December 1995

Visualization of STructured Modeling

Jian Hu
National University of Singapore

Gee-Kin Yeo
National University of Singapore

Visualization of Structured Modeling

Jian HU Gee Kin YEO
Department of Information Systems and Computer Science
National University of Singapore

Visual programming is the use of graphic representations in the process of programming[Rathnam and Mannio, 1995]. Likewise, visual modeling is the use of graphical representations in the process of modeling. In order to realize visual modeling, SOOM, a representaion for visual models based on structured modeling is proposed and then two kinds of visual modeling approaches: generic visual modeling (GVM) and specific visual modeling (SVM) for MS/OR models are presented in this paper. By using structured modeling conceptual framework and object-oriented concept, the visualization of structured modeling is discussed and the construction of visual models for production and transportation system is described. As a result, a feasible and effective schema of general visual modeling from a meta model to specific model is established.

Keywords: Modeling, visualization, structured modeling, object-oriented concept, generic visual modeling, specific visual modeling.

1 Introduction

It is well known that doing MS/OR tends to be a low productivity activity[Geoffrion, 1989]. Even seasoned practitioners are repeatedly dismayed by how much effort is needed to achieve useful results. This is due to the lack of effective, intuitive and general modeling approaches to facilitate modeling. Usually, MS/OR model is represented by an executable language such as GAMS, AMPL, SML, etc. This needs user to understand all the unintuitive command syntax and abstract the model in mind. It is an onerous task for users, especially end-users, to construct and reuse a MS/OR model.

Graphical or pictorial representations are very useful for modeller and end-user to understand complicated relations and structures of decision models and can be used intuitively to construct them. This is the reason that visualization of decision model on computers is becoming more and more important for modeller and end-user. When the executable language used to describe analytic models is a kind of visual language which consists of a set of visual units, modeling becomes visual modeling. Visual modeling allows the user to create and manipulate graphical objects (models) and perform actions on them that will be understood by the system.

Visual units may include icons, menu-items, dialogue-windows, lines, boxes, circles and text strings. When these are arranged under certain rules, a visual language is formed. Visualization, in general, can be regarded as the translation from a textual language into a visual language[Kamada, 1989]. Applying a modeling paradigm means adhering to certain rules in decision modeling i.e. adopting a definite way of thinking about and describing models. This suggests that based on the modeling paradigm a visual language for modeling can be formed. Thus, visualization for modeling can be regarded as the translation from a kind of modeling paradigm into a visual language.

Structured modeling(SM), proposed in [Geoffrion, 1987], is a way of thinking about analytic models and the system which supports them. It can be thought of as a style of modeling as well as a formal framework[Lenard, 1988]. There are three levels of model abstractions in SM: elemental structure, generic structure and modular structure[Geoffrion, 1987]. With these three level structures, the four levels of model abstractions identified in [Geoffrion, 1989] can be covered as below:

Abstraction Level	Structured Modeling Counterpart
Specific model	Elemental detail tables (together with a schema)
Model class	Schema
Modeling paradigm	Definitional system
Modeling tradition	MS/OR (by genesis)

As SM is a systematic approach for modeling and has structured rules to construct a decision model, it is an obvious choice as the modeling paradigm to be visualized. Based on the discussion of visualization of SM, in this paper we propose a presentation for visual models: SOOM and two levels of visual modeling: generic visual modeling (GVM) and specific visual modeling (SVM), focusing on the formulation of problem domain oriented generic model and specific model through these two kinds of modeling approach. The subsequent sections are organized as follows.

Section 2 discusses the model representation for visual modeling, which is the foundation of visualization and visual modeling. Section 3 relates this representation to the visualization of structured modeling. In section 4 and section 5 two kinds of visual modeling GVM and SVM are proposed and discussed. Meanwhile an example of formulation of visual model for production and transportation system is described. In section 6 we present a feasible and effective schema of visual modeling from meta model to specific model based on the foregoing discussion, and section 7 has our concluding remarks.

2 Representation of Visual Models

Visual modeling is a user-oriented approach for creating and manipulating visual models. It refers to an interactive process during which an user makes use of visual units pre-designed in the modeling system under certain rules to construct and manipulate models.

A visual model is a system of graphical representations that is used in visual modeling. The graphical representations are those parts of visual units representing model components. Visual units may include icons, menu items, dialogue-windows, lines, boxes, circles and text strings that either represent model components or instructions to operate on the model components under certain rules. So the representation of *a visual model is the composition of a set of visual units and the relations between them.* With this consideration, we propose a general model framework of a visual model, SOOM, the structured and object-oriented model. In SOOM, *all the components comprising the model are of object-oriented objects and the relations between components are structured. The components and their relations are all represented by corresponding visual units.* A formal expression for this model is as follows.

$$M = \{e_1, e_2, \ldots e_N\}; \qquad (1)$$
$$D(e_i) = \text{Class}\ (\text{id}[e_i], \text{md}[e_i], \text{attr}[e_i]) \quad i=1,2, \ldots N; \qquad (2)$$
$$R(e_i, e_j) \quad i,j=1,2, \ldots N \text{ and } j\neq i; \quad e_i, e_j \in M; \qquad (3)$$

Where M: visual model; e_1, e_2, . . . e_N: model components and also visual units; $D(e_i)$: object-oriented definition of model components (In our object description, there will be identification, methods and attributes); $R(e_i, e_j)$: direct relations between the components. The components and their direct relations can best be expressed as an acyclic graph.

There are three levels of model abstraction in SOOM. They are the *meta model*, the *domain model* and the *model instance* or *specific model*. A *meta model* is a generic model based on a certain modeling paradigm with its components expressed in object classes. It is the model for defining the other two models. A *domain model* is a schematic representation of a model after it has been specialized to a particular application and/or domain[Lazimy, 1993] through GVM. It inherits from a meta model and includes domain/application-dependent features and elements. A *model instance* is the instantiation of domain model through SVM. It is a specific model that inherits from a domain model.

In SM an elemental structure is used for defining model instance with either acyclic graph or SML. There are five types of elements in the elemental structure level: *primitive entity; compound entity; attribute (variable attribute); function* and *test*. Here we view these five types of elements as the five fundamental 'blocks' to build a meta model which can be used to generate domain oriented generic model. The dependencies among the five types of elements actually determine certain structured relations among the model components that are really needed in SOOM.

Using the object-oriented concept, a meta model or a model for generating domain model based on the structured modeling framework can be given in Figure 1 or expressed in the textual formulae of (1) to (3). The five basic elements which are used for constructing models can be viewed as five meta classes/objects which have their own O-O definitions. Figure 1 also shows the fixed dependencies among the meta classes/objects in the formal expression as well as in the acyclic graph. The arrows show the direction of the existence dependencies. For example, an object ATTR can be defined only after, at least, one object PE or CE has been defined.

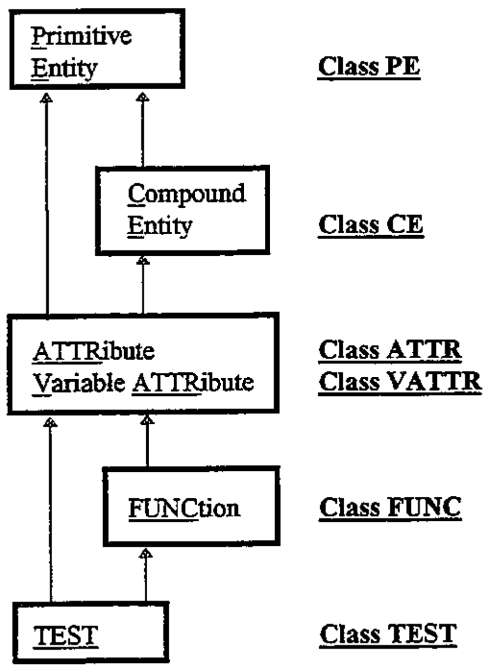


Figure 1. The meta model in SOOM

M={PE, CE, ATTR/VATTR, FUNC, TEST}; (1)-1
D(PE)=Class (id[PE], ,); (2)-1
D(CE)=Class (id[CE], md[CE], attr[PE, CE]); (2)-2
D(ATTR/VATTR)=Class(id[ATTR/VATTR], md[ATTR/VATTR], attr[PE,CE,ATTR/VATTR]); (2)-3
D(FUNC)=Class(id[FUNC],md[FUNC], attr[ATTR/VATTR,FUNC]); (2)-4
D(TEST)=Class(id[TEST],md[TEST], attr[ATTR/VATTR, FUNC, TEST]); (2)-5

R(PE,CE) = CE → PE; (3)-1
R(PE,ATTR/VATTR) = ATTR/VATTR → PE; (3)-2
R(CE,ATTR/VATTR) = ATTR/VATTR → CE; (3)-3
R(ATTR/VATTR,FUNC) = FUNC → ATTR/VATTR; (3)-4
R(ATTR/VATTR,TEST) = TEST → ATTR/VATTR; (3)-5
R(FUNC,TEST) = TEST → FUNC; (3)-6

For the domain model, we use a Production and Transportation System model (PTS) as an example. The PTS model described in SML[Geoffrion, 1987] is shown as below:

&SDATA SOURCE DATA
PLANTi /pe/
There is a list of PLANTS.
SUP(PLANTi) /a/ {PLANT} : R+
Every PLANT has a SUPPLY CAPACITY (tons).
&CDATA CUSTOMER DATA
CUSTj /pe/
There is a list of CUSTOMERS.
DEM(CUSTj) /a/ {CUST} : R+
Every CUSTOMER has a non-negative DEMAND (tons).
&TDATA TRANSPORTATION DATA
LINK(PLANTi, CUSTj) /ce/ Select {PLANT}x{CUST} where i covers {PLANT}, j covers {CUST}
There are some transportation LINKS from PLANTS to
CUSTOMERS. There is at least one LINK incident to each PLANT, and at least one LINK incident to each CUSTOMER.
FLOW(LINKij) /va/ {LINK} : R+
There can be a nonnegative transportation FLOW(tons) over each LINK.
COST(LINKij) /a/ {LINK} : R
Every LINK has a TRANSPORTATION COST RATE ($/ton).
$(COST,FLOW) /f/ SUMi SUMj (COSTij * FLOWij)
There is a TOTAL COST associated with all FLOWS.
T:SUP(FLOWi.,SUPi) /t/ {PLANT};
SUMj(FLOWij)<=SUPi
Is the total FLOW leaving a PLANT less than or equal to its SUPPLY CAPACITY? SUPPLY TEST.
T:DEM(FLOW.j, DEMj) /t/ {CUST};
SUMi(FLOWij) = DEMj
Is the total FLOW arriving at a CUSTOMER exactly equal to its DEMAND? DEMAND TEST.

Schema for Production and Transportation Model

With SOOM, the above model can be expressed as a domain model by using a set of visual units. The representation shown in Figure 2 is an acyclic graph which is similar to the genus graph of SM[Geoffrion, 1987]. But here every visual component of model which is inherited from a meta object also has an object representation. From Figure 2, by direct manipulation[Lee, 1993] of the domain model a specific model of PTS shown in Figure 3 can be derived interactively through inheritance and instantiation of the domain model.

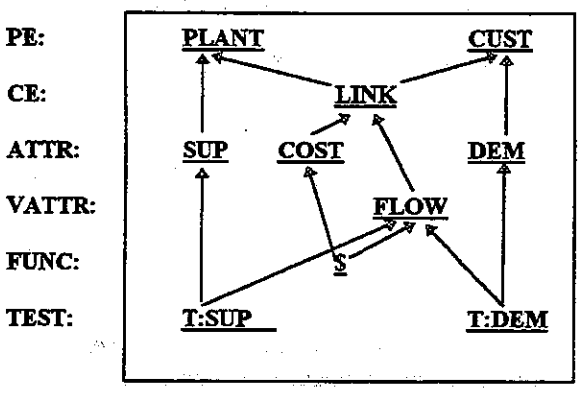


Figure 2. A domain model of PTS

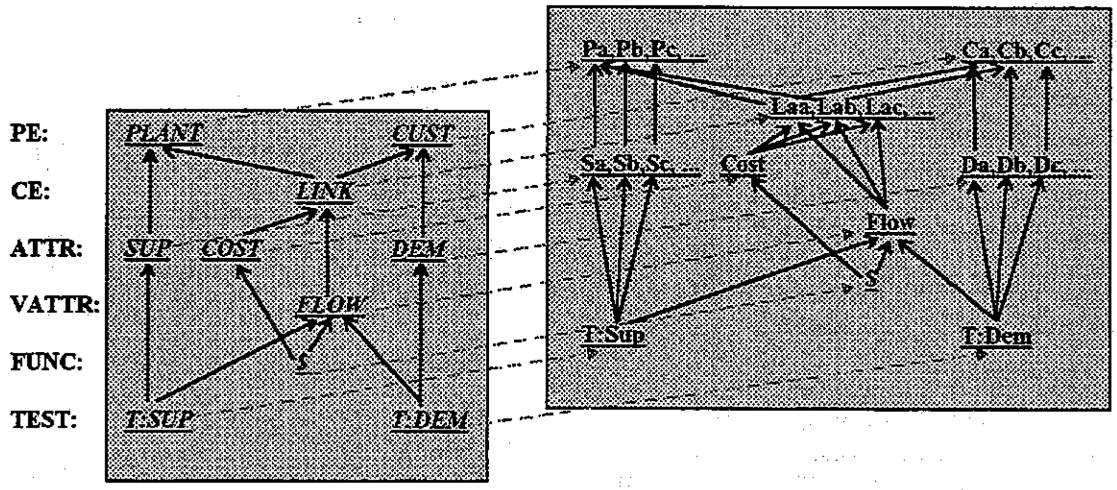


Figure 3. The domain model & specific model of PTS

3 Visualizing structured modeling

Structured modeling is usually implemented by using structured modeling language (SML). Visualizing structured modeling to some extent is to visualize SML. A general visualization process described in [Kamada, 1989] is translating the original textual representation into visual structure representation through a semantic structure representation. A visual structure expresses the relationships among the constituents in a visual representation, and is represented by graphical objects. The semantic structure represented by abstract objects and relations can be derived from the textual representation. From SML, we can abstract a semantic structure expressed in a formal form which is also the context-free grammer or the deep-structure of SML:

$$S = \{E, A\ [, D, V, R]\} \qquad (4)$$

In (4) *S* is a sentence of SML; *E* is an objective element of the sentence; ***A:*** are descriptive attributes for ***E***, which are PE, CE, ATTR/VATTR, FUNC and TEST. A particular ***A*** determines the existence of ***D,V*** and ***R***; ***D*** is dependency between two ***Es; V*** are values given to ***E; R*** is a rule or mathematic formula that defines ***E***.

This rigorous and coherent conceptual framework of SML is for both semantic and mathematic use. Based on the semantic structure and the models presented in SOOM, we can realize the visualization of structured modeling in the following steps:

- make all the *As* as pre-designed visual units (icons or menu-items);
- write a procedure to determine how to provide *As* to users based on *D;*
- establish dialogue-windows for definition of *Es* when they are required;
- analyse the types of *V* and the rules or mathematic formulae of *R*, induce all the possible types and rules which will be used for defining models;
- decompose the rules or mathematic formulae into elementary functions and operators which are often used and can be composed to various rules;
- generate auxiliary visual units based on the types, operators and functions which are graphical representations of them;
- beginning with *As*, provide all the visual units to users according to the semantic structure in (4) during a modeling process.

4 Generic visual modeling

Generic visual modeling or GVM, is domain independent visual modeling. It is used for formulating and manipulating a diversity of domain models. The visual units used for GVM which are based upon a modeling paradigm are general tools that allow users to do modeling-in-the-large[Geoffrion, 1994]. With these domain independent visual units, users can create and manipulate domain oriented generic models. The visual units for GVM, except for the auxiliary visual units mentioned in section 3 which are available for both GVM and SVM to define *attributes, functions* and *tests*, are called primary visual units or generic visual units. All the components of a meta model, or meta objects, are primary visual units.

With these primary visual units the components of domain models can be generated through inheritance from the meta objects. First, modeling system uses the five meta objects to enter the model definition. Every meta object is represented by a visual icon. When a model construction is required an interactive process will be started. By operating the menus or icons and keying in text in dialogue windows provided by the system, users can define all the components of domain models step by step under the guidance of the system.

In GVM, the components defined are model genera and are also visual object classes. Through the definition of object classes (model genera), a domain oriented generic model can be derived. At the beginning *primitive genera* of a model are defined from the meta object class PE and have the properties of PE as in Figure 4:

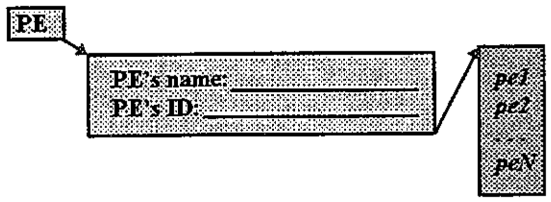


Figure 4. Defining PEs

By following this operation the **PLANT** and **CUST** of PTS domain model can be defined. When the definition of a *primitive genus* is finished a visual unit for the genus is generated. It is both a component of the model and an icon

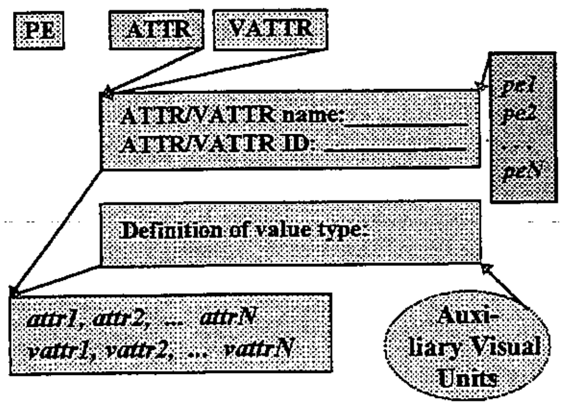


Figure 5. Defining ATTR/VATTR of PEs

for operating on the model. Based on the icons generated from PE the *attributes* (or *variable attributes*) of the *primitive genera* can be specified in Figure 5.

For example, based on the definition of **PLANT** and **CUST**, the **SUP** and **DEM** attributes can be defined. The dependency relationship between **PLANT** and **SUP** , and between **CUST** and **DEM** inherit from the meta model.

Similarly *compound genera* and their *attributes* (or *variable attributes*) can be defined respectively as in Figure 6 and Figure 7. In the PTS model the **LINK** and its *attributes* **COST** and **FLOW** are defined at this stage.

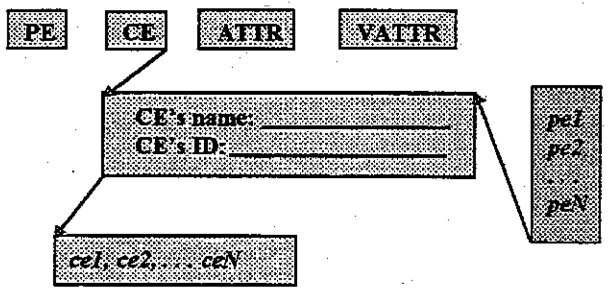


Figure 6. Defining CEs

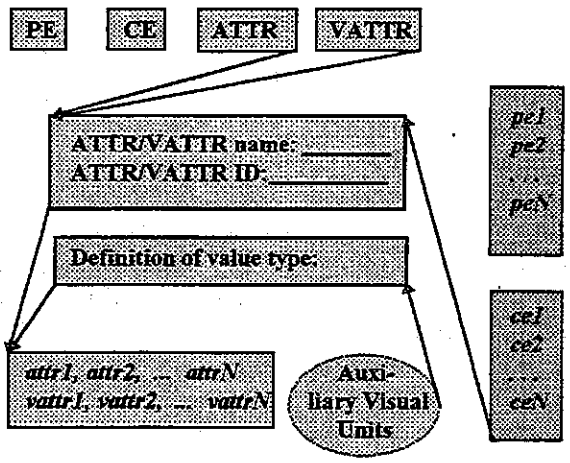


Figure 7. Defining ATTR/VATTR of Ces

After *attributes* (*variable attributes*) have been defined definitions of *function genera* and *test genera* can be carried out as in Figure 8 and Figure 9. When all the definitions are finished a domain model shown in Figure 2 is obtained. At this stage, only a general model frame is formed. The components of the model will have values only during model instantiation.

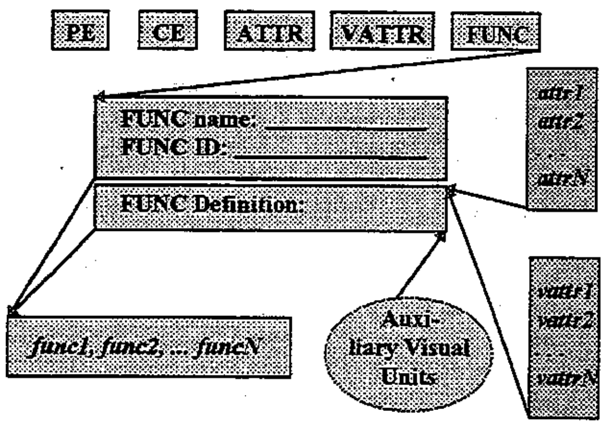


Figure 8. Defining FUNCs

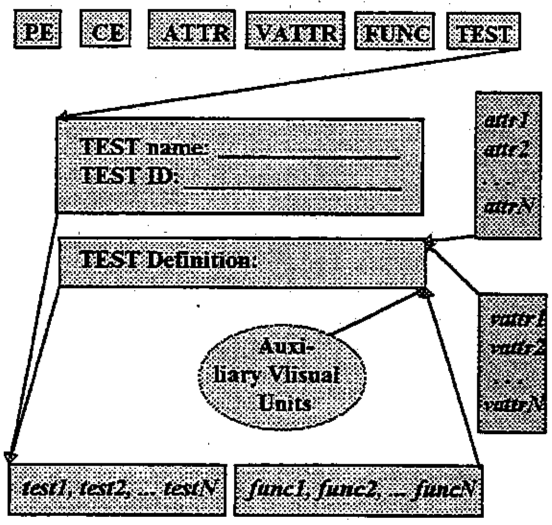
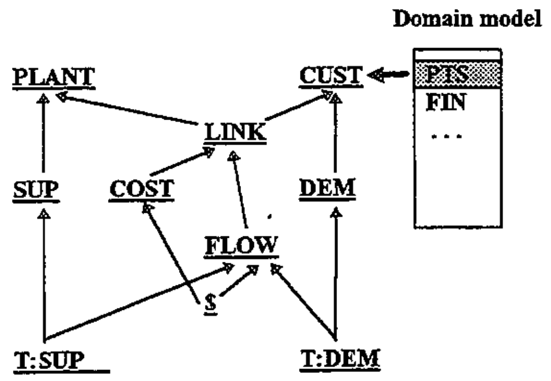


Figure 9. Defining TESTs

5 Specific visual modeling

Specific visual modeling SVM, which is derived from GVM through instantiating a domain model, is domain dependent visual modeling. It is targeted at the end-users categorized in [Bharadwaj, etc., 1992]. The visual units used for SVM are derived from GVM. With these visual units, users directly manipulate a domain model to form a specific model. The visual units for SVM except for the auxiliary visual units mentioned in section 3 are called secondary visual units or domain oriented visual units.

When doing SVM a domain model should be chosen first. That will make all the required visual units (including secondary visual units) available for generating specific models. For example, the PTS model is selected in Figure 10. Then, by direct manipulating and instantiating the object/classes (components) of the domain model, a model instance shown in Figure 3 is obtained.



PLANT	⇒ Pa, Pb, Pc, . . .
CUST	⇒ Ca, Cb, Cc, . . .
LINK	⇒ Laa, Lab, Lac, . . . , Lba, Lbb, Lbc, . . . , Lca, Lcb, Lcc, . . .
SUP	⇒ Sa, Sb, Sc, . . .
DEM	⇒ Da, Db, Dc, . . .
COST	⇒ Cost
FLOW	⇒ Flow
$	⇒ $
T:SUP	⇒ T:Sup
T:DEM	⇒ T:Dem

Figure 10. Instantiating the PTS Model

6 A general visualization schema for modeling

Visual modeling is an effective approach for the three parties of user MS/OR modelers, domain experts and end-users[Bharadwaj, etc., 1992]. To different modeling users there should be a different visual modeling for them so that they can easily learn and use the system. For this purpose and taking into account of the previous discussion we present a general visualization schema and a hierarchy of visual models shown in Figure 11 and Figure 12 seperately for all the three kinds of users.

A visual modeling system starting from meta model with the aid of GVM is a *general visual modeling system*. Equipped with a specific domain knowledge, a user who uses the system to derive a specific visual modeling system is both a modeler and a domain expert. A visual modeling system starting from domain models with the aid of SVM is a *domain oriented visual modeling system*. It is used by end-user to generate specific model or model instance.

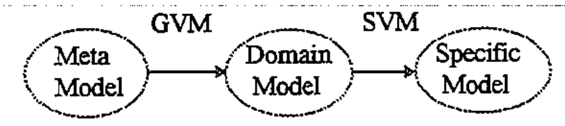


Figure 11. A general visualization schema of modeling

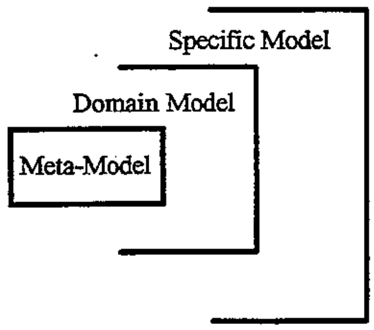


Figure 12. A hierarchy of visual models

7 Conclusion

By referencing structured modeling language SML and making use of object-oriented concept, we can use GVM and SVM to generate domain models and specific models from a meta model intuitively in a step-wise manner. Visualization of SM actually is to form a visual language that is used to facilitate the four phases of modeling described in [Bharadwaj, etc., 1992]. This kind of language is not only a model definition language but also a model manipulation language. It is much easier for users to learn and use as compared to those textual languages. As the entire process of modeling and the generated models are visible throughout the modeling, it will give users much more confidence about using the models to make decision. Also it is much easier for users to reuse and refine models.

With the general modeling schema and the hierarchy of visual model the three groups of user can obtain the corresponding modeling system that fits them. All of these systems are only established on a kind of modeling paradigm or a meta model. We have demostrated here for SM. A similar modeling of visual modeling systems can be obtained with a different modeling paradigm.

References

Angehrn, A. A. and Lüthi, H. J. "Intelligent Decision Support Systems: A Visual Interactive Approach," *Interfaces* 20:6 Nov.-Dec. 1990 (pp. 17-28)

Belton, V. and Elder, M. D. "Decision support systems: Learning from visual interfactive modelling," *Decision Support System* 12 (1994) 355-364, North-Holland.

Bharadwaj, A., Choobineh, J., Lo, A. and Shetty, B. "Model Management Systems: A Survey," *Annals of Operation Research* 38 (1992) 17-67.

Blanning, R. W. "Model Management System An Overview," *Decision Support Systems* 9 (1993) 9-18, North-Holland.

Dolk, D. R. "Model Management and Structured Modeling: The Role of an Information Resource Dictionary System," *Communications of ACM*, Vol. 3, No. 6, June 1988

Geoffrion, A. M. "An Introduction to structured Modeling," *Management Science* Vol. 33, No. 5, May 1987.

Geoffrion, A. M. "Computer-based Modeling Environments" *European Journal of Operational Research* 41 (1989) 33-43, North-Holland.

Geoffrion, A. M. "Integrated modeling system," *Computer Science Economics and Management* 2 (1989) 3-15.

Geoffrion, A. M. "Reusing Structured Models via Model Integration," In B. W. Blanning and King D.R. (Ed.), *Current Research in Decision Suport Technology* (pp.25-55). IEEE. 1992.

Geoffrion, A. M. "Structured Modeling: Survey and Future Research Directions," *ORSA CSTS Newsletter* Vol. 15, No. 1, Spring 1994.

Kamada, T. *Visualizing Abstract Objects and Relations--A Constraint-Based Approach,* World Scientific Series in Computer Science, Vol. 5, 1989

Kendrick, D. A. "Parallel Model Representations," *Expert System with Applications,* Vol. 1, pp. 383-389, 1990.

Kendrick, D. A. "A Graphical Interface for Production and Transportation System Modeling: PTS," *Computer Science in Economics and Management,* 4, 229-236., 1991.

Lazimy, R. "Object-Oriented Modeling Support System: Model Representation, and Incremental Modeling," *Hawaii:* IEEE, 445-459, 1993.

Lee, R. M. "Direct Manipulation of Graph-based Decision Models," *Decision Support System* 9 (1993) 393-411, North-Holland.

Lenard, M. L. "Fundamentals of Structured Modeling," In G. Mitra (Ed.), *Mathematical Models for Decision Support* (pp. 695-713), Springer-Verlag, 1988.

Lenard, M. L. "Structured Model Management," In G. Mitra (Ed.), *Mathematical Models for Decision Support* (pp. 375-391). Springer-Verlag, 1988.

Rathnam, S. and Mannio, M. V. "Tools for Building the Human-computer Interface of a Decision Support System," *Decision Support System* 13 (1995) 35-59, North-Holland.

Ting, P. L. "A Graph-based Approach to Model Management," *Proceedings of International Conference on Information Systems,* 136-151, 1986.

Turban, E. and Carlson, J. G. *Interactive Visual Decision Making. Decision Support Systems Putting Theory into Practice,* Second Edition, page 170-182, Ralph H. Sprague, JR. & Hugh J. Watson Editors.