VIRTUAL PRODUCT MODEL

Danijel Rebolj[1]

[1] University of Maribor, Faculty of Civil Engineering, Construction IT Centre

ABSTRACT: Product models are currently recognized as the relevant solution for the problem of disintegrated islands of automation in many engineering fields. Some researchers working in the field of Construction IT have, however, expressed doubt about the ideality of this solution in Civil Engineering and have described many deficiencies and new problems, introduced by the concept of product models. The article summarizes some deficiencies and then introduces a solution, called Virtual product model, which is based on decomposition of a conventional product model. The concepts, basic components and an example of the Virtual product model are described.

KEYWORDS: Product models, process models, harmonization, intelligent agents, virtual product model

1. INTRODUCTION

From the physical scientific experience we are seeing the cosmos as an interaction of particles and (or) energies, which are subject to simple rules, known as physical laws. Because there exist many combinations of particles the outcome of interactions are not simple to predict. It seems that the many combinations form more and more complex structures, as we observe them in more detail.

Today many researchers, working in the field of engineering information technology, recognize the problem of modelling complex structures, and many are asking themselves whether an all-including-product-model is a solution for an integrated information environment that should efficiently support the life-cycle of a product. It seems that rich experiences in product modelling in the last decade lead not to better and better models but rather to the awareness that the more complex the product models are, the more rigid and the less usable they become in reality. These recognitions already led to some suggestions for the future integration methods and product modelling.

Before we continue to analyse the deficiencies of complex product models, let us briefly browse through the short history of product modelling. Probably everything started when the first data interface has been implemented, which has linked the output of one computer program to the input of another. After that successful integration, researchers started to develop more sophisticated integration methods. According to the principle we can divide them in the following groups:

- Integration of different stand-alone programs with the help of information interpreters, as for example in the "software fixing" method (Syal et. al. 1991). These methods have two main deficiencies: they don't enable fluent information flow, and it is necessary to implement a new interpreter for every new program we want to include.
- The use of a common medium for information exchange between programs. "Blackboard" is one such method (Yau et. al. 1991), which enables a fluent information exchange through a common "blackboard". The "Object shell" method (Rebolj 1993)

supports a fluent information exchange as well, however all these methods still require implementation of new interfaces to include a new program.

- The integrated database concept, where all included programs use a common data repository. There have been many projects, which have developed and used this concept: RATAS (Björk 1989), ATLAS (ATLAS 1992), COMBINE (Augenbroe 1993), COMBI (Ammerman et. al. 1994), and in the last years SPACE and OSCON, which were the fundamentals of the probably technologically highest developed integrated environment in civil engineering, described in (Faraj et. al. 1999). Among earlier, but less known systems, the CIS, Construction Information System (Rebolj 1990) introduced an integrated geometry-construction database. Many authors published more detailed reviews of relevant projects and systems, including the listed ones (e.g. Amor 1998, Eastman and Augenbroe 1998).

Nowadays the integrated database concept is recognized as the most effective method for integration of computer programs in the life cycle of a building object. The integrated database contains the complete description of a product, therefore such data models are known as product models.

2. DEFICIENCIES OF COMPLEX PRODUCT MODELS

Present examples of product models show a tendency to build a unique all-including complex model for a specific engineering field (like shipbuilding, car industry, building industry, road building, etc.). However, none of these attempts has been generally accepted in the civil engineering practice. Rather, the past development of building product models led to a question, whether a definition and use of a standard total product model has sense at all. To overcome the need to have a single product model some authors have proposed inter-model linking schemes (like in Spooner and Hardwick 1997, and Pfennigschmidt et.al. 1997), in this way, however, the complexity of the whole system hasn't decreased, even worse, it grew.

Another problem is arising from the necessity for standard building elements. The history of mankind shows that in communication the only "standard" is the diversity of standards. In other words, it seems most unlikely that the whole mankind would use a single standard language. Even if such a language would exist, it is very likely that soon many dialects would appear, since every individual or group is seeing the same thing in its own perspective.

This problem is even extended in civil engineering and construction, where many different views have to be considered through a product life cycle. Different views are leading to more or less different descriptions (data structures) representing the same entity. Notable progress has been made by the International Alliance for Interoperability with the development of the Industrial Foundation Classes, which can be seen as applicable building blocks (IAI 1996), but which are still not resolving the problem of views (as evident from Yu et.al. 2000).

A conflict between the concept of a single integrated model and the need for individuality also showed up. Companies (and individuals) have a strong affection to fully control their own data, which also form the company's "memory" (Larson 1998), a vital part of every company.

Such and similar problems have already been recognized by some authors, who have expressed their hesitation either between lines (e.g. Graves 1998, and Amor 1998) or directly (as in Eastman and Augenbroe 1998, and Turk 1999). Appending author's own experiences, the main deficiencies of product models could be summed up into the following essential points:

- product models are based on clearly defined semantics and demand unique standard basic elements, however, such elements don't exist,
- computers are not (yet) capable to fill up semantic inconsistencies and holes, which show up in the integration of computer programs (a human is adapting daily in communication with other humans with different mental models, and is capable to reconceptualize parts of information, which don't fit into the whole),
- product models are subjective interpretations, not objective representations of the real world, therefore an effective uniform product definition is not possible,
- product models only include parts from the building process and disregard some important views (social, environmental, etc.), which form the process in the real world,
- models are restricting creativity due to their complexity and rigidity,
- when implementing prototype models into the real environment they fail due to the inability to consider the rich knowledge and experience of the people,
- although product models are basically open, they get stiff and hardly upgradeable in the real world,
- in an integrated database each partner's control over his own data is limited.

In (Eastman and Augenbroe 1998) and (Turk 1999) authors also propose some solutions to the problems they described:

- product models should be rather small and limited to specific areas; coexistence of more models in the same field is not necessarily bad,
- implementation of middleware tools between applications and models, which will help humans to navigate between the islands of automation,
- gradual implementation of small models into industry,
- development of a richer set of language constructs for model description,
- product and process models should be linked more closely,
- new integration concepts should be tried, which would not reside on integrated semantics,
- it is necessary to allow coexistence of structured information and unstructured data and leave their interpretation to the human,
- programs should not limit but extend the engineers being in the world (virtual reality, telepresence, multimedia, etc.)
- pure information exchange should be upgraded with communication software for collaboration support.

3. VIRTUAL PRODUCT MODEL

3.1 Concept

On the basis of the bad and the good experiences in modelling of building products the author proposes a concept of the Virtual Product Model (VPM), which could preserve positive, and avoid some negative characteristics of product models.

The virtual product model is represented by a network of loosely coupled particle models, interconnected by relatively simple but strong rules (like gravity in the macro-cosmos). The neighbourhood of a particle model is in a logical sense defined through a process model, which also determines relations between particles. So the main point lies in communication network between particles.

Let us recall here the concept, which solved the complexity problems of computer networks. The problems of how to get together many different communication technologies seemed unsolvable, until a cut has been made in decomposing the network into clearly defined functional layers – a solution nowadays known as the 7-layer ISO model (CTRC 1989).

Decomposition has also become a magic word in software development. Huge monolithic systems tend to evolve into open and flexible structures of software components (IEEE 1997).

The virtual product model can be explained as a decomposed product model, consisting of three main layers (see figure 1):

- particle models or particles (data structures used by applications),
- a process model, which determines the particle interconnection scheme (the "higher sense" of particles), and
- communication network, which is responsible for harmonization of particles and implements the "rules" on them.

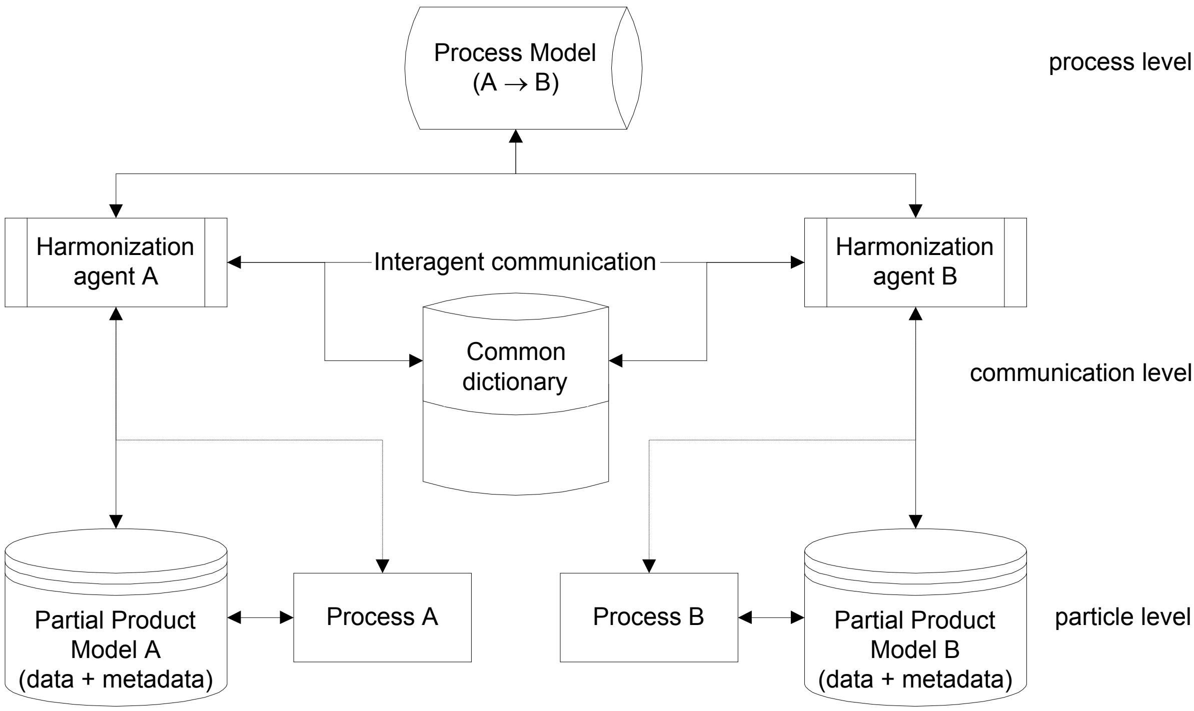

Figure 1. Virtual Product Model basic scheme

It is believed that such decomposition will decrease complexity of the product model to a manageable level and increase it's flexibility through the autonomy of applications and partial product models - particles.

3.2 Basic Components

As already noted, relations between particles are of most importance. From the aspect of the model as a whole the proper relations should assure the integrity of the model. Therefore special attention has been given to the harmonization of the content of particle models, which are representing parts of the virtual product model. The mechanism is based on harmonization agents, which are leaving the particles their individuality but also bind them to the whole.

Harmonization agents do not require uniform semantics of particle models, but only common basic primitives. It is therefore possible to allow different structuring and representation techniques and standards for particle models. While communicating harmonization agents use their own knowledge about structures, which they gathered and saved in common dictionaries, whereby in insolvable situations agents establish contact with humans. Actually, agents will in the first stage act as assistants, then as advisors and at the final stage as autonomous agents. (More details are described in Tibaut 2000.)

The common dictionary is a repository of basic element (term) descriptions in a semantic domain. There is a domain dependent starting set of terms with relations between them, which assure starting the communication between agents. It is however supposed that agents will soon come into situations where basic terms and relations won't be enough to exchange views in a new situation. In such cases, agents will have to ask an expert - either a human, as mentioned earlier, or another, more experienced software agent. For the second case, agent "chat rooms" will be the place, where "inexperienced" agents will have the opportunity to learn, and then improve their "native" dictionary. Building a dictionary automatically from very simple starting terms should avoid coming into the known trap of defining complex view mapping schemes (see Spooner and Hardwick 1997).

To become a part of the VPM the particles (applications + data structures) have to fulfil certain conditions regarding data representation:

- all exchange data has to be available in an external representation in a text form (which also implies database systems able to communicate in text form),
- a data description (metadata) has to be available,
- data has to be structured in an object-oriented way, using Express, XML, but also non-standard languages.

Having in mind that it is not necessary to adapt the semantic and the data structure of a particle to integrate it with the product model, even old (but good working) computer programs can be upgraded to suit the conditions. It is, however, a good idea to redesign the data structures to give particles a better ability to interact with others.

On the other hand, the VPM concept, which supports the flexibility and autonomy of applications, is a good accelerator of application's (particle's, component's) self-intelligence and adaptability.

3.3 Example

Figure 2 shows the use of the VPM concept on the example of a part of the road life cycle (the road has been in the focus of our research group in the last few years; see Rebolj 1999). A simplified scheme of the process model shows a chain of tasks, with the information about the program(s) and the external data representation used in a specific task. The process model is built, or adopted, for each specific project, because every project can include slightly different tasks, carried out by different programs.

The scenario (shown on figure 2) starts with the activation of the task "Emission analysis", which is supported by the program named Dynem. When the user specifies the project name he wants to work on Dynem tries to read relevant data. The read request is intercepted by the Dynem's harmonization agent, which checks the status of the data in the project model (implemented as a project database). If the requested data is harmonized with the predecessor particles, the agent releases the reading request. Otherwise, it locates the data source in the process model and establishes communication with the responsible harmonization agent. In our case it is the Plateia-agent, responsible for the geometrical design. When Plateia-agent gets the description of the requested data structure, it tries to find it in its particle model and returns the data in the agreed form (XML is proposed). Now the Dynem-agent can update data in "his" particle model and release read request, and the user can work with harmonized data.

From this example it can be seen that data in the VPM is not harmonized all the time, but only on demand. This mechanism simplifies the harmonization of the model as a whole, but still assures correct data when it is needed.

It is believed that the principle of the VPM will be especially effective in civil engineering, where processes and partners, as well as applications used, are changing from project to project. However, same applications (particles) are used more often, which makes it possible for their agents to improve.

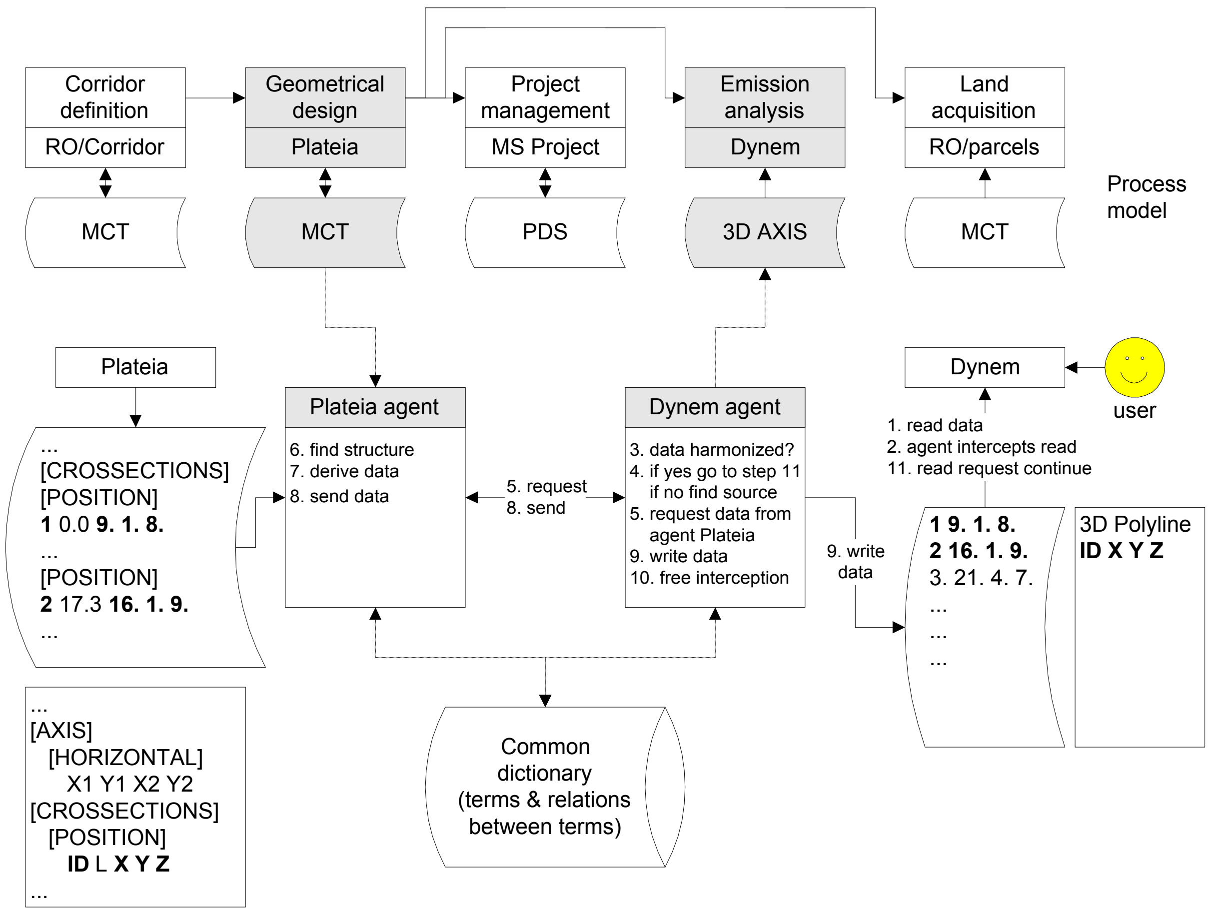

Figure 2. Example of VPM mechanism operations

4. CONCLUSION

The concept of the product model is a result of human's mental activity and desire of mastering the whole to the smallest possible detail. Especially their development in civil engineering shows that the human has, as so often before, ignored the natural laws as well as himself. He only relied on his own mental constructs and has equated them with the objective reality. His models work only under special circumstances, but they are not generally applicable. This does, however, not mean that the concept of product models is useless.

Through the concept of the virtual product model it is believed that it is possible to preserve the independence and flexibility of particles - existing island models and applications, and the simplicity of mastering them, but also to preserve the positive integration effects of complex product models. The reason for this conviction lies in the simplicity of used principles and in their closer relation to natural mechanisms (basic laws), which also includes the ability of implicit evolution. The evolution and improvement is supported by harmonization agents, which not only communicate, but through the communication also gain new knowledge and develop adaptability. In short, we have tried to find a mechanism to avoid complexity, considering the words of German philosopher Oswald Spengler "Everything complex is of short lifetime".

REFERENCES

Ammerman E., Junge R., Katranuschkov P., Scherer R.J. (1994). Concept of an object-oriented product model for building design, Technische Universität, Dresden, Germany.

Amor R. (1998). A UK survey of integrated project databases, Proceedings of the CIB W78 conference The life-cycle of construction IT innovations, The Royal Institute of Technology, Stockholm, Sweden, pp. 67-76.

Augenbroe G. (1993). COMBINE, Final Report, Delft University, Delft, The Nederlands.

ATLAS (1992). Architecture, methodology and tools for computer integrated large scale engineering – ESPRIT project 7280, Technical Annex Part 1, General Project Overview.

Björk B.C. (1989). Basic structure of a proposed building product model, Computer Aided Design, 21(2), pp. 71-78.

CTRC (1989). International Standards for the Computer. Computer Technology Research Corp., New York.

Eastman C., Augenbroe F. (1998). Product modeling strategies for today and the future, Proceedings of the CIB W78 conference The life-cycle of construction IT innovations, The Royal Institute of Technology, Stockholm, Sweden, pp. 191-208.

Faraj I., Alshawi M., Aouad G., Child T., Underwood J. (1999). Distributed Object Environment: Using International Standards for Data Exchange in the Construction Industry, Computer-Aided Civil and Infrastructure Engineering, 14(6), pp. 395-405.

Graves G. (1998). Industry requirements for data standards harmonization, Proceedings of the Global Business Solutions for the new millenium, CD ROM.

IAI (1996). End User Guide to Industry Foundation Classes, Enabling Interoperability in the AEC/FM Industry. International Alliance for Interoperability (IAI).

IEEE (1997). Engineering meets the internet: how will the new technology affect engineering practice?, IEEE Internet Computing, 1(1), pp. 30-38.

Larson M. (1998). AF integrated digital environment, Proceedings of the Global Business Solutions for the new millenium, CD ROM.

Pfennigschmidt S., Kolbe P., Pahl P.J. (1997). Integration von Datenmodellen, Proceedings of the IKM conference, Weimar, CD-ROM.

Rebolj D. (1990). Graphic Modelling of Superstructures, Automatika, 31(1-2), pp. 147-156.

Rebolj D. (1993). Computerunterstützter integrierter Straßenentwurf in einer objekt-orientierten Umgebung. Verlag für die Technische Universität Graz.

Rebolj D. (1999). Integration of computer supported processes in road life cycle, Journal of transportation engineering, 125(1), pp. 39-45.

Spooner D.L., Hardwick M. (1997). Using views for product data exchange, IEEE Computer Graphics and Applications, 17(5), pp. 58-65.

Syal M.G., Parfitt M.K., Willenbrock J.H. (1991). Computer integrated design/drafting, cost estimating, and construction scheduling, Housing Research Center Series Report No. 11, The Pennsylvania State University, Dept. of Civil Eng.

Tibaut A. (2000). Intelligent agents for better information management process in construction, submitted to Construction Information Technology 2000, Reykjavik, Iceland.

Turk Z. (1999). Constraints of product modelling approach in building, Proceedings of the 8th International conference on Durability of Building Materials and Components, NRC Research Press, Vancouver, Canada, pp. 2776-2787.

Yau N.J., Melin J.W., Garrett J.H., Kim S. (1991). An environment for integrating building design, construction scheduling, and cost estimating," in ASCE Seventh Conference on Computing in Civil Engineering and Symposium on Databases, Washington, D.C.

Yu K., Froese T., Grobler F. (2000). A development framework for data models for computer-integrated facilities management, Automation in Construction, 9(2), pp. 145-167.