4 SIGraDi 2000 Rio de Janeiro
constructing the DIGITAL space

Towards a Working Design Environment: from Enterprise to Functionality Model

Ann Hendricx
ann.hendricx@asro.kuleuven.ac.be
K.U.Leuven, Faculty of Applied Sciences, Department of Architecture, Urban Design and Planning

Herman Neuckermans
herman.neuckermans@asro.kuleuven.ac.be
K.U.Leuven, Faculty of Applied Sciences, Department of Architecture, Urban Design and Planning

Abstract

Several product-modelling initiatives have produced static descriptions of the architectural and geometrical objects capable of describing architectural design projects. Less attention is paid to the development phase in which these static models are transformed into workable architectural design environments. In the context of the IDEA+ research project (Integrated Design Environment for Architectural Design) emphasis lies on the systematic development of both phases. The result is an analysis model that consists of two submodels. On the one hand, the enterprise model defines the architectural and geometrical objects, their methods and their relation with other objects. On the other hand, the functionality model organises the functionality objects – ranging from single-event objects to complex-workflow objects – in a layered and easily expandable system. As such, it closes the gap between the static enterprise model and the dynamic design environment as a whole.

Resumen

Diferentes iniciativas en el área de 'product modelling' han producido descripciones estáticas de los objetos arquitecturales y geométricos que constituyen un proyecto arquitectural. Pero el procedimiento en el cual un modelo estático se transforma en un sistema factible está menos investigado. El proyecto IDEA+ (Integrated Design Environment for Architectural Design) se enfoca hacia el desarrollo sistemático de las dos fases complementarias. Esto resulta en un modelo de análisis dividido en dos submodelos. El modelo de empresa define los objetos arquitecturales y geométricos, sus acciones y las relaciones con otros objetos. El modelo de funcionalidad organiza los objetos de funcionalidad - una gama de acciones simples hasta muy complejas - en una estructura en capas y fácilmente extensible. Este modelo provee la transición entre el estático modelo de empresa y el factible y completo área de diseño.

Introduction

A uniform building representation

A uniform building representation can be of great help during the entire course of the design process. Ideally this building representation is located in an integrated design environment (Neuckermans 1992).

For the architectural designer on the one hand, he can call the assistance of modelling and evaluating tools situated in the environment. And by using the computer from the early stages in the design process, the unproductive translation of a design made by hand into a digital version would be avoided.

For all partners involved in a building's life cycle on the other hand, they can use and contribute to one single building representation, instead of producing their own. Next to avoiding the waste of time and manpower, the communication between the building partners will improve and inconsistencies between models will be avoided. To make this possible, the digital building representation should be located at a central and accessible place (using web technology).

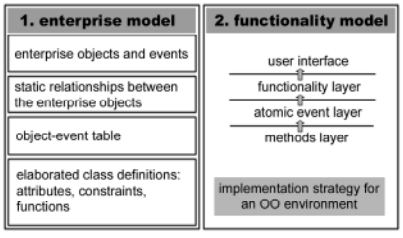

figure 1 - Enterprise model and functionality model

MERODE, an object-oriented analysis method

Several product-modelling research initiatives have produced static descriptions of the architectural and geometrical objects capable of describing architectural design projects. Less attention is paid to the development phase in which these models are transformed into workable design environments. In the IDEA+ research project (Integrated Design Environment for Architectural Design, Hendricx et al. 1998, Hendricx and Neuckermans 1999, Hendricx 2000) emphasis lies on the systematic development of both phases. This goal was achieved by leaning on the object-oriented analysis method MERODE (Model-driven Existence-dependency Relationship Object-oriented Development, Snoeck et al. 1999).

To manage complex problems, analysis objects are systematically partitioned in objects of the problem domain and objects that embody functionality. The result is an analysis model that consists of two submodels: enterprise model and functionality model *(figure 1)*. The enterprise model acts as a solid kernel on top of which functionality is modelled, leading to a set of independent function object classes. For both modelling phases the specification techniques are clearly defined.

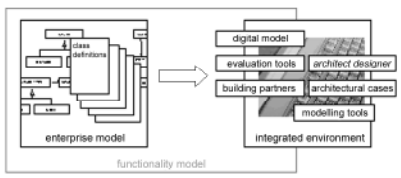

figure 2 - The functionality model closes the gap between the enterprise model and the design environment

The enterprise model

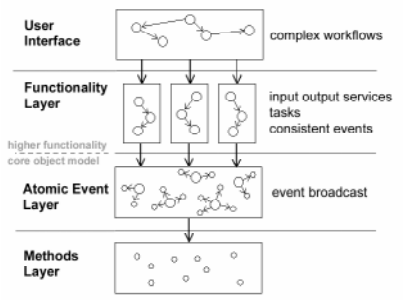

figure 3 - Functionality in MERODE's layered system architecture

The first column in figure 1 shows the steps to be taken during enterprise modelling. Pointing out the relevant **enterprise object types** and the static relationships between them leads to a model apt to give a static description of a design project. In addition, **event object types** describe the fundamental behaviour of the enterprise object types. The associations between enterprise object types and event object types are represented in an object-event table. The class definitions (of both enterprise objects and event objects) gather all information acquired in the previous modelling phases. In the last step, these class definitions are further refined by adding attributes and constraints, and by elaborating the object methods.

At this stage, the core model developed in the context of the IDEA+ project gives the detailed descriptions of both enterprise object types and event object types (Hendricx 2000). Prototype implementations both in legacy software (using a relational database) and using object-oriented technology have been built and validated by means of actual design cases. As to the envisaged global design environment, the underlying structure has been developed, i.e. a graphical kernel and a simplified data structure kernel have been implemented and are used in the development of new packages. Current efforts focus on the development of a geometrical modeller and a fully interactive lighting tool for the early design phases (Geebelen 2000), both examples of newly-developed en-suite tools that make use of one and the same core object model.

The functionality model

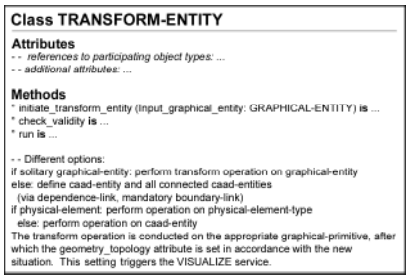

figure 4 - Textual description of the TRANSFORM-ENTITY function

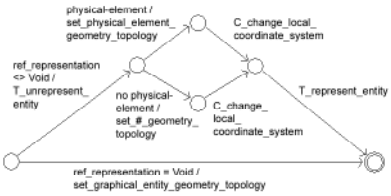

figure 5 - Final State Machine of the TRANSFORM-ENTITY function

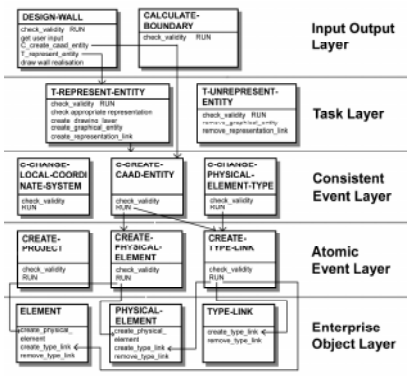

figure 6 - An impression of the functionality built on top of the core object model

Towards an implementation strategy

The functionality model organises the functionality objects – ranging from single-event objects to complex-workflow objects – in a layered and easily expandable system. It is created on top of the enterprise model and closes the gap between a static enterprise model and the dynamic design environment as a whole *(figure 2)*. For instance, our above-mentioned daylight tool can communicate with the core object model through functionality objects elaborated in this phase.

From now on, the model's description is no longer completely independent from its later implementation. MERODE presents implementation schemes towards both object-oriented technology and more legacy technologies. The presented functionality model follows an object-oriented implementation strategy, since this is the route we are taking in the IDEA+ project.

MERODE's layered approach

MERODE's presents a layered system architecture *(figure 3)*. Higher-level events and functionality are composed of lower-level events, leading to a system of cascading event definitions. The basic behaviour of the enterprise objects is located at the two lowest levels and is described in the enterprise model. The object methods - defined in the class definitions of the enterprise objects - are located in the Methods Layer. In the Atomic Event Layer, an atomic event triggers all methods with the same name by broadcasting its message across all enterprise objects.

In the Functionality Layer we find:

- consistent events: a mandatory series of atomic events to ensure the object database's consistent state

- tasks: a voluntary series of atomic events to improve efficiency

- input and output services: input and output functions for the communication between core model and outside world

At the highest level MERODE situates the complex workflows, which make up the interface between the user and the software system.

Describing functionality object types

The techniques for describing functionality objects are illustrated by an example: the input and output service TRANSFORM-ENTITY (Hendricx 2000). An application program built on top of the core object model can use this function to change an entity's geometry or topology by a rotate, move, scale or stretch operation. These are operations that may start a series of cascading events.

The final goal of the functionality-modelling phase is a collection of class definitions describing the functionality objects. These descriptions should be as complete as possible to improve later implementation. Several notation techniques may complement each other:

- textual description *(figure 4)*

- Final State Machine *(figure 5),* with the alternative sequences of events

- Service Specification Diagram, focusing on the events and the involved enterprise objects

Functionality in an architectural design environment

By following the MERODE methodology, we end up with a layered system of event types, tasks and services. Higher–level event types may be composed of lower-level ones. By working with encapsulated objects, a function's contents can change with minimal effects on other functionality object types. Figure 6 gives an impression of possible functionality building blocks. Here the Functionality Layer of figure 3 is subdivided for the three above-mentioned functionality objects.

An event such as CREATE-TYPE-LINK in the Atomic Event Layer is broadcast across all the enterprise objects in the lower layer and triggers all methods sharing the same name. And the same event CREATE-TYPE-LINK is used as a building block to compose the consistent event C-CREATE-CAAD-ENTITY. For efficiency, frequent sequences of consistent and atomic events can be defined as tasks, e.g. T-REPRESENT-ENTITY and T-UNREPRESENT-ENTITY.

In the Input Output Layer, a service such as DESIGN-WALL combines lower level tasks (e.g. T-REPRESENT-ENTITY) and consistent events (e.g. C-CREATE-CAAD-ENTITY).

Since one can rely on existing building blocks, adding new functionality becomes easy. This stepwise-elaborated functionality is an attractive MERODE feature, especially since adding functionality is often a point paid less attention to in product modelling initiatives.

Conclusion

A systematically developed functionality model is the stepping-stone between a static product model for architectural design and a workable integrated design environment. The use of the object oriented analysis method MERODE leads to the development of a layered and easily expandable system of function objects. As it is now, prototypes of such function objects in the context of architectural design have been elaborated. Following work will include the testing of the above structure by elaborating the functionality for a new natural lighting design tool.

Keywords: integrated design environments, product modelling

Palabras claves: sistemas integradas de diseño, product modelling

References

Geebelen, B. (2000) "IDEA-I, An Early-Stage Architectural Design Tool for Natural Lighting". In *Proceedings of IBPC2000*, Eindhoven, The Netherlands, September 18-21, 2000.

Hendricx, A., Geebelen, B., Geeraerts, B. et al. (1998) "A methodological approach to object modelling in the architectural design process". In *Proceedings of the 4th International Conference on Design and Decision Support Systems in Architecture and Urban Planning*, Maastricht, The Netherlands, July 26-29, 1998, CD Rom publication.

Hendricx, A., Neuckermans, H. (1999) "About objects and approaches: a conceptual view on building models". In *Computer in Building/ Proceedings of the CAADfutures'99 Conference*, Proceedings of the 8th International Conference on Computer Aided Architectural Design Futures, Atlanta, Georgia, June 7-8, 1999, AUGENBROE, G., EASTMAN CH. (eds.), pp.133-148, Norwell, MA: Kluwer Academic Publishers.

Hendricx, A. (2000) *A core object model for architectural design*. Ph.D. Thesis, Faculty of Applied Sciences, K.U.Leuven university, Leuven.

Neuckermans, H. (1992) "A conceptual model for CAAD". In *Automation in Construction*, vol 1, nr 1, pp. 1-6.

Snoeck, M., Dedene, G., Verhelst, M. et al. (1999) *Object-oriented Enterprise Modelling with MERODE*. Leuven: K.U.Leuven University Press.