New Generation CAD in an Integrated Design Environment: A Product Model Approach

Richard Junge and Thomas Liebich

Computeranwendung in der Bauplanung
Osterwaldstrasse 10 D-80805 Muenchen Germany, Tel No: +49-89-369030, Fax No: +49-89-363801
e-mail addresses: junge@cab-muenchen.de and liebich@cab-muenchen.de

Summary: Product Modeling is considered to be an established concept not only for semantically based data exchange, but also for the specification of models, dealing with specific application requirements. The product model approach is regarded to be one step towards a new generation of Computer Aided Architectural Design, and to provide underlying means for enabling communication between different applications on a semantic level. After on overview about the background and the basis principles of product modeling, the authors discuss how product models can be used in commercial developments and in applied research projects.

Keywords: Product Modeling, STEP, Computer-Aided Design, Data Integration

1. Introduction

There are projects dealing with new generation CAAD and with integrated or intelligent design environments at many places. Most of them are scientific research projects based paradigms, like shape grammar, artificial intelligence in design, case based design, or object model based expert systems. The approach presented in this paper is based on the idea of building product model as the basis for CAAD systems. Of course, this is not the newest idea in regard to the scientific background, but the objectives of the paper are different. The work does not aim at an experimental proof of feasibility. It aims at the question, what are the tracks of research that are applicable for implementation in commercially used systems. Such a "new" commercial CAAD system has to compete with existing ones, that are based on the paradigm of geometric objects. On the other hand these systems are very mature due to their long existence and development. That leads to further questions:

a) What are the long term strategies for implementation of a new system and for competition with existing systems?
b) What are the foundations that would allow to include stepwise all of the above mentioned paradigms later on in a stepwise approach?
c) What are the limitations that are set on such a system by their potential future users. What is their understanding of doing their business. How will the understanding of new technology within our society influence the development?

First of all such a building product model has to provide a kind of skeleton that is strong enough to bear enhancements depending on how the answers to those questions will turn out in future.

2. Background of Methods being applied

A certain degree of consensus, e.g., that an integrated building design system has to be structured around building parts and not around drafting objects was already reached in the early years of building product model research [Eastman, 1978]. Nevertheless it took more then a dozen further years until the structures of building product models has become clearer.

One of the first foundation was the idea of the 'Product Definition Unit' (PDU) and its subtypes 'functional unit' and 'technical solution' of the GARM [Gielingh, 1988]. The Building Systems Model [Turner, 1990] shows a top down strategy to model a building using functional systems as, e.g., enclosure, structural, mechanical, etc. and their entities. The Neutrabas project [NEUTRABAS, 1991] and the ship distribution reference model [NIDDESC 1990] are dealing with the question of entities,

that belong to more than one system. Important contributions have been made by the IMPACT Reference Model [IMPACT 1993] e.g., the orthogonalization of the principles: life cycle view (e.g., production, usage, demolition), specification (generic, specific, occurrence), concretization (required, proposed, realized) and aspect view (e.g., cost, strength, safety).

There was no answer on how to structure the task of defining a building product model that would cope with the needs of all parties involved. It is the belief of nearly all researchers that such a task is in deed impossible. Recently, a number of projects have proposed more realistic approaches. They all aim at dividing the entire model into smaller parts. Terms used to describe such partial models are 'local product model' [Wright et al. 1992] and 'topical model' [van Nederveen and Tolman 1992]. The ESPRIT project ATLAS has 'view type models' [ATLAS 1994] and ESPRIT project COMBI uses 'partial models' and 'application models' [COMBI 1994]. Although the solutions are slightly different a way has been found that leads to practical building product models. Currently, the discussion focuses on the question, how to connect these partial models?

The term 'aspect model' is used in a strategy where these aspect models should be compatible with each other. The idea is that there is a "central" part of the modeling domain that would be shared by all aspect models [Luiten et al. 1991]. These central parts are often called kernel or core models. The core model concept is an underlying principle of the framework for Building/Construction developments in STEP, that is the B/C Application Protocol Planning Project [Junge and Storer 1993]. This framework structures AP developments for Building/Construction into families, such as building services, architectural system, structural system, etc.. These families are integrated by family cores. The Building/Construction core provides integration among these families. There is also the idea of an AEC core to integrate among all disciplines in the AEC domain. The building construction core is currently under development as Part 106 in STEP [Tolman and Wix 1995]. The AEC core is undergoing a feasibility study.

There is no place for core models in the current STEP architecture. The belief that STEP is dealing with data integration is a misunderstanding of the 'STEP integration' task. STEP integration means here the necessity to map all application reference models to the STEP resource parts. This provides the standardized definition for product data exchange on a semantically rich level. STEP's current architecture is only suitable for data exchange, whereas the core model methodology aims at data integration. It seems that STEP is strong enough to renew its architecture. The term core model does not imply any limitations to the size, but there is a consensus that cores should be small. That leads to the concept of 'minimal kernel' as it is used in the NICK II project [Tarandi 1993]. Such a minimal kernel is the highest abstraction level useful for integration of other levels in a layered model architecture.

Almost all building product models are static constructions, they provide valid definitions of data for a given point in time. A building however goes through various valid stages from conceptual design, workshop design, construction, use, refurbishment, etc.. The same has to apply to the product definition data. There are requirements for dynamic change of definitions, extendibility of the model, unforeseen type changes, etc.. On the geometry level that would call for multiple presentations of the modeled objects and for implicit representations. Current building product models try to (or have to?) define the domain completely and in detail. Current techniques does not seem to be able to cope with these demands. There is a gap between modeling activities and data definitions. Since some years a modeling technique, EDM, is under development [Eastman, 1991 and 1994] that aims exactly at overcoming these shortcomings.

It seems that most building product models assume the existence of a homogeneous model world. Is this a realistic estimation? If not, there will be a need to communicate in an heterogeneous model world. One possible answer to that is given by formal languages, that establish the correspondence between different models in order to allow the integration across several dictinct models. Those translation languages provide a method to describe the correspondences between models and the conditions under which the correspondences can be used. For an environment that consists of conceptual models it is important to describe the correspondence between the models at the same high conceptual level. This allows to react on changes within one of the participating models, that should communicate with others, by modifying the translation specification rather than by re-developing the hard coded converters.

An established translation specification is the conceptual source of a translation implementation, that allows to send and receive data between models having different abstraction of the design objects. The ability to move data back and forth also helps to guarantee the consistency of the two models in respect to each other.

3. Basic ideas

The product modeling approach is considered as a sound foundation for developments that should lead to data exchange, data integration and to new design applications, such as future CAAD systems [Junge, 1994, Liebich, 1994]. Both the new applications and the data exchange and integration facilities should be enabled to deal with semantically meaningful elements rather that with geometric primitives [Figure 1].

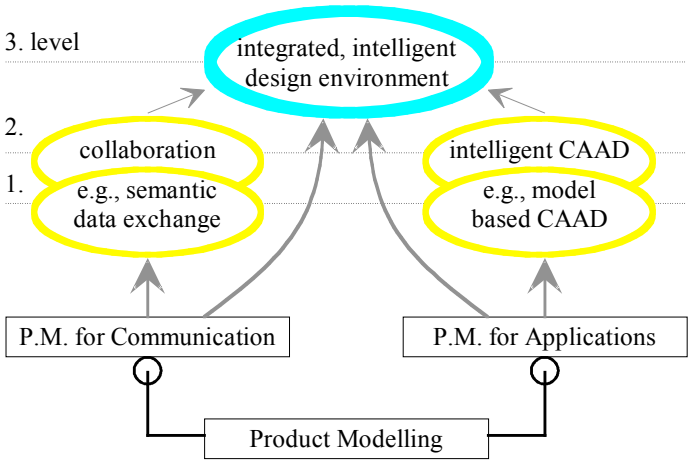

Figure 1 two intentions of product modeling

3.1 Abstract design object

The basic construct of the product modelling work, the abstract design object, is independent of its geometrical representation. An abstract design object is the highest abstraction of any architectural object, for example; building component, space, grid, or catalogue item. Each of these architectural objects is later regarded as a specialisation of this root object. It is described at different levels of complexity by the definition of its attribute classes, that is by its name and, if available, by its geometrical and physical properties, relationships, dependencies and constraints [Figure 2]. Thus, the abstract design object allows the equal treatment of form descriptions and specifications, e.g., material, costs, building instructions, or performance attributes.

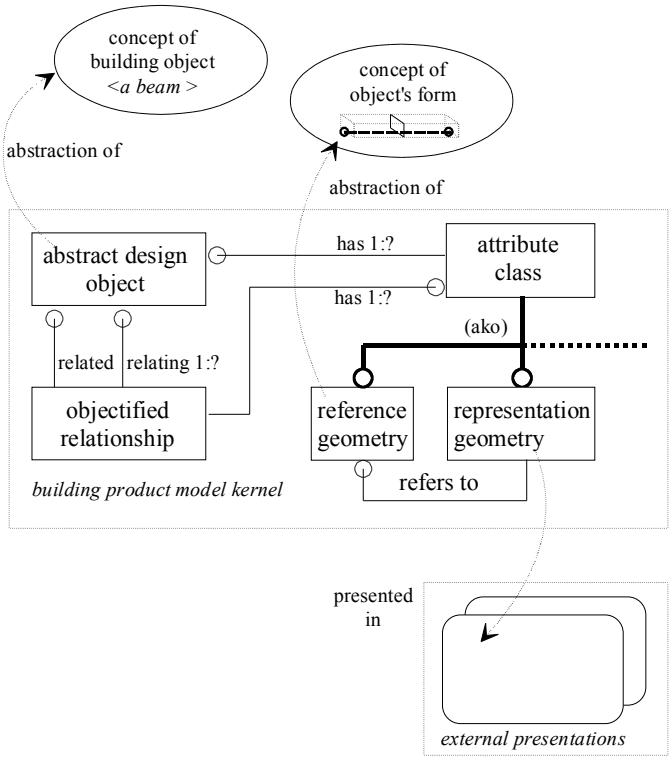

Figure 2 the definition of the abstract design object

The design object has a logical and a physical state, whereas the logical state refers to abstract definitions, such as the reference geometry or the logical relationships, such as connected_by, bounded_by, or adjacent_to relationships. Those definitions establish the network of relationships among the design objects, and they influence but not directly describe the form of design objects. The form of a wall being connected to other walls at its ends is geometrically defined by an axis and a cross section. The polyhedra forms a particular three-dimensional representation, that is developed from the reference geometry and the functionality of the connected_by relationship. The physical state refers to the detailed development of the representation form, such as the representation geometry. Thus, the logical and the physical states can partly be compared with the GARM approach, that distinguishes between the functional unit and the technical solution of design objects.

3.2 **Objectified relationships**

Beside the design objects the objectified relationships are considered as root objects as well. Objectified relationships deal with different kinds of interrelationships between design objects, such as dependencies, connections, or aggregations. They are objects by their own rights. The objectified relationship is described by attribute classes to keep the specific properties. 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. Objectified relationship has a logical and a physical state as well. The comparison to GARM applies particularly to the linkage relationship, as a logical relationship can be further decomposed into several physical relationships. This process can be recursive, if any of the physical relationships acts as a source of a new logical linkage [Figure 3]. At the logical state the bounded_by relationship points to a space and defines a virtual space boundary, with a connection to the space behind. At the physical state, however, this relationship is objectified into a space enclosing element, being a view type object of the building element that actually forms the physical boundary, such as a wall.

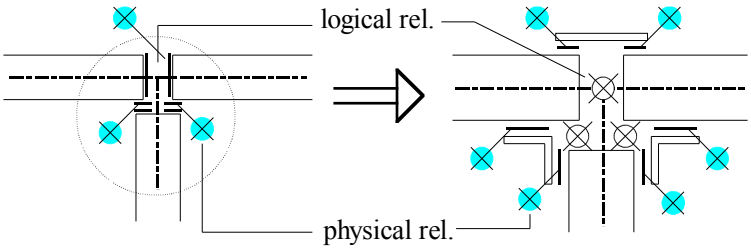

Figure 3 recursive decomposition of logical and physical relationships

3.3 **Reference geometry**

A prerequisite for the logical description of design objects is the reference geometry. The reference geometry contains the basis geometric definition of a design object and places it into the spatial context of a building. It can be considered as an implicit geometric definition, as it defines a geometric reference item to which either a parametrized description or an explicit geometric representation item refers [Figure 4]. Whenever a design object, e.g., a building element or space, is placed into the coordinate system of the building project, exactly one reference geometry item is created. Its type, however, can change during the design object's life cycle. The sum of reference geometry of spaces builds the skeleton topology of the building. The Relational Model Topology of COMBINE follows a similar approach [Combine II, 1995].

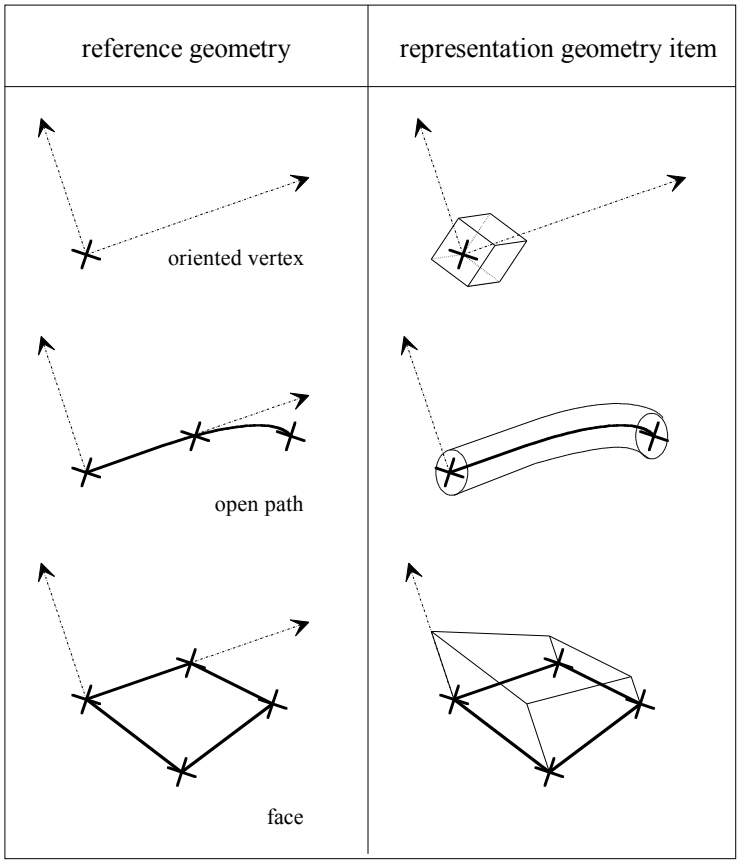

Figure 4 different types of reference geometries

The reference geometry is the point, to which all different geometric representations refer. It is either possible to develop a reference geometry under a given context into geometric representation items, or

to connect any explicit graphic or geometric representation form to the reference geometry. In an application, the geometric representation can now change from vague and fuzzy symbols to sketch-like two-dimensional geometry and again to exact two- and three-dimensional geometric entities. All those geometric representations are valid design object descriptions, as they refer to the reference geometry and thus to the same location within the spatial context of the building [Figure 5].

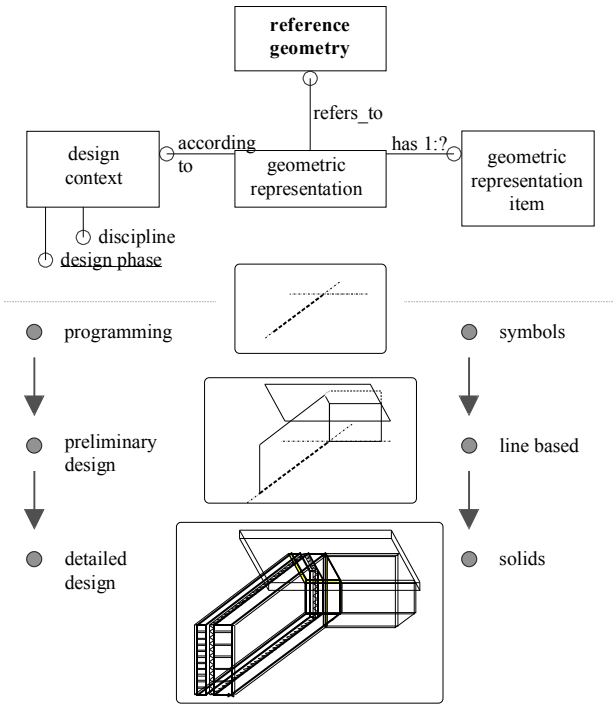

Figure 5 different context dependent geometric representations

3.4 **Building product model decomposition**

The abstract design object, the objectified relationship, and the reference geometry are fundamental concepts for defining a building product model applied to design applications. Another prerequisite is the proper structuring of the building product model. The layered approach is certainly the favored way to decompose the whole modeling universe into levels of specific information contents. According to the core model philosophy, the following structure is chosen [Figure 6]:

- minimal kernel level
- system level
- system component level
- application level

It is the task of the minimal kernel to establish a common agreement for all models, that are being incrementally defined. The minimal kernel establishes the basic description of the root objects, such as the design object and the objectified relationship. Those objects are further specified in partial data models at the system and system component level. Another task of the minimal kernel is to keep track of the links between the different partial data models.

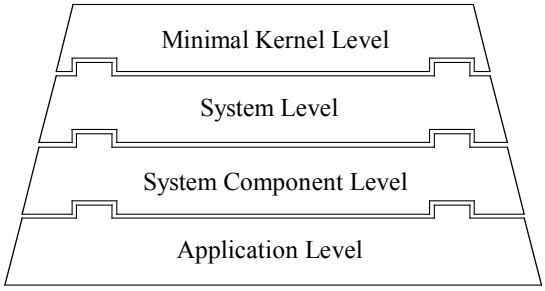

Figure 6 layered structure of the modelling framework

At the next level, the different systems are described, e.g., structural system, enclosing system, spatial system and distribution system. Those partial models for systems contain the core description, that applies to all components of the system. The third level comprises schemas for the components, that play their major role in the according partial model for systems. Those components are fully described in terms of their attributes and relationships. As the definition of system components, such as spaces, significantly varies during the life cycle, a further decomposition into life cycle stage dependent partial models for system components might be appropriate. The forth layer of application models uses modeling constructs from the layers above and adds the application specific functionality. The application layer should also allow to further expand the design object hierarchy to create more detailed and user specific subtypes of design objects.

A problem arises with the multidisciplinary nature of many system components, playing a role in several partial models for systems. Each partial model manifests a different way to cope with the specific views on physically identical design objects. This reflects the human way, as the various participants of the design process have different conceptualizations of the design object. There are several relationships between the different abstractions, based on different views onto the physically identical design object [Figure 7]:

a) Different attribute sets of the design object are kept only once in that partial model, where the design object plays its major role.

b) Different design objects refer to the same physically identical design object, but they have distinct abstraction in two or more different partial models. The relationship between them can be a one-to-one or a many-to-one relationship. The differences, made within the partial models, can require:

- a complete relationship, i.e., the design object within the first partial model is fully decomposed into many design objects within the second partial model
- a incomplete relationship, i.e., the design object within the first partial model is also decomposed into sections of other design objects within the second partial model
- an attribute to object transformation, i.e., the same information item is described as an attribute of another design object within the first partial model, whereas it is an own object within the second partial model

Those view type objects can dynamically evolve as type changes of the original design object during the design process. If the original object should be kept, since it reflects a specific view or it manifests the design history, a relationship between the two view type or version objects has to be established.

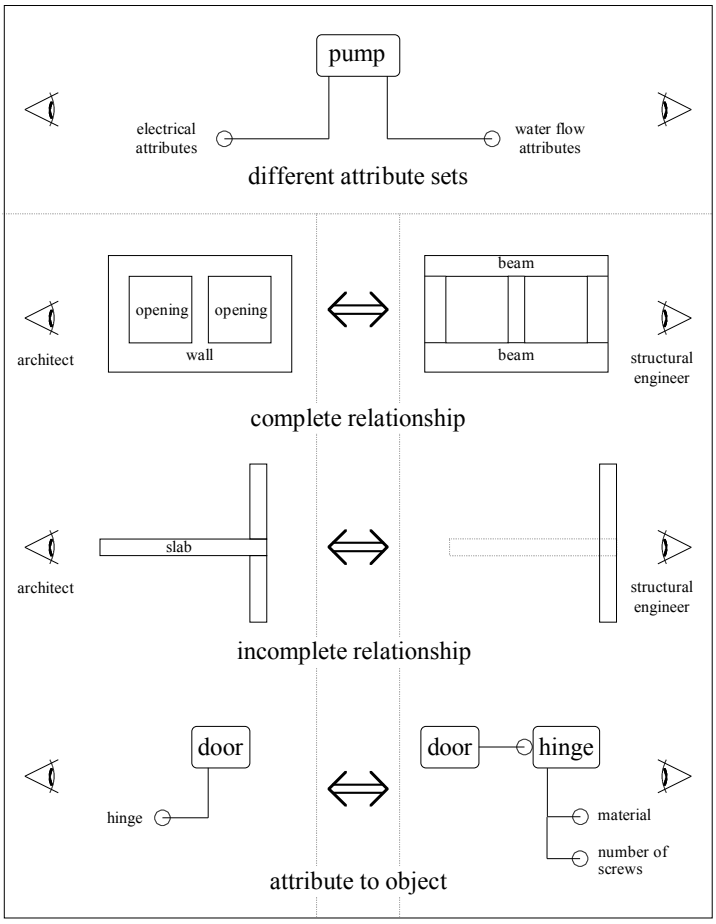

Figure 7 example of different views onto physically identical design objects

3.5 **Product model integration**

The establishment of a collaboration between diverse application or partial models requires a model translation, both at conceptual and at implementation level. The translation process needs the input of technical knowledge to relate the semantic description of the source object to the description of the target object. Thus, the conceptual model is necessary to allow the modeler to specify the model correspondence on a conceptual level as well. Another advantage is that changes can be made at the specification level, and the actual converter is then automatically generated.

There are several formal translation languages being currently under development. Most of them accept two schemas, source and target usually specified using EXPRESS, and establish the transformation process either by imperative statements or by rules, e.g., in languages based on Lisp or Prolog derivatives. Those conceptual translation languages have to cope with converting mapping primitives (entities and attributes) in any combination and with any possible cardinalities [Figure 8]. They should provide for translations in one or both directions. Functions and constraints, associated with entities, should also be converted in a more advanced stage.

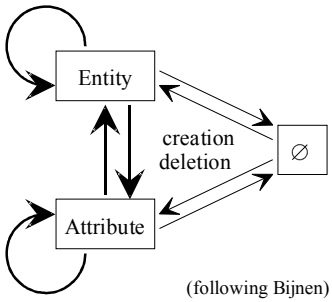

Figure 8 mapping primitives

The complexity of even simple translation problems is shown in the following example. An entity 'column' within the architectural tool has to be converted into the different abstraction of column within the structural engineering tool. This requires the translation of the unique identification, the material information, and the different geometric descriptions, whereas only the first two are now described and given in the schema descriptions [Figure 9]. The translation of the identification needs a mechanism for the handling of synonyms, since the attributes have different names. The second translation problem comprises the conditional conversion and merges a set, containing embodied subsets, into a single set. Only those specified_by of the column in the source schema, which refer to a material, have to be converted into consists_of of the column in the target schema. As the source entity material is described by a list of strings, it has to be translated into many target entities material, being described by a single string.

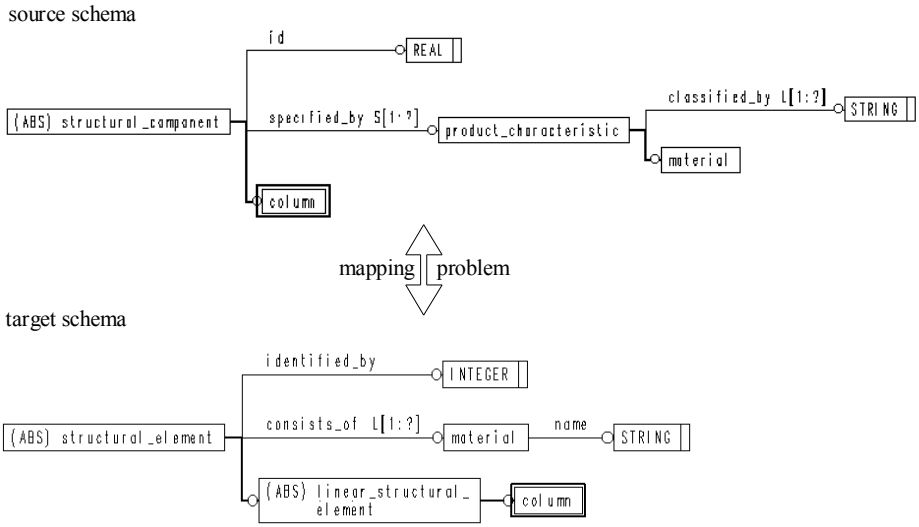

Figure 9 EXPRESS-G diagrams of the translation example

The decomposition of the data integration framework into small and easily manageable schemas already reflects the different views of design actors. Model translation can be used to realize data integration within such a heterogeneous model world. The differentiated model structure, however, also provides a background for collaboration, that goes behind the bulk exchange of design information. At a next stage, the design actors should be enabled to collaborate by the frequent exchange of distinct information on request, realized by the transformation of required data between the according partial models.

4. **Project Developments**

The authors are mainly involved in two interrelated projects. The first one, internally called NextCAAD [Junge et al., 1994], has two levels. The first level deals with applied research at the field of building product modeling for design applications. The second level aims at realistic developments for CAAD companies, leading to a commercially available product. The framework for the software technology within the project is, among others, the application of a total and consistent object orientation. It does mean applying all characteristics of object oriented analysis, design and implementation [Rumbaugh et al., 1995] to the development of CAAD software. The main characteristics of the project are:

a) Building product model applied to commercial CAAD development
b) Abstract design object defined inside the minimal kernel
c) Definition of specific design objects in different, but interconnected, schemata
d) Awareness of interoperable building design tools surrounding CAAD

The second project is an applied R&D project under the ESPRIT framework, called COMBI. It envisions an intelligent environment based on the idea of "integration by communication" [Junge, 1991]. The first focus is the development of four expert systems for structural design. The second focus is on integrating these four plus linking them with an external traditional CAD system. Two prototype developments of COMBI are shown in the paper.

a) The COMBI communication manager (CCM) uses model translation techniques. It knows the origin of the entities respectively their attributes within the sending and receiving systems. The CCM now establishes the exchange of instances between neutral files, based on STEP physical file format, from different application models.

b) The COMBI visualization manager (CVM) provides an access to drafting tools by the translation of relevant application data into the format of the new international standard for explicit drafting [ISO DIS 10303-201, 1993].

5. NextCAAD: a new generation of commercial CAAD systems

Some of the results, that are already achieved, or their foundations, that are currently established, will be shown. Beside this, some remarks on lessons learnt will be given.

5.1 Intelligent relationships among design objects

Within a NextCAAD realization, the design objects are enabled to know about their interrelations or dependencies. If one design object has been modified, the system can now keep track of the consequences to other design objects. If the constraints of a relationship are violated, the system will ask the user for a decision. In the next realization phase a constraint propagation mechanism will be called to solve the problem. On the other hand, the relationships are also the point, where design rules can be kept in the CAAD system. A special door might be obliged to have a distance to the next corner of more than 60 cm. If the corner is modified by moving an adjacent wall, the door is moved accordingly. In the case, that there are other restrictions on the moveability of that door, which prevent the movement (a simple example of overconstraining) the user has to reset some of the rules.

The implementation example shown in Figure 10 demonstrates how structural components are interconnected with themselves and with the building grid. Any modification of the building grid leads to a rearrangement of connected structural components, and furthermore to an adjustment of spaces. Openings are connected with structural components and may be filled, e.g., by windows or doors. If, for instance, a wall is changed from a straight wall to an arc wall, the opening will be changed accordingly. The inserted window will be adjusted to fit, or, if it is not possible to automatically resolve the conflict, a message will be displayed, asking the user to manually modify the window. That is the point, where the CAAD system starts to behave intelligently.

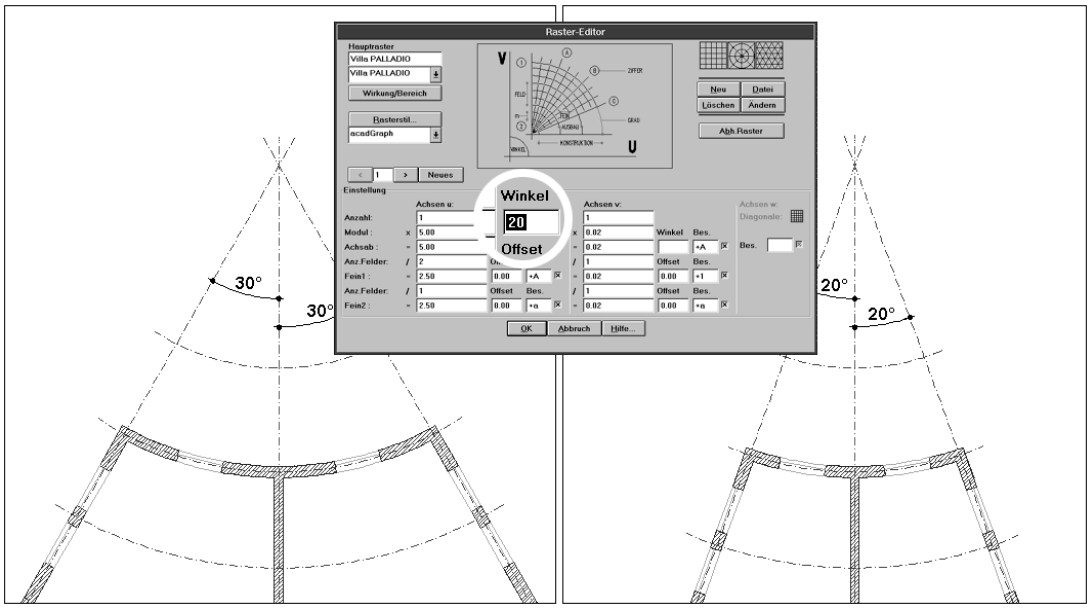

Figure 10 building elements associated with the building grid

5.2 **Case study: Evolution of space**

The new CAAD system has to handle space objects in the same way as building components. This is an important step in regard of the support of building life cycle information, since spaces are one of the first objects being defined and spaces are used as carriers of information until the later phases of usage, e.g., in facilities management. The proper handling of spaces is now in the realization phase.

The first definition of spaces in a building design process appears in the design brief. In the design brief there is at least information about the name, the required area, and possibly about requirements for spatial layout. Considering the design brief view type model, the space objects can now be topological arranged. Following this way, the definition of spaces starts at a moment, when no building elements exist. Thus, spaces are bounded by space enclosing elements, which later on can be redefined as being view type objects of walls, floors, or other structural components. The designer now gains enhanced functionality from the system and its BPM, since after each modification of structural components the space information is automatically updated. It comprises not only the area but also the finish, the floor level height, and constraint checking against requirements, given in the design brief, such as minimal width, or specific adjacency. During these life cycle phases, the geometric definition and representation of spaces have to be transformed several times.

For the project, another study has been undertaken, asking on how to incorporate a space allocation program. The pragmatic approach led to the usage of an algorithm, that comprises two procedures, one for stacking, i.e., assigning each space to a floor, and one for blocking, i.e., arranging the spatial layout of each floor. The merits were mainly seen in checking the requirements of a given design brief against the actual situation of a chosen building envelope. The generated floor layouts, however, were not sufficient and treated only as by-products. Sophisticated design layout generators based on AI techniques, such as SEED, are more powerful solutions [Flemming et al., 1994]. It would be nice to have the opportunity to study how those solutions could fit into a commercial system.

5.3 **Lessons learnt**

The following lessons have been learnt during the project and they give a guidance to the methodology:

a) Modeling and documenting with EXPRESS

EXPRESS is a frequently used language for the purpose of data definition in product modeling, and there are a lot of modeling tools available in the environment of STEP/EXPRESS [ISO IS 10303-11, 1994]. For reasons of easiness of readability for human beings a graphical method of displaying product models is of high importance. The graphical capabilities are provided by the subset EXPRESS-G.

b) EXPRESS to formally describe model consistency
The language provides facilities to define rules to restrict dependencies between entities and to define the valid range of attribute values by using where clauses. Thus, the consistency of the model can be described on the same conceptual level as the data, which allows the acquired knowledge to contain some basic intelligence.

c) Realization of the total CAAD functionality as methods on the objects
The role of the CAD software itself will be transformed step by step. In addition to the traditional role focused on generating and manipulating geometry it now has to fulfill functions concerning the other behaviors of design objects. This well-known concept seeks for support by the modeling language.

d) Limitations of current EXPRESS version
Beside its merits, EXPRESS still lacks some principles of object orientation. The behavioral aspects of objects can not be described in EXPRESS. Otherwise, new developments on EXPRESS, e.g., Version 2.0 or EXPRESS -C [Staub and Nieva, 1994], are pulling into the direction of strict object orientation.

e) Generating code from EXPRESS models
A product model written in EXPRESS contains all definitions of the data in a computer readable form. It is obvious that there must be a way of automated interpretation, conversion or interfacing to other computer usable forms. The availability of such tools is a real progress, and with some restrictions applicable for application developments.

f) Use of relational data bases
To permanently store the design objects, an interface to a RDBMS was implemented. This concept was not approved because of the insufficient runtime performance in regard to CAAD application. On the other hand it is supposed that OO DBMS are suitable tools for integrated CAD data bases. Thus, the use of OO DBMS remains as a main task of the next phases of the project.

6. **COMBI: a prototype for data integration**

COMBI is developing a prototype environment for co-operative design, which is mainly focused on the domain of structural engineering. It envisions an intelligent environment based on the idea of "integration by communication". The first focus is the development of four application tools for structural design. These tools form a chain beginning with soil mechanics and foundation design going to structural preliminary design and to structural analysis and ending with reinforcement design. The description of these tools is out of scope of the paper. The second focus is on integrating these four expert systems. The third focus is put on linking these tools with an external traditional CAD system, mainly for visualization.

Both prototypes, for integrating the four expert systems and for visualization, make use of different translation mechanisms in order to establish semantically meaningful exchange of information.

6.1 **Communication manager**

COMBI's Application Models simply contain all possible output/input entities and their attributes from the corresponding COMBI application tools. With help of an intelligent center, a tool capable of filtering and translating the incoming information for use in the receiving system, however, the desired communication can be achieved in a more flexible way. The model mapping technique used by the COMBI Communication Manager (CCM) knows the origin of the entities respectively their attributes within the sending systems. Therefore the CCM performs the translation between different specification forms of design objects and attributes within the participating systems. The mapping rules, that describe the input and output data format and the transactions, are realized and executed in

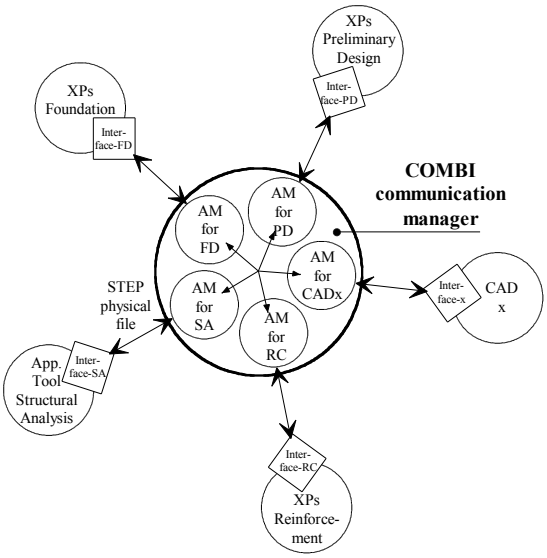

Figure 11 COMBI Communication Manager

KEE, an AI shell. The CCM now establishes the exchange of instances between neutral files, based on STEP physical file format, from different models [Figure 11]

6.2 **Visualisation manager**

The CAD integration currently under development allows the external graphical visualization of instances from application tools via a neutral file, based on the STEP AP 201 definition.

The results from any specific COMBI agent are managed by the COMBI Visualization Manager (CVM). The data, sent by an agent in STEP physical file format, are transformed into the 201-format, using the XP-rule tool within the XPDI platform [1]. During the translation process most of the semantics is preserved. The entity type information, for instance, is converted into sub strings of layer names, that follow the layer and attribute structure of COMBI, and that is provided by the AP 201 layer construct. The same applies to important collaboration information, such as the owner of data and the time of data exchange. On the other hand the unique object identifier is mapped into AP 201 group information. The output is displayed by the STEPviewer [2] and can be read in a CAD system, e.g., via the 201 converter for AutoCAD [Figure 11].

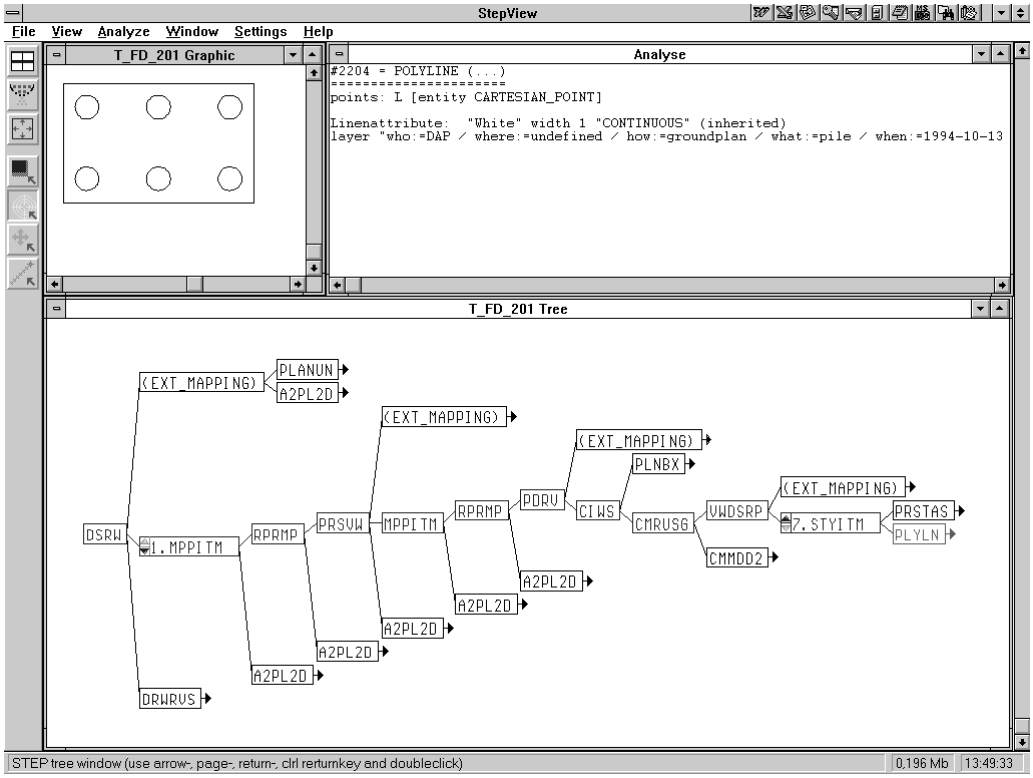

Figure 12 visualization of application mode data in the STEP viewer

7. Conclusions

The first phase of the NextCAAD project realization phase is now considered to be a successful step into the direction of intelligent CAAD. During the next phases the emphasis will be put on incorporating intelligent modules, which will be based on the object oriented description of design objects, provided by the underlying product model. Finally, new generation CAAD can be more easily included into an integrated design environment.

For data integration, the authors assume a loose integration of relatively independent modules will be particularly suitable for small and medium scale design tasks, which is the average of the daily work by architects and engineers and which are done by the help of commonly available software tools. The model mapping appears to be an appropriate concept to achieve an easy as well as effective integration of design data.

8. Endnotes

[1] XPDI is a STEP based tool offering software components to enable the modeller to develop EXPRESS schemas, both in graphical and in textual form. Beside that, a Lisp late binding of SDAI enables the dynamic interpretation of a rule based language. The XP-RULE module is now able to interpret rules which are developed to convert source schemas (and their instances) to target schemas. The grammar of XP-RULE is closed to EXPRESS-M and one of the future development is to be 100% compliant with EXPRESS-M. XPDI is being developed at CSTB Sophia Antipolis, France.

[2] The STEPviewer is a STEP based tool, that accepts a STEP physical file according to AP 201 and displays the logical structure as well as the contained 2D geometry. The tool is developed by CONCAD in Bonn and was kindly put to the authors' disposal.

9. Bibliography

ATLAS, 1994. ESPRIT 7280 project: various deliverables

COMBI, 1994. ESPRIT 6909 project: deliverable A2

COMBINE II, 1995. EC joule project. final report

Eastman, C., 1978. The representation of design problems and maintenance of their structure. Latcombe (ed.), Application of AI and PR to CAD, Elsevier, North-Holland

Eastman, C. , 1991. Use of data modelingin the conceptual structuring of design problems. Schmidt, G. (ed.), CAAD futures '91. Vieweg, Wiesbaden

Eastman, C., 1994. A data model for design knowledge. Carrara, G. and Kalay, Y.E. (eds.), Knowledge-Based Computer-Aided Architectural Design, Elsevier, Amsterdam

Flemming, U., Coyne, R., Fenves, S., Garret, J., Woodbury, R., 1994. SEED: A software environment to support the early phases of design. Abstracts of IKM, Univ. HAB Weimar, pp. 5-10

Gielingh, W., 1988. General AEC Reference Model, ISO TC 184/SC4/WG1 DOC N.3.2.2.1

IMPACT, 1993. ESPRIT 2165 project: final report

ISO DIS 10303-201, 1993. Application Protocol: Explicit Draughting. ISO TC184/SC4/WG3 N251

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

Junge, R., 1991. Integration by Communication. 1st international symposium Buillding System's Automation - Integration, Conference Paper, Univ. of Wisconsin, Madison

Junge, R., 1994. Produktmodellierung im Architekturbereich, Teil 1: Techniken und Anwendung fuer den computergestuetzten Entwurf. Abstracts des IKM, HAB Weimar -Universität-

Junge, R. and Storer, G., 1993. B/C Application Protocol Planning Project. ISO TC 184/SC4/WG3/T12 AEC Working paper

Junge, R., Ammermann, E., Liebich, T., 1994. Product Model: a Basis for Next Generation CAAD. CIB W78 Workshop on Computer Integrated Construction, Helsinki, Finnland

Liebich, T., 1994, Produktmodellierung im Architekturbereich, Teil 2: wissensbasierte Entwurfsunterstützung auf der Grundlage von Produktmodellinformationen. Abstracts des IKM, HAB Weimar -Universität-

Luiten, B., Luijten, B., Willems, P., Kuiper, P. and Tolman, F.P., 1991. Development and Implementation of Multilayered Project Models. Mangin, J-C., Kohler, N. and Brau, J. (edts), Computer Building Representation for Integration, Pre-Proceedings of the second international workshop, Ecole de polytechnique federale de Lausanne, Lausanne,

NEUTRABAS, 1991. ESPRIT 2010 project: Outfitting Information Model, Deliverable 4.2.2

NIDDESC, 1990. Ship distribution reference model. ISO TC 184/SC4/WG1 Working paper

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., 1991. Object-Oriented Modeling and Design. Prentice-Hall, Englewood Cliffs

Staub, G., Nieva, A. (eds.), 1994. PISA Information Modelling Language: EXPRESS-C, ISO TC184/SC4/WG5 working paper

Tarandi, V., 1993. Object oriented communication with NICK. Mathur, K., Betts, M and Tham, K. (eds), Management of Information Technology for Construction, World Scientific and Global Publication Services, Singapore

Tolman, F.P. and Wix, J., 1995. ISO TC 184/SC4/WG3/T12 AEC, Part 106 Working paper

Turner, J., 1990. Building Systems Model. ISO TC 184/SC4/WG1 Working paper

van Nederveen, G.A. and Tolman, F.P., 1992. Modeling multiple views on buildings. Automation in Constuction, vol 1, 1992 nr. 3

Wright, A.J., Lockley, S.R., and Wiltshire, T.J., 1992. Sharing data between aplication programs in building design, pruduct models and object-oriented programing. Building and Envieronment, vol 27, 1992, no. 2

10. **Biographical Sketch**

Richard Junge
Education: Diploma in architecture, Technical University Munich, 1969
Position: Managing director of CAB
Short curriculum vitae:
- 1973- Partner of Schmidt-Schicketanz + Partner, Architects
- 1985-91 Projectleader of the EDV-LABOR Munich
- 1995- Professor for CAAD at Technical University Munich

Experience:
- cost-planning methodology and software for CAD
- information flow for construction process (BPRIM)
- EDIFACT messages development in WEEB-MD5
- ISO/STEP/TC 184-SC4 - Teamleader of AEC Team (WG3/T12)
- building product models for applications and data integration

Liebich, Thomas
Education: Diploma in architecture, HAB Weimar,University, 1989
Position: researcher at CAB
Short curriculum vitae:
- 1993 Doctor degree from HAB Weimar,University
- 1993-94 EC research fellowship at Univ. of Strathclyde, Glasgow

Experience:
- object-oriented data models
- knowledge based systems in architectural domains
- building product models for applications and data integration