지식 표현 기법을 이용한 모델 구조의 표현과 구성: 단편구조 유연생산 시스템 예

Model Structuring Technique by A Knowledge Representation Scheme: A FMS Fractal Architecture Example

조 대 호*
Cho, Tae Ho

Abstract

The model of a FMS (Flexible Manufacturing System) admits to a natural hierarchical decomposition of highly decoupled units with similar structure and control. The FMS fractal architecture model represents a hierarchical structure built from elements of a single basic design. A SES (System Entity Structure) is a structural knowledge representation scheme that contains knowledge of decomposition, taxonomy, and coupling relationships of a system necessary to direct model synthesis. A substructure of a SES is extracted for use as the skeleton for a model. This substructure is called pruned SES and the extraction operation of a pruned SES from a SES is called *pruning* (or *pruning operation*). This paper presents a pruning operation called *recursive pruning*. It is applied to SES for generating a model structure whose sub-structure contains copies if itself as in FMS fractal architecture. Another pruning operation called *delay pruning* is also presented. Combined with recursive pruning the delay pruning is a useful tool for representing and constructing complex systems.

1. Introduction

Computer simulation is one of the most widely used techniques in manufacturing systems study. The value of simulation increases constantly due to improvements in computing power. However models of large-scale systems tend to be very complex, and writing simulation programs to execute them can be an arduous task [1]. FMS is a manufacturing system formed by tieing flexible manufacturing cells, or workcells. A workcell is an independent unit which consists of group of machine tools and associated material handling equipment that is managed by an intelligent

* 경남대학교 전자계산학과

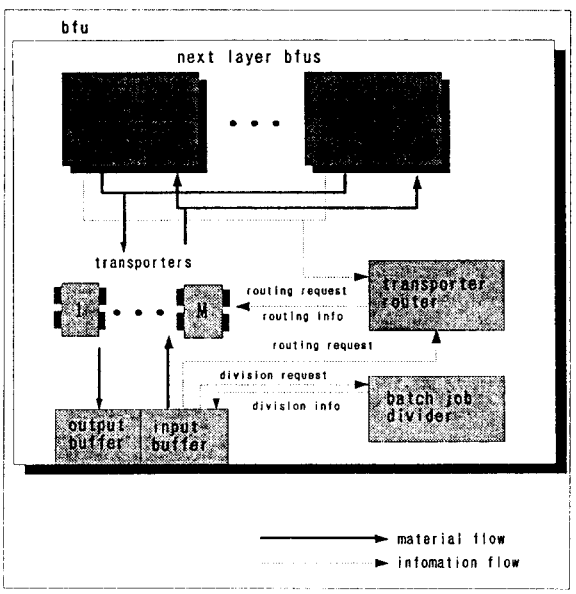
supervisory computer. Typically, it consists of a hierarchy of several workcells, each containing one or more transporters, sub-cells, etc. The term flexibility in FMS implies the ability of the system to process a wide variety of parts or assemblies without outside intervention. Rapid modeling of such a system can play a significant role in the selection of a manufacturing strategy [2, 3, 4, 5].

Cast in a fractal architecture, the model of a FMS admits to a natural hierarchical decomposition of highly decoupled units with similar structure and control. As a consequence, this model manages the structural complexity and coordination of an FMS hierarchy by maximizing local functionality and minimizing global control. The FMS fractal architecture model represents a hierarchical structure built from elements of a single basic design called basic fractal unit (BFU). The design of BFU incorporates a set of pertinent attributes that can fully represent any level in the hierarchy [4].

〈Figure 1〉 gives a example of the essential structure of the basic unit used to construct a fractal architecture representation of the fractal model. This so-called BFU is specifically designed to embody the elements which fully describe the structure of any level in the model hierarchy. Included within in a BFU is a set of next layer BFUs whose internal detail is essentially hidden. The routing controller (or transporter router) sees these units as stations to which the transfer batches should be delivered. Its upto transporter routers within the next layer BFUs for controlling (or routing) the batches delivered to them.

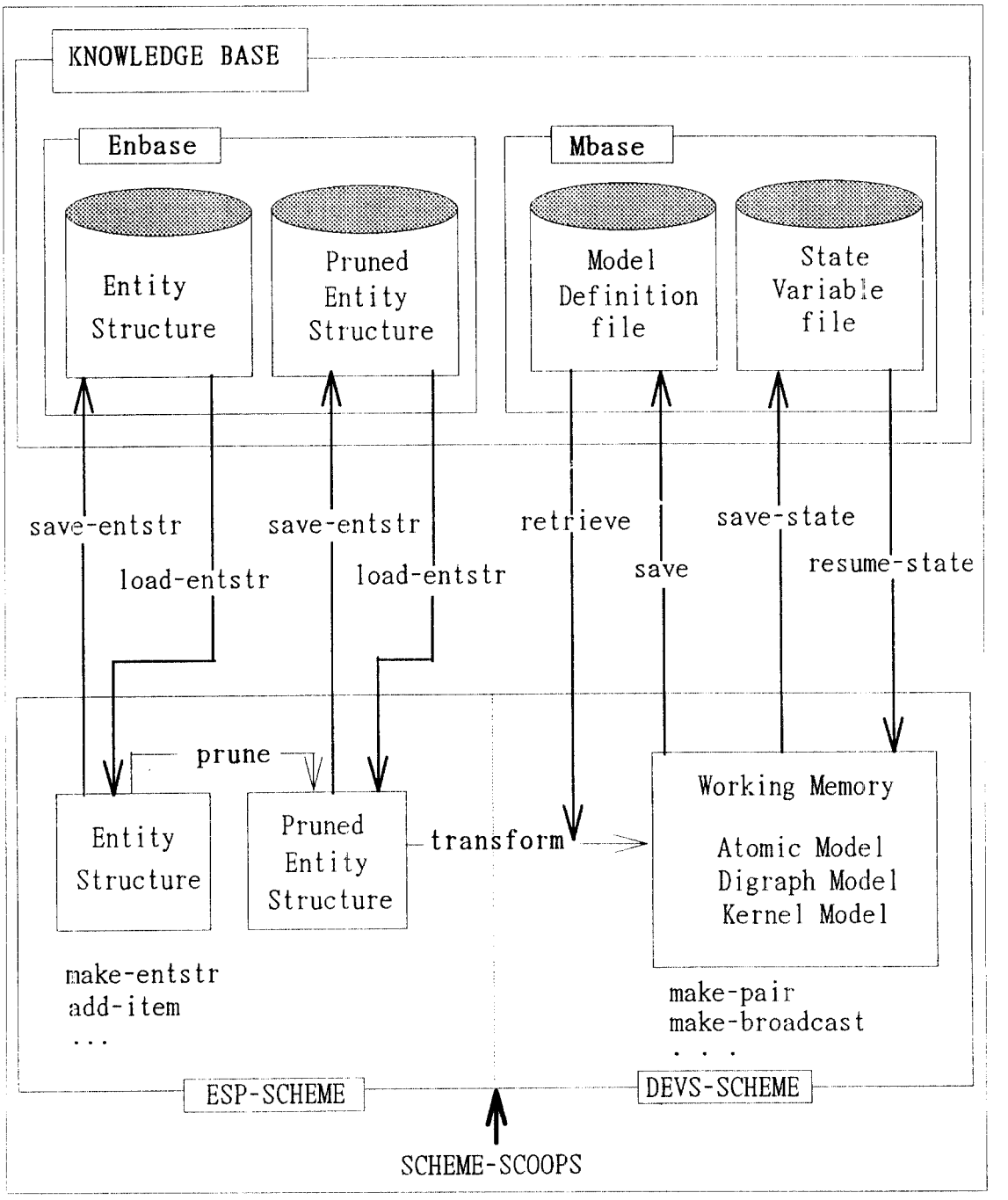
DEVS-Scheme is a modular hierarchical discrete event simulation environment implemented in object oriented Scheme language which runs on unix and DOS compatible computers. The environment realizes the DEVS formalism [6, 7, 8, 9] a theoretically well-grounded means of expressing hierarchical, modular discrete event simulation models. DEVS-Scheme is implemented as a shell that sits upon Scheme language in such a way that all the underlying Lisp-based and object oriented programming language features are available to the users. The result is a powerful basis for combining AI and simulation techniques.

DEVS-Scheme allows a modeller to keep models in an organized library in a modular form, enabling hierarchical assembly and disassembly as required in investigating design alternatives. The environment is based on two formalisms: discrete event-system specification (DEVS) and system entity structure (SES) formalism [6, 7, 10]. The environment allows the modeller to specify explicitly the structure of a simulation model using SES formalism and its behavior using DEVS formalism. Structural and behavioral specifications of a model can be saved in a structural knowledge base called entity structure base (ENBASE) and behavioral knowledge base called model base (MBASE), respectively (refer to Figure 2). DEVS-Scheme can be used to develop DEVS models and save them in a model base for later use. To organize such models into a model base, a model base management system is highly desirable. The SES (system



〈Figure 1〉 Basic fractal unit architecture.

entity structure) formalism is one such tool for model base management. ESP-scheme is a realization of the SES formalism developed by Zeigler in a LISP-Based, object-oriented programming environment. The ESP-Scheme supports specification of the structure of a model, pruning the structure to the reduced one, and transforming the structure



〈Figure 2〉 DEVS-Scheme simulation environment

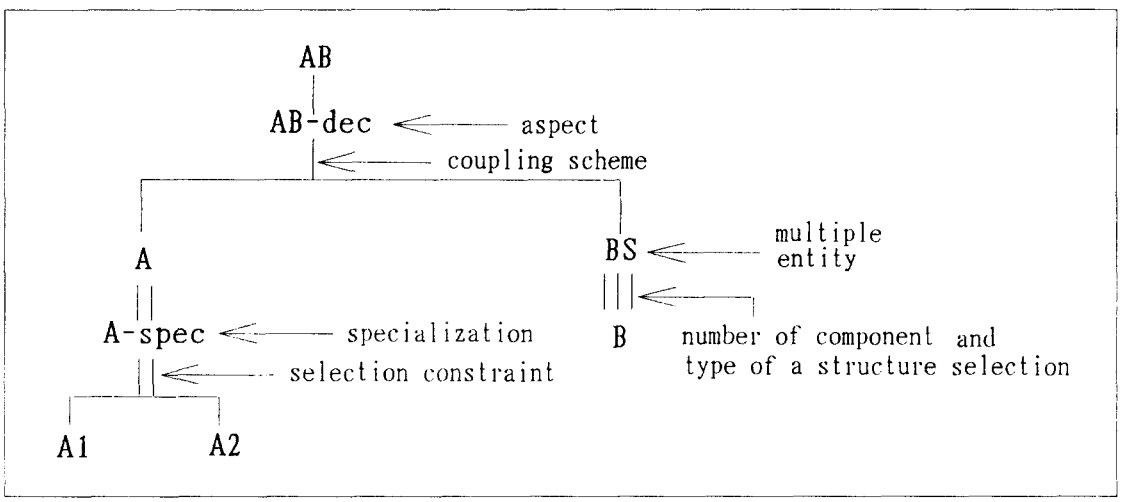
to a simulation model by synthesizing components models in the model base developed by using DEVS-Scheme [11, 12].

A substructure of a SES is extracted for use as the skeleton for a model. This substructure is called pruned SES and the extraction operation of a pruned SES from a SES is called pruning (or pruning operation).

In this paper, we present a new operation called *recursive pruning* and its application to the FMS fractal architecture model structuring. Recursive pruning is a procedure applied to SES for generating a model structure whose sub-structure contains copies if itself as in FMS fractal architecture. For the background knowledge SES formalism is presented first in the following section.

2. SES Formalism

A SES is a structural knowledge representation scheme that contains knowledge of decomposition, taxonomy, and coupling relationships of a system necessary to direct model synthesis. There are three types of nodes in the SES - *entity, aspect,* and *specialization* - which represent three types of knowledge about the structure of systems. The entity node, 〈Figure 3〉 represents one decomposition of an entity. Thus the children of an aspect node are entities, distinct components of the decomposition. The specialization node (within a double vertical line in the labeled tree of 〈Figure 3〉 represents way in which a general entity can be categorized into special entities. A *multiple entity* represents the set of all members of an entity class and it is a special entity that consists of a collection of homogeneous components. Such components are a *multiple decomposition* of the multiple entity. The aspect of such a *multiple* entity is called a multiple aspect (triple vertical lines in the labeled tree of Figure 3). A substructure of a SES is extracted for use as the skeleton for a model. This substructure is called pruned entity structure (PES) and the extraction operation of a PES from the SES is called *pruning (or pruning operation)*. There are only aspect type nodes and entity type nodes in a PES. All the multiple decomposition nodes and specialization nodes in SES are traversed and processed in pruning process. The process done at the specialization node is to select a special entity (either A1 or A2) and the process done at the multiple aspect is to



〈Figure 3〉 System entity structure (SES)

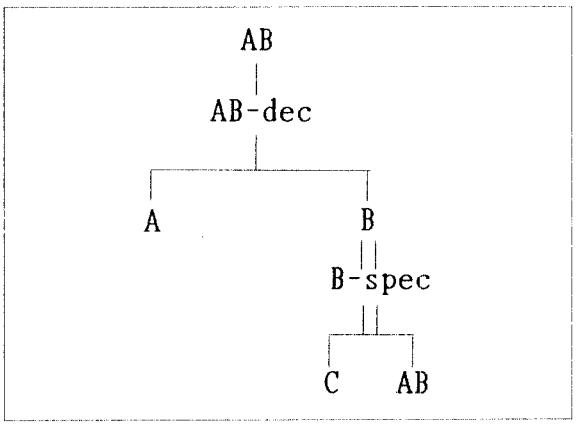
having several aspects and/or specializations, corresponds to a model component that represents a real world object. The aspect node (a single vertical line in the labeled tree of decide the number of multiple decomposition (number of components in BS) and type of kernel models (broadcast models, controlled-models, etc.) that this multiple decompo-

sition forms. For more on DEVS formalism and SES formalism refer to [6, 7, 13].

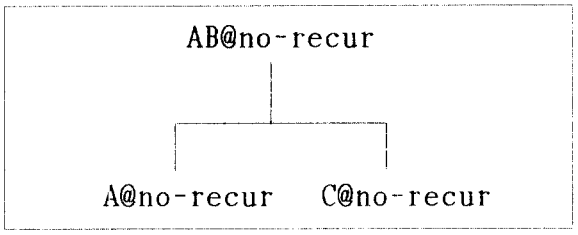
3. Recursive Pruning and Delay Pruning

Zeigler [6, 7] introduced the *system entity structure* (SES) which directs the synthesis of models from components in a knowledge base. The SES is a knowledge representation scheme that combines the decomposition, taxonomy, and coupling relationships. Recursive pruning is a procedure applied to the SES for generating a model structure whose sub-structure contains copies of itself.

〈Figure 4〉 shows a SES which specifies the recursive structure of model AB. AB consists of A and B, where B is specialized into C or AB. Here AB is the root model itself, which represents the repetition of same structure in the SES.

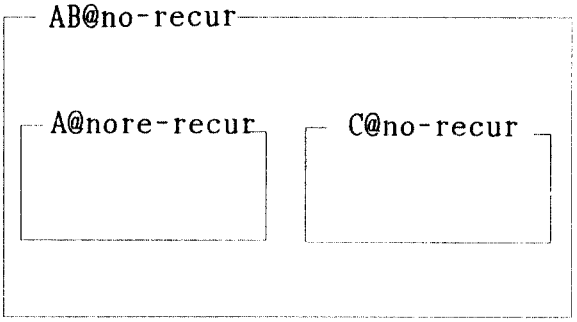


〈Figure 4〉 System entity structure for recursive pruning.



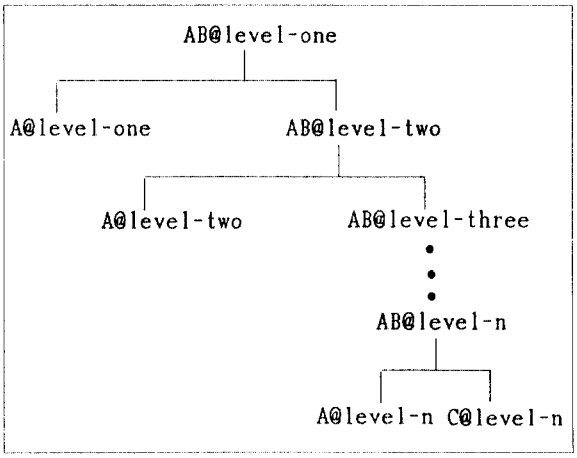
〈Figure 5〉 Pruned SES (PES) without recursive pruning.

〈Figure 5〉 represents the PES when C is selected at B-spec node of the SES in pruning process. Its corresponding model structure is shown in 〈Figure 6〉. In this case recursive pruning is not invoked.



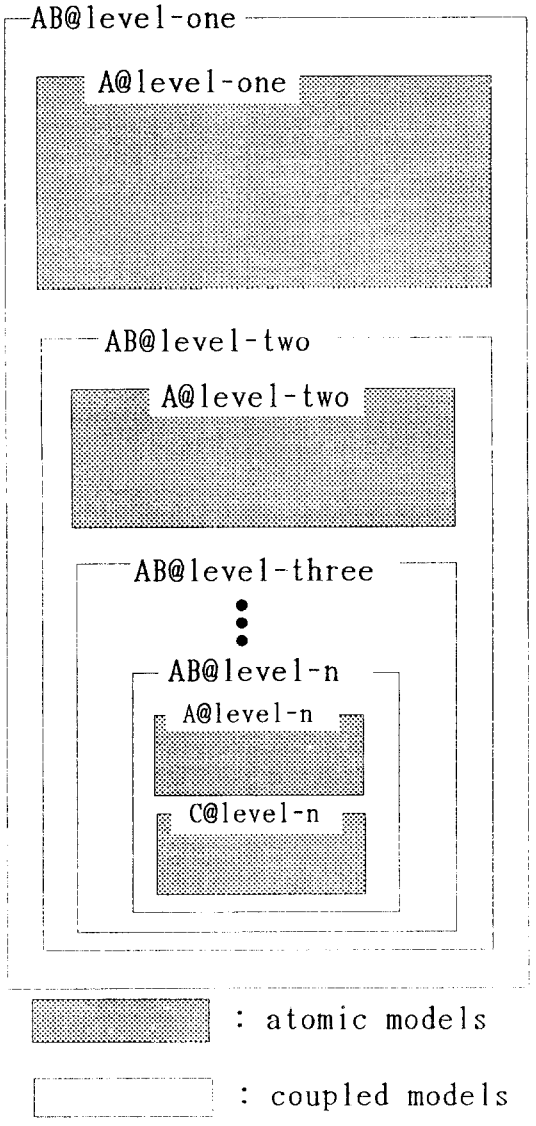
〈Figure 6〉 Model AB pruned to be a non recursive structure.

Whereas, if AB is selected at B-spec node during pruning process recursive pruning is invoked. It is invoked as long as AB is selected at B-spec node. 〈Figure 7〉 represents the PES with n layers of recursion. The figure also shows the extensions, level-one, level-two, ..., of model names. These extensions are prompted each time B-spec node is encountered during pruning process so that models in different levels can be distinguished. The recursion of AB stops when type C is selected at B-spec node during pruning process. 〈Figure 8〉



〈Figure 7〉 Pruned SES (PES) with recursive pruning.

shows the recursive model represented by the PES in 〈Figure 7〉.



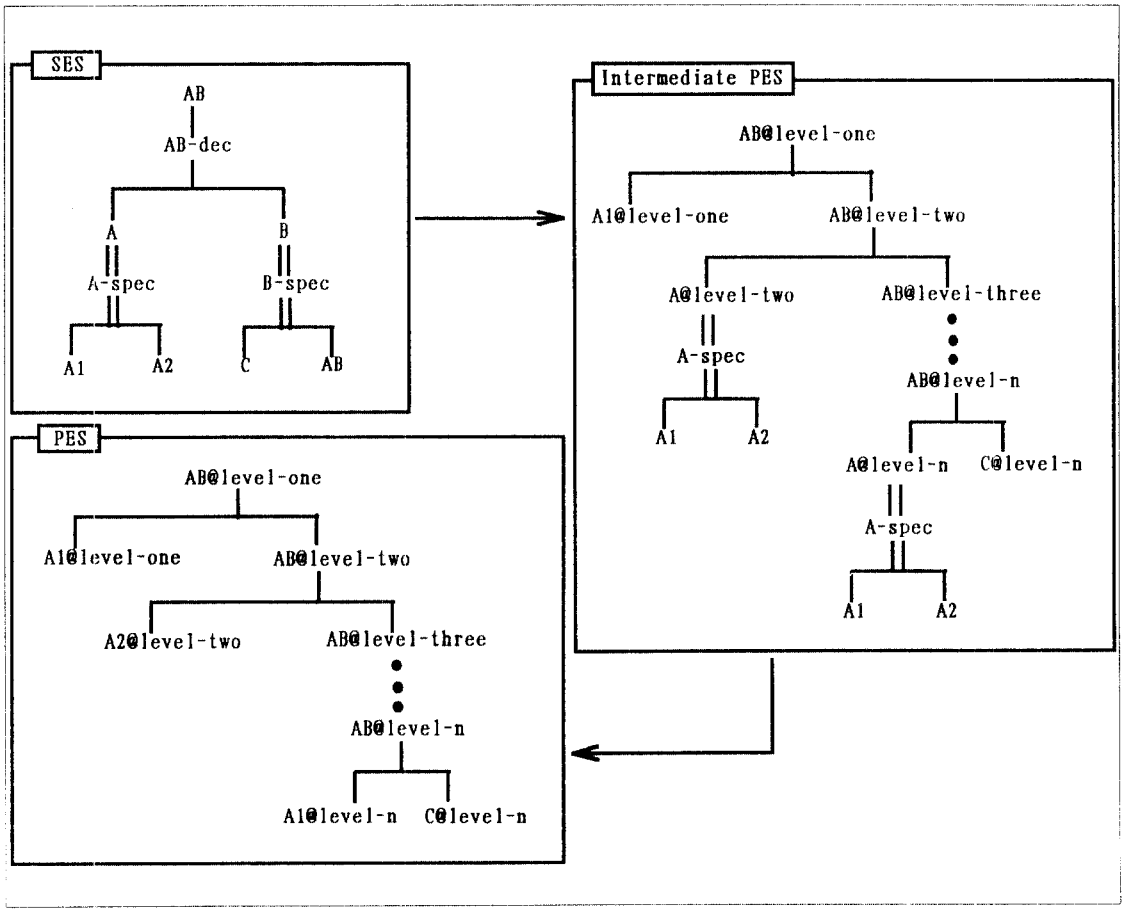
〈Figure 8〉 Model AB pruned to have a recursive structure.

When there are several recursions in pruning process the process gets complicated. We can delay some of the pruning decisions for a later pruning process and perform pruning only on part of the SES. This type of pruning process is called *delay pruning*. 〈Figure 9〉 shows delay pruning process. In the first pruning process only the number of recursions of AB and type A at A-spec node of level-one are decided. Rest of type selections at A-spec node other than level-one are left undecided and delayed for later pruning. The intermediate PES generated by delay pruning is shown at the right side of the figure. It is further pruned to become the final PES (left lower side of the figure) where every nodes are pruned and the PES is ready for transformation, i.e. ready for constructing a model for simulation. Delay pruning is also useful in constructing several similar model architecture where only small differences in pruning are needed, compared to the first pruning. For example, construct an intermediate PES for a model by delay pruning (and recursive pruning) and perform further prunings on this PES for generating those similar model architectures. Recursive pruning, combined with delay pruning, is a useful tool for representing and pruning a complex system such as the one described in next section.

4. FMS Fractal Architecture Structuring

An example FMS fractal architecture with three levels of recursions is shown in 〈Figure 10〉 (detailed function of each model is described in [14]). This architecture is one of many possible structures that can be generated by recursive pruning applied to the SES shown in 〈Figure 11〉. The top of the hierarchy is a factory. The factory is composed of two shop floors. The first shop floor (shopfloor1) in turn has two workstations, workstation1 and workstation2. Whereas, the second shop floor has two machines without having any workstation. Workstation1 has two machines and workstation2 has just one machine. Architecture of all BFUs are same except the number of transporters, if the next layer BFUs are looked at as black boxes.

The transporter router is a model with expert system capability which controls routing of transporters in each BFU. It routes transporters among next layer BFUs, *output buffer* and *input buffer*. There are total of five expert system models (one for each BFU) inferencing based on their own set of facts, i.e., based on different dynamic conditions of each BFU



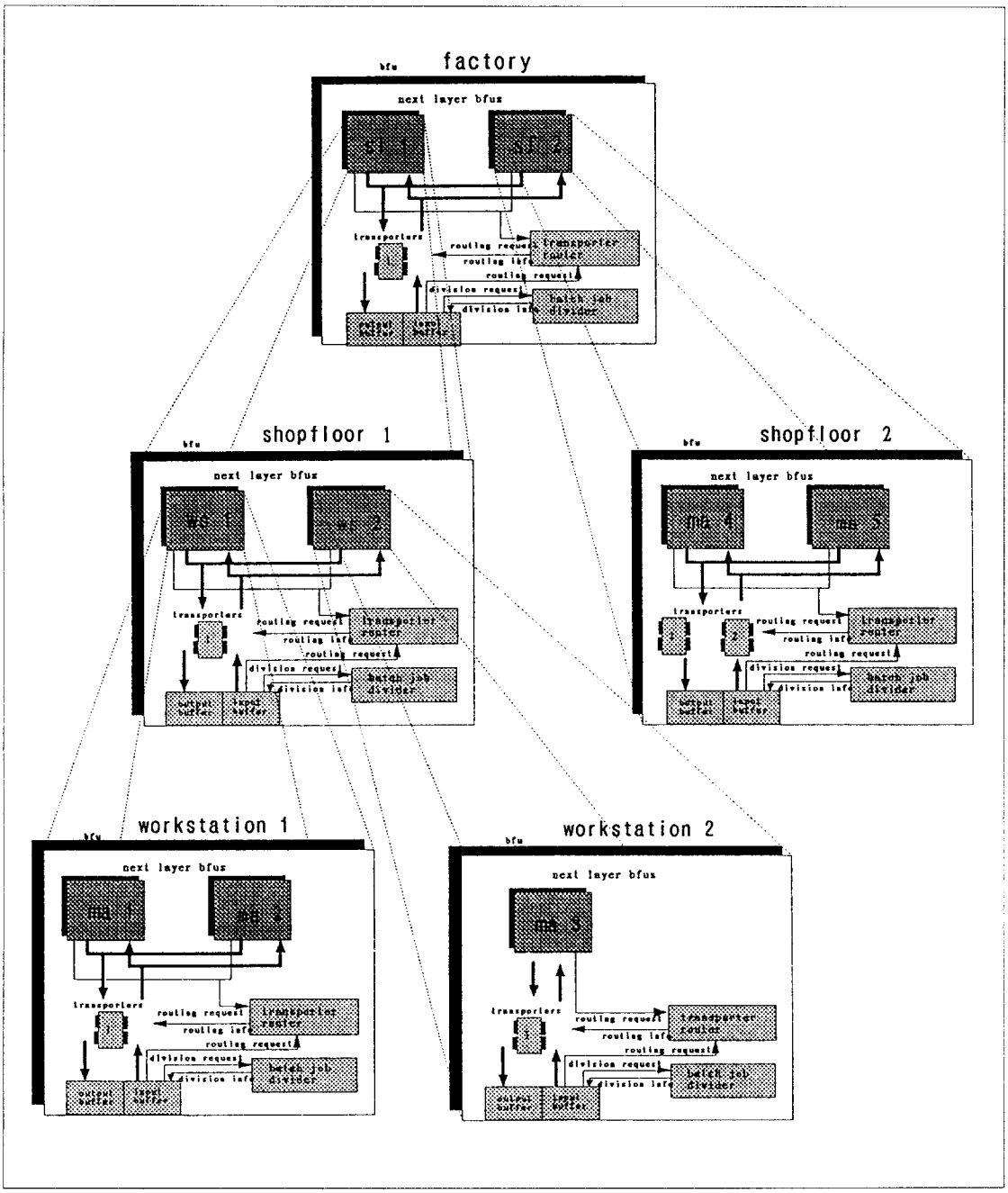
〈**Figure 9**〉 **Delay pruning.**

(Figure 10).

4.1 SES of Fractal Architecture

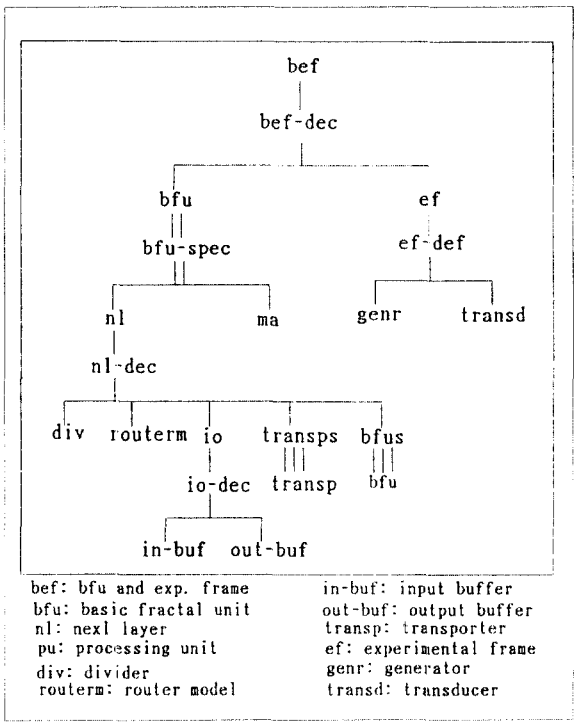
The SES for fractal architecture is shown in 〈Figure 11〉. The top of SES is *bef* which represents the model being composed of BFU (or *bfu*; *bfu* is the actual model name) and *ef* (experimental frame). The experimental frame consists of *genr* (generator model) and *transd* (transducer model). Generator provides inputs to *bfu*, and *transd* collects the outputs from *bfu* for the calculation of output statistics on *bfu*. The *bfu* is specialized into two models, the *nl* (next layer) and *pu* (processing unit). *nl* is decomposed into *div* (transfer batch job divider), *routerm* (transporter router), *io* (input/output buffer), *transps* (transporters) and *bfus* (BFUs). Where *transps* and *bfus* are broadcast models, which implies that the number of these models can be selected during pruning. The *io* is again decomposed into *in-io* (input buffer) and *out-io* (output buffer).

Notice that the left subtree of *bef* is the recursive SES. This SES fully represents the desired model architecture shown in 〈Figure 10〉 once pruned in the proper way, using recursive pruning. During recursive pruning, the recursion of *bfu* stops when pu is selected otherwise the recursion continues. 〈Figure 12〉 shows PES for the architecture in 〈Figure 10〉. This architecture is just one of many possible



〈Figure 10〉 Fractal architecture pruned to have five bfus.

architectures that can be generated from the SES for *bef* (Figure 10). We can have any number of recursions (hierarchical levels) of BFUs in constructing an architecture from the SES. This capability is shown as the PES having



〈Figure 11〉 System entity structure for bef (basic fractal unit with experimental frame) architecture.

three layers of recursion for the left side of the tree and just two layers for the right side. Also the number of BFUs at each layer can be selected to be any number since *bfu* model, except the one at the top of the SES for *bef*, is multiple decomposition of multiple entity *bfus*.

5. Conclusion

The structural knowledge of a target system (e.g. FMS) is expressed by SES knowledge representation scheme. The pruned SES is extracted by applying the pruning operation to SES to construct the desired architecture. The proposed *recursive pruning* is a pruning operation for constructing the recursive structure of a model where sub-structure contains copies of itself as in FMS fractal architecture.

The fractal architecture built by applying recursive pruning ensures ease in the reconfiguration of the overall structure and the reusability of the architecture when used as a sub component of the more complex system. Alternate structures (reconfigured structures) of the FMS model can be generated rapidly through the recursive pruning process. The alternate structures can have different hierarchical levels, different number of sub-BFUs and transporters within a BFU.

REFERENCES

[1] M. L. Law and W. D. Kelton, *Simulation Modeling and Analysis*, second edition McGraw-Hill, Inc., 1991.

[2] K. R. Anderson and G. W. Diehl, "Rapid Modeling: In the Design of A New PCB Manufacturing System," *Proceedings of the 1989 Winter Simulation Conference*, vol. 48, pp. 818-826, 1989.

[3] S. D. Wu, "Artificial Intelligence and Scheduling Applications," *Artificial Intelligence: Manufacturing Theory and Practice (ed. S.T. Kumara)*, 1989.

[4] T. M. Tirpak, S. M. Daniel, J. D. LaLonde and W. J. Davis, "A Note on a Fractal Architecture for Modelling and Controlling Flexible Manufacturing Systems," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 22, pp. 564-567, June 1992.

[5] J. A. Buzacott, "The Fundamental Principles of Flexible Manufacturing Systems," *in Proceedings of the 1st International Conference on Flexible Manufacturing Systems*, Brighton, U. K., pp. 1-12, Oct. 1982.

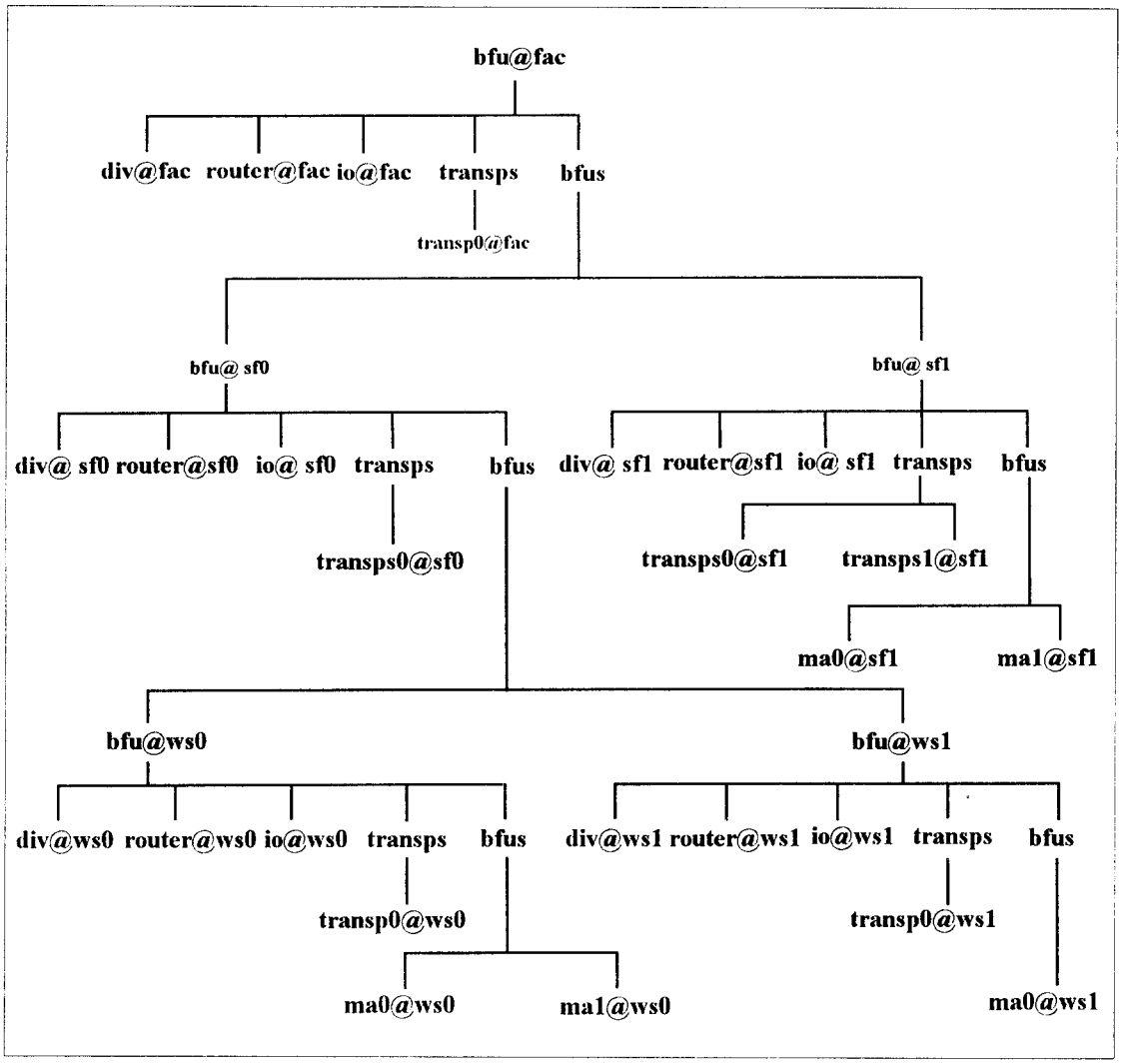
[6] B. P. Zeigler, *Multifacetted Modeling and Discrete Event Simulation*, Orlando, FL, USA: Academic Press, 1984.

[7] B. P. Zeigler, *Object-Oriented Simulation with Hierarchical, Modular Models*. San Diego, CA, USA: Academic Press, 1990.

[8] B. P. Zeigler, *Theory of Modelling and Simulation*, NY, USA: John Wiley, 1976.

[9] A. I. Concepcion and B. P. Zeigler, "The DEVS formalism: hierarchical model development," *IEEE Transactions on Software Engineering*, vol. 14, no. 2, pp. 228-241, 1988.

[10] B. P. Zeigler, "Hierarchical, Modular Discrete Event Modeling in an Object Oriented Environment," Simu-



〈Figure 12〉 Pruned system entity structure of bfu.

lation Journal, vol. 49, no 5, pp. 219-230, 1987.

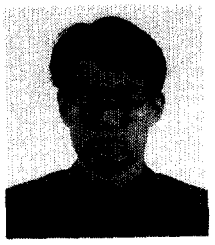
[11] J. W. Rozenblit, W. Hu, T. G. Kim and B. P. Zeigler, "Knowledge-based Design and Simulation Environment (KBDSE): Foundation Concepts and Implementation," *Journal of The Operational Research Society*, vol. 41, no. 6, 1990.

[12] T. G. Kim and B. P. Zeigler, "Knowledge-based environment for investigation multicomputer architecture," *Information and Software Technology*, vol. 31, no. 10, 1989.

[13] J. W. Rozenblit, "A Conceptual Basis for Integrated, Model-Based System Design," Ph.D. dissertation, Department of Computer Science, Wayne State University, Detroit, Michigan, 1985.

[14] T. H. Cho, "A Hierarchical, Modular Simulation Environment for Flexible Manufacturing System Modeling," Ph.D. dissertation, Univ. of Arizona, Tucson, AZ, 1993.

● 저자소개 ●



조대호

1983 성균관대학교 전자공학과 학사
1987 알라바마대 전자공학과 석사
1993 아리조나대 전자 및 컴퓨터공학 박사
현재 경남대학교 전자계산학과 전임강사
관심분야 : 모델링 및 시뮬레이션, 지능형 시스템