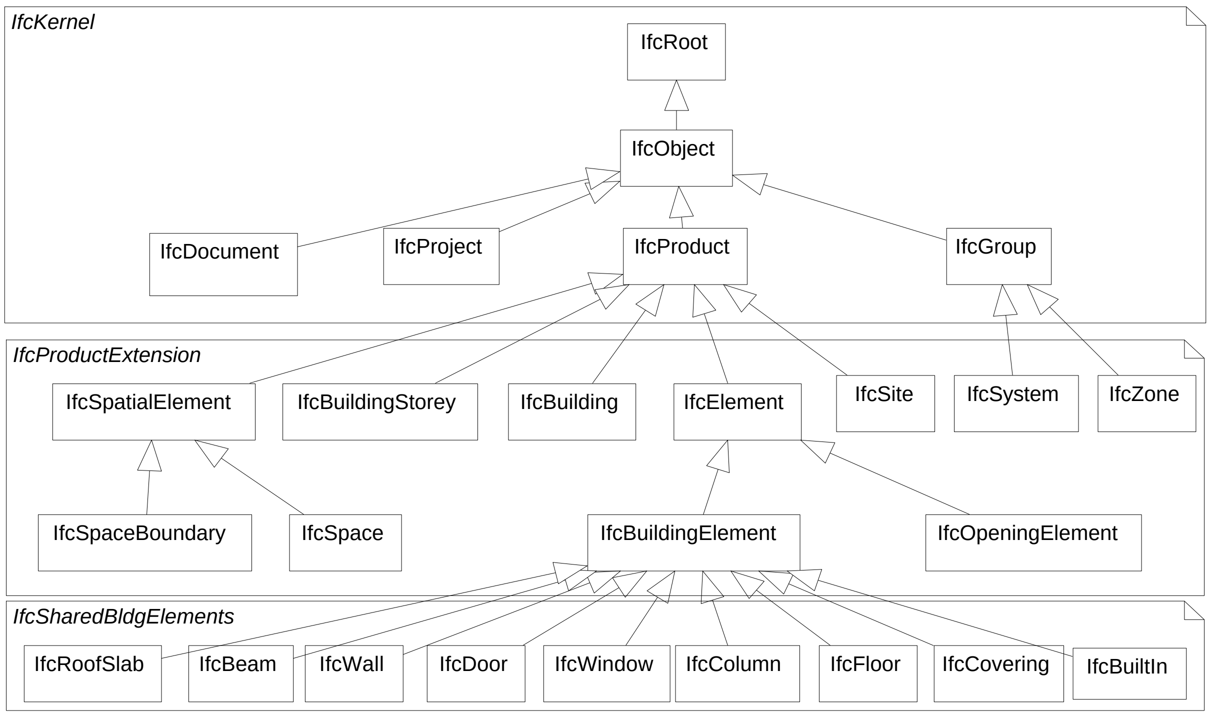


FIG. 2: A subset of IFC1.2 Objects.

The root cause of the different problems in construction projects is the information or data upon which all participants depend. Because of fragmentation within the industry, there are many different interpretations of the semantics of the data in use. The formal ways in which to describe information semantics are referred to as models, which represent a formal description of a view of a domain of interest. A model is always a representation of the real world to some acceptable degree of approximation for the viewpoint we hold (Wix and Storer, 1996). In practice, the word ‘modelling’ gives rise to many interpretations and the solution lies in the fact that there is only one actual product to be conceived, designed, manufactured and maintained. Therefore, there is a need for only a single model of the real world product from which the other consistent representations can be derived and through which different presentations are generated as user views. In the ICON methodology, this concept is referred to as a "canonical model" (Cooper et al., 1994) and also termed a "product model" in (Wix and Storer, 1996). Modelling design information is the most debatable aspect of designing the product model. It involves dealing with graphical and non-graphical data and many complex issues such as representing form, function, topology, appearance and many other aspects. Major efforts have been undertaken by researchers in the STEP community and outside it in order to define a generic model that can describe a building. The Technical Research Centre of Finland has made multiple studies about a building product-model (RATAS) using relational databases and hypermedia (Pentilla, 1989). Several researchers have suggested models for space and space enclosure and have also developed object-oriented product models that could be used to store richer kinds of data and knowledge about a product, including knowledge about its design, manufacturing and operational parameters (Svensson and Aouad, 1997).

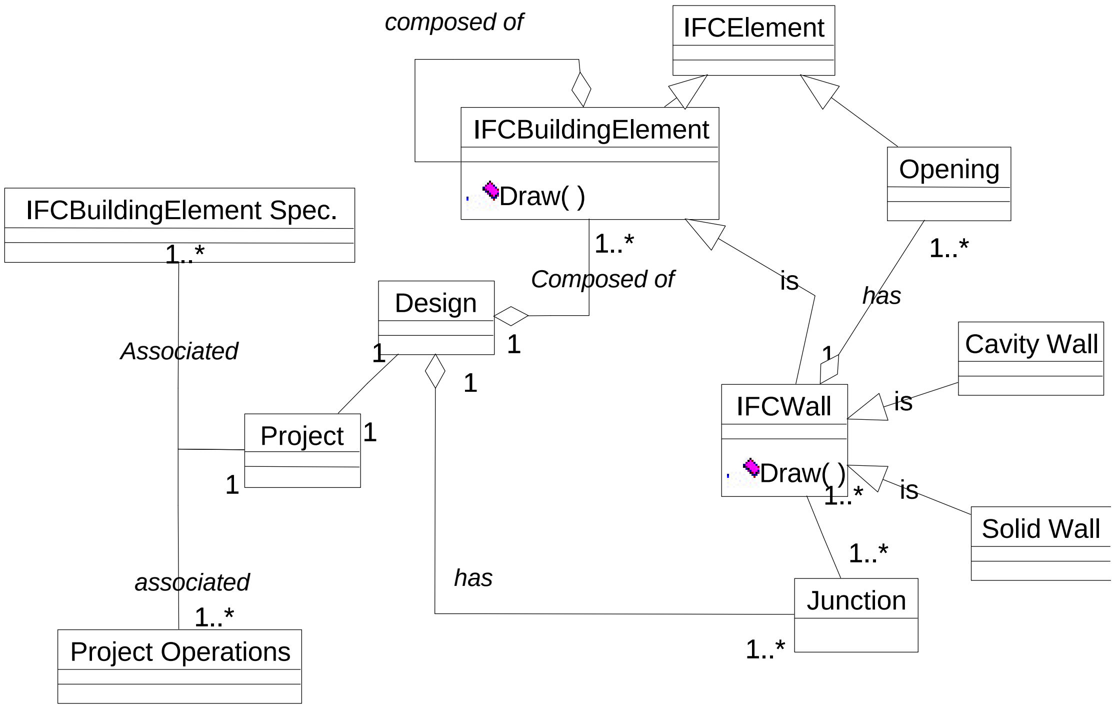


FIG. 3: A subset of the OSCONCAD architectural design model.

None of these models has a universal acceptance. This is due to the fact that a data model must be readily accessible to all construction-related applications and users. It must also be compatible with systems across AEC and other industries (Froese and Paulson, 1994). Recently a group of software companies and users have come together to form the Industry Alliance for Interoperability and define specifications for a set of Industry Foundation Classes (IFC, 1997). "Fig 2" shows a subset of the classes of the IFC 1.2. The IFC is an industry object model, proposed as the standard for construction integrated systems. By its compliance to the IFC, the OSCONCAD models definition allows for a common interpretation of real world construction objects, whereby software applications using such objects do so in a consistent and interoperable manner. In this model, an object will incorporate physical and non-physical information such as weight, colour, electrical or thermal properties, cost, as well as a definition of how it should appear in space (from which applications software can derive an appropriate display representation). An object will also encapsulate procedures that can be used to describe how it should behave and relate to other objects. Thus, for instance, an object describing a light switch might give its height above a floor and the distance from a door and an object describing a window might 'cut itself' into a wall and create cavity closures and lintels automatically (IFC, 1997). The aim of the OSCONCAD system is not to recommend another design model but to derive classes from the IFC Generic classes and enhance them with knowledge about the detailed design stage that can be shared with other applications, such as cost and time planning. The detailed design model is a description of the design components and their specifications with very limited topological properties incorporated as shown in Fig. 3. The model offers the representation of several structural members:

- Foundation: Raft, pad and strip.
- Slab: Ground slab, floor slab
- Columns,
- Beams,
- Walls: solid and cavity walls,
- Roof, and
- Non-structural members, such as Windows and Doors.

3.3 The Architecture of the OSCONCAD System

The OSCONCAD Architecture adopts the Client/Server approach as shown in Fig. 4. The server is composed of the OSCONCAD objects. Two kinds of objects are provided:

1. Objects that reflect the database schema of the OSCONCAD architectural design model where instances of the design component are stored and
2. A set Abstract Design Classes, which provide abstractions that the design model classes can use to render themselves in a given display environment.

The AutoCAD and VRML front-end applications are discussed in individually in the fourth and fifth sections. They use the Abstract Design Classes to derive concrete classes to display instances of the design in the AutoCAD and VRML environments. However, the cost estimator (Esteem), the project manager (Mentor) and the project planner (Planner) applications, shown in Fig. 4., are discussed in the seventh section. These client applications are designed to access and manipulate instances of the specific domain and the OSCONCAD design distributed objects through CORBA Object Request Brokers.

3.3.1 The OSCONCAD Distributed Design Objects

AEC CAD systems date back to the early 1960's (Port, 1989). However, after thirty-five years of research and development, these systems are still considered as drafting rather than design packages. This is also the case in the construction industry, where most available AEC CAD systems are used as drafting tools. Regular CAD drawings are usually simple graphic representations of projects. They are collections of points, lines, and other primitives that visually portray project components and have no meaningful information associated with them. These drawings, however, have the advantage of offering maximum drawing flexibility. Primitives can be easily copied, deleted, stretched, or edited in a variety of ways. Taking the advantage of drawing flexibility, the OSCONCAD adds meaningful information by grouping these drawing primitives into objects with architectural design information associated to them. The objects are encapsulations of a set of operations, which are invoked externally from custom applications using the OSCONCAD model. They allow the existence of two different views of the drawing. The graphic view, which is the usual three-dimensional CAD drawing, and the textual view which is the collection of all design information captured. These objects required by the detailed design stage revolve mostly about concepts, such as components and component specifications. The component specifications within a design model play a major role in defining design information and ultimately in generating cost and time data. Also, as objects, design components are handled as single entities, making them direct representations of the elements being symbolised. Besides making the association of design information possible, objects offer the potential for speeding up the design process, particularly in the later stages, since they can be made readily available to the user by the suppliers of building products, or by third party software suppliers.

At present, the data model of object-oriented systems are dictated by their underlying languages, databases, or programming environments. However, several major object-oriented data model standards currently are being developed. These standards provide all the characteristics needed to support integrated systems, and most commercial object-oriented databases are likely to adopt these standards as their fundamental data models. The OSCONCAD object model has adopted the CORBA (Common Object Request Broker Architecture) standard (CORBA, 1995) developed by the Object Management Group (OMG), an international organisation of information systems vendors, users, and researchers dedicated to promoting industry standards. By adopting CORBA standards, the OSCONCAD system allows inter-working between object-oriented software components within networked computing environments (Atwood, 1991). The compliance of the OSCONCAD objects with CORBA standards will allow not only the sharing and interchange of design data with linked applications of the integrated framework, but also the sharing of objects distributed amongst all the applications. To achieve this aim, CORBA IDL (Orbix, 1996) interfaces have been written for each of the OSCONCAD objects. The use of the IDL neutral interface for the OSCONCAD objects promotes language and platform independence, location transparency, modularity, and robustness of the system, which facilitates the integration of other construction applications that access and manipulate the OSCONCAD distributed objects.

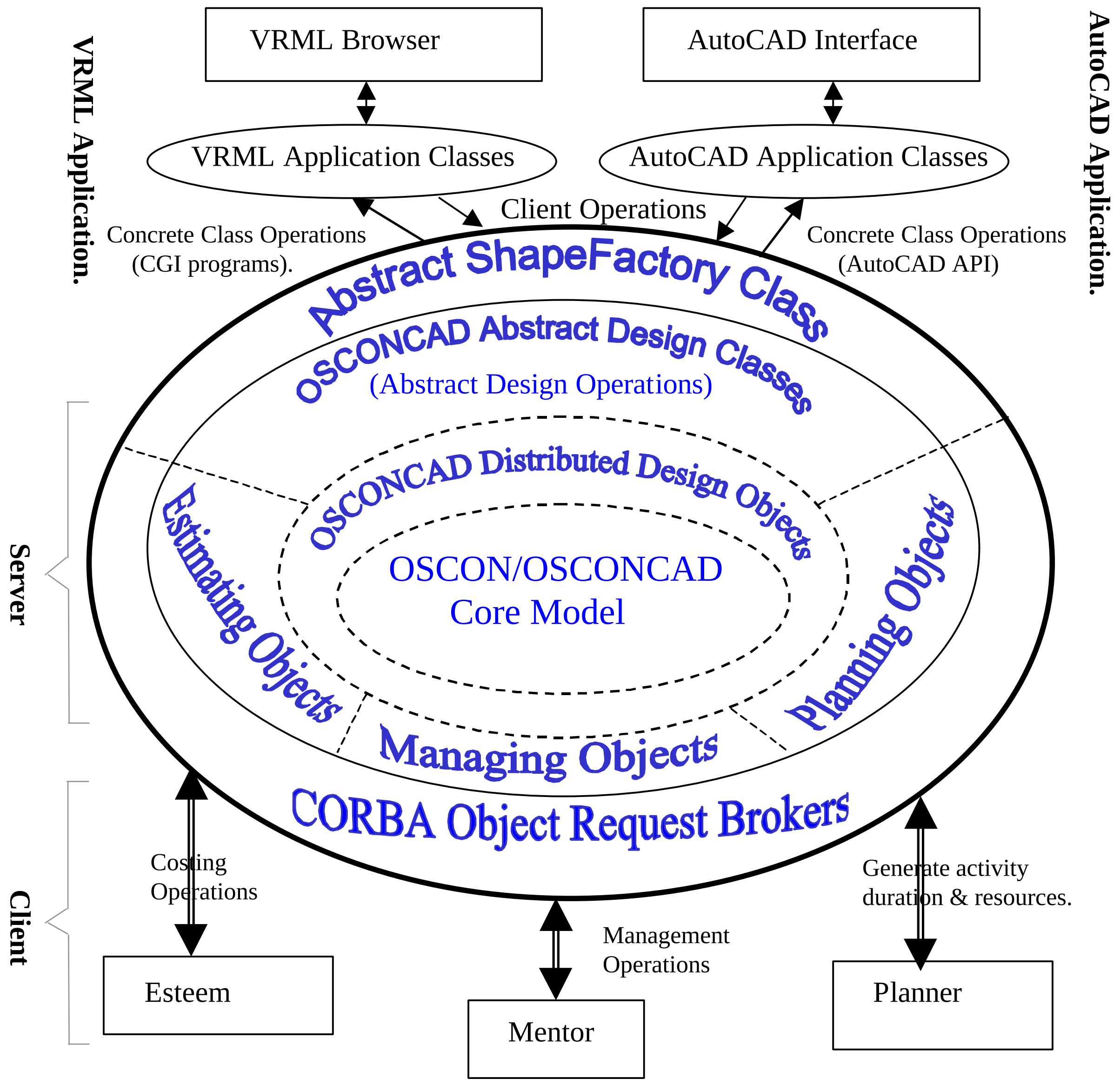

VRML Browser
AutoCAD Interface
VRML Application.
AutoCAD Application.
VRML Application Classes
AutoCAD Application Classes
Client Operations
Concrete Class Operations (CGI programs).
Concrete Class Operations (AutoCAD API)
Abstract ShapeFactory Class
OSCONCAD Abstract Design Classes
(Abstract Design Operations)
OSCONCAD Distributed Design Objects
OSCON/OSCONCAD Core Model
Estimating Objects
Planning Objects
Managing Objects
CORBA Object Request Brokers
Server
Client
Costing Operations
Generate activity duration & resources.
Management Operations
Esteem
Mentor
Planner

FIG. 4: OSCONCAD Architecture

3.3.2 The OSCONCAD Abstract Design Classes

The OSCONCAD model is designed to be generic and abstract enough to allow separation between the design model and the specific implementations of commercial CAD tools. For this it uses the Abstract Factory Design Pattern which provides an interface for creating families of related objects without specifying their concrete classes (Gamma et al., 1994). As shown in Fig. 5., the OSCONCAD system defines:

- an Abstract ShapeFactory Class that declares the interface for creating each basic kind of design element,
- an Abstract Design Class (IfcBuildingElement) that represents an abstraction for each kind of architectural design element and provides abstractions that the OSCONCAD design model classes can use to draw and render themselves in any CAD environment
- a set of Concrete Subclasses of the ShapeFactory Class which implement the design elements for a specific CAD or graphical display environment. These subclasses can be used by the model classes to 'draw' themselves without knowing which application is actually doing the drawing. In OSCONCAD application the AutoCAD_Drawer and VRML_Drawer are the concrete classes designed specifically for the AutoCAD and VRML applications respectively, and
- a set of client classes that encapsulate the commands the users will want to issue. Instances of these classes can then be used to implement command functionality in an environment independent manner.

The ShapeFactory's interface has an operation that returns a new design object for each Abstract Design Class of the design component. The client class (from the user application) uses the IfcBuildingElement Abstract Design Class, which calls operations to obtain instances of a design element and draw themselves without being aware of the Concrete Classes they are using. In other words, the client class has to commit to an interface defined by the Abstract Design Class, not to a particular Concrete Class.

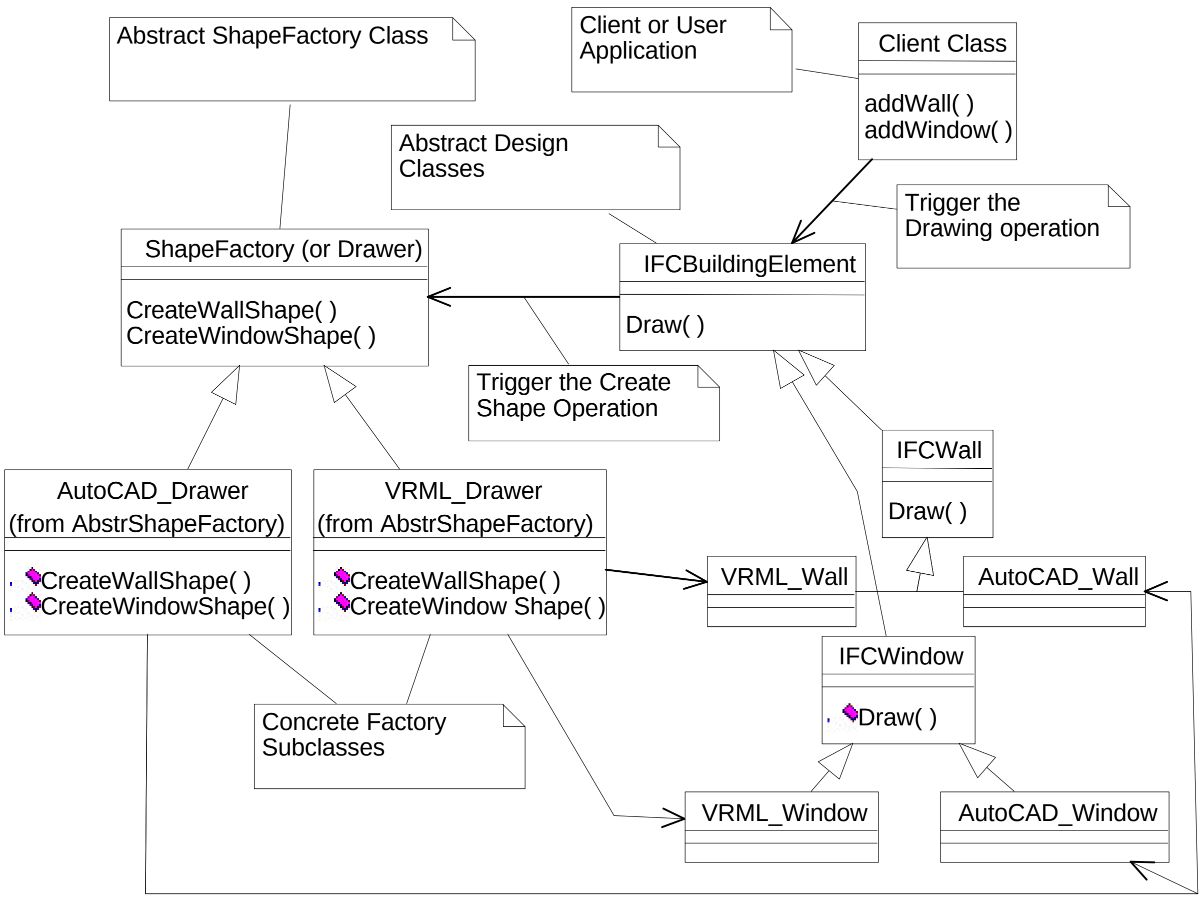


FIG. 5: The OSCONCAD Abstract ShapeFactory and Abstract Design Model.

In practice, all classes in the OSCONCAD design model inherit from the base class *IFCBuildingElement* that defines a common interface. This interface includes a *draw()* operation which is redefined in subclasses to allow instances of the design classes to draw themselves in a given display environment. The *draw()* operation passes a pointer to the ShapeFactory (or Drawer) object which encapsulates a set of simple 3D drawing operations as virtual member functions. These functions are implemented in the ShapeFactory concrete subclasses *i.e.* *AutoCAD_Drawer* and *VRML_Drawe*r, to display the drawing in specific environments. As a result of this modelling approach the design of building components stay independent of the prevailing CAD and other display environment and the AutoCAD 13 and the VRML applications discussed below become graphical front ends for the instances of the OSCONCAD architectural design model.

4. THE AUTOCAD APPLICATION

The OSCONCAD Application is an AutoCAD Application Development System (ADS) prototype, which is loaded by a user of an AutoCAD session. It allows the user to create and manipulate architectural components of a building. The created components are stored as instances of classes of the OSCONCAD design model in the OSCONCAD object database through which they are made available to other AEC computer-aided applications in computer-interpretable and standard form.

The application provides a simple environment for designing building components. The design primitives are walls, windows, doors, columns, beams, foundations etc. As design objects are created through the AutoCAD interface, corresponding objects are created in the object-database. The information that is used to draw these objects on the AutoCAD screen via an instance of the *Drawer* class comes directly from the instances in the database. This ensures that the screen reflects the design database. This also means that changes in a design can be immediately propagated to other areas of the project database. For instance, if an on-line project-planning package were accessing the database then it would be possible to change a part of the design and immediately see the impact these changes would have on the project schedule. The prototype provides the following functionality:

- creation and manipulation of foundations, walls, windows, doors, floors, stairs and roofs,
- creation of Space Diagrams and expansion into building elements,
- creation of specifications and specification sets for application to a design, and
- texture mapping of building elements based on component specifications and finishes

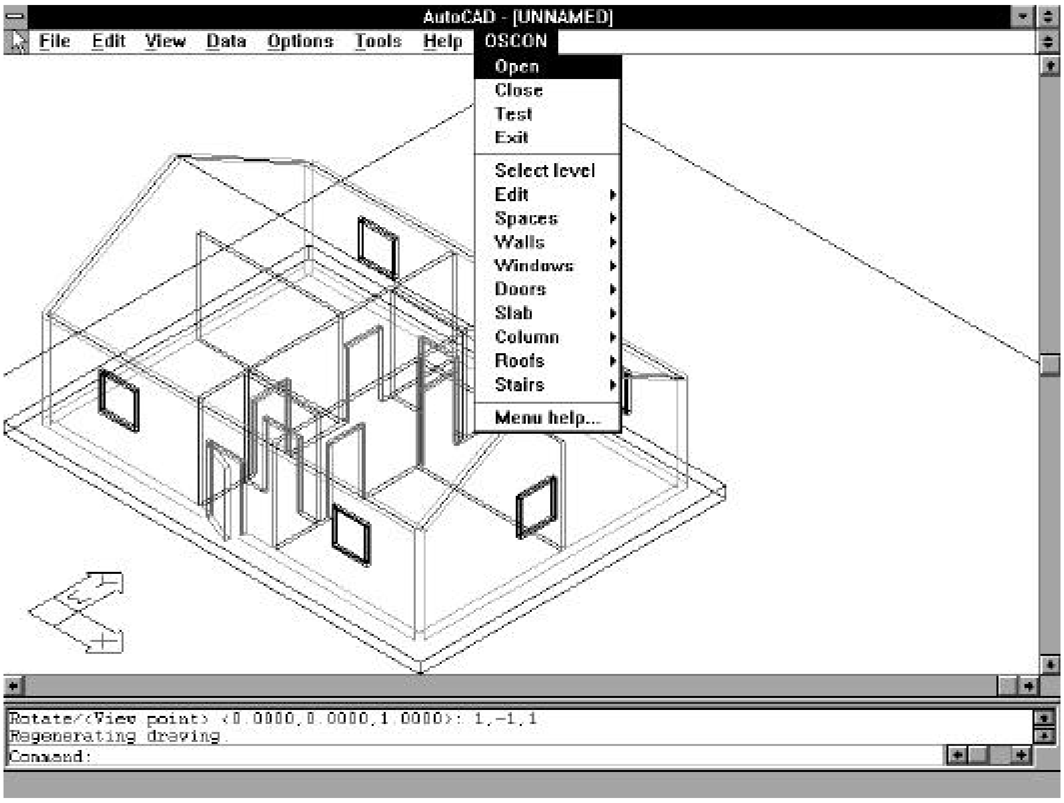
The prototype is being developed using Microsoft Visual C++ v4.0 on the PC under Microsoft Windows NT. The database is implemented using the ObjectStore OODBMS in conjunction with the Rationale Rose case Tool. It is designed on top of AutoCAD 13 which provides tools for manipulating low level drafting primitives and for producing realistic 3D views of designs by the use of hidden line removal and shading. It provides two main programming environments for implementing such applications. AutoLISP, which is an interpreted language accessible from the AutoCAD, interface itself, and AutoCAD Developers System (ADS) which is a C language library that provides an Application Program Interface (API) for manipulating AutoCAD and its internal database from a C or C++ program. ADS applications are loaded by issuing a command from the AutoCAD interface. As the OSCONCAD system uses the ObjectStore object database, which provides a persistent storage mechanism for C++ objects, it was more appropriate to use the ADS C library to build a demonstration application.

4.1 User Interface

A command driven model of interaction was chosen for the application. This is the standard way of implementing an AutoCAD application, with extra sets of menus being added to the standard AutoCAD menu hierarchy to invoke the new commands. These menus as shown in Fig. 6. provide commands to create and manipulate building elements such as walls, windows, foundations etc. Levels have been defined for the design components so they can be drawn at the locations specified by the user. The user has the option to draw in 2D or 3D. Editing facilities have also been provided. For instance the user can move junctions, delete and copy objects and any changes made to the design is reflected in the OSCONCAD object database.

When the application is started it opens the ObjectStore database for storing the design data. As various commands are called to add design elements, each command invokes a function in the application which starts a transaction on the database, prompts the user for parameters and positioning of the design element and creates an instance of the element in the database. The display is then updated to reflect this addition, and the transaction is committed to the database. The graphical representations of the building elements are displayed in the AutoCAD drawing, with the user still having access to all the usual AutoCAD commands for changing the viewpoint, zooming, performing hidden line removal etc. Thus the user creates a design using design elements instead of creating a drawing using low level drawing primitives such as lines and 3D-faces. The advantage of this approach is that the application, and indeed the database objects themselves, can contain knowledge about how buildings are designed and can thus aid the designer during the design process. An example of this kind of knowledge built into the OSCONCAD database is that a column, which is supporting a beam, will be moved automatically if we stretch the beam. This knowledge is actually built into the database itself (in the implementations of the operations on beams) and so does not have to be programmed directly into any CAD applications that manipulate beams.

When the user leaves the application, the database is closed and the current drawing is discarded. There is no need to save the drawing as a standard AutoCAD file, because the next time the application is started it will recreate the drawing from the OSCONCAD object database.



```
Rotate/<View point> <0.0000,0.0000,1.0000>: 1,-1,1
Regenerating drawing.
Command:
```

FIG. 6: OSCONCAD AutoCAD 13 Design prototype.

4.2 The Concrete Design Class: The AutoCAD_Drawer

The part of the application that is responsible for displaying the design information through the Abstract ShapeFactory (or Drawer) class makes use of the concept of a AutoCAD_Drawer. An Auto*CAD_Drawer* is an application level class. It is responsible for displaying a part of the database design via an instance of the Abstract Design Class for a given CAD package. It may perform its task by itself, or delegate it to one or more sub-drawers. In the case of the AutoCAD building design application the Auto*CAD_Drawer* class is at the root of the hierarchy with classes such as *Wall_CADDrawer*, *Door_CADDRawer* being further down. Each of these *concrete design classes* contains a reference to the part of the database for which it is responsible for presenting, and updates the CAD representation when this information changes. A *concrete design class* may use more than one drawing primitive by performing multiple calls to the Auto*CAD_Drawer* operations. The AutoCAD_Drawer maintains the set of IDs returned by these functions so that it may update the associated drawing primitives.

4.3 Accessing and Populating the OSCONCAD Object Database

From the OSCONCAD architecture design model, a set of C++ classes defining design components e.g. slab, wall, door and window, are generated. The code that implements the operations defined on these design classes is then compiled into a library.

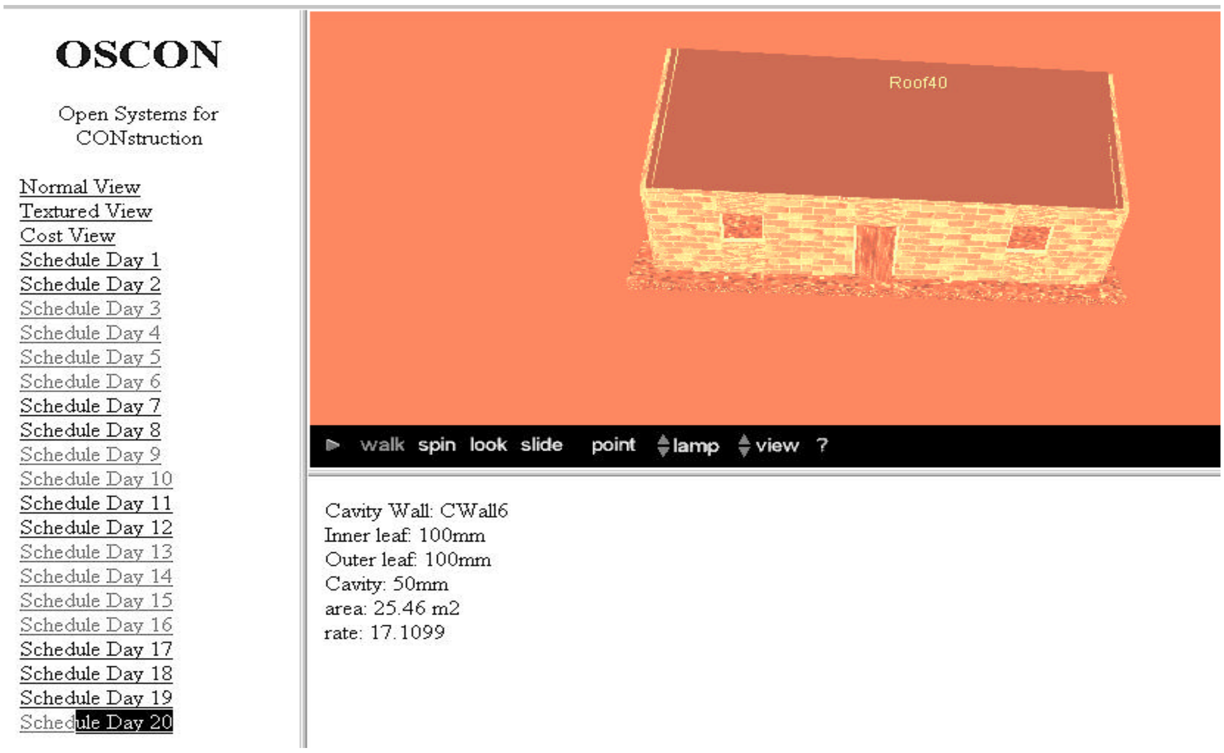
Most of the previous research on design-construction integration uses CAD database facilities, such as AutoCAD Block structures to store design information and also provide expensive and specialised interfaces to decode and transfer this information into an object-oriented database. The same process is repeated when the user performs any manipulation on the drawing. The difficulty with block structure is more apparent when performing an operation that involves more than one building element. For instance, to model the intersections between building elements, it is unavoidably necessary to explode the blocks, perform the editing task and then use the block facility once again. For complex projects, such as buildings, the use of blocks is not appropriate as the layout, orientation, shape, geometry, etc. of a building is not well defined. The OSCONCAD system avoids these complex and expensive interfaces. It uses exclusively the object-oriented database directly to store design information and to display the associated drawing in a CAD environment. To achieve this the OSCONCAD provides an alternative to block structure is used. An ID protocol is used to refer to objects previously created in the CAD environment. Each operation, which creates or updates a drawing entity, takes an ID string as a parameter. If this string is empty the operation creates a new drawing entity and returns a unique ID string for that entity. The ID string may then be used as a parameter to the same operation in which case the previously created entity is updated.

5. THE VRML APPLICATION

This application is a web-based application using VRML (Virtual Reality Modelling Language). It is used as a means of remotely interrogating information stored within the integrated database and allowing the user to visualise and manipulate the design components in a 3D environment. VRML is a developing standard for describing interactive three-dimensional scenes across the Internet. Based on the design information stored in the OSCONCAD object database, the VRML_Drawer generates a program for the VRML environment. The generated programme includes nodes such as WWW anchor (which allows links to web pages), material, rotation, and cube and is used by the VRML browser to create a 3D environment that can have links to web pages containing information about the design components. The VRML standard allows links between different worlds to be established on the web. Links are anchored to specific objects, such as walls, beams, etc. By clicking on an anchored object, you request information from other worlds using a URL (Universal Resource Locator) that specifies the address of a file on the web. Worlds loaded from the Internet are delivered by the web server, which is running on the remote host at the remote Internet site. In our case, the URL specifies a CGI script, which is a C++ program to run on the remote host under the control of the web server. The CGI program returns information about objects being queried in the VRML browser.

In OSCONCAD, VRML is not only used as a visualisation tool, but as a user interface as well. For instance, the user could interact with a 3D column in VRML rather than a column in a traditional database environment. This will allow the construction practitioners better access to information, which will motivate them to use integrated databases. Fig. 7. shows a screen shot taken from the OSCONCAD VRML application of a house and its related schedule from a VRML representation generated by the OSCON database. The user can navigate inside the building clicking on design objects and retrieving information about their properties, which include geometric, cost and time data.

This direct interaction with a VR environment has many advantages over the use of a CAD package. In traditional CAD systems, the design is relatively static which makes changes costly and time consuming as new images are re-generated from sequences of fixed frames (Larijani, 1994). In a VR environment, changes are handled efficiently by the technology. The ultimate benefit is the ability to create a walk-through, which can facilitate collaboration between clients, designers, contractors and suppliers. Such collaboration is the main objective of the OSCON database and VR is a powerful medium for communication and convergence.



Cavity Wall: CWall6
Inner leaf: 100mm
Outer leaf: 100mm
Cavity: 50mm
area: 25.46 m2
rate: 17.1099

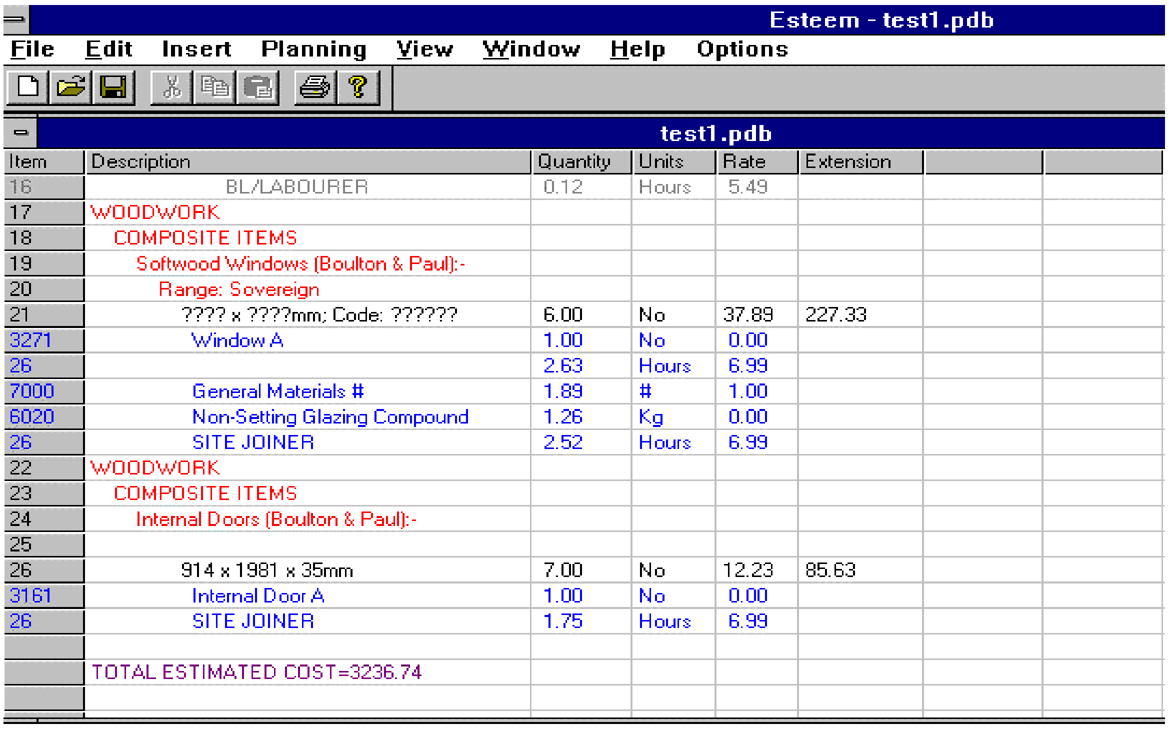
FIG. 7: A VRML house model.

6. APPLICATIONS LINKED TO THE OSCONCAD SYSTEM

As shown in Fig. 4., the major components of the OSCONCAD system are the OSCON core integration model (Aouad et al., 1997), the models for the OSCONCAD architectural design, cost estimating and construction planning domains. From these conceptual models a distributed object database schema is established as a means of integrating the information used by a number of participants within a construction project. The OSCONCAD design instances created in the database provide the different construction experts the information they require and serve as the main medium for the integration in the OSCON project. To demonstrate the feasibility and practicability of this integration approach, three OSCON construction applications have been developed to access and share the building design instances generated by the OSCONCAD system and stored in the object database (Aouad et al., 1997). Esteem for cost estimation, Mentor for project management, and Planner for construction planning. As a result of the use of CORBA standard, other construction applications could be plugged in and access the OSCONCAD distributed objects.

6.1 Esteem

Esteem is an object oriented application “Fig 8”developed after producing an object model that incorporates object types such as resource, work item, work specification, etc. The model is implemented in the object-oriented database (ObjectStore). Within Esteem, the user can change the rate, productivity, etc of a resource, the information is then changed in the database. Once a design is created, Esteem will read that design and create instances of work items in the database and populate the attributes of quantities and rates. Quantities are derived from the design components whereas rates are established from the library of a company’s work items and resources. Esteem can then establish the building project’s total cost. Any changes to the design model are notified to objects within the Esteem costing model in order to ensure consistency of information. The figure below shows the user interface for Esteem. It contains a list of work items, quantity and rates generated automatically by the object oriented database. The user can view the resources associated with work items and change some of the costing parameters such as price, productivity, etc.



Item	Description	Quantity	Units	Rate	Extension		
16	BL/LABOURER	0.12	Hours	5.49			
17	WOODWORK						
18	COMPOSITE ITEMS						
19	Softwood Windows (Boulton & Paul):-						
20	Range: Sovereign						
21	???? x ????mm; Code: ??????	6.00	No	37.89	227.33		
3271	Window A	1.00	No	0.00			
26		2.63	Hours	6.99			
7000	General Materials #	1.89	#	1.00			
6020	Non-Setting Glazing Compound	1.26	Kg	0.00			
26	SITE JOINER	2.52	Hours	6.99			
22	WOODWORK						
23	COMPOSITE ITEMS						
24	Internal Doors (Boulton & Paul):-						
25							
26	914 x 1981 x 35mm	7.00	No	12.23	85.63		
3161	Internal Door A	1.00	No	0.00			
26	SITE JOINER	1.75	Hours	6.99			
	TOTAL ESTIMATED COST=3236.74						

FIG. 8: Cost displayed in Esteem.

6.2 Planner

Once a design is created by the design module, a series of construction tasks associated with that design are instantiated in the object oriented database as there is an object model that supports the planning application. These tasks contain information about building duration. They also have attributes about starting date, finishing date, dependency, which are populated once the plan is processed by the superproject planning application. File import/export is used to update the information in the database. Any changes to the design model are notified to the planning system Fig. 9. in order to ensure consistency of the information. The user can change the resource requirements, dependency, etc.

6.3 Mentor

Mentor is an object oriented process management system Fig. 10. It contains objects such as process, processor, right, etc. The concept “process” itself is treated as an object. The users can instantiate processes in the object database. He/she can freeze, delete, update, etc that process. The object oriented model supporting mentor is fully integrated into the other models and applications. For instance, if the process design is instantiated in the database with a freeze operation, this will disallow the AutoCAD design application to operate. For a comprehensive description of the aforementioned applications refer to (Aouad et al.,1997).

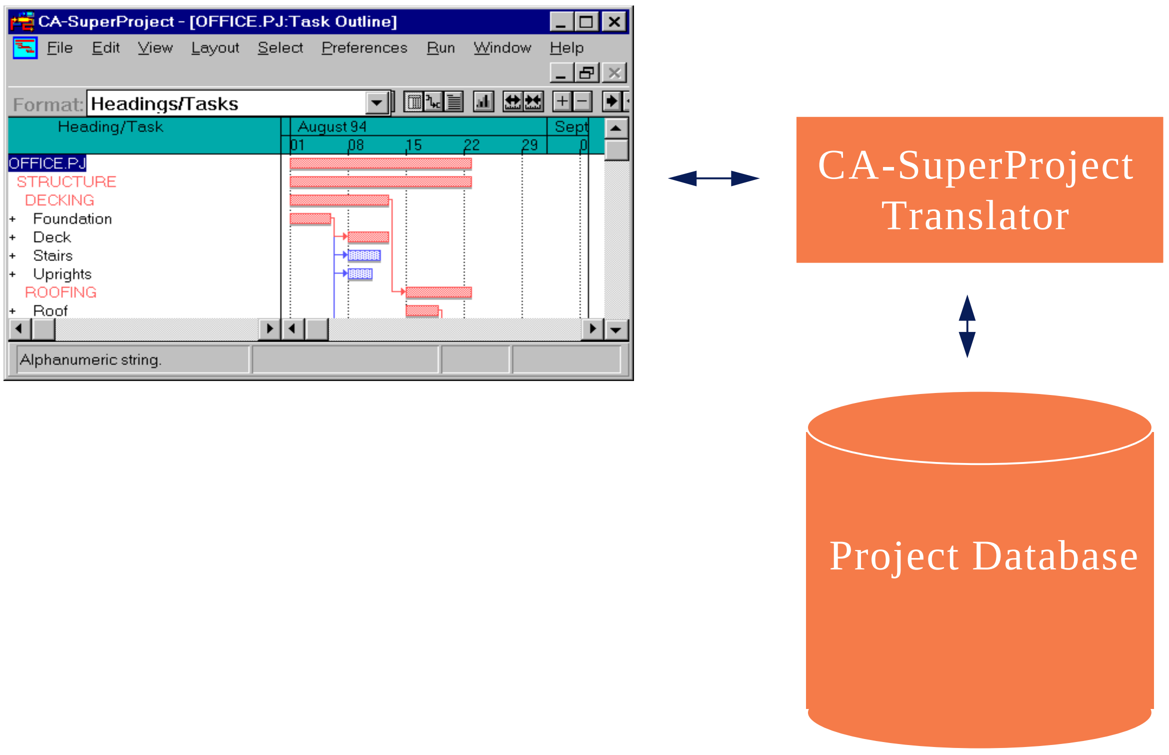
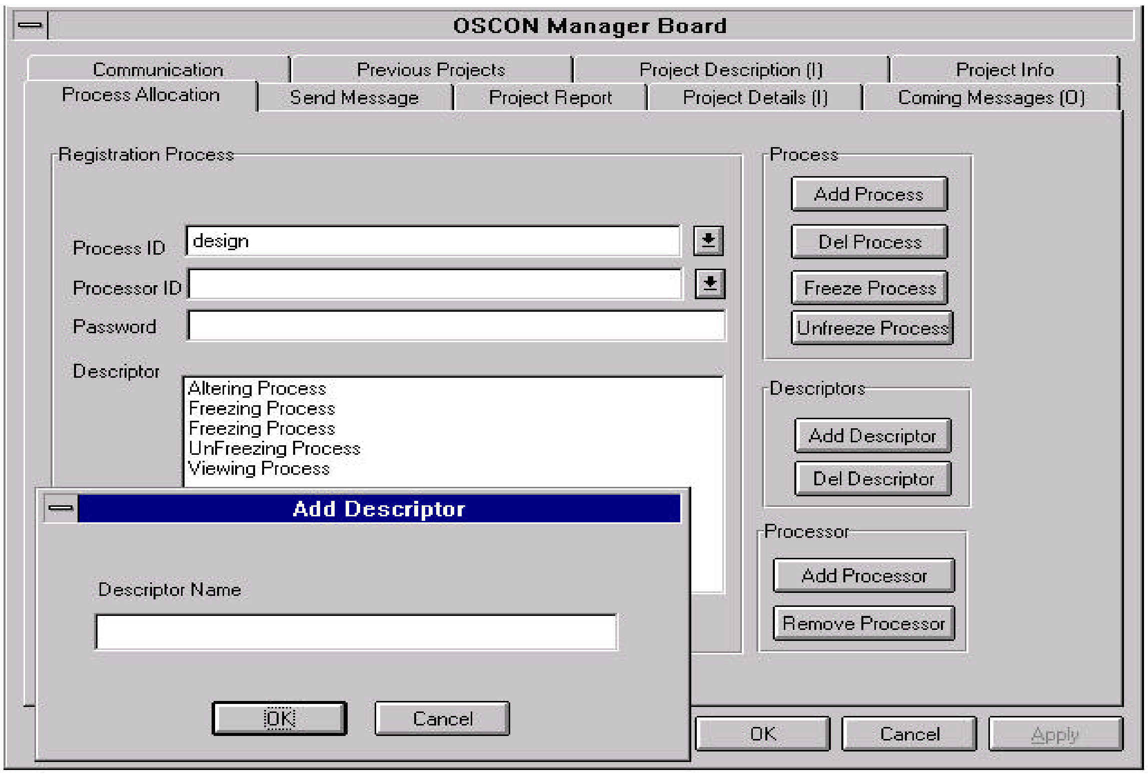


FIG. 9: A legacy application accessing the project database.



OSCON Manager Board

Communication | Previous Projects | Project Description (I) | Project Info
Process Allocation | Send Message | Project Report | Project Details (I) | Coming Messages (0)

Registration Process

Process ID: design

Processor ID:

Password:

Descriptor:
Altering Process
Freezing Process
Freezing Process
UnFreezing Process
Viewing Process

Process: Add Process | Del Process | Freeze Process | Unfreeze Process

Descriptors: Add Descriptor | Del Descriptor

Processor: Add Processor | Remove Processor

OK | Cancel | Apply

Add Descriptor

Descriptor Name:

OK | Cancel

FIG. 10: The Mentor process manager.

7. CONCLUSIONS

This paper presents the development of OSCONCAD system that provide a vehicle for storing architectural design information in an object-oriented database shared across a range of construction computer-based applications. This integrated database is developed using the object-oriented modelling approach to establish standard models for architectural design OSCONCAD and other specific domains, e.g., estimation, planning etc. This research has demonstrated that the integration of CAD and computer based construction systems using object-oriented technology can be utilised to reduce design fragmentation and to bridge existing gaps between disciplines within the AEC industry. In designing the software architecture, the use of design patterns (Gamma et al., 1994) has been found extremely helpful in arriving at robust solutions such as the Shape Factory approach, and is to be recommended.

The OSCONCAD research described in this paper attempts to overcome the limitations of the previous research development in the area of design-construction integration. This is by using a model-based approach whose emphasis is on capturing graphical and textual information about building components and directly storing them into an object oriented database rather than the complex and ill-structured CAD database. This approach will avoid the expensive process of translating drawing data from the CAD database into the object-oriented database. Moreover, this approach allows the separation of the design data from the specific CAD package that is used to manipulate it. A consequence of this is that new CAD products will be much easier to incorporate and also other none CAD applications can have direct access to the design data from such a database. In addition, the OSCONCAD model classes compliance with the IFC will facilitate data exchanges between construction-related applications and allow common interpretation of the design objects amongst the applications in a consistent and interoperable manner. Also, in the context of project data interchange the OSCONCAD not only concentrates on the problem of integrating applications via data sharing but goes further into sharing objects distributed amongst applications by complying to CORBA standards.

This work does not address issues relating to management of the updating of objects. However, the authors believe that mechanisms to handle issues such as versioning, authorisation, and aggregation of changes will need to be addressed in relation to object-based approaches. Some of these issues are being addressed in a related COMMIT project (Cooper et al., 1997).

A system like OSCONCAD evolving in an integrated framework will have an impact on the ability of a construction project to meet client needs. It will also contribute to the quality of a construction project by improved design and better communication between the design and construction stages of the construction project. This will also reduce the AEC industry fragmentation and increase competitiveness. It also provides a vehicle for sharing different types of graphical and non-graphical data among the construction participants and across stages in the building life cycle. By using standards for different types of data, that data can be transferred among the project participants. By using the object-oriented technology, the OSCONCAD favour the reuse of object and applications and avoids expensive islands of automation and interfaces between construction applications. This will reinforce the important industry restructuring moves taking place and the trend towards greater use of design and build contracts as an organisational attempt to overcome the construction problems.

8. ACKNOWLEDGEMENT

The authors would like to thank the members of the OSCON steering committee for their kind contributions and the UK Department of Environment, which supports the work, presented here.

9. REFERENCES

Adams S. (1989). Practical Buildability. *CIRIA, Butherworths*, London.

Ahmed S., Wong A., Sriram D. and Logcher R. (1991). A comparison of object-oriented management systems for engineering applications. *Resaerch Report No R19-12, Intelligent Engineering Systems Laboratory, Dept. of Civil Engineering*, Massachusetts Institute of Technology, May 1991.

Alshawi M. (1996). SPACE: integrated environment. *Internal Paper*, University of Salford.

Aouad G. (1991). Integrated Planning Systems for the Construction Industry. *PhD thesis*, Loughborough University.

Aouad G., Marir F., Child T., Brandon P. and Kawooya A. (1995). A Construction integrated databases- Linking Design, Planning and Estimating. *International Conference on Rehabilitation and Development of civil engineering infrastructure systems*, June 9-11, 1997, American University of Beirut, Lebanon.

Aouad, G,.Child, T, Marir, F & Brandon, P (1997). Developing a conceptual model for an integrated database, Proceedings of intelligent information systems, Bahamas, December, pp 316-320.

Aouad G., Betts.M, Brandon.P, Brown.F, Child.T, Cooper.G, Ford.S, Kirkham.J, Oxman.R, Sarshar.M, and Young.B. (1994). Integrated databases for design and construction: Final Report. University of Salford

Atkin B. and Gill M. (1986). CAD and Management of Construction Projects. *Journal of Construction Engineering and Management*, Vol 112, No 4, December, ASCE, pp 557-565.

Atwood T. (1991). Why the OMG object request broker should mean good news for object databases. *Journal of Object-Oriented Programming*, Vol. 4 No4, pp8-11.

Bjork B-C. and Wix J. (1991). An introduction to STEP. *VTT and Wix Mclelland* Ltd. 1991.

Björk B-C. (1994). RATAS Project - Developing an Infrastructure for Computer-Integrated Construction, Journal of Computing in Civil Engineering, Vol. 8, No. 4, 400-419. http://www.vtt.fi/cic/ratas/index.html

Bohms M., Tolman, F. and Storer G. (1994). ATLAS, a STEP Towards Computer Integrated Large Scale Engineering, Revue internationale de CFAO, Vol. 9, No. 3, 325-337. http://www-uk.research.ec.org/esp-syn/text/7280.html

Booch G. (1994). Object-Oriented Analysis and Design with Applications, 2/e. ISBN 0-8053-5340-2. Addison-Wesley.

Brandon P.S. (1993). Integration of Construction Information. *Proceedings of N+N Workshop. Lake District. UK.*

Cherneff J., Logcher R. and Sriram, D. (1991). Integrating CAD with construction schedule generation. *J. Comput. Civil Engineering.*   5(1), 64-84.

Cooper G.S., Aouad. G, and Brandon. P. (1994). ICON Methodology Report, *IT Institute and Department of Surveying*, University of Salford.

Cooper G.S. (1995) Object-Oriented Databases from an Information Technology Viewpoint: Knocking Down Some Walls, in Object Technology and its Application in Engineering. ISBN 0902 376 209, Ed. Professor James Powell, proc. Conf. on Object Technology and its Application in Engineering, Glasgow, March 1995.

Cooper G.S., Rezgui Y., Yip J., and Brandon P. (1997) Notification and Change Propagation Support in a Multi-actor Environment, In proceedings of the CIB W78 Workshop on Information Technology Support for Construction Process Reengineering, Cairns, July.

CORBA (1995). The: Common Object Request Broker: Architecture and specification. Revision 2.0.

Dubois, A.M., Flynn, J., Verhoef, M.H.G. and Augenbroe, F. (1995) Conceptual Modelling Approaches in the COMBINE Project, presented in the COMBINE final meeting, Dublin. http://erg.ucd.ie/combine/papers.html

Gamma, E., Richard Helm, Ralph Johnson, and John Vlissides. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley.

Gorti, S., Sriram, D., and Ram, D. (1996). From Symbol to Form: A Framework for Conceptual Design, Journal of CAD, Vol. 28, No. 11, 1996.

Ferguson I. (1989). Buildability in Practice. *Mitchell's Professional Library*, London.

Froese, T. and Paulson B. (1994). OPIS: An object model-based project information system. *Microcomputers in Civil Engineering*, 9, pp 13-28.

Howard, H.C. (1991). Linking design data with knowledge-based construction systems. *CIFE Spring Symposium.*

IFC (1997). Industry Foundation Classes, Release 1.0,Vol 1-4.

Ito, K. (1991). Design and construction integration using object oriented project model with multiple views. *In Proceedings of the Computer-Integrated Design and Construction Symposium, Construction Congress II*, Cambridge, MA, April 1991.

Ito, K., Ueno, Y., Levitt, R. and Darwiche, A. (1989). Linking knowledge-based systems to CAD design data with an object oriented building product model, *CIFE Technical Report* No. 17, Stanford University.

Kartam, N. (1994). ISICAD: interactive system for integrating CAD and computer-based construction systems. *Microcomputers in Civil Engineering* 9, pp 41-51.

Larijani, L.C. (1994). The virtual reality primer. McGraw-Hill, USA.

Latham, M. (1994). Constructing the team, H.M.S.O.

Orbix (1996). White Paper. IONA Technologies Ltd.

Pentilla, H.(1989).A Scenario for The Development and Implementation of A Building Produce Model Standard. *CIFE(Center for Integrated Facility Engineering-Stanford University) Symposium Proceedings*.

Port, S. (1989). The management of CAD for construction. *BSP Professionals*.

Rumbaugh, J., Blaha M., Premerlani W., Eddy F. and Lorensen, W. (1991), Object-Oriented Modelling and Design. Englewood Cliffs, New Jersey: Prentice-Hall.

Sriram, D. and Logcher, R. (1993). The MIT DICE Project, IEEE Computer, Special Issue on Concurrent Engineering, pp. 64-65, January 1993.

Sriram, D., Ram D., Wong, A., and Li, H. (1995), GNOMES: An Object-Oriented Non-Manifold Geometric Engine, Journal of CAD, November.

Svensson K. and Aouad G. (1997), Developing standardised architectural building product models for spaces and space enclosures. Submitted to Automation in Construction.

Tah J., Thorpe A.and McCaffer A.(1991) Decision making within large computer based construction management systems. *SERC Report GR/E76957/E.*

Timberline (1990). Precision Estimating, Timberline Software Corporation, USA.

UML (1997) UML Document Set, Version 1.0, 13 January, 1997 (Rational Software Corporation). http://www.rational.com/uml/index.html

Wix, J. and Storer, G. (1996). Building core models. Occasional paper.

Zabilski R. and Hall H.E. (1989). Presented in 3D. *Civil Engineering*, June, pp 48-50.