PRODUCT DATA MODEL AND IMPLEMENTATION STRATEGIES FOR A DISTRIBUTED ENVIRONMENT

Delopments in the VEGA Project

RICHARD JUNGE
Faculty for Architecture, Professorship for CAAD
Technical University Munich, Germany

KLAUS BEETZ
Dipl. Math.
Nemetschek Programmsystem, Munich, Germany

THOMAS LIEBICH
Dr. Ing.
CAB Research & Consult, Munich, Germany

This paper is based on work carried out in a series of research and development projects. The latest of it is the ESPRIT Project VEGA. VEGA stands for 'Virtual Enterprises using Groupware tools and distributed Architectures. The result will be an IT platform based on extended CORBA technologies, the 'COAST', adaptations and extensions of work done in the Workflow Management Coalition, WEB technologies and Product Data Technology. Product model architectures and implementation strategies and for proof of feasibility prototype implementation for enabling such an environment that will surely be based on distributed product models including parts of such models are in the focus of this paper.

Introduction

The story surely started somewhere in the early 70th with the development of systems like GLIDE, BDS and others of that type (Richens, 1978). Today one would say they where using Building Product Models, a term that was not invented at those days. But a certain degree of consensus, e.g., that an integrated building design system has to be structured around building parts and not around geometrical objects was already reached in the early years (Eastman, 1978). Nevertheless it took more then a dozen further years until the structures of building product models have become clearer (Junge, 1994)

The time of what we now regard as 'Product Model' started with the development of two generic models. The GARM (General AEC Reference Model) (Gielingh, 1988) and the Building Systems Model (Turner, 1990). In projects like ESPRIT project IMPACT (IMPACT 1991) the technique of 'layered models' was introduced. The ESPRIT project NEUTRABAS (Neutrabas, 1991) started to deal with the question of, respectively solutions, for objects, belonging to more than one system and are playing different roles in them. The discussion on this problem is still open today.

Today the belief is that there is no way leading to a homogeneous single central building product model is accepted nearly unanimously. In a number of current projects one finds various terms used for strategies dividing the universe of discourse into partial models. Terms used to describe such partial models are 'topical model' (van Nederveen and Tolman, 1992). The ESPRIT project ATLAS has 'view type models' (ATLAS, 1994) and

Period	Focus of development	Example Model
1975-85	‘pre-product models’	
1975-80	basics of building models	GLIDE
1980-85	models for expert systems	
1985-	building product data models	
1986-90	generic product models	GARM, BSYSM
1990-95	aspect and layered models	COMBI, ATLAS
1995	‘consolidated models’	O.P.E.N

Figure 1 Model history

ESPRIT project COMBI uses 'partial models' and 'application models' (Junge et al, 1994). Although the solutions are slightly different, a way has been found that leads to practical building product data models.

The discussion is on strategies where these partial 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 partial models (Luiten et al., 1991). These central parts are often called kernel or core models. Again one finds this concept in ATLAS and COMBI, as LSE Project Type Model or as Central Neutral Model The main question still remains: What is the focus of a core model? Is it the sum of parts used in at least one application or domain? Or should it be a small but generic kind of ‘overhead model` which aims at staying stable when the scope of the overall model expands? A proposed solution for the first is e.g. the minimal kernel model as used in the NICK project (Tarandi, V. 1993)

The Building Construction Core Model (BCCM, ISO STEP Part 106, Tolman and Wix, 1995) is based on the opposite concept. The basic assumption is that all objects that are used by more than one system should go into the core model. This at first glance simple concept implies that the core model will grow into a central model with some but small satellites. The question is how many objects do not belong to more than one system? This seems to be a concept that bears the danger in it that the model will ‘explode’ because of the shear number of entities it will have to deal with and it will not be a stable core for the suite of models it is intended to serve until all these models are being developed themselves. Such a model for Computer Integrated Construction (CIC) could easily reach a number of some thousand entities. Even if most of the attributes would go into the connected ‘domain models’, thus leaving itself to be a semantically flat model, it seems very questionable if such a model would be manageable. A comparable approach, the IPIM, was discussed in STEP during 1989.

One could draw a kind of time table showing the main focuses of research and development of product modeling technology. A variation of such a time table presented by Bjoerk (Bjoerk 1993) is shown in fig 1.

In 1994 Bjoerk presented a structure of how the different product modeling concepts very briefly and incomplete described above can be grouped. He called it a ‘Layered Framework Architecture’ for grouping data structures which are used in building product data models’ (Bjoerk 1994). Bjoerk writes: ‘The set of data structures that make up a fictive comprehensive building product model is partitioned using an onion- like overall architecture. ....a decomposition into five layers would seem appropriate’.

Model layer	Example data structures	Example models
Fundamental data model	objects, attributes, relationships, generalisation- specialisation, rules, methods, messages	Relational data model Entity-relationship -model The object as in objectoriented programming The frame
Generic product description model	Data structures for describing aggregation- decomposition, type objects, versions, shape and location information, abstraction levels etc	STEP Resource Parts EDM GARM OOCAD
Building product data model kernel	Generic object classes which are particular for a specific design or construction discipline and or phase in the construction process	Building Systems Model GSD Model RATAS Framework KBS MODEL NICK
Aspect models	Detailed object classes which are particular for a specific design or construction discipline and or phase in the construction process	COMBINE IDM RATAS quantity take off prototype Structural Steel Model CIMsteel LPM
Aplication models	Detailed object classes which constitute the conceptional model of one particular application	Not subject for this framework

Figure 2 The layered Framework Architecture

Following Bjoerk all these layers combined would form a comprehensive building data model. The models quoted in the framework where more or less focused on the necessary exploration of only one the layers. From today's level of knowledge it is obvious that none of those 'layers' as standalone solution can provide the necessary basis for a comprehensive Building Product Data Model.

The authors therefore propose the 'Consolidated Model' as the attempt to combine the pieces developed in previous projects to an architecture mature for an commercial implementation of the paradigm of the 'Building Product Model'.

1. **Steps towards the 'Consolidated' Model**

The Consolidated Model architecture emerged out of a more or less consequently and gradually development in a series of projects in which the authors where involved during the last five years. These are COMBI, an ESPRIT project, NextCAAD I, NextCAAD II, both are industry funded development projects and VEGA, again an ESPRIT project. One common basis for all these projects was an as much as possible and consequent application of so called 'STEP methodologies' and the search for a model architecture flexible and comprehensive enough to serve as a basis for commercial applications in architecture and engineering.

COMBI

The COMBI project (1993-95) developed a prototype environment for cooperative design focused on the domain of structural engineering. It envisions an intelligent environment based on the idea of 'integration by communication' (Junge 1991). The first focus is the development of four application tools for structural design. These tools form a chain

beginning with beginning with soil mechanics and foundation design to structural analysis end ending with reinforcement design. The second focus is on integrating these four application tools, while the third focus is on linking tools which are developed on the basis of a product model with an external traditional CAD system. For these three tasks a product model framework is being developed.

This framework has been created under consideration of the evolutionary multi stage and multi agent nature of the design process and for unified integration approach. The architecture of the framework can schematically be represented by four hierarchically structured levels (Figure 5). It can be envisioned as a network of computers and users, where communication and coordination is achieved through a shared information medium and a control mechanism which provides facilities for the integration of loosely coupled design tools.

The application tools, that support the work of the individual designers shown on the bottom level have their own application dependent product model, which need not necessarily conform to a standardized product model specification. Thus, application data are processed and stored only within their application domain, which helps to avoid information explosion and unnecessarily complicated data structures. The three main reasons for this decision had been:

- the application data structure must be organized according to the needs of the application methods in order to achieve maximal run time performance,
- existing tools must be integrated without internal modifications, and
- specific data extensions needed by the application methods are often only for temporary use and can be generated automatically, e.g. by a finite element mesh generator, as it is the case in the COMBI prototype.

The two middle levels are forming the common kernel of the integration framework. They are divided into a central neutral model and several COMBI partial models designed to serve specific engineering domains. Communication within one domain is supported by the corresponding partial model, communication between different domains need additionally to their partial model the Central model and thus increasing the need as well as the complexity of the necessary mapping mechanisms. The uppermost level serves as a common source, containing resource parts from STEP as well as small COMBI proprietary resources. The framework can be classified as a layered architecture using a central kernel and partial models.

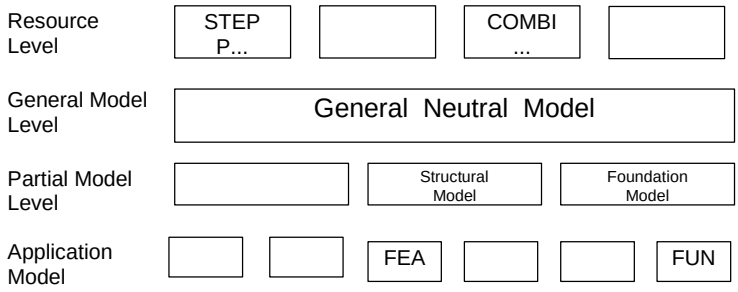
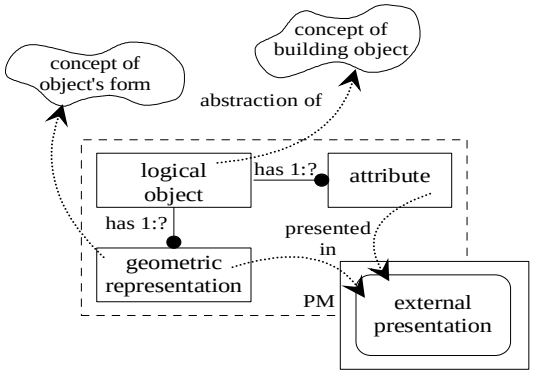


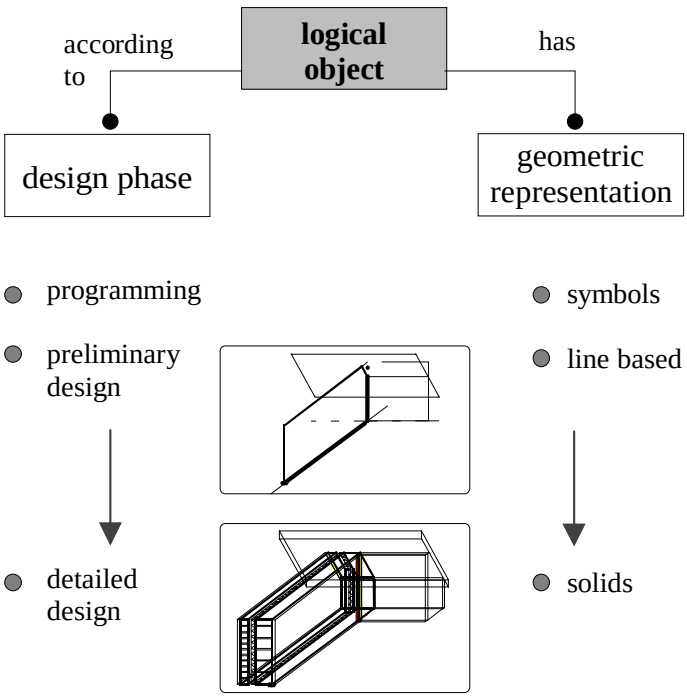
Figure 3. COMBI Framework

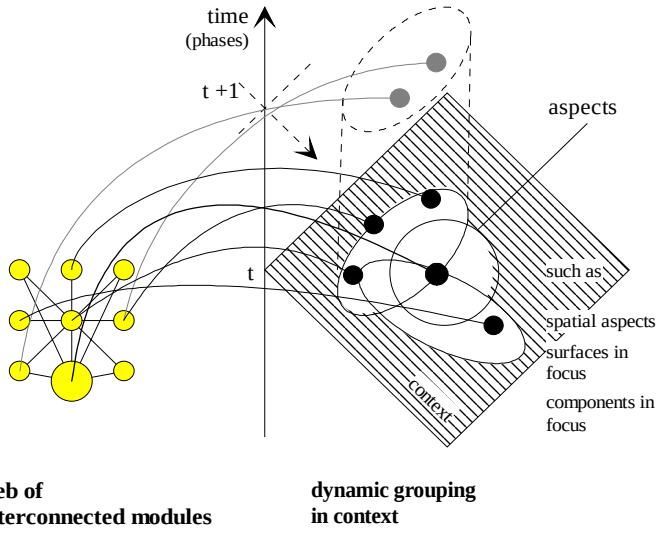


THE NextCAAD I PROJECT

The objective of the project is to develop a CAD system that is not primarily oriented towards the graphical and geometrical processing as current systems are. the basic idea is that the architectural design object is considered as an logical entity within the product model. Architectural design objects can be rooms, spaces, building components of different complexity, grids, etc.. Parts of the objects' definitions are dealing with its physical shape, physical properties, others with its representation, which is always an abstraction of the real design object.

The existence of an object is not based on the representation of its geometry on the screen respectively the object's definition in geometry data. This is the fundamental step, which creates the basis for a consistent application of the product model philosophy. An object is a more abstract entity that can be viewed from different viewpoints. Thus it has different representations, e.g. different geometric representation, characterized by different sets of attributes.





An application covering a whole area, such as CAAD the discipline of architecture, has to be dynamic by its own nature. It should be possible to shift the focus from, e.g. the spatial layout, to the building components and again to the bill of quantities.

Therefore the modules of the system, and accordingly the partial models of the product model, should be interconnectable in a dynamic way. Thus, the underlying product model is structured as a web of interconnected modules, in opposite to the layered approach, e.g., in STEP.

The next generation CAAD has to deal with many phases of an object's life cycle, from programming phase to detailed design. Objects may occur and disappear during the life cycle, usually they are transformed or refined many times. In particular the geometric representation change from vague and fuzzy symbolic descriptions to sketch-like two dimensional geometry and again to exact two- and three dimensional geometric entities. Thus, multiple geometric representation is of special concern for the project.

Design is a highly dynamic process with many diverse particular activities and constantly shifting focuses between them. Translated into a product model this means that there are many individual focuses. This could be modeled as a huge singular model. Disadvantages are discussed in the introduction. Even if it where theoretically possible to model such a thing, practically it is not feasible, because one has to have particular results in a foreseeable range of time. All this leads to modular models. The conceptual idea for bringing the dynamic nature of the design process into a product model was that of a number of modules that could be connected together in dependence of the momentary context requirements of the CAAD system. This is the idea of a web of dynamically interconnected modules.
The concept of this model architecture could not be realized for many reasons. The resources of the NextCAAD I project where reduced and it was focused to architectural design only. This goal could be reached with much simpler model architectures that or based on more proven ground. Secondly it still is doubtful if it would have been possible to implement such an architecture with the implementation technique of that time and project. Today with developments from the VEGA project a realization would be more likely

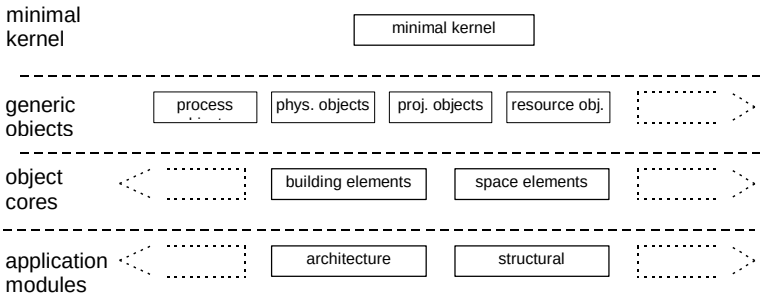


Figure 7 NextCAAD II, modules linked by a minimal kernel

THE NextCAAD II PROJECT

The model architecture of the NextCAAD II project in a certain way is a step back. However the model is built upon experiences from the COMBI project as well as on concepts realized in the ATLAS project. Parts that in Combi constitute the 'general model' are now splitted into two levels and they are extended at the same time. A small bur generic part is the minimal kernel. it consist out of definitions of generic objects, attributes, association, etc.. The new level, the generic objects level specializes and extends definitions of the kernel in a way that the model finally could be used as an integrated project model as it was the aim of IRMA (Luiten et. al. 1993) and others attempted for example. The partial model level from COMBI in way can be compared with the object_core level. The prototype implementation is a slice through the model form the minimal kernel to physical objects and to building_elements down to an architecture application module. The prototype implementation of the model is quite promising.

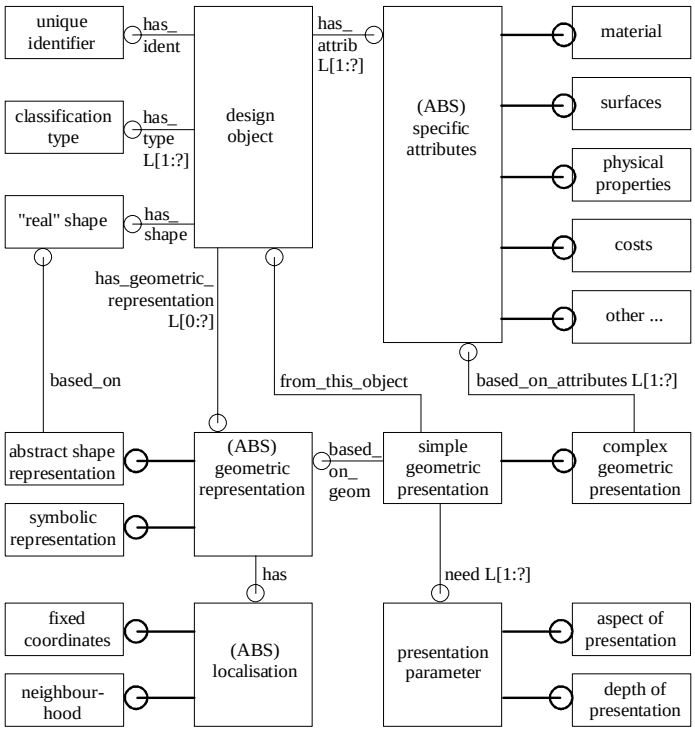


Figure 8 Design object as root of the minimal kernel

Figure 10 is showing a small view on the minimal kernel of NextCAAD II. The design object can be regarded as the root object of the model.

The ESPRIT project VEGA

The author is convinced that the only computer based environments really excepted and freely integrated in daily working habits will be those that are working in a similar manner as the human experts of a design and/ or construction team do. Such a team consists of various experts from different domains and expertise. They are highly dependent on each others working results as basis for own work. A strong interactive communication using all kinds of documents and the spoken word is a necessity. The accessibility of information as basis of each others work is based on interpretation which transforms the received information into the experts domain world. The experts are working on the same building elements but having very distinct understandings, semantics of those elements.

One could say all those experts are having different 'Brain Implemented Models' (BIM) compared with the models we are used to talk about. There is no person existing with an above all central knowledge, each is understanding their own domain often (or normally?) using different 'expert languages'. See example of architect and structural engineer talking about a concrete column or load bearing wall. In addition to their own BIM they have to have a filtering and translation knowledge enabling them to find the mutual implications of a certain decision on their tasks. Without such 'mechanisms' it would be necessary for each and every time to communicate in a lecture style manner, surely not a useful type of communication between experts during a design task.

Transferred to an research issue in the IT world this picture of the design/ construction team leads to topics that are very actual today: IT for virtual enterprises, CSCW (Computer Supported Collaborative Work) and distribution of objects ore data models in an network environment to name only a few. Solutions enabling virtual enterprises collaboration in an distributed product data model environment are the objective of the ESPRIT project VEGA (Virtual Enterprises using Groupware tools and distributed Architectures). The task structure of VEGA gives an answer to the question which ingredients are needed for an implementation of the 'BIM' idea in an computer environment.

The VEGA project can be structured into five main technical tasks. These are:

1. Conceptual model architectures and product models for a distributed environment (PDM)
2. Technologies enabling communication between applications in an distributed model environment (COAST)
3. Distributed information services (DIS)
4. Work- and information flow management and control (WFL)
5. Implementation architectures and techniques

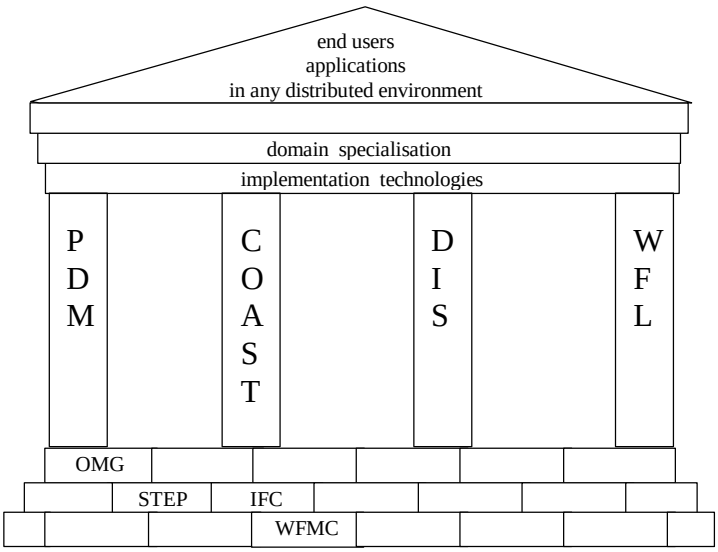


Figure 9 The structure of the VEGA project

To use a picture the first four main tasks are forming columns standing on a basement and bearing an architrave and a gable. The basement bearing the columns are standards like ISO's STEP, OMG's CORBA, WFMC's Workflow interface definitions, IAI's IFC and the Internet with HTML and VRML. And it is not only that these standards or supporting these tasks, there is also a counter reaction back into the standards. OMG has accepted a proposal for extensions of CORBA coming out of VEGA's COAST, there are influences into the workflow interface definitions as well as into IFC's architecture(Junge et.al. 1997).

The architrave is formed by implementation technologies and domain specialization. These are consisting of two parts: first a generic implementation environment for EXPRESS product data models schemata, the Dynamic Product Model. This PDM among other features allows dynamic evolution of a schema under runtime of the application. The second is a generic product data model which is the necessary task for a specialization of the PDM for the domain of architecture and engineering. The following is dealing with the 'VEGA model'.

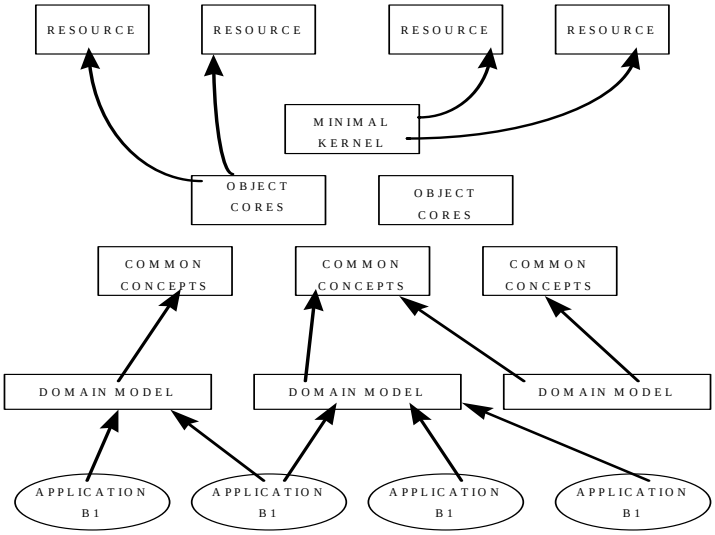


Figure 10 The VEGA model architecture

The architecture of the VEGA model is building again upon COMBI and especially extending NextCAAD II model prototype capabilities. As in the COMBI Framework ‘recourses’ are used. These can be ‘borrowed’ and/ or adapted ones, for example from STEP, or native ones, developed for the project. The idea simple is that parts used in several places of the schema should be referenced and not copied into the parts.

The minimal kernel is quite identical with its forerunner, but off course results from prototype implementation are introduced. The two levels which where named generic objects and object cores are now being coupled together under ‘object cores’. It is always a little bit fuzzy to distinct one level from the other or to clearly identify the need for a ‘level’ to be introduced. So one could discuss this separation or not separating of the two. The objective however is similar as before: to extend and specialize definitions of the minimal kernel, which are held very generic into the direction of domain model objects.

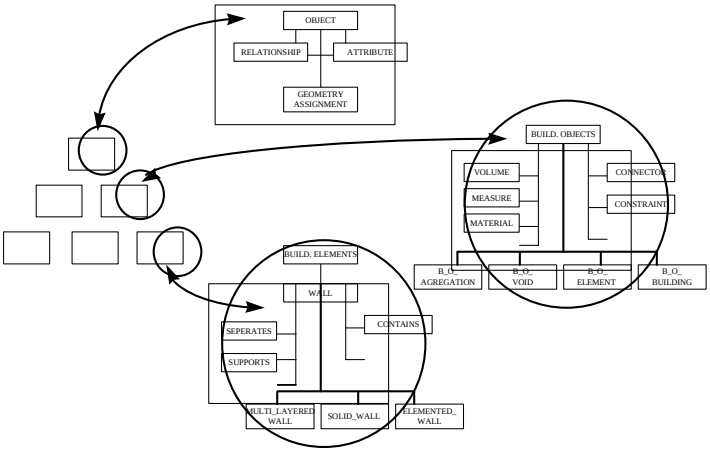


Figure 11 Components of the VEGA Kernel

A level introduced in the VEGA model architecture is that of ‘common concepts’. The idea behind this is, using the example of a wall for explanation, is the following: In most building product definitions of building elements like wall are being found at the level of aspect or domain models. This results in the fact that there are different wall definitions with respect to their domain requirement or aspect. The level atop those domain models end with definitions of the general building element of which wall is a subtype. The fact that walls are to be defined on domain model level only results in what can be called horizontal mapping or better translations between the different ‘walls’, e.g. the ‘architects wall’ and the ‘structural engineers wall’. These mappings are much more pain making tasks compared with the horizontal mappings at the transitions between one model level to the next lower or higher one. The goal of the ‘common concepts’ is to avoid these horizontal translations and replace them with vertical mappings, in the case of wall from either ‘architects wall’ or ‘structural engineers wall’ to the common concept of wall.

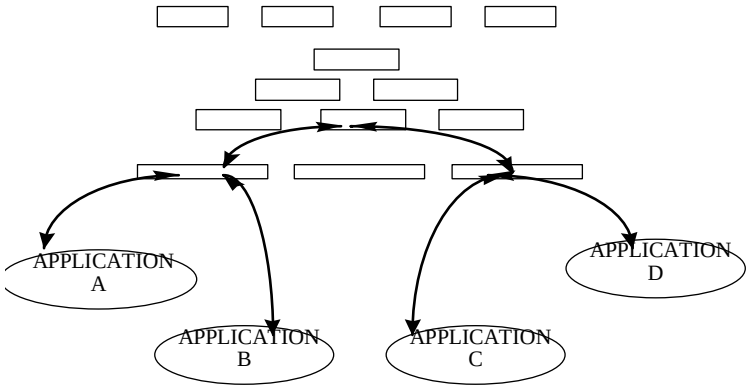


Figure 12 Communication through the VEGA Model

Figure 14 is showing how applications of the same domain (A with B or respectively C with D) are communicating through their domain models. While for communication of A with D, applications of different domains, both using their domain model plus the next higher level which is the common concepts.

By the time of 'CIB '97' VEGA will be at its 'mid-term'. Some parts that today are still concepts will be prototyped. Thus more open questions on the way to a 'consolidated building product data model' will be answered. Others, off course, will stay open. Results from VEGA up to date are quite promising. VEGA will be a mayor step towards the authors concept of 'integration by communication' (Junge 19991) and it is extending it to collaboration of human- inter- actors. Up till today the project was concentrated on working on the technological foundations. Future work will use these for building of a new type of applications enabling architects and engineers not only to communicate through their applications in an distributed environment modeled after their habits, it also will be a major step to the new CAAD generation.

2. Implementation Strategies

The core technology for the software application part of the project is pure product modeling technology as such. The advances made, lie in a solution to overcome the normally rather static then flexible structures of product models. This kind of product model has named DPM Kernel (Dynamic Product Model Kernel). This name should indicate that while the application is directed to fulfill requirements for its use in the building domain it generally is a generic solution from an IT point of view. The DPM Kernel consists of the following four basic components:

- Instances — are the concrete entities as stored in the data base.
- Patterns — are describing the properties of the instances. The properties may be attributes, methods or the ability to be part of an association.
- Methods — are describing the behavior of a pattern's instance, e.g. when the value of an attribute or an associated object changes.
- Filters —are enabling the user to request and find any information stored in the model.

In order to realize how these four parts are helping to gain the high flexibility, openness and extensibility of the loaded schema, that mark the main benefits of the approach, it is important to understand the generic base technology of the implementation. This

technology allows object oriented modeling of the knowledge of a specific domain, such as architecture, as well as Dynamic Schema Evolution (DSE). Dynamic Schema Evolution means that the once modeled part of the world is not kept in a static and determined schema. DSE is modification, creation and deletion of schema information at runtime.

The DPM is a meta model and because of that the components of the DPM Kernel are not 'hardcoded' classes of the entities of a specific domain. There is for example no class 'wall' or a class 'room' in the DPM. The traditional 'hardcoded' implementation strategy would result in the implementation of a class together with its attributes as data members and additionally its methods for every entity of a domain. As a consequence, the entities could only be changed in a new version of the application (Figure 2). Thus the whole model is static and can not be expanded or modified after loading it to the DPM Kernel.

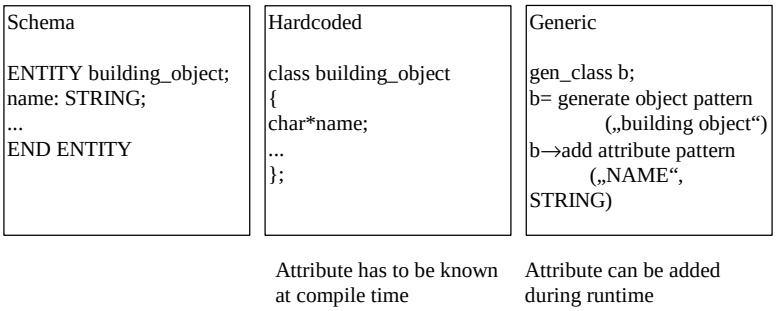


Figure 13 Generic vs. hardcoded approach

But the fact that the project didn't stick to the traditional 'hardcoded' concepts doesn't mean the paradigm is a non object-oriented one. In contrary, all important principles of the object oriented paradigm (Stoutrup 1992), such as abstraction, inheritance, information hiding or polymorphism are also fully available in the DPM Kernel.

The main constructs of the DPM Kernel are the patterns. Patterns are equivalent to classes in the traditional 'hardcoded' concept. They are providing the functionality to have attributes, methods and to be part of associations. Such a pattern gets its specific meaning of being, e.g. a wall, by a process called Dynamic Data Typing (DDT). Dynamic Data Typing is a functionality provided by the DPM-Schema-Loader.

With these four basic components in place the semantics of any specific domain can be described and stored. Moreover, description of the domain is not only a static one. It is dynamic in two respects. First, methods are used to describe the behavior of the instances and associations are used to propagate the changes of an instance to the associated instances. Secondly, with the generic approach the model can be extended even at runtime.

The base technology described above is an abstract technology for modeling the knowledge of any specific domain. The problem is reduced to a appropriate formalized description method for the knowledge of the domain. This is the place for the product modeling technology itself.

A specific schema of the building domain is loaded in the application as it currently is, which includes the construction and the semantics of a building. But, it has to be emphasized that in principle any schema formulated in EXPRESS can be loaded.

To get real instances for the model, standards from STEP and IAI are used. In the current version the AP225 protocol is used to read in a whole building from any CAD system, which supports the AP225. An IFC interface is also existing as a first prototype.

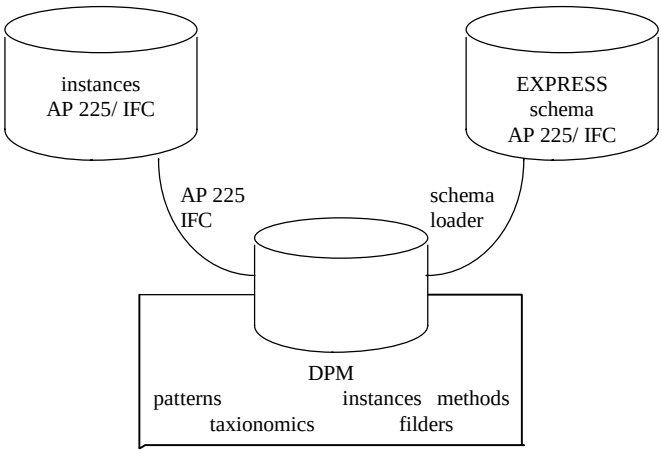


Figure 14 The base technology

The next step will be to distribute the information stored in and managed by the DPM Kernel over wide area networks, so that architects and engineers can work concurrently on the building model and have access to it regardless where they are working. We see the forthcoming COAST Platform from the VEGA project to be the appropriate way to do this. Therefore the DPM Kernel provides also an API, through which it can communicate with the COAST-platform.

3. The application

One can discuss the value, the potential extensions of functionality that DSE could have to end user' applications against the dangers of creating chaotic situations in an end user environment and the dangers DSE could have in information exchange. Without any doubts, however, are the advantages it opens in application development. The DPM constitutes a powerful and flexible basis for application development on top of a product data models. Together with foundations described in 'Product Data Model for Interoperability in an Distributed Environment' (Junge and Liebich. this volume) and 'The VEGA Platform' (Junge, Koethe, Schulz, Zarli, Bakkeren. this volume) it provides possibilities for new application software in the architectural domain. Only a few of those possibilities can be touched in the following.

The DPM itself is a neutral implementation method of any Express schema totally independent of any specific domain. What is needed to make it's use specific and useful to the architecture and building engineering domain is a core model of the kind described in 'Product Data Model for Interoperability in an Distributed Environment'. It follows a strategy to provide a core of semantics to allow domain models to be ' plugged in' to it. This core, in the project called 'BPM Kernel' (Building Product Model Kernel) provides the necessary semantic for communication between these domain models. Without such a level of commonly agreed semantics a technique like DPM would be useless in an implemented environment. The dynamically under runtime created extensions of a product model schema would not make any meaning to anybody besides the creator himself.

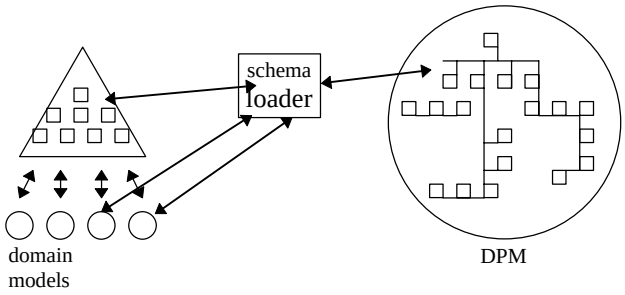


Figure 15 BPM Kernel and DPM

The applications using the DPM, as well as the VEGA platform together with the described product data model, currently under investigation are pointing in different directions, which are:

- CAAD kernel
- Coordinator Workstation
- A tool set (the current prototype implementation)

Ideas behind 'CAAD kernel' are being worked in the 'NextCAAD' projects, and where presented at CIB Conference in Stanford '95 (Junge 95a) (Junge 95b). Ideas behind 'Coordinator Workstation' very briefly are described in the following.

The distributed environment envisaged to become reality by the VEGA Platform lets arise some questions to be answered before it can be successfully implemented in design offices. How can such a distributed environment be managed, be made transparent for the user? What are the functions needed? It seems to be an approach appropriate to be followed again is a recollection of processes and functions as in today's design offices. In today's practice a very important role in project work is on coordination. In larger projects a specific role is that of coordinating all participants, may it be in the home office itself, or with all engineering offices participating in the project. The tasks that such a coordinator has on his agenda have to be translated to this new computer tool. Candidate functions of the Coordinator Workstation are:

- Distribution and collection of models or model parts .
- Control of communication, network and workflow.
- Coordination of design tasks.
- Conflict detection and management.

Prototype Implementation

The prototype applications which are implemented today upon the BPM/ DPM-Kernel contain, among other features, three main parts. These parts are giving access to the stored building for the user:

- The Project Explorer
- The 3D-Window
- The Spreadsheet Window

The Project Explorer

The Project Explorer is making the DPM-Taxonomies provided by the BPM/ DPM-Kernel visible for the user. Apart from the functionality described in section 2.3.4 the Project

Explorer allows different kinds of documents to be attached to every taxonomy node. These documents are also DPM-instances. Because of this, they have the same functionality as a common DPM-instance, e.g. they can have attributes. In the prototype version the following kinds of documents are available:

- 3D-View documents include a 3D view of the whole building or a part of it, with a specific camera position.
- Report documents include the result sets of filters in a spreadsheet, which may be designed by the user.

The User can browse through the whole building taxonomy as well as view and modify the attributes of the building elements at the click of a button. Moreover, it is possible to import the same building in different versions in the same project. Executing the same reports on the different versions will show the consequences of the changes done in the CAD application.

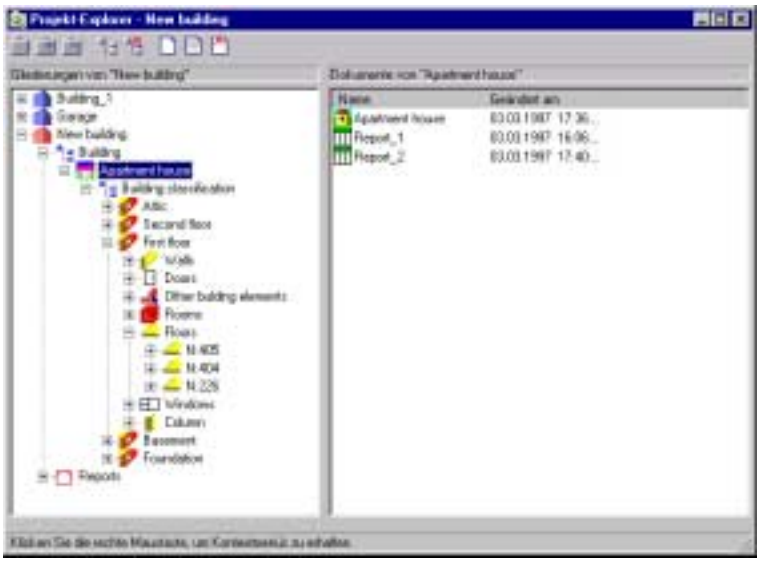


Figure 16: Example of the Project Explorer with a building taxonomy

The 3D-Graphic-Window

The 3D-Graphic-Window visualizes the geometric attributes. This means that the client gets a 3D-image of the whole building or of a part of it. It allows an animated navigation through the building. Identifying and picking of one or more elements is possible. However, the attributes of identified elements can be viewed and modified. Furthermore, it is possible to highlight all elements that are the results of a filter execution or that are identified in the project explorer. The attributes of these highlighted elements can also be viewed and modified.

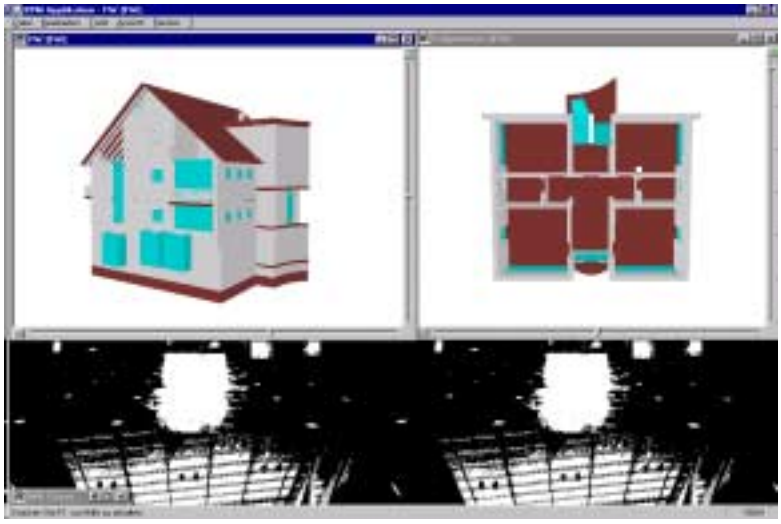


Figure 17: 3D-Window with the whole building(perspective view)
and 3D-Window with one floor (bottom view)

The Spreadsheet-Window

The Spreadsheet-Window shows the results of filters in a very comfortable way. The spreadsheet is fully EXCEL-compatible and offers all abilities like formatting, calculation etc., that are known from EXCEL. In the design window the user can define the whole layout of a report specifying which attributes it should include, defining how attribute values are sorted and so on. All defined filters can be stored in a catalog categorizing them under specific groups e.g. “All room filters”. So a previously defined filter can easily be used in another project.

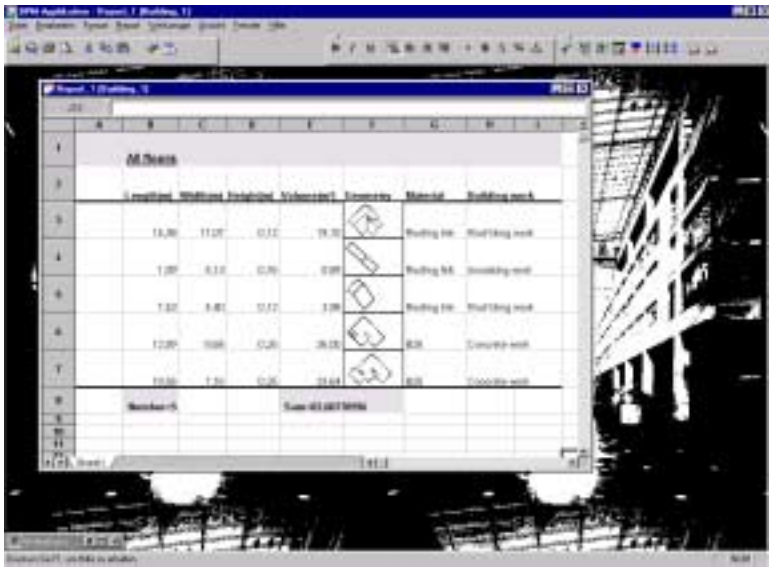


Figure 18: Spreadsheet-Window with the result of a filter execution

References

ATLAS, [1994]
ESPRIT Project 7280: various deliverables

Bjoerk, Bo-Christer [1993]
Byggproductmodeller- Nulaege Byggforkskningraedet, R27:1993

Bjoerk, Bo-Christer [1994]
Contributions to the theory of building product data models

Bjoerk, Bo-Christer [1995]
Requirements and information structures for building product data models, VTT Publications no245, Technical Research Centre of Finland, Espoo

Eastman,C. [1995]
Structure of a Product Database supporting Model Evolution. CIB Workshop proceedings, Stanford University.

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

IMPACT [1991]
IMPACT Reference Model. Deliverable, ESPRIT Project Impact

COMBI, [1994]
Junge, R. Ammermann, E. Katranuschkow, Scherer, R. ESPRIT Project 6909: deliverable A2, A3, B3

Junge, R. [1991]
Integration by Communication. 1st International Symposium on Buillding System's Automation - Integration Conference Proceedings, Univ. of Wisconsin, Madison

Junge, R, [1994]
Bauproduktmodelle: Eine Einführung Arbeitspapier, OFD Berlin, Bundesliegenschaftsverwaltung

Junge, R. and Liebich, T., (1995)
Product modelling for applications: Model for next generation CAAD, Computing in Civil and Building Engineering, Pahl & Werner (eds), Balkema, Rotterdam

Junge, R. (1995a)
Aspects of New CAAD Environments. CIB Workshop proceedings, Stanford University.

Junge, et. al. (1997)
Junge, R. Köthe, M. Sschulz, K. Zarli, A. Bakkeren, W. The VEGA Platform, CAAD futures '97, Junge, R. ed. Kluwer. in print

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

Luiten, G., Froese, T., Bjoerk, B-C., Cooper, G., Junge, R. Karstila, K. and Oxman, R. 1993
IRMA, An information reference model for architecture, engineering and construction. Mathur, K., Betts, M. and Tham, K. (eds.) Managementof Information Technology for Construction., World Scientific & Global Publication Services, Singapore 1993

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

Richens, P. [1978]
The Oxsys system for the design of buildings. 3rd International conference on Computer in Engineering and Building Design CAD78., Ipc science and technology press, UK

Stroustrup, B.. (1992)
The C++ programming language, Addison-Wesley Publishing Company, Inc.

Tarandi; V. [1993]
Oblect oriented communication with NICC, neutral intelligent CAD communication. Mathur, K., Betts, M. and Tham, K. (eds.) Managementof Information Technology for Construction., World Scientific & Global Publication Services, Singapore 1993

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