# ID'EST: An Integrated Modelling Framework for Management of Architectural Data

*Thomas Liebich*

CAB, Osterwaldstrasse 10 D-80805
Muenchen
Germany

*Inhan Kim*

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

*An Integrated Design Environment, IDE, facilitates cooperation between different disciplines. The paper investigates the data modelling framework, distinguishes between homogeneous and heterogeneous model worlds, discusses the formal mapping mechanisms available to establish a heterogeneous model world, and introduces a way to incorporate CAD systems into IDE. A prototype IDE has been developed to prove these methods. The ID'EST prototype comprises its own core data model, different schemas to cope with several design views, and interfaces to incorporate external CAD systems. A prototype architectural data model has been defined, that includes core data models and aspect models for enclosure system and spatial system. Conventional CAD systems can be integrated into ID'EST, if they are able to map data from the aspect models into their own data structure, and vice versa, on a high semantic level. The inherent methods of classifying data in CAD, layers, macros and attached attributes, have been used to retrieve product data from CAD data files. The usability of conventional CAD systems as data instantiation tools for IDE has been proved and a path has been shown, by which existing tools can be integrated into new technology solutions.*

*Keywords Product Modelling, Formal Mapping Specification, Computer-Aided Design*

## 1 Introduction

The vision of Integrated Design Environments (IDE) promises collaborative support for the multi-disciplinary nature of the building design process. IDE should enable design teams to efficiently cooperate within a virtual design office, and to easily predict the performance of buildings in order to improve the quality of design. Therefore IDE has to accommodate the multiple views of design disciplines and, vice versa, to support the generation of multiple design representations. Building design and evaluation tools have to work interoperably via IDE, which controls their consistency. This approach requires semantically meaningful descriptions of all relevant building data, high-level data interfaces, and the breakdown of IDE into a modular structure of manageable pieces for instantiation, storage and retrieval, as well as for classification, propagation and projection.

An IDE basically relies on a core data model and different specific models, often being structured according to product model definition of its domain. The basic concept,

used in product models to describe objects of interest, is the entity within an inheritance hierarchy. The entity is specified in terms of its attributes and constraints. All entities are related to each other, mainly by objectified relationships. Thus, product modelling incorporates certain characteristics of well-known object-oriented techniques, such as data encapsulation and inheritance. An homogeneous model world, however, where all partial models use the same set of integrated resources is hardly to be achieved in a domain, such as building and construction. Taking the variety of different abstractions made by the various actors of the design process into account, a heterogeneous model world is more likely to reflect the nature of building design. Heterogeneous models use different conceptual definitions for their views on the same design object. The general definition of objects and relationships are established by the core data model, that reflects a common agreement for the IDE models. Consequently, the correspondence between the aspect models has to be formally defined in order to establish the integration across the aspect models.

The data management system of IDE has to control and interpret the collection of design data instances. The data management system has a design tool control module, which facilitates the processing of data. All issues in relation to the control mechanisms for the data communications between the various design stages and design actors are described in [Kim and Liebich, 1995] in more detail.

CAD systems are treated as one of the several computer-based design tools acting around an IDE. Their specific usage relates to the instantiation process, as they are mainly used to create design instances, which now have to conform to the semantic level of the IDE data models. The new generation of proposed object-oriented CAD systems will certainly be able to provide those design instances [Junge and Liebich, 95], in the meantime, however, conventional CAD systems can be used through special conventions.

## 2 Shortcomings of conventional CAD systems

Conventional CAD systems have inherent bottlenecks, which diminish possible achievements for architectural practices. First, almost all conventional CAD systems rely on a pure geometric data model. All non-geometric information about objects of architectural interest has to be attached to these geometric entities. This restricts the ability to describe semantically dependent relationships. Second, the data exchange remains restricted, since it is based on a fairly low semantic level of a document-based exchange of information, such as geometric representation in DXF or IGES, rather than on a high semantic level of a model-based exchange. In consequence, the integration of different design tools for building and construction is still very limited. Thus, different information about the same building object, such as a wall, (e.g., drawing symbols in CAAD systems, cost calculations in spreadsheet programs, tender documents in word processors, and results of compliance checks in simulation tools) cannot be exchanged between the different design tools. CAD systems fail to provide an integrative role due to deficiencies in their underlying database.

### *2.1 Use of the inherent structure of CAD*

In IDE, conventional CAD systems can be used as data instantiation and modification tools with special conventions and some restrictions. CAD systems are able to create the geometrical representation of building objects, and allow additional semantic information as needed for input into the data model. Therefore their inherent structure of classifying data has to be used. Methods of structuring CAD data are:

(a) layers

Layers are mainly used to control visibility. In addition they offer the possibility to group sets of data, e.g., all load-bearing walls of ground floor. Many CAD systems still restrict the use of layers so that any entity is only permitted to be assigned to one particular layer.

(b) macros (e.g., blocks or parameterised macros) Macros can be used to label and control all geometric entities, that are representations of one building element. Thus, they offer grouping mechanisms on instance level, necessary to maintain unique identifiers for the bi-directional exchange of meaningful descriptions of building elements.

(c) attached attributes (e.g., extended entity data)

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

Those structures have to be used in a standardised manner to allow for the establishment of a semantically rich data exchange. Naming conventions labelling layers and macros provide such a method to enable the extraction of appropriately structured CAD data for IDE. Some restrictions still remain, since semantic and mainly bi-directional relationships can not be sufficiently expressed in conventional CAD. The support of a beam on a footing is a relationship having its own attributes, e.g., the type (free or restraint) and the bearing pressure. Their expressions in extended entity data sets do not have the power of the corresponding IDE data model constructs, and they are hard to achieve and to manage.

## 3 Data Modelling Framework

There is now a broad consensus that a conceptual or more precisely a product model is the key to intelligent data exchange and collaboration between different applications. It is particularly the case that the development in relation to ISO-STEP standardisation efforts [Boyle, 1992] pushes the process ahead, as it provides appropriate methods, languages, and standardised resources, as well as encouraging the production of necessary tools.

The principles of conceptual modelling are assumed to be known [Gielingh, 1988, Wilson, 1994 or Junge and Liebich, 1995], thus the discussion is now focused on the different ways being proposed to model a complex universe of discourse. The domain of building and construction is a signific example of this kind of complexity, as

(a) it includes several disciplines with very different views on the elements of buildings,
(b) it copes with many design, construction and maintenance stages during the life cycle,
(c) it deals mainly with one-of-a-kind products,
(d) it establishes several and varying relationships among the participants, and
(e) it depends heavily on collaboration among the actors of independent organisations.

The first approach in defining one global data model that includes all aspects of any element of buildings has to fail. It seems impossible to deal with the whole universe in a single model and to derive the entire spectrum of views from it.

The second approach tries to establish an integrated data model, being structured according to different modelling layers. The layered structure of the STEP model world follows this approach, since the outermost level of Application Protocols has to use and to specialise only constructs being already modelled within the inner layers of generic and integrated resources. Within the COMBINE project, an Integrated Data Model, IDM, has been defined, consisting of a variety of schemas, and an interface kit has been developed that provides data exchange facilities, through which the design tool prototypes can communicate with the IDM [Augenbroe, 1993].

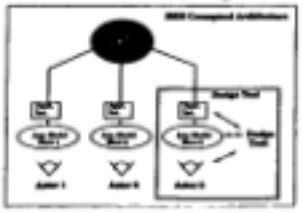

Figure 1 the COMBINE 1 integrated data model and interfaces to the applications

These projects have in common, that they establish communication between several design tools through definition of a homogeneous world of models, that allows these tools to send and receive data, if they are able to map their own data abstraction into the data abstraction of the integrated data model [Figure 1]. This approach certainly leads to the desired results, especially under non-commercial circumstances. Several questions, however, remain to be answered, such as how does the whole model framework reacts on changes that might be done in some partial models? If the model world is not established

within a standardisation project, like STEP, where results are per se public, another question arises; who owns the indivisible integrated data model? An answer would certainly be to divide the entire model world into several relatively independent pieces of partial or aspect models.

This third approach tries to establish a more flexible structure of loosely integrated aspect models. Within the COMBI project the focus has been shifted toward such a modelling framework, that includes translation mechanisms between partial models, if design data have to be exchanged between disciplines [Junge et al., 1995]. The several aspect models, each dealing with a specific view of design disciplines, are relative independently defined. They all rely on the common agreement of basic modelling constructs described within the core data models, that preserves the minimum of necessary commonality. That includes the root objects and basic hierarchies of product items, relationships between them, and the systems in which the product items play a role. The aspect models then focus on views, specified according to the needs of design disciplines and life cycle stages. In order to establish this heterogeneous model world, the correspondence between the diverse models has to be defined at the same high conceptual level, as the core and aspect models themselves.

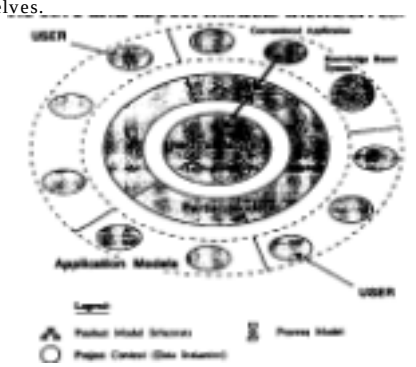


Figure 2 the COMBI modelling framework, neutral (core), partial (aspect) and application models

This understanding has led to many research efforts, seeking for formal methods to describe the correspondences between heterogeneous models, in order to utilise the communication within the integrated design environment. Formally specifying a mapping mechanism allows the developer to determine the conversion of all required data between the data models. With a mapping defined, it is then possible for a mapping implementation to extract data from a source model for a target model and, additionally, to move data back and guarantee the consistency of the two models in respect to each other [Figure 3].

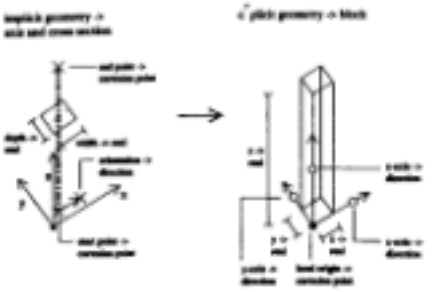

Figure 3 example of a mapping problem between two different geometric abstractions

Mapping methods are not only applicable to the common integrated environment layout, where several application tools are grouped around a central integrated model.

Even for a loose integration without any established integrated model or for the exchange of information by request [Genesereth and Fikes, 1992], mapping can still work as the backbone of the integration. A third field of integration occurs with the requirement to update an integrated model to accommodate the needs of more advanced design tools. Then, mapping provides a mechanism to denote how to propagate data back and forth between older and newer versions of an integrated model [Amor and Hosking, 1994]. A forth and specific field to apply mappings is the CAD integration. Not all principles of the formal mapping specification are applicable here, as the CAD data structure does not follow the principles of product models. The paper will discuss this aspect later.

## 4 Prototype Architectural Data Model

ID'EST (acronym for Integrated Design Environment using STep methodology) is a prototype IDE developed to test the principles of integrated environments (such as data models, data interfaces, view dependencies) as well as to implement the CAD integration into the environment. Within this paper the main structure of ID'EST is only briefly discussed the emphasis being placed on the proposed prototype data model and CAD integration part [Figure 4].

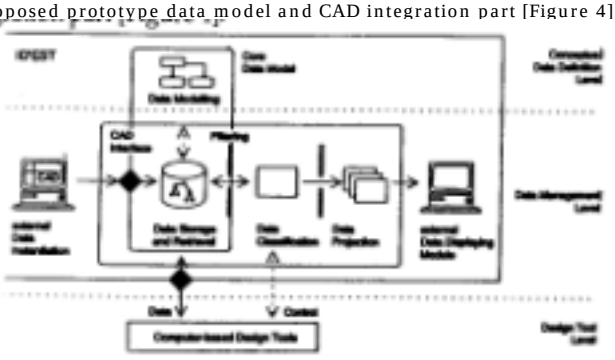

Figure 4 ID'EST system overview

ID'EST uses a set of product models principally developed according to the methodology of STEP to define the Prototype Architectural Data Model (PADM). In particular it uses the layered approach to structure the model. The PADM defines the structure of the data instantiation tool, as well as the syntax of the exchange format. The design tools, which are connected to ID'EST, communicate on the basis of the STEP physical file, the common exchange format. The permanent storage of the design data within the environment has been realised by an object-oriented database [O2 Technology, 1994].

During the first phase of analysis the PADM was divided into two main layers, each consisting of several schemata [Figure 5]:

the layer of building and construction comprises the schemata building_project and building_system,

the layer of specific building systems comprises the schemata enclosure_system and spatial_system.

All schemata make use of integrated resources following the STEP methodology. Therefore a subset of the generic resources for topology, geometry, and geometric_model [ISO DIS 10303-42, 1993] has been defined as application integrated resources. Development during analysis has been performed using EXPRESS-G [ISO IS 10303-11, 1994a]

Figure 5 Schema level diagram of ID'EST

During the second phase of design more detailed specifications, such as constraints, functions, and rules, have been added to the model. The graphical notation has to be mapped into a textual form. EXPRESS offers the functionality to include these specifications, although there are still some restrictions. In particular, EXPRESS does not support the definition of behavioural aspects of objects, and in this sense it does not provide all features of the object-oriented paradigm. The design model also takes some implementation requirements into consideration.

During the third phase of implementation the EXPRESS definition was mapped into an application form. The processing tools, developed at NIST [Clark, 1993] were used during the development process of ID'EST. The layered data model specifies all entity types that represent objects and relationships within the model. Based on this specification, the data instantiation module allows the definition and the exchange of unique entity instances. In consequence, every time the data model has changed, the data instantiation module has to be recompiled.

This consequence led to a new concept of PADM, that is now in a pre-development stage. In order to restrict the influence of modifications and to ease the process of adding new aspect models to the PADM, the more flexible structure of loosely integrated data models and mapping mechanisms established among them has been chosen. A new aspect model for preliminary structural design should be included within the modelling framework. Therefore, the aspect model has been defined according to the core model definition, but it includes different abstractions for e.g., the geometric representation items and the structural element connectors. In order to allow for data integration across the disciplines (here: aspect models) a mapping has to be defined that translates the STEP physical file according to the architectural enclosing system of the sender into the STEP physical file according to the preliminary structural system of the receiver. The mapping itself can be established at a formal level with the help of specification languages, such as EXPRESS-M [ISO, 1994b] or EXPRESS-C [ISO, 1994c], and there already exist generators to compile an executable code [ECCO] or to interpret the mapping rules [XPDI station.

### *4.1 Data instantiation, classification and projection modules*

The PADM is the prerequisite for the implementation form of ID'EST, which consists of several modules:

(a) Data instantiation module
(b) Data classification module
(c) Data projection module

The main objective of the instantiation module is to have an instance manipulation environment for the semantically structured entities defined in the suggested data model. An instantiation module should provide a mechanism to input data in a graphic way and to classify data.

In ID'EST, DataProbe [Sauder, 1993] has been used to create, edit, or view instance data corresponding to the unified data model for which it was created. DataProbe also facilitates the reading, merging and writing of STEP physical files [Figure 6]. On the other hand all project data are usually created by an external CAD system. In this case, it is necessary to use an accessory tool to convert the CAD data format to the STEP physical file format, in order to import data to the instantiation module. This interface is described next.

Figure 6 Data instances in ID'EST

## 5 CAD integration in ID'EST

The creation of building design instances is normally done using conventional CAD systems for the input of geometry. Beside the geometrical representation, more semantic information is needed to extract data for input into the instantiation model. Therefore, the inherent structure of CAD systems plays a key role in enabling the extraction of appropriate data. Naming convention gives the opportunity to map this implicit information into the object-oriented data representation being used in ID'EST.

Within ID'EST AutoCAD was chosen as the particular CAD system and the AIA layer guide [Schley, 1990] as the main naming convention. In order to maintain consistency between building elements within the data instantiation module and their appropriate CAD representation, the macro structure was intensively used to instantiate unique instances of building elements within the external CAD package. In addition the layer system was used for major grouping, e.g., all building elements within a specific floor level. Each macro in the external CAD file is further defined by attached attributes, describing properties, such as material, building code, price, etc. To import such a structured CAD data file into ID'EST a STEP/DXF interface had to be built.

### *5.1 The STEP/DXF interface*

The STEP/DXF interface, which was developed as a part of the proposed environment, enables the reading of DXF file and translates it into STEP physical file [ISO DIS 10303-21, 92], according to the given aspect model. The converter of the STEP/DXF interface incorporates several mechanisms to map structured CAD data into the relevant structure of the data model:

(a) mapping code for geometric entities
(b) mapping tables for entity type information
(c) mapping rules for non-geometric attributes

### *5.2 Mapping of explicit geometry*

This element of the STEP/DXF interface is faced with converting the geometric entities representing a building element in CAD into an entity data type for the has_representation attribute, which is part of the entity definition of all building elements, as an result of the core data model agreement. This mapping part includes one-to-one mappings as well as one-to-many mappings [Figure 7]. The mapping of geometric entities follows an early binding philosophy, i.e., the converter has to be modified and recompiled whenever the internal structure of either the CAD data format or the schema definitions of geometry or topology within the product model changes. This is, however, an acceptable approach, as the definitions of geometric items are unlikely to be frequently changed.

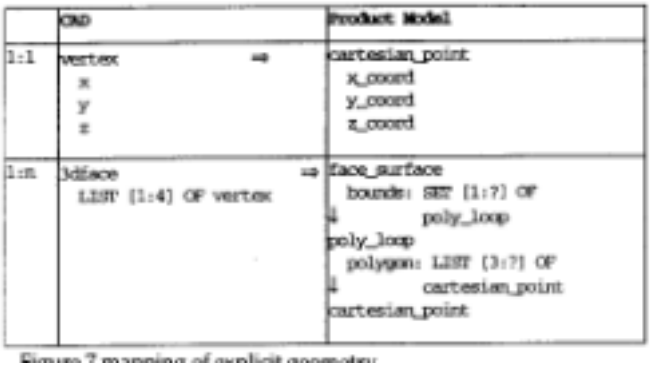


Figure 7 mapping of explicit geometry

*5.3 Mapping tables for entity type information*

In order to facilitate a semantically meaningful data exchange between CAD and ID'EST the geometric entities in CAD have to be characterised as representation of building elements. Therefore the macro and layer structure is used in connection with the AIA naming convention. According to AIA layer guide a layer name using the short format, has the principle form xyyzz-zz, where x is the major group, representing the different disciplines (e.g., A = architecture, S = structural engineering), yy is the minor group, designating construction systems (e.g., wa = wall, co = column, cl = ceiling), and zz is an user-defined minor group. The same convention was adopted for labelling macros. In order to map the building element macro definitions into specific entity type instances DXF/STEP interface makes use of user defined mapping tables [Figure 8].

Figure 8 CAD macro structure and mapping table

These mapping tables are read during runtime. They follow a late binding philosophy, since their definitions could be sent together with the physical file. Thus other naming conventions can be easily adopted within ID'EST.

*5.4 Mapping rules for non-geometric attributes*

Non-geometric information, usually stored in attached attributes is also converted as. Contrary to the fixed structure of geometric entities in CAD and to the standardised naming conventions for layer and macros, there is no standard by which attributes may be specified. Thus a more flexible approach to define the mapping between attributes in CAD and entities or entity attributes in ID'EST is required. The final aim is to create an universal mapping language to define rules, which control the mapping during runtime.

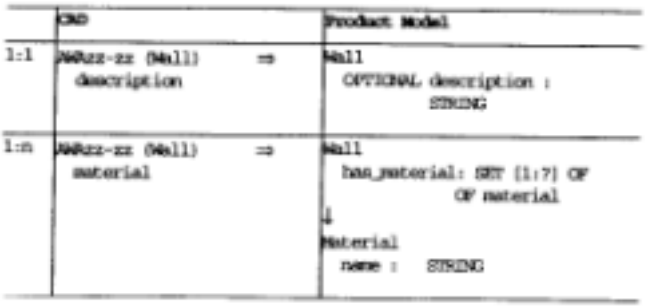


Figure 9 mapping of attached attributes

Again there are two kinds of mapping: one-to-one and one-to-many, and for each attribute-to-attribute and attribute-to-entity mapping [Figure 9]. In the case of one-to-one, an attached attribute is mapped into a defined data type having the same name or an alias for that name. In the case of one-to-many, an attached attribute is mapped into an attribute pointing to an entity having the same name or an alias for that name and in addition a defined data type to carry the value of that attached attribute [Figure 10]. The next figure shows the final output of a STEP physical file according to the PADM [Figure 11].

Attribute mapping is controlled by rules. They have the principle structure of:

attribute::entity.entity_attribute{.entity_attribute}

The rules for the two mapping pairs, as given in figure 10, are:

description::SELF.description
material::SELF.has_material.name

whereas SELF refers to that entity type, which was found with the help of the mapping table.

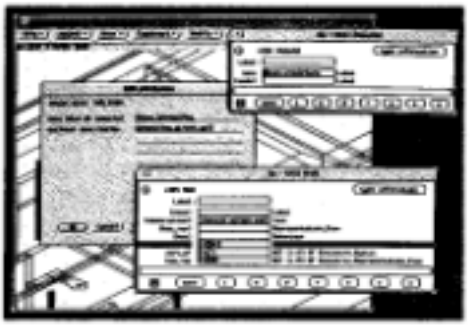

Figure 10 mapping of attached attributes in DXF/STEP interface

These rules reflect only a subset of possible and necessary rules usually included in a formal mapping language, since on the left hand side only simple attributes can exist according to the limitation inherent in the conventional CAD data base. Similarly, other methods, e.g., allowing for conditional mappings, are not included. If this functionality is needed, a twofold mapping process has to be implemented in the DXF/STEP interface. First, the CAD data is mapped into an external format, structured according to product model definition, such as the application protocol for explicit shape representation (ISO, 1995), and second the mapping is defined between the external format and any of the aspect models, the data should be mapped to.

Figure 11 the output of the DXF/STEP interface

## 6 Conclusions

The author argues that the human designer should have all the flexibility possible to facilitate his or her work and full responsibility for the project in any design system. The designer should receive all possible support for the crucial parts of the design work, such as collaboration and consistency. The suggested design environment should lead the way to the "ideal CAAD environment" a designer might want to use in future.

Mapping specifications, in the general scope of model mapping as well in the more narrow scope of CAD integration, are certainly a way to allow for the integration of heterogeneous models. The requirements coming from practise will put the emphasis on non-centralised and diverse model frameworks. Another requirement is the further use of existing technology solutions and skills of the professionals, even if new integrated design environments will be commercially available. The CAD integration shows a smooth integration path for the intermediate time. Until the new generation of object-oriented CAD systems for the ideal environment is available, conventional CAD systems can be used as data instantiation tools with special conventions.

## 7 Acknowledgement

This work was partially funded by the E.C., Directorate General XII under the Human Capital and Mobility Program at the host institute: ABACUS unit, Dept. of Architecture, Univ. of Strathclyde, Glasgow.

## 8 Endnotes

[1] The ECCO Tool kit is an implementation of the EXPRESS-C language, developed at the University of Karlsruhe. The ECCO Tool Kit can be used to generate an executable directly from the EXPRESS-C or EXPRESS specification. Although the use of EXPRESS-C for mapping has been considered as a side effect, ECCO can also be used to implement mappings, being specified in EXPRESS-C.

[2] The XPDI station is a STEP based tool being developed at CSTB Sophia Antipolis, France. It includes components to develop EXPRESS schemas in graphical and in textual form. A Lisp late binding of SDAI enables the dynamic interpretation of a rule based language, that is also used to convert source models into target models.

[3] The data classification and projection modules are described in [Kim and Liebich, 95].

## 9 References

Amor, R.W., Hosking, J., 1994. Mappings: the glue in an integrated system. 1st European Conference on Product and Process Modelling, Univ. of Dresden, Germany

Augenbroe, G.L.M. (ed.), 1993, COMBINE, Final Report JOUE-CT90-0060, Delft University of Technology, Netherlands

Boyle, A., 1992, Step: An Introduction. Technical Report: SERC CIPM/LU/ TP/1, Department of Civil Engineering, Universtiy of Leeds, U.K.

Clark, S.N., 1993, An Introduction to the NIST PDES Tool Kit. NISTIR 4336, National Institute of Standards and Technology, USA

Fenves, S.J., Flemming, U., Hendrickson, C., Maher, M.L. and Schmitt, G., 1990, Integrated software environment for building design and construction. Computer-aided Design 22, pp. 27-36

Genesereth, M.G. and Fikes, R.E., 1992. *Knowledge Interchange Format Version 3.0 Reference Manual.* Computer Science Department, Stanford University, CA

Gielingh, W., 1988, General AEC Reference Model. Proceedings of CIB W74 + W78 Seminar, Lund Sweden, pp. 165-178

ISO DIS 10303-21, 1992, STEP Part 21: Clear text encoding of the exchange structure. ISO TC184/SC4/WG5

ISO DIS 10303-42, 1993, STEP Part 42: Geometric and Topological Representation. ISO TC184/SC4/WG3

ISO IS 10303-11, 1994a, STEP Part 11: EXPRESS Language Reference Manual. ISO TC184/SC4/WG5

ISO, 1994b. WG5 "PISA Information Modelling Language: EXPRESS-C". Working Draft, TC184/SC4/WG5

ISO, 1994c. EXPRESS-M Reference Manual. Working Draft N51, TC184/SC4/WG5

ISO, 1995. STEP Part 225: Building Elements using Explicit Shape Representation. TC184/SC4/WG3Project Draft

Junge, R. and Liebich, T., 1995. New Generation CAD in an Integrated Design Environment: a Path towards Multi-Agent Collaboration, accepted paper for Tan, M. (ed.), CAAD futures '95, Singapure

Junge, R. and Liebich, T., 1995. Product modelling for applications: Models for next generation CAAD. VI. Int. Conference on Computing in Civil and Building Engineering, Berlin, Germany

Junge, R., Liebich, T., and Ammermann, E., 1995 Product modelling for communication: the COMBI approach. VI. Int. Conference on Computing in Civil and Building Engineering, Berlin, Germany

Kim, I., Liebich, T., 1995, Representations and Control of Design Information in an Integrated CAAD Environment, accepted paper for Tan, M. (ed.), CAAD futures '95, Singapure

Liebich, T., 1994, Managing Design Data: Including CAD in Integrated Environments. 1st European Conference on Product and Process Modelling, Univ. of Dresden, Germany

O2 Technology, 1994, O2C Reference Manual. O2 Technology Ltd., Horsham, UK

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

Schenck, D., Wilson, P., 1994, Information Modeling the EXPRESS way. Oxford Univ. Press, New York

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