Promoting Computer Integrated Construction Through the Use of Distribution Technology

RECEIVED: June 1996
REVISED: September 1996
PUBLISHED: October 1996

Mr. Alex Brown
Email: a.j.brown@iti.salford.ac.uk

Yacine Rezgui, Dr.
Email: y.rezgui@iti.salford.ac.uk.

Grahame Cooper, Dr.
Email: g.s.cooper@iti.salford.ac.uk

Jim Yip, Dr.
Email: y.j.yip@iti.salford.ac.uk
all: Information Technology Institute, University of Salford

Peter Brandon, Professor
Department of Surveying, University of Salford
Email: p.brandon@surveying.salford.ac.uk

SUMMARY: *This paper discusses how distributed object technology can profitably be put to use in the construction industry. It begins by outlining the potential benefits distributed object technology and standards can bring to construction, particularly when considering the integration of existing and future information systems. It then describes research that has applied this technology to the COMMIT (COnstruction Modelling and Methodologies for the IntelligentinTegration of information) project. This project aims to provide new levels of support for integrated building information systems by dealing with issues such as versioning, notification, the representation of intent and the definition of object rights. To this end, a generic information management model has been proposed, a brief description of which is given here. The architecture and distributed implementation of this information management model are then described, and a framework proposed in which the model acts as a bridge between services provided by the CORBA (Common Object Request Broker Architecture) distribution standard and objects in a computer integrated construction environment. Examples, illustrating the implementation of information management concepts as distributed components complying to the CORBA standard, are given. Several techniques for integrating existing CIC systems and legacy applications are proposed before finally presenting the user interface through which project participants interact with the distributed components of the integrated environment. The research presented in this paper is ongoing; a primary aim is to refine the prototypes that demonstrate the new levels of integration we can expect in the coming component-based era.*

KEYWORDS*: Distributed, Computer Integrated Construction, Information Management, Component, CORBA, COMMIT*

1. INTRODUCTION

It is widely recognised that business and social trends are driving the construction industry through a period of radical change. Downward cost pressures, coupled with ever more specialised building trades and the increasing technical complexity of projects all create a demand for the integration of construction project information. At the same time, the state of the art in information technology continues to move forward. The development and deployment of new construction industry software applications, improvements in network technology, the

application of robotics to the building process, the development of new modelling methodologies and languages and the definition of standards for information exchange all create new opportunities for integration. Of course, the desire to integrate is not unique to construction. Other industries, such as manufacturing and the IT industry itself, are arguably even more powerful levers for change. Indeed, construction has yet to achieve the levels of design process integration found in the automotive and aerospace industries.

The next few years will undoubtedly see major changes in the use of computers. Improvements in hardware technology - in particular, the advent of inexpensive, high performance CPUs - are resulting in increasingly powerful desktop machines. The computing resources of an organisation or project team are now spread across many (often heterogeneous) platforms in different locations. At the same time, business application end users are increasingly seeking more local autonomy and responsibility. These changes are being reflected by a growing interest in distributed object technology. Many computer industry analysts see this as the next evolutionary step in a process which has already moved from monolithic applications to the client/server era (Orfali et al., 1995). These changes will inevitably have an impact on the construction industry and, in particular, the potential for the integration of information.

Researchers have addressed information integration in construction in a variety of ways: communication between applications (achieved through specific software integrating a unique and invariant set of chosen applications), integration through geometry (often the case in commercial CAD packages where integration is based on and limited to geometrical information), knowledge-based interfaces linking multiple applications and multiple databases, and integration through central project databases holding all the information relating to a project according to a common conceptual model. The latter category includes ATLAS (Bohms et al., 1994) (for large scale engineering), COMBINE (Dubois et al., 1995) (for HVAC and building design), RATAS (Bjoerk, 1994) and ICON (Aouad et al., 1995) (for building design and construction management).

The aim of this paper is to illustrate how research carried out by the COMMIT project can be married with IT developments in the field of distributed object technology to help to provide new levels of intelligence and integration in construction projects. The paper first discusses the benefits of adopting distributed object technology, both to the construction industry itself and the software industry which serves it. An overview of the COMMIT project is then given. The latter parts of the paper present the architecture and distributed implementation of the COMMIT Information Management Model.

2. THE USE OF DISTRIBUTION TECHNOLOGY IN CONSTRUCTION

There are currently two major emerging standards for distributed objects, Microsoft's Object Linking and Embedding (OLE) (Brocksmidt, 1995) and the Object Management Group's Common Object Request Broker Architecture (CORBA) (OMG, 1995a, 1995b). Without going into detailed comparisons of OLE and CORBA which have been made elsewhere (Orfali et al., 1996), this paper will refer exclusively to CORBA for the following reasons. At the time of writing CORBA would seem to have more to offer projects aimed at computer integrated construction; it fully supports the principles of object-orientation whereas OLE does not support inheritance. It also offers a wider range of facilities and services for integration; one estimate puts CORBA about two years ahead of OLE (Orfali et al., 1996). If OLE (or some other) standard prevails, we assume it will offer services along similar lines to those currently offered by CORBA.

The nature of the construction industry is such that virtual teams are often brought together for projects before being broken apart again on completion. The software applications used may vary from one construction project to another. Organisations and individuals participating in a team will bring their own unique skills and resources, which may include legacy applications and data. Any integrated environment should provide a means by which the component objects provided by organisations and application vendors can interoperate in a seamless way. This interoperability should not be limited to those components which have a prior knowledge of each other; components should "plug and play" and so be usable in ways which were unanticipated by the original developers.

There is therefore a need to concentrate on the interfaces between objects as well as standardising the objects themselves. Instead of integration being achieved through static models that define the structure of shared information (in the form of files or databases), integration should be made through frameworks which define semantic relationships between the interfaces of separate components. The CORBA architecture provides for these frameworks through business object facilities (also known as vertical common facilities) (OMG, 1996) and distinguishes them from basic services (such as naming, persistence, transactions, etc.) and horizontal common facilities (such as user interfaces and system management). Business object facilities are already under development for several business areas including computer integrated manufacturing and banking. It is to be

expected that work such as STEP, which aims to define standards for the exchange of product model data (ISO/TC184/SC1, 1994), will play an important part in the definition of a CORBA facility for computer integrated construction. The Industry Alliance for Interoperability is also developing the Industry Foundation Classes (IFC's) - a common set of intelligent building design objects that will enable the sharing of information in all stages of a construction project. The IAI is currently investigating the possibility of making IFC's CORBA compliant. Other researchers have attempted to enable the integration of product modelling and distribution standards by proposing languages which combine the features of the STEP and CORBA modelling languages (Su et al., 1995). It remains to be seen as to whether this work will impact upon the standards bodies. Work carried out at Stanford University investigates interoperation at a more fundamental level in order to support concurrent engineering, although without adhering to standards such as CORBA (Chedro et al., 1994).

One advantage that CORBA frameworks have over many existing methods of integration is that they are highly flexible and designed for reuse. Business objects within the framework are capable of recognising their environment and interacting with other business objects. They can therefore take part in a wide range of integration scenarios, perhaps even across industrial sectors. They can also easily be specialised to meet the needs of particular organisations or projects. This is of particular relevance to the construction industry where organisational structures and processes may vary greatly from one project to another. Where appropriate, relationships between business objects will be transparent and modifiable by end users. An organisation may also define its own standard frameworks for object interworking and collaboration. Object collaborations also occur at many different levels of granularity, for example:-

- a line in a CAD drawing should be cut off at the edge of its bounding frame window,
- the contents of a room should move when the room is moved,
- a estimate of the cost of building a wall might be generated from the wall dimensions, material and unit rate and
- the structural engineer might need to be notified in case of any modifications to the dimensions of a given load-bearing wall.

Such collaborations and relationships can be placed within the appropriate framework of a well defined distributed object standard. This breaks down the problems of integration into more manageable, logically discrete areas.

The integration of existing construction industry software applications is also facilitated by the language neutral nature of CORBA. In a CORBA system, distributed objects are first specified in IDL (Interface Definition Language). From IDL, which only describes object's interfaces and not their implementation details, various language specific implementation skeletons can be automatically generated. CORBA currently specifies language bindings for C, C++ and SmallTalk (Ada, COBOL and several other languages are forthcoming). Existing construction industry software applications are coded in a variety of implementation languages; IDL therefore provides a means by which they may be more easily "wrappered".

A typical large scale construction project involves hundreds of participants, thousands of decisions and huge volumes of data. Clearly, centralising such large amounts of data in a single database poses technical difficulties. Distribution technology, however, provides "network transparency" - data in a single logical repository may be physically stored in many different locations.

In addition to the fundamental advantages mentioned above, a component-based approach to integration has benefits when one considers the current technological and socio-economic state of the construction industry. For example, commercial pressures often dictate that existing construction industry software applications satisfy a very broad range of user requirements. This results in large monolithic applications. Component based applications could be assembled on the basis of user needs and could potentially be much smaller. They would run on hardware that is less expensive and more portable, thereby creating the opportunity to involve construction industry practitioners who have hitherto been excluded. This might, in particular, promote the use of computers on-site. Smaller, component based applications are also easier to distribute over networks, particularly with the advent of interpreted platform independent languages such as Java (Gosling and McGilton, 1996).

3. THE COMMIT PROJECT

The overall aim of the COMMIT project is to improve the long-term effectiveness of the construction industry by the provision of intelligent integration of information to support concurrent engineering and decision making for the effective management and realisation of all stages of a construction project's lifecycle. In order to further improve support for computer-integrated construction, project information needs to be conceptually modelled

throughout its lifecycle, along with the events that impact upon it. The project intends to deal with issues not covered fully by either previous work in integrated building information systems, or previous work in standardising for the exchange of product model data such as STEP. It is proposed that an intelligent integrated building information system needs to address the following four areas: rights and responsibilities, notification, versioning support and the representation of intent. The COMMIT Information Management Model (CIMM) consists of generic models of the concepts needed to address these areas.

A pure Object-Oriented methodology has been chosen for the analysis and design stages of the COMMIT project. It is widely accepted that Object-Orientation has a stronger equivalence with the real world than other methodologies: encapsulation, abstraction and multiple inheritance are all features which help us to model the complex and dynamic systems found in the construction industry (Turk, 1993). It is worth noting that the emerging standards for distribution are also deeply rooted in Object-Oriented methodologies. Only a brief description of the CIMM can be given here, for further details the reader is referred elsewhere (Rezgui et al., 1996a).

3.1. Rights and Responsibilities

As a construction project progresses, actors' rights and responsibilities with respect to specific construction project objects change. Such rights can be captured through the use of a role; as an actor performs different roles through the life of a project, so their rights over project information change. Defining roles in this way is essential for effective management of concurrent multi-actor engineering and needs to be tightly integrated with support for versioning. The performance of all project roles involves actions on both object types and objects. Each action takes place according to a certain pre-defined object right (an authorisation given to a role to perform a given action on an object). The set of *ObjectRights* that a *Role* has on a given *Object* is defined as *ObjectAuthority* (Fig. 1). Similarly, the set of rights a role has on a given *ObjectType* is defined as an *ObjectTypeAuthority*. An object authority might consist of an ordered set of other object authorities. This is represented by the concept of *OrderedCompositeAuthority* (an approval procedure involving several roles and a common object might be specified in a given order). The object authority is specialised into default and specific authority. The default authority is related to an object type and is thus inherited by its object instances, whereas the specific authority only concerns the object itself.

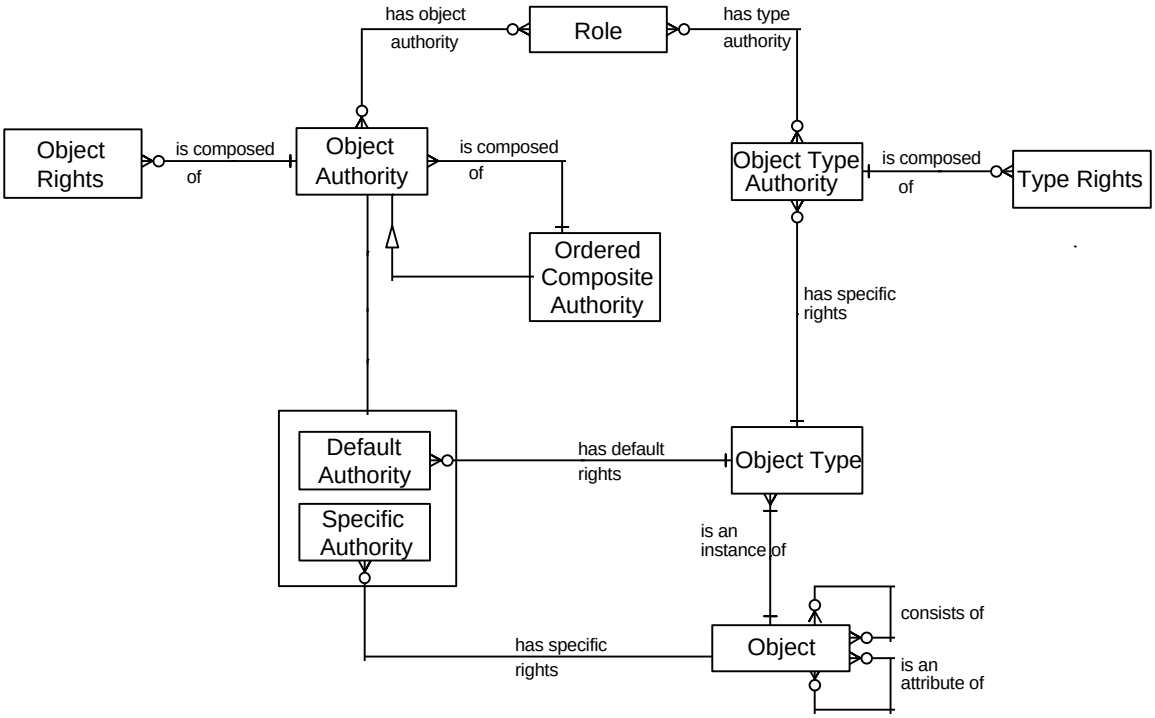

FIG. 1: The CIMM model of rights

Diagramming standards modelling notation: rectangles represent object types which describe concepts. Lines joining object types represent bi-directional relationships. Relationships with a cardinality of one are represented by a single line crossing the relationship. Relationships with a cardinality of many are represented

by a crow's foot. Optionality is represented by a small circle meaning "sometimes" or by a single line meaning "always". A box enclosing more than one object type denotes a complete partition that expresses a full list of its partitioned subtypes.

3.2. Notification

Notification is the process by which amendments made to objects are notified to the various roles concerned with that object. The spread of electronic mail and fax means that the CIMM can not only represent this notification, but also to a large extent automate it. Many individuals will be unused to working in an integrated construction environment in which shared project information changes at a rapid pace. A notification mechanism is therefore essential for keeping actors aware of project changes and also supports the automation of other information management processes such as approval.

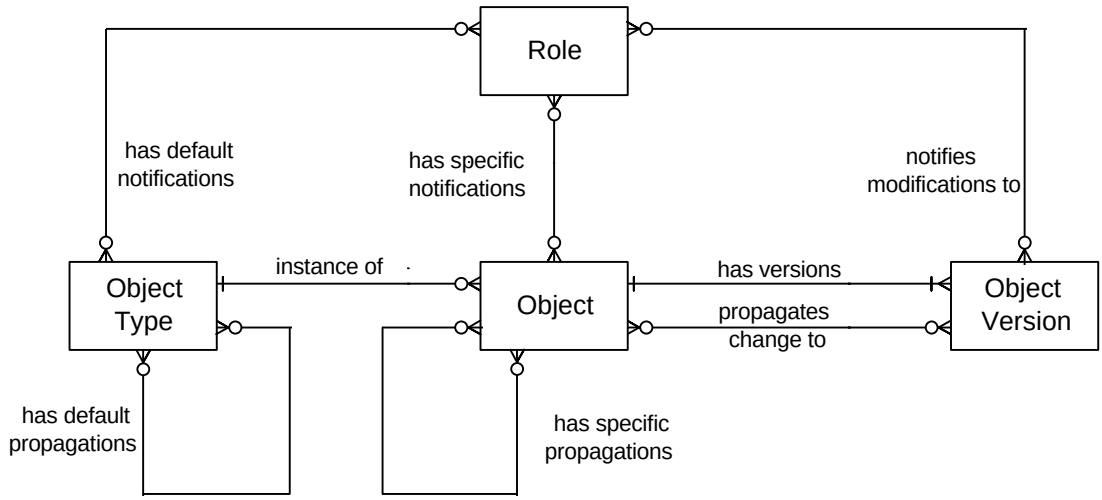

FIG. 2: The CIMM model of Notification and Propagation

As illustrated in Fig. 2, an *Object Type* is related to a default set of roles which will typically need to be notified in case of any modification to instances of that type. *Objects* inherit this set but, as the set of roles can vary from one instance to another, objects have their own specific set of roles to notify. The CIMM also supports the specification of notification lists at the version level in order to accommodate the complex interaction of roles and proposed information changes in construction activities such as design. Therefore, a modification to an *Object Version* notifies a set of roles, which may sometimes differ from those attached to the object and object type.

Propagation involves the CIMM making inferences on the basis of project information. It is modelled in a similar way to notification; an object type, object and object version may all have a unique set of objects to which a modification is propagated. Propagation relationships typically exist in knowledge based systems in the form of rules.

3.3. Versioning Support and the Representation of Intent

Construction projects frequently involve multiple actors making changes and working simultaneously. The primary reasons for providing a structured environment in which such changes are recorded are firstly to provide the client with a complete project history, and secondly to facilitate backtracking. In addition, because actors often refer to previous versions of objects, a structured historical versioning environment reduces ambiguity by standardising version numbers and allowing the examination of previous versions of objects.

An intelligent construction information management system should also record the intent behind construction project decisions. A record of intent is one component necessary for providing the client with a complete project history. It could also be useful if disputes occur, not only for resolving litigations, but also to help actors to see the project from other viewpoints. Intent representation promotes actor's understanding of the reasons for project changes, which in turn reduces problems arising from misunderstanding intentions. A complete record of project intentions could be useful for decision support in future projects in the form of a case-based reasoning analysis of objects and intentions.

In the CIMM, any *Object* may exist in one or more *Object Versions* (Fig. 3). The existence of versions influences a *Decision* which is implemented by an *Object Operation.* In turn, this operation generates more object versions. The intent behind a decision is recorded explicitly by a *Statement of Intent,* but also implicitly by the relationships which exist between a decision and the object versions which influenced it.

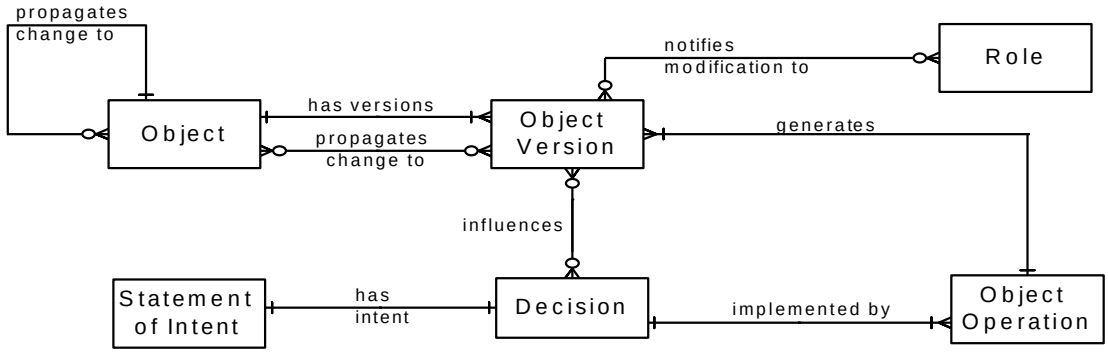

FIG. 3: Object Versioning and the Representation of Intent

4. A DISTRIBUTED IMPLEMENTATION OF THE CIMM

The COMMIT project has chosen to implement the CIMM as a set of CORBA compliant distributed components which make use of CORBA services. This section describes an architecture that makes this possible, and gives details of how CORBA service objects can be specialised into CIMM objects. Three strategies for integrated existing construction industry software and CIC systems are also given.

4.1 Architecture

The CORBA standard defines various services which currently comprise Naming, Events, Life Cycle, Transactions, Concurrency, Persistence and Queries. Relationship, System Management and Security services are in the process of being finalised. It is worth noting that, in the past, these services have been provided by a range of different vendors and technologies. Persistence - the capacity to have data that survives the process which created it - has typically been provided by relational and object-oriented databases. Naming, on the other hand, has been provided by services such as ISO's X.500. The rationale behind much of CORBA is that it can transparently encapsulate these existing services by providing a standard interface to distributed objects.

CIC projects have tended to expend some of their effort researching and implementing these features, for example, some of the system level elements in OPIS (Froese and Paulson, 1994) or the work of Eastman and Kutay (1991). Although useful results have often been produced, we argue that the implementation of a distributed CIC system can be considerably simplified by re-using the services CORBA provides. Where services do not match the particular requirements of the construction industry, existing services may be specialised or, in the worst case, new services provided. Research efforts can focus on issues more directly related to construction while at the same time re-using the efforts of technologists devoted purely to computing. The COMMIT implementation architecture proposes that concepts in the CIMM, which facilitate the management of information in a distributed CIC environment, make use of CORBA services by specialisation (Fig. 4). Objects in the CIC environment can then be specialised from CIMM objects. In this way, the CIMM provides a bridge between basic CORBA services and CIC systems - in CORBA terms, a vertical market common facility.

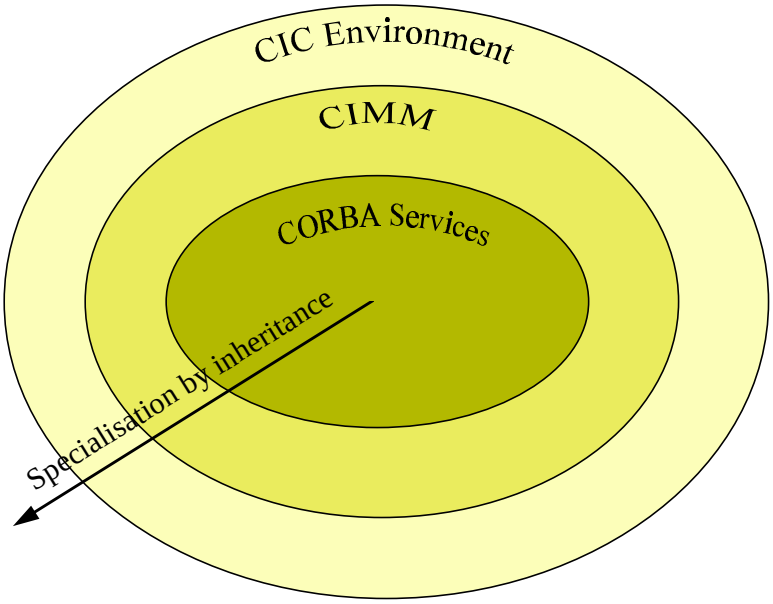

FIG. 4: The CIMM as a bridge between CORBA and CIC environments.

The backbone of the COMMIT integrated construction environment is an ORB (Object Request Broker) which manages the low level interoperability between components (Fig. 5). The ORB also provides the low level CORBA services necessary for distributed computing and provides transparency of platform and location (as indicated by platforms A, B and C). Components participating in the distributed environment are specialised from CIMM objects, whether they reside independently or in applications. The CIMM browser is a specialised application that allows project participants to interact with CIC objects at the information management level. It is described more fully in section 6. Any number of applications may be running on a given platform at any one time.

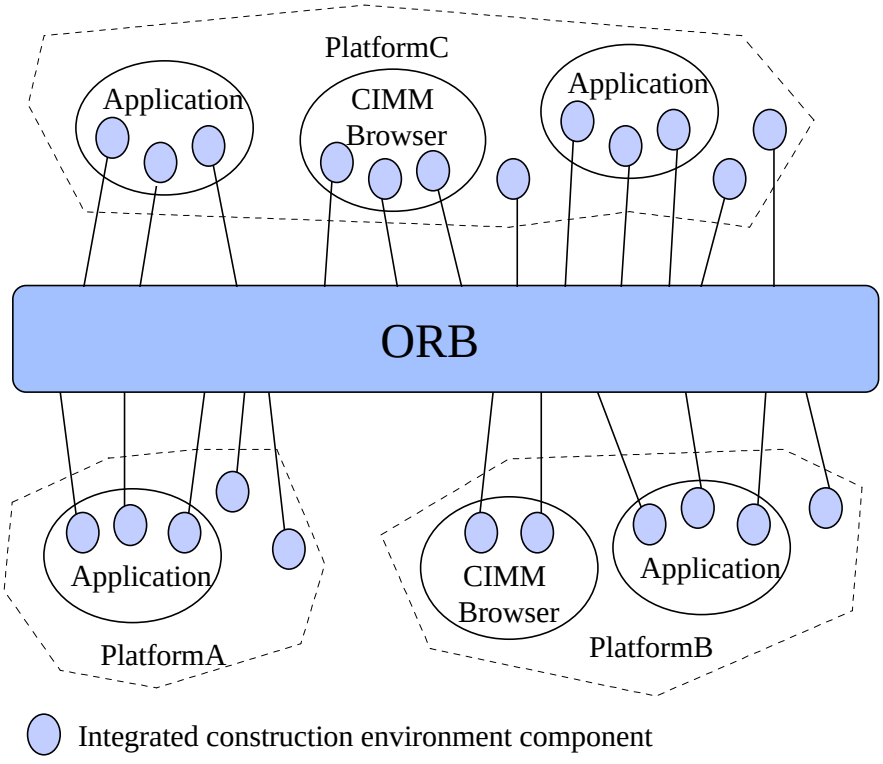

FIG. 5: COMMIT Implementation Architecture

4.2. Implementing a CORBA Compliant CIMM

The procedure for the basic implementation of objects and relationships in a distributed environment does not differ greatly from their implementation in an object-oriented database. In the CORBA methodology, object interfaces must be described in an Interface Definition Language (IDL); they can then be implemented in the language of the developer's choice.

All CIMM objects use the services provided by three basic CORBA objects; *Persistent Object*, *Transactional Object* and *Life Cycle Object.* Clearly, implementing the CIMM requires the use of the CORBA persistence service; CIMM objects must remain in the construction environment after the programs which initiated their creation have terminated. The *Transactional Object* provides consistency, concurrency and recovery in a distributed object system. While the underlying CORBA implementation of services may often be complex (as in the case of transactions and concurrency), this complexity is largely hidden from the developer. The *Life Cycle Object* provides services for creating, copying and deleting objects while maintaining referential integrity constraints.

The CIMM also contains several concepts that require the use of CORBA collection service. For example, an *Object* is related to the set of *Object Authorities* (one for each *Role*). These will need to be stored in a collection class which can be traversed with an iterator if the information manager is specifying the rights that different roles have over core objects. In addition to the CORBA objects that all CIMM objects inherit from, there are several CORBA services which CIMM objects make use of through method invocations. All CIMM Objects make use of the *Naming Context* object (provided by the CORBA naming service) in order to give CIMM objects a unique system wide identity by which they can be located. The implementation model of the CIMM concept of Object, with possible specialisations into CIC environment objects such as Building, Room and Wall, is shown in Fig. 6.

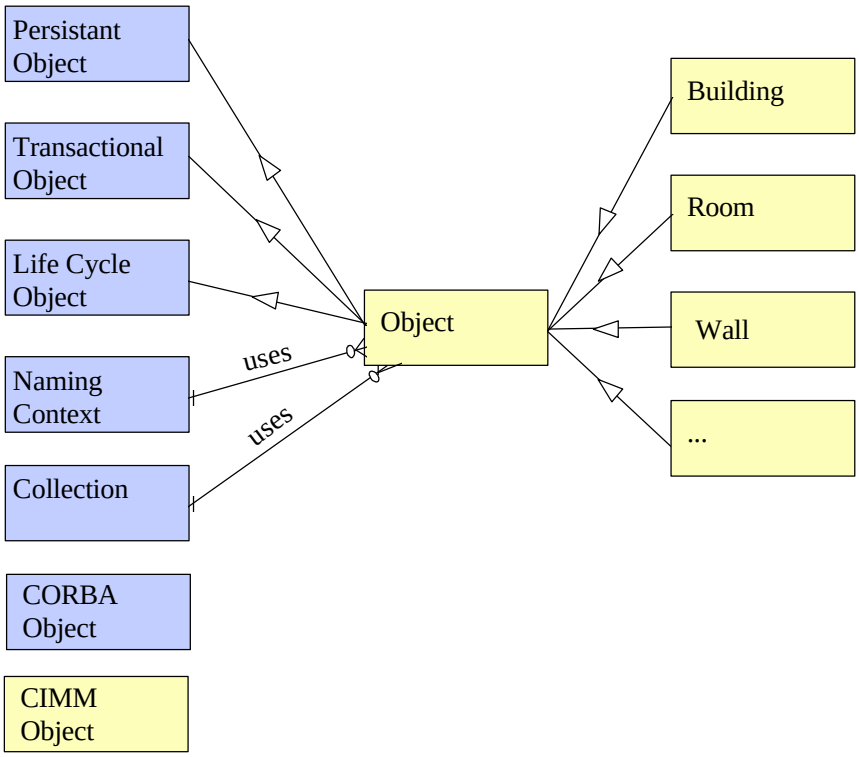

FIG. 6: Implementation Model for CIMM Object

The CORBA event service allows objects to register their interest in specific events by invoking an appropriate method of the *Event Channel* object. This service allows the invocation of a distributed object's method to cause the transparent invocation of a CIMM object's method. This is used to implement notification and rights access in the CIMM by de-coupling the communication between objects. It is not necessary, therefore, for an object version to maintain a list of roles to notify. Instead, it simply has methods that can be invoked in order to register a role's interest (using the *Event Channel*). Any subsequent invocations of a relevant object method

cause a message to be sent to the role object. This message can then be acted upon - the role object creates an appropriate message box on the workstation of the actor performing that role. The implementation model for object version is shown in Fig. 7.

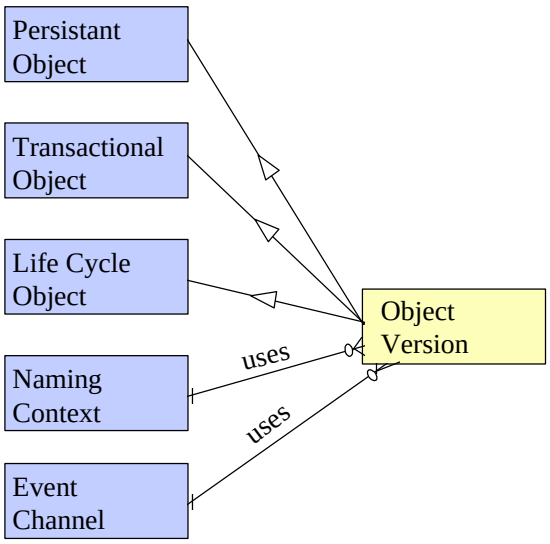

FIG. 7: Implementation Model for CIMM Object Version

To write CORBA compliant applications, the boundaries and interfaces of objects must be declared in IDL. IDL is only concerned with the declaration of components and does not contain any implementation details. For the purposes of implementation, IDL code may be mapped into one of several languages, for example C++. Fig. 8 shows some sample code for the IDL definition of the CIMM's *Object* and *ObjectVersion*.

```
// Example IDL interface definition of Object and Object Version from COMMIT
// Information Management Model

module CIMM
 {
       interface Role;    // Forward declaration

       // Note - In IDL :: is a  scoping mechanism to refer to other modules.

       interface  ObjectVersion:CORBA::LifeCycleObject,
                                CORBA::TransactionalObject,
                                CORBA::PO
       {
              // Name of object version.  A shorthand would have been
              // to declare Name as an attribute.  This would
              // automatically cause the generation of   GetName and  SetName
              // implementation methods.

              // Implementation of  GetName and  SetName uses  NamingContext
              void SetName(in string  newname);
              string  GetName();

              // Roles to notify in case of modification.  Implementation
              // uses  EventChannel.
              void AddRoleToNotify(in Role  addrole);
              void RemoveRoleToNotify(in Role  remrole);

              // Called in case of  ObjectVersion modification.
              // Implementation uses  EventChannel to notify roles.
              void OnModify();
       };


       typedef sequence<ObjectVersion>   OVList;

       interface  Object:CORBA::LifeCycleObject,
```

```
                        CORBA::TransactionalObject,
                        CORBA::PO
        {
                // Implementation of  GetName and  SetName use  NamingContext
                void SetName(in string  newname);
                string  GetName();

                // Object Versions of this Object.  Sequence used because
                // CORBA collection service not finalised.

                // Create a new version
                 ObjectVersion  NewVersion();

                // Delete object version
                void  DelVersion(in  ObjectVersion  delv);

                // Get entire version list
                OVList  GetVersionList();

                // Get most recent v ersion
                ObjectVersion  GetRecentVersion();
        };
 };
```

FIG. 8: IDL definition of Object and Object Version

The development of other CORBA services is ongoing; we expect the examples of re-use given here to be the first of many. It is expected that CIMM objects will re-use additional CORBA services as they are finalised, particularly in the area of rule-based, trader and system management services. At present, there are around 20 concepts in the CIMM which are construction industry independent and are concerned purely with information management. These make use of the small set of CORBA objects already described and are being tested with around 30 construction specific concepts.

4.3. Integrating Existing Software with the CIMM

For the CIMM to be effective, it should be possible to integrate it with both existing CIC systems and legacy applications. This existing software could then take advantage of the CIMM's information management functionality. Three categories of existing software and systems that have the potential to be integrated are described below.

4.3.1. CIC systems which use shared databases

One of the most favoured methods of information integration in construction research is to use shared object-oriented database (OODB) technology. This type of CIC system can be integrated using inheritance; concepts in the database model are specialised from concepts in the CIMM. This has the effect of converting the CIC system to a set of CORBA compliant distributed components which are managed by the CIMM. Although the existing CIC models must be translated to IDL interfaces (a process which can be made easier by the use of CASE tools), much of the original implementation and integration code can be retained. This approach has advantages - the full functionality of the CIMM is utilised to produce a system which is easily extended by adding new CORBA compliant components. However, potential difficulties arise in cases where the existing CIC functionality overlaps with the CIMM. This strategy has been chosen to test both the feasibility of integrating the CIMM and its effectiveness for managing information. The existing CIC system to be integrated has been developed by a companion project (the OSCON Project) and uses an integrated core model and the ObjectStore object-oriented database (Tracey et al., 1996).

4.3.2. Construction Industry Software Applications

It is not unreasonable to describe most construction industry software packages as monolithic legacy applications. For such systems, the integration strategy is similar to that adopted by CIC systems based on OODBs. Object wrappers for the legacy application are modelled and coded, but, rather than being in the language of the OODB, they are described in IDL. The IDL object wrappers can then be specialised from CIMM concepts. Again, this uses the full functionality of the CIMM and effectively converts the application into a set of distributed components.

4.3.3. Component Based Applications

Applications based on distributed components are still in their infancy, but will undoubtedly appear over the next few years. CORBA's DII (Dynamic Invocation Interface) makes it possible to integrate such applications at run time (that is, with no code changes or recompilation). It would, for example, be possible for the CIMM to use the DII to query a component, and subsequently control actor's rights to perform operations on that component. Full CIMM functionality would, however, only be available if components were specialised from CIMM concepts.

5. THE CIMM BROWSER

In order to make a distributed CIC system effective, it is necessary to provide users with a uniform user interface through which they can access project information. This information consists of not only the usual concepts found in construction integration models (for example, buildings, walls, and tasks) but also the CIMM's information management data (for example, roles, actors, rights and responsibilities). The CIMM organises project information into discrete units accessible by a project's actors. It also contains the basic concepts for implementing a distributed Electronic Document Management (EDM) system. The CIMM's user interface reflects this organisation. As there is an overlap between information management and project management, the CIMM browser shares some of its functionality with project management systems. It differs, however, in that it provides a means of controlling the distributed components of an integrated construction environment.

Ideally the CIMM browser would be implemented as a set of distributed user interface components. Unfortunately, there are two reasons why current technology does not offer sufficient support for this approach. Firstly, distributed user interface components (for example OpenDoc or OLE) are not as yet sufficiently mature or platform independent. Secondly, as most ORB's use the Internet to communicate with each other, low bandwidth often renders the transmission of data intensive user interface components impracticable. The CIMM user interface has therefore been implemented as a standalone application which can interact with the distributed components of the CIMM through the use of object proxies. The software is being developed using Visual C++ Version 4.0 for the Windows NT and Windows 95 platforms.

Fig. 9 shows the CIMM user interface panel in which the project's actors are described and managed. Through this interface any actor may browse the details of other actors, or, in the case of the information manager, specify project participants. In addition to describing the contact information, physical location and main discipline of each actor, the panel describes the roles each actor performs. Within the distributed CIC, it is therefore possible to see exactly who is responsible for carrying out a given activity.

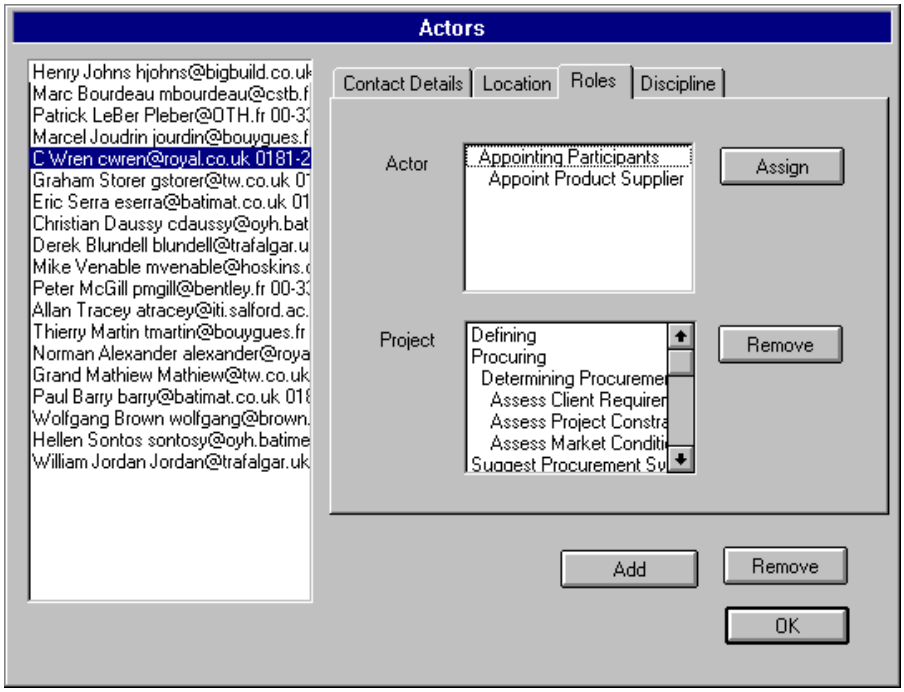

FIG. 9: CIMM User Interface - Actors

In addition to managing the actors involved in a project, it is also important to manage the roles or tasks they perform (Fig. 10). For each role, the CIMM user interface describes the actors who will be involved, the resources this role uses (hardware, software, equipment or human), the timeframe for the performance of the role, the costs of the role and any quality standards that need to be adhered to.

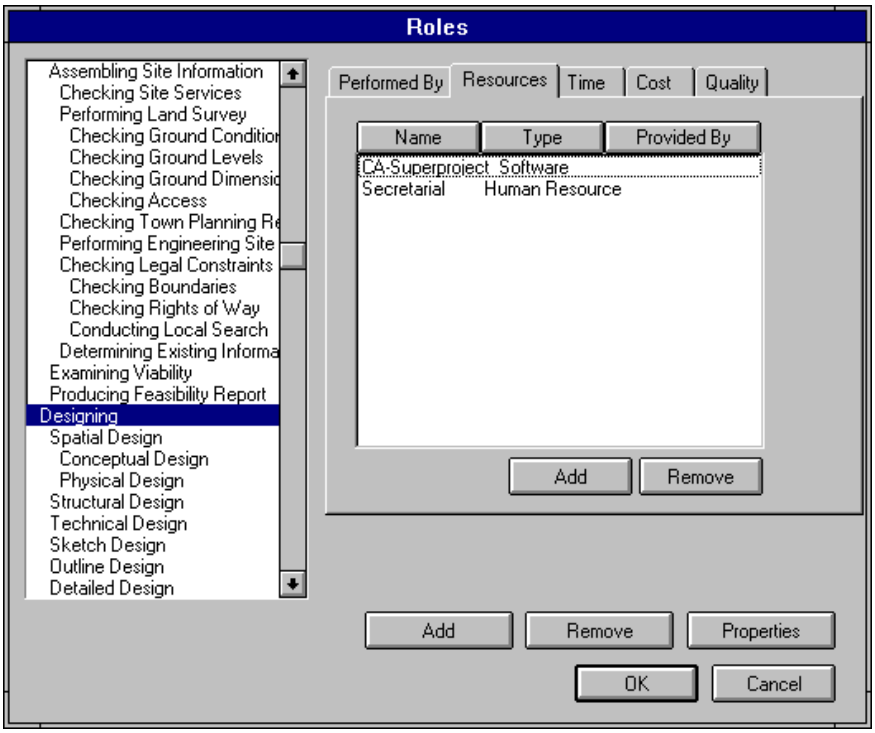

FIG. 10: CIMM User Interface - Roles

Fig. 11 shows the user interface panel through which actors browse managed project information. This information includes the building core concepts (walls, beams, etc.) which constitute the project information base - it is the CIMM's view of the distributed components which make up the whole CIC environment. Of course, much of this information will be created and amended through the use of construction industry software applications. It is, however, important to allow actors to interact with information in a way that is application independent. The management properties of information must also be made explicit. For example, a given object has a creator and a creation date and an intent behind its creation. Note that the panel describes objects at both the type and instance level; the prototype supports schema evolution through the creation of new versions of object types. This is described more fully elsewhere (Rezgui et al., 1996b).

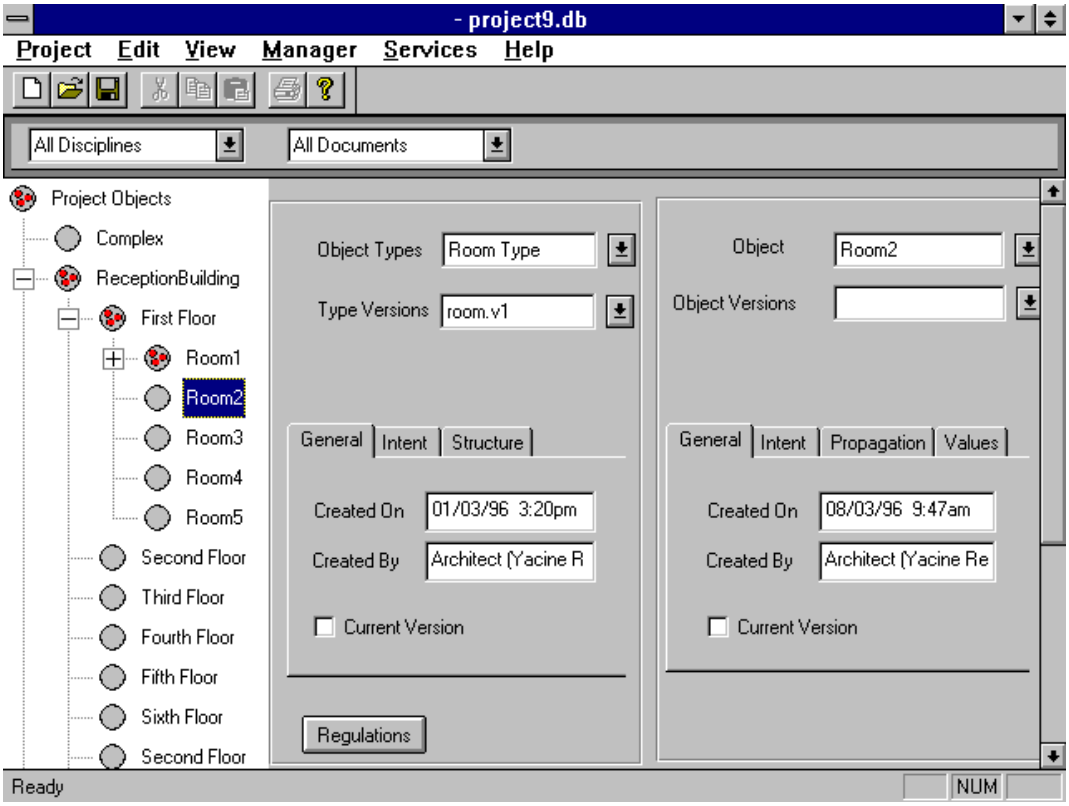

FIG. 11: CIMM User Interface - Objects

One of the aims of the COMMIT project is to provide a migration path for the introduction of intelligent distributed CIC. Project participants should be able to access information in a familiar way, and current practices evolved to fully take advantage of the benefits of integration and distribution. To this end, the CIMM models include the concept of interpreted object, which may represent a document or other interpretation (e.g. multi-media) of a core concept. Fig. 12 shows the CIMM user interface for accessing interpreted objects. Treating documents as another form of managed information has several additional benefits; for example, all the project's participants share documents and document organisation and access to documents may be controlled according to rights.

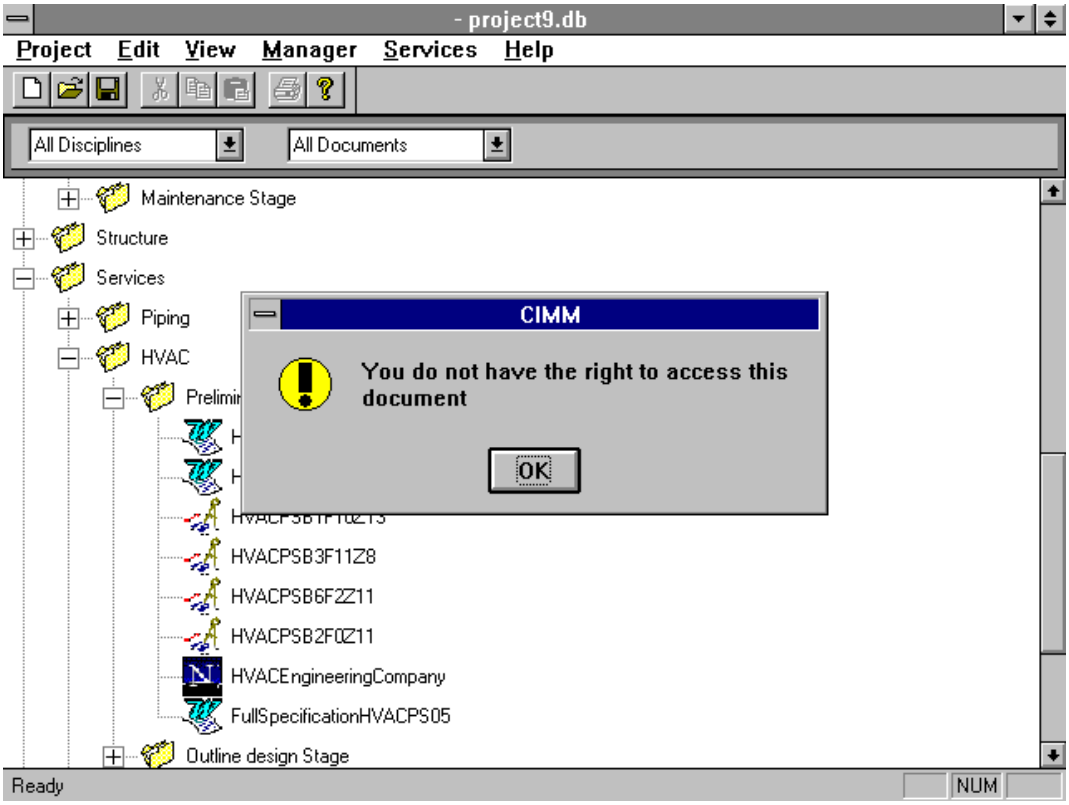

FIG. 12: CIMM User Interface - Documents.

6. CONCLUSIONS

To conclude this paper we would like to stress the potential of distributed objects in computer integrated construction. The paper has outlined the benefits of adopting distribution standards such as CORBA, and described a framework for distributed computer integrated construction. In this framework the COMMIT Information Management Model acts as a bridge between objects in an integrated construction system and CORBA services. In the CORBA methodology, the CIMM is equivalent to a business object facility for the construction industry.

This paper has presented a distributed implementation of the CIMM, and shown how CIMM objects can be specialised from CORBA service objects. Techniques for integrating the CIMM with existing CIC systems and legacy applications have been presented. The CIMM user interface, which allows project participants to manage distributed project information, has also been described.

The work presented here is ongoing; the prototype demonstrating the ideas described in this paper is currently being refined. In the longer term, it is hoped that the COMMIT project, which is supported by a UK steering group comprising regulation bodies, research institutions and industrials, will successfully demonstrate the technical, economic and sociological benefits that would be gained by adopting an approach to information integration founded on an object-oriented information management model and distributed object technology. While the work presented here is specific to the construction industry, many of the ideas behind it are generic. Future work also includes generalising the CIMM across industries.

7. ACKNOWLEDGEMENTS

The authors would like to thank Dr G. Aouad, Mr J. Kirkham and Dr Z. Turk for their helpful comments on this paper. The authors also wish to acknowledge the financial support of the EPSRC, and the assistance provided by members of the steering committee throughout the duration of this research project.

8. REFERENCES

Aouad, G. et al., (1994), *ICON Final Report*, University of Salford. http://www.salford.ac.uk/docs/depts/survey/staff/GAouad/pubs.html

Bjoerk, 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

Brocksmidt, K. (1995), *Inside OLE-2*, 2nd Edition, Microsoft Press.

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

Eastman, C. and Kutay, A. (1991), Transaction Management in Design Databases, *Concurrency in Engineering Data Management*, eds. D. Sririam R. Logcher and S. Fukuda, Springer, New York.

Froese, T. and Paulson, B. (1994), OPIS: An Object Model-Based Project Information System, *Microcomputers in Civil Engineering*, No. 9, 13-28. http://maillist.civil.ubc.ca/~tfroese/pubs/

Gosling, M., and McGilton, H. (1996), The Java(tm) Language Environment: A White Paper, Sun Microsystems.
http://java.sun.com:80/doc/language_environment/

ISO/TC184/SC4 (1994), STEP Part 1: Overview and Fundamental Principles, International Standard, ISO, Geneva, (11). http://www.igd.fhg.de/www/igd-a2/hyperstep/iso-10303/part1/gen.html

Khedro, T., Genesereth, M. and Teicholz, P. (1994), Concurrent Engineering Through Interoperable Software Agents, Technical Report, Stanford University. http://www-leland.stanford.edu/group/CIFE/cifepubs.html

OMG (1995a), The Common Object Request Broker: Architecture and Specification, OMG.
http://www.omg.org/corbask.htm

OMG (1995b), The Common Object Request Broker: Services, OMG. http://www.omg.org/public-doclist.html

OMG (1996), Common Facilities RFP-4: Common Business Objects and Business Object Facility, OMG TC Document Number 96-01-04. http://www.omg.org/public-doclist.html

Orfali, R., Harkey, D. and Edwards, J. (1996), *The Essential Distributed Objects Survival Guide*, John Wiley & Sons.

Orfali, R., Harkey, D. and Edwards, J. (1995), Intergalatic Client/Server Computing, *BYTE April 1995*.
http://www.byte.com/art/9504/sec11/art1.htm

Rezgui, Y., Brown, A., Cooper, G., Yip, J., Brandon, P. and Kirkham, J. (1996a), An Information Management Model for Concurrent Construction Engineering, to be published in *Automation in Construction*.
http://www.salford.ac.uk/iti/projects/commit/#Papers

Rezgui, Y., Brown, A., Cooper, G., Brandon, P. and Betts (1996b), M., Intelligent Information Versioning Support In The Context Of Collaborative Construction Engineering, *Proceedings of the 1st international conference on computing and information technology for AEC*, Singapore.
http://www.salford.ac.uk/iti/projects/commit/papers/iivs/abstract.html

Su, S., Lam, H., Yu, T., Lee, S. and Arroyo, J. (1995), On Bridging and Extending OMG/IDL and STEP/EXPRESS for Achieving Information Sharing and System Interoperability, *Proceedings of the 5th annual express user group international conference*, Grenoble, France.

Tracey, A., Child, T., Aouad, G., Brandon, P. and Rezgui, Y. (1996) Developing Integrated Applications for Construction: The OSCON Approach, *Proceedings of the 1st international conference on computing and information technology for AEC*, Singapore.
http://www.salford.ac.uk/iti/att/oscon.html

Turk, Z. (1993). Object Oriented Modelling Techniques and Integrated CAD, *Automation in Construction*, Vol. 1, 323-337. http://www.fagg.uni-lj.si/~zturk/biblio.htm

Available electronically at http://itcon.org/