ILL: 79091778

Call Number: QA76.9.C65 M623 1989
Location: Storage
Maxcost: $50IFM

DateReq: 6/18/2011
Date Rec: 6/20/2011
Borrower: AZU

☐ Yes
☐ No
☐ Conditional

Request Type:
Source: ILLiad
LenderString: *IWA,RCE,ORE,IXA,TXA
OCLC Number: 20169008
Affiliation: AZNET ; GIF ; GWLA ; SHRS
Staff Email: askddt@u.library.arizona.edu
Billing Notes:

Title: Modelling and simulation methodology : knowledge systems' paradigms /

5/7

Uniform Title:
Author:
Edition:
Imprint: Amsterdam ; New York : North-Holland ; New York, N.Y., U.S.A. : ~~Distributo~~

Article: Rozenblit, J.W. and Hu, J.F.: Experimental Frame Generation in a Knowledge-Based System Design and Simulation Environment

Vol: No.: Pages: 451-466 Date: 1989

Dissertation:
Verified: <TN:951829><ODYSSEY:150.135.238.6/ILL> OCLC 9780444880444 (U.S.)

Borrowing Notes: For Book Chapter requests, scan the chapter only, do not lend book. If over your page limit, please CONDITIONAL request. Thank you.
ShipTo: ILL/UNIVERSITY ARIZONA LIBRARIES/1510 E UNIVERSITY/TUCSON AZ 85721-0055

E-delivery Addr: (520) 621-4619 //ODYSSEY PREFERRED/ILL // ARIEL: 150.135.238.50
Ship Via: Ariel, Odyssey or Library Mail

ShipVia: **Ariel, Odyssey o**

Return To:

Interlibrary Loan
204 Parks Library
Iowa State University
Ames, IA 50011-2140

Ship To:

ILL
UNIVERSITY ARIZONA LIBRARIES
1510 E UNIVERSITY
TUCSON AZ 85721-0055

NeedBy: 7/18/2011
Borrower: AZU

ILL: **79091778** Lender: IWA
Req Date: **6/18/2011** OCLC #: 20169008
Patron: **Napalkova, Liana**
Author:
Title: **Modelling and simulation methodology : knowle**
Article: Rozenblit, J.W. and Hu, J.F.: Experimental Frame Generation in a Knowledge-Based System Design and Simulation Environment
Vol.: No.:
Date: 1989 Pages: 451-466
Verified: <TN:951829><ODYSSEY:150.135.238.6/I
Maxcost: $50IFM Due Date:
Lending Notes:
Bor Notes: For Book Chapter requests, scan the chapter only, do not lend book. If over your page

MODELLING AND SIMULATION METHODOLOGY
Knowledge Systems' Paradigms
M.S. Elzas, T.I. Ören, and B.P. Zeigler (Editors)
Elsevier Science Publishers B.V. (North-Holland), 1989

CHAPTER V.4

EXPERIMENTAL FRAME GENERATION IN A KNOWLEDGE-BASED SYSTEM DESIGN AND SIMULATION ENVIRONMENT [1]

Jerzy W. Rozenblit and Jhyfang Hu

AI and Simulation Group
Dept. of Electrical and Computer Engineering
The University of Arizona
Tucson, Arizona 85721
U.S.A.

The focus of this chapter is the development of experimental frames in an advanced system design and simulation environment. The research presented here is concerned with laying a groundwork for the formulation and realization of simulation experiments in design studies. Two following issues are discussed: a.) investigation of methods for defining experimental conditions (experimental frames) based on the set of design model performance objectives; b.) definition of procedures for the aggregation of experimental frames and specification of trade-off frames under multiple performance evaluation criteria.

1. KNOWLEDGE-BASED SYSTEM DESIGN AND SIMULATION

Modern engineering design is a highly complex process involving consideration of a multiplicity of objectives, constraints, materials, and configurations. Despite great strides in computational tools such as high performance workstations intended to help to cope with this rising complexity, the design process remains error prone. Given the often severe constraints imposed by cost, environmental impacts, safety regulations, etc., designers are forced to make compromises that would not be necessary in an ideal world. Simulation is increasingly recognized as a useful tool in assessing the quality of suboptimal design choices and arriving at acceptable trade-offs. However, the working hypothesis of this presentation is that simulation and other advanced computational tools are of limited effectiveness without a methodology to induce a systematic handling of the multitude of goals and constraints impinging on a design process.

Our research aims to develop and implement a methodology of design in which design models can be synthesized and tested within a number of objectives, requirements, and constraints. This framework, termed knowledge-based system design and simulation, is presented in detail in (Rozenblit and Zeigler, 1985, 1988; Rozenblit, 1985). Here, we summarize its basic tenets.

[1]Research reported here was supported by NSF Grants DCR 8407230 for "Distributed Simulation of Hierarchical Multilevel Models" and DCR 8514348 for "Variant Families of Hierarchical Discrete Event Models: Distributed Simulation", and Semiconductor Research Corporation contract 87-MP-086 for "VLSI Packaging Research".

As illustrated in Figure 1, the design objectives (understood here in a broader context that includes requirements and constraints of the design process) drive two fundamental processes in the methodology: first, they facilitate the construction, retrieval, and manipulation of design entity structures (Rozenblit and Zeigler 1986, 1988). The design entity structure is based on a tree-like graph that encompasses the boundaries, decompositions and taxonomic relationships that have been perceived for the system being modelled. An entity signifies a conceptual part

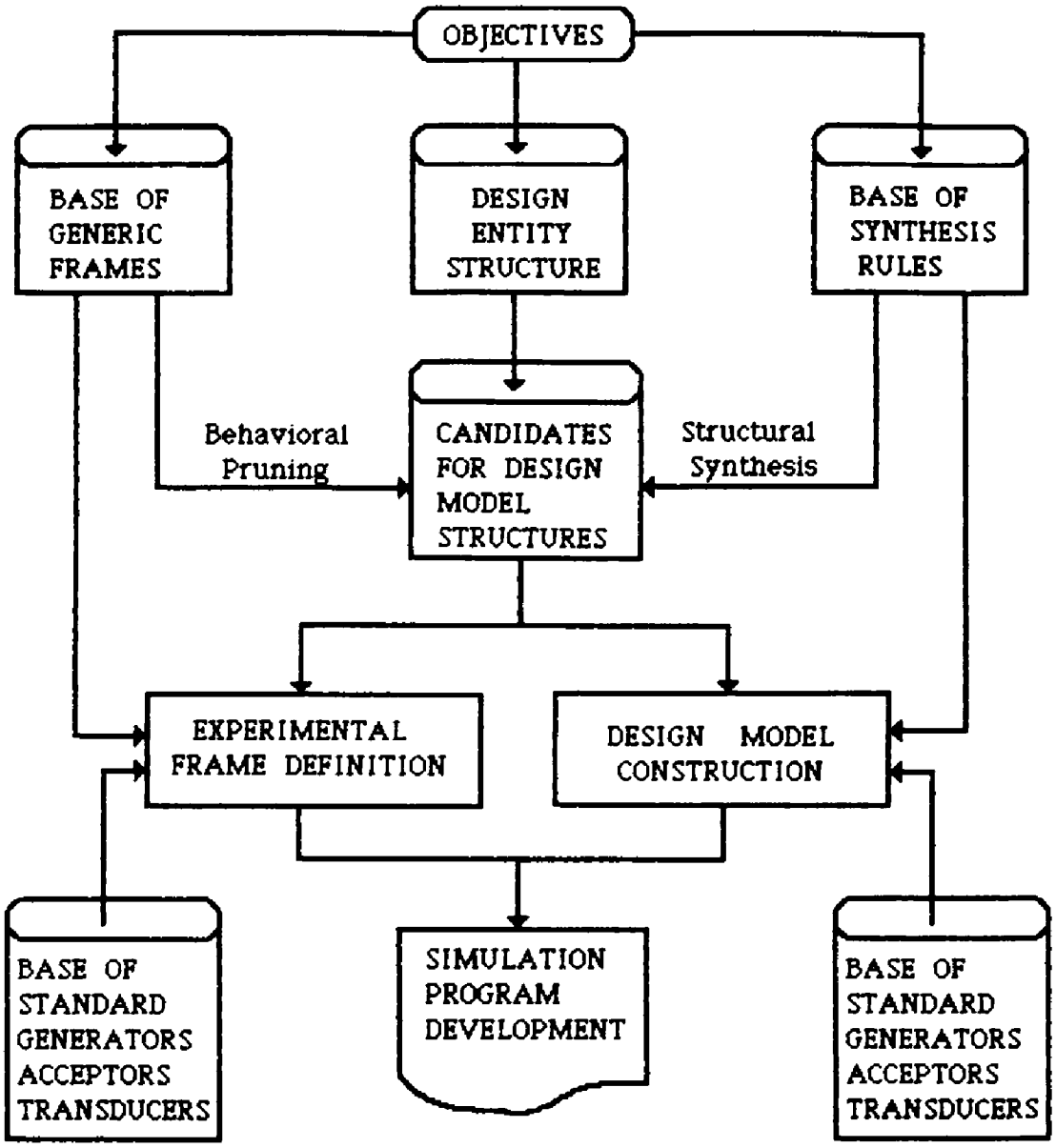

Figure 1. Knowledge Based Design Support Environment

of the system which has been identified as a component in one or more decompositions. Each such decomposition is called an aspect. Thus entities and aspects are thought of as components and decompositions, respectively. In addition to decompositions, there are relations termed specializations. A specialization relation facilitates representation of variants for an entity. Called specialized entities, such variants inherit properties of an entity to which they are related by the specialization relation.

Aspects can have coupling constraints attached to them. Coupling constraints restrict the way in which components (represented by entities) identified in decompositions (represented by aspects) can be joined together. In addition to coupling constraints, there are selection constraints in the system entity structure. Selection constraints are associated with specializations of an entity. They restrict the way in which its subentities may replace it in the process of model construction (Rozenblit et. al., 1986, Rozenblit and Huang 1987).

Secondly, the objectives serve as a basis for the specification of the generic observation frames and experimental frames (Zeigler 1984, Rozenblit and Zeigler 1988). Generic frames consist of input, output, and summary generic variable types. The variable types express performance indices associated with a given modelling objective. Experimental frames are instantiated generic frames wherein variable types are qualified with model components and execution run conditions are defined in experiment initialization, continuation, and termination sets (Zeigler, 1984).

Given the system entity structure the modeller has a choice of a number of model alternatives. This is due to the multiplicity of aspects and specializations. In our previous research we have developed algorithms that prune the system entity structure with respect to a generic experimental frame and design constraints (Rozenblit, 1985, Rozenblit and Huang 1987). We employ the production rule problem solving approach (Winston 1984) to support automatic selection of entities from specializations and synthesis of structures underlying the design model. The modeller invokes an inference engine which, through a series of queries based on the constraint rules, allows him to consult on an appropriate model structure for the design problem at hand. The results of such consultations are stored in the base of candidates for design model structures (Figure 1). They are represented as model composition trees (Zeigler 1984).

At this point a simulation environment is invoked for evaluation of the proposed design model. A simulation shell called DEVS-Scheme is used as the evaluation tool. DEVS-Scheme (Zeigler 1986, 1987) is a knowledge-based simulation environment for modelling and design that facilitates construction of families of models in a form easily reusable by retrieval from a model base. The environment supports construction of hierarchical discrete event models. It is written in the PC-Scheme language which runs on IBM compatible microcomputers and on the Texas Instruments Explorer. Model specification and retrieval in the DEVS-Scheme simulation environment is mediated by a knowledge representation component designed using the system entity structuring concepts. A user prunes the entity structure obtaining a reduced structure that specifies a hierarchical composition tree. Upon invoking the transform procedure, the system searches the model base for model components specified in the model composition tree and synthesizes the desired model by coupling the components together in a hierarchical manner. The result is a discrete event simulation model expressed in DEVS-Scheme which is ready to be executed to perform simulation.

In the ensuing sections, we describe efforts towards incorporating methods for the experimental

frame construction into the above environment.

2. KNOWLEDGE REPRESENTATION FOR EXPERIMENTAL FRAME DEFINITION

The basic structure of an experimental frame is defined as a coupling of a generator of input segments, an acceptor for monitoring the model run control variables, and a transducer for observing model outputs and processing them into summary variables. Rozenblit and Zeigler (1986) proposed a scheme for formulating an experimental frames based on the system entity structure and the generic observation frame concept. In that scheme, generic frame's variables are instantiated with model component names in a manner consistent with the model's I/O specification. This specification is derived from coupling constraints represented in the system entity structure.

Rozenblit (1985) has further extended the specification of simulation experiments by defining the distributed experimental frame architecture. In his approach, a frame composition tree isomorphic with the model composition tree is defined. Thus, a hierarchical, multicomponent model is evaluated in a hierarchical frame where each atomic model has an atomic frame and each coupled model has a coupled experimental frame, respectively. The distributed experimental frame has been realized in DEVS-Scheme environment by Duh (1988). This realization is limited in that the modeller has to define each atomic and coupled frame based on the modelling domain specification and check the frame/model coupling consistency. We propose to extend the distributed framework by providing methods for automatic generation and retrieval of experiments from a base of atomic generic experimental frame components.

Our first task in enhancing the environment depicted in Figure 1 is to establish the base of atomic frame components. There are two levels of the frame base design as shown in Figure 2. At a generic level, the data base will contain basic frame components from which coupled frames can be assembled. Basic frame modules will include:

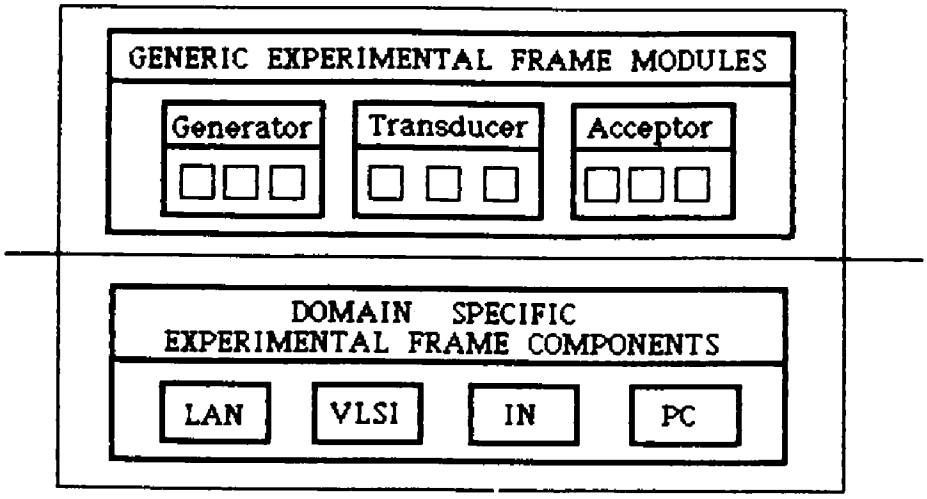

Figure 2. Organization of the Experimental Frame Base

Generators:
- Function: SIN, COS, or user-defined function
- Distribution: NORMAL, BINOMIAL .. distribution
- Constant: random numerical generator
- Symbol: random symbol generator

Acceptors:
- Reached Constant: control based on accumulated values
- Comparator: control based on comparison of variables
- Timer: control based on execution time
- Debugger: control based on error
- Satisfaction: control based on level of satisfaction

Transducer:
- Count: count the number of variables
- Elapsed Time: accumulate the total elapsed time
- Sum: perform addition of variables
- Integrate: accumulate time integral of variables
- Maximum: determine the maximum value
- Minimum: determine the minimum value
- Mean: compute the mean of variables
- Median: compute the median of variables

The domain specific frames constructed for particular applications will be stored for possible re-use in future simulations of specific design models.

To facilitate the automatic frame generation given atomic frame modules we augment the system entity structure with an entity information frame (EIF). As illustrated in Figure 3, each entity will have a frame structure called EIF associated with it.

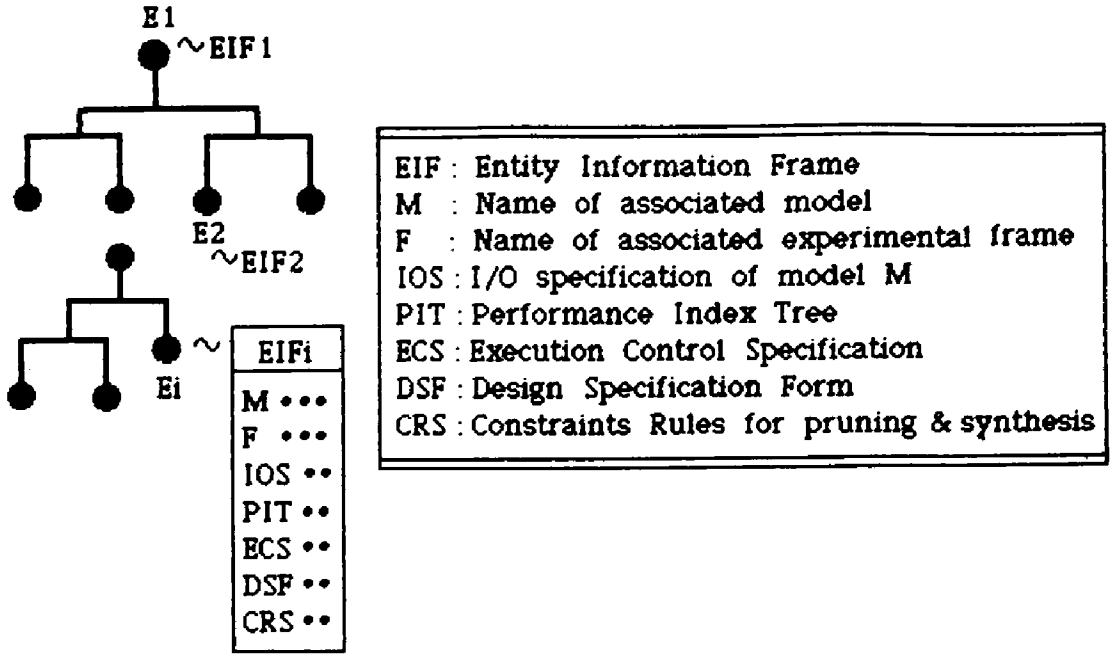

Figure 3. System Entity Structure with Entity Information Frames

Components of an entity information frame are first used in in the model development process. CRS is applied in rule-based pruning. After the model composition tree is generated, M serves as an index for retrieval of an atomic DEVS model. If such a model does not exist, M is instantiated to a model newly constructed.

The components used for the experimental frame generation are: the design specification form (DSF), the performance index tree (PIT), and the execution control specification (ECS). The I/O specification (IOS) is used for assuring coupling consistency between the model M and experimental frame F. Again, the frame F may already exist in the frame base. However, if F cannot be retrieved, then it has to be generated based on the knowledge provided by EIF. We shall describe the frame generation process in Section 3.

The execution control specification includes definition of initialization, continuation, and termination conditions. For example, typical termination rules might include:

IF Queue.Length > Max.Length
THEN stop simulation

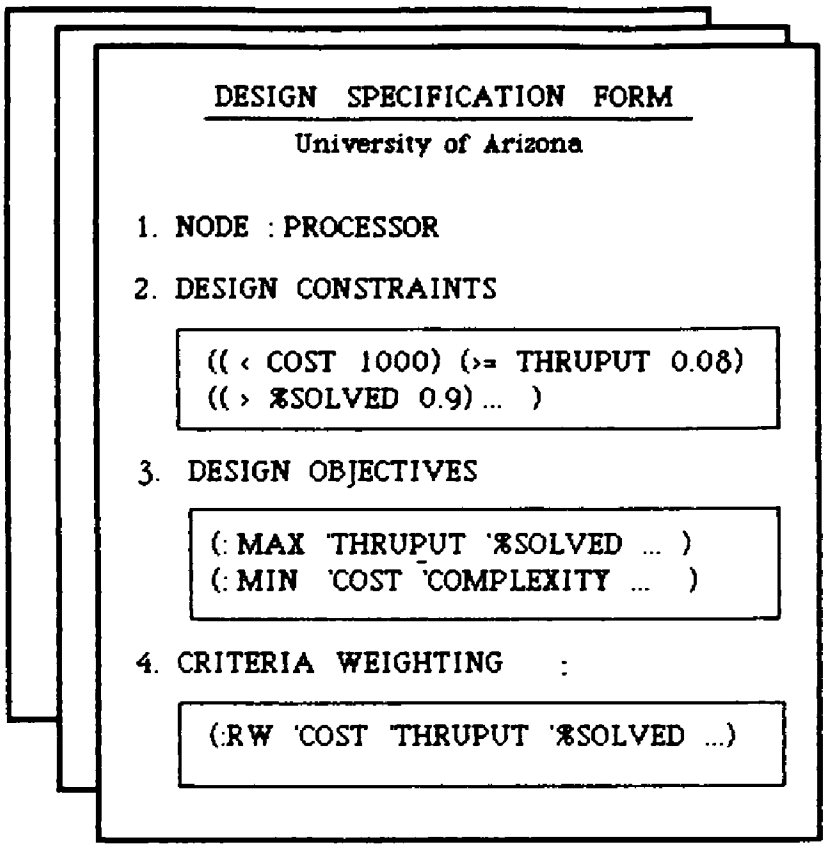

Figure 4. Design Specification Form

The design specification form (DSF) in Figure 4 serves as a means of representing knowledge about design objectives, constraints, and evaluation criteria for a model associated with the entity E of EIF. The node field of DSF is used to indicate the associated entity node in system entity structure. The design constraints field of DSF implies the requirements that must be satisfied by the resulting system. Each performance constraint is expressed by a triple as shown below:

(Relation Performance-Index Value)

For example, in the design of a local area network (LAN), the constraint on a ring network can be expressed as:

(< RING.Average-Transfer-Delay 280 μ sec)

The "relation" indicates the requirement of the performance index over the specified value. The "performance index" is composed of "object" and "generic frame". The "object" indicates the associated model. The "generic frame" provides variable types, such as Throughput, Efficiency, Cost, etc.. The "value" is the index used to indicate the quality (good, fair, bad, high, medium, low, etc.) or quantity (100, 0.9, etc.) of the associated attribute. The performance constraints can be classified into two categories:

- Static performance constraints: This type of constraints is expressed by performance indices which are essentially design parameters used to characterize the physical properties of the design object. The static performance indices are usually predefined during the construction of models. The static performance constraints can be directly used as criteria to prune the entity structure without invoking any simulation since their values are known and are not modified by the system's dynamic behavior. Typical examples of static performance indices are the unit cost, maximum capacity, area, volume etc.
- Dynamic performance constraints: Dynamic constraints are expressed by performance indices which are relevant to the behavior of the system being designed. Simulation methods must be invoked to measure the performance of model components. For example, typical dynamic performance indices in the design of local area network are network throughput, average transfer delay, effective service time etc.

Design objectives imply the design goal. Specification of design objectives will be used to conduct the design optimization. Typical design objectives are to maximize or minimize one or more performance indices. For example, in the design of local area network, the objectives of a ring network may be expressed as:

((MAX Thruput Utilization ...) (MIN Cost Average-Transfer-Delay ...))

Finally, the criteria weighting conveys the designer's preference over the set of evaluation measures. In Section 4, we shall give examples of typical criteria weighting scheme.

3. AUTOMATIC FRAME RETRIEVAL AND GENERATION

As we have indicated in Section 2, our current simulation environment does not provide facilities to automatically generate and couple experimental frames. This section describes our approach to developing a procedure for frame development. The procedure involves the following steps:

1). Identification of performance indices and control strategies

Relevant EIF slots of the related entity (i.e., DSF and ECS) are used to extract knowledge about performance indices and control strategies. The execution control must be defined before the DEVS simulation. A user-friendly menu-driven interface will be designed to provide selection of execution controls. Since on- line modification is allowed in the DEVS simulation environment, the execution controls may be changed during simulation.

2). Extraction and Aggregation of Atomic Frames

The first step toward the automatic frame generation is to identify the generic frame type from the design specification. Each generic frame will activate the extraction of one or more atomic frames represented in DEVS-Scheme code.

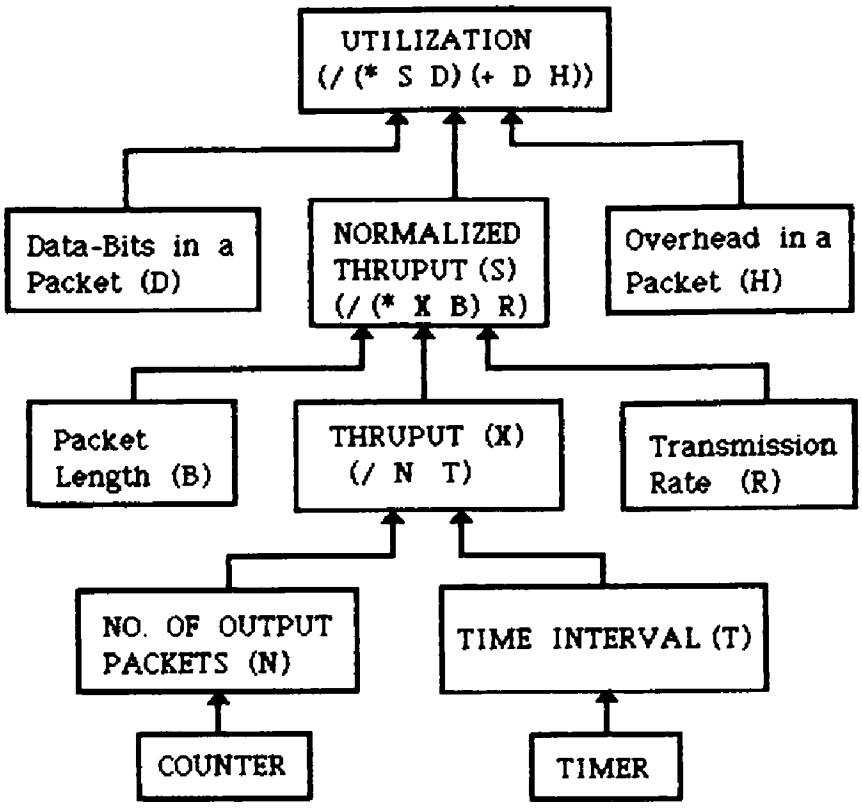

Figure 5. Performance Index Tree for Aggregating Utilization Transducer

The extraction of atomic frames starts from the reference to the Performance Index Tree (PIT). The performance index tree is a semantic data structure indicating how the related performance index is evaluated within design parameters, atomic frames, and/or lower-level performance indices. A typical performance index tree used for evaluating the utilization of a local area network is illustrated in Figure 5. Each node of the performance index tree may correspond to a design parameter, an atomic frame, or a performance index. The Transducer Aggregation Key (TAK) is used to indicate how the related performance index can be aggregated from its child nodes. By referring to the performance index tree, all relevant atomic frames can be identified, extracted, and aggregated by the system. As shown in Figure 5, to build an experimental frame for utilization, the system starts searching its sub-level frames. Since the number of data bits in a packet (D) and number of overhead bits in a packet (H) are known as design parameters, only the frame of normalized throughput (S) will be extracted for aggregation. If the frame of normalized thruput (S) is not contained in the frame base, the system must aggregate the frame of normalized thruput before the frame of utilization is aggregated.

Aggregating the normalized thruput frame (S) will require that a frame thruput (X) be available in the frame base. If (X) is not available, it will be aggregated from two frames: number of output packets (N) and time interval (T). These two frames are constructed form the generic components counter and timer, respectively. Thus, the transducer component of the frame thruput (X) will take the form depicted in Figure 6a. The TIH and TOH modules of this transducer are transducer input and transducer output handlers. They assure consistency of the frame/model coupling as we shall explain shortly. The pseudo-code realization of this transducer is presented in Figure 6b.

The aggregation of components for a generator and acceptor definition will proceed in a similar fashion. We are currently developing procedures for defining a semantic tree structure for input and control segment representation.

3). Coupling of Experimental Frame Components

After valid frame components (generator, transducer, and acceptor) are generated, the coupling scheme between frame components must be generated automatically. The coupling scheme between frame components is determined by the selection of performance indices and control strategies. The system will check the specified performance indices and control strategies to generate appropriate coupling schema. A pseudo-code representation of the coupling scheme can take the following form:

```
(:FC (G.Oports F.Oports) ;;gen -> frame
     (G.Oports T.Iports)   ;;gen -> transducer
     (T.Oports A.Iports)   ;;transducer -> acceptor
     (F.Iports T.Iports)   ;;frame -> transducer
     (F.Iports A.Iports)   ;;frame -> acceptor
          .....
     ) ;; end of frame coupling
```

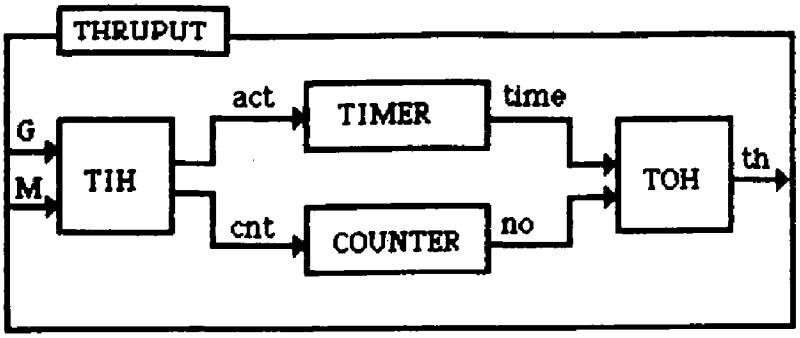

Figure 6a Structural Realization of the THRUPUT Transducer

```
Module: TIH
When receive message on input port-G
  Send 'activate to output port-ACT
When receive message on input port-M
  Send 'increment to output port-CNT
```

```
Module: TIMER
When receive message on input port-ACT
    T-START = SYS-CLOCK
Time = SYS-CLOCK - T-START
Send Time to output port-TIME
```

```
Module: COUNTER
When receive message on input port-CNT
  Out-Number = Out-Number + 1
Send Out-Number to output port-NO
```

```
Module: TOH
When receive message on input port-TIME
 Time-Interval = Message.Time
When receive message on input port-NO
 Out-Number = Message.Out-Number
Thruput = Out-Number/Time-Interval
Send Thruput to output port-TH
```

Figure 6b Pseudo-Code Realization of the THRUPUT Transducer

Since the aggregated frame is automatically generated based on the consideration of design specification and I/O consistency, we should be able to couple the aggregated frame with the design model without any difficulties. Each output port of the design model has a corresponding input port in the aggregated frame. In the same way, each output port of the aggregated frame has a corresponding input port at the designed model. A pseudo coupling between the design model and the aggregated frame is shown below:

```
(:MFC (MODEL.Oport-I FRAME.Iport-I) ;; model -> frame
      (FRAME.Oport-J MODEL.Iport-J) ;; frame -> model
      ) ;;end of model/frame coupling
```

4). Coupling of model and experimental frame.

During the construction of an experimental frame, two special function modules called Input Packet Distributor (IPD) and Output Packet Formatter (OPF) will be installed in the frame. The IPD is used to distribute the contents of input packets to the relevant atomic frames according to the generic frame type of performance indices. The evaluations of performance indices are accomplished by operating the aggregation key over the relevant outputs of atomic frames. The OPF is used to pack the experimental results into a valid packet based on I/O specification of the design model. Finally, these packets will be sent to the appropriate output ports for further propagation. The schematic structure of an aggregated frame is shown in Figure 7.

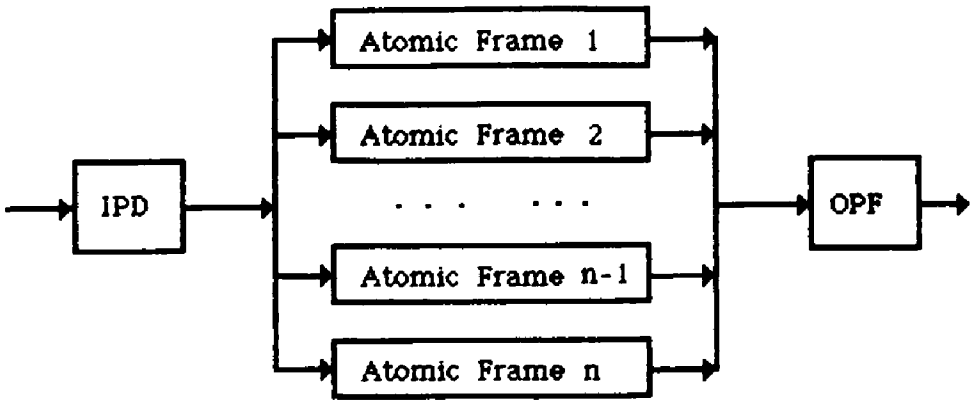

Figure 7. Structure of an aggregated frame

Since the original model configuration is changed due to the generation of local frames, the coupling scheme of the design model must be changed consistently. Following is an example of coupling scheme transformation:

Before local frame assignment:

```
(:MC (A.IN B.IN)
    (B.OUT C.IN)
    (C.OUT A.OUT)
    ) ;; serial coupling recipe
```

After local frame assignment on model B:

```
(:MC (A.IN BF.IN) ;; BF: coupled model of B and F
    (BF.IN B.IN)
    (F.OUT B.IN) ;; F: local frame assigned to B
    (B.OUT F.IN)
    (B.OUT BF.OUT)
    (BF.OUT C.IN)
    (C.OUT A.OUT)
    ) ;; end of coupling transformation
```

In other words, whenever a local frame is assigned to the model, an extra co-ordinator must be declared for the coordination between model and its local frame. One schematic explanation is shown in Figure 8 for reference.

Sucessful aggregation and coupling of an experimental frame will lead to a simulation study of a design model. Given a set of design constraints and objectives, there may exist more than one design alternative which conforms to the design specifications. Therefore, we ought to provide procedures for meanigful evaluation of design models under multiple criteria.

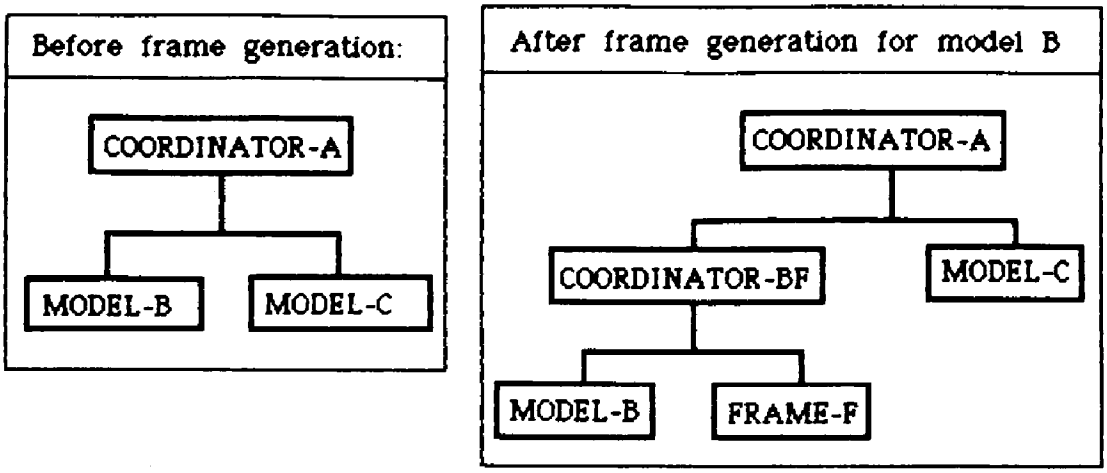

Figure 8. Coupling transformation with frames

4. MODEL EVALUATION UNDER MULTIPLE CRITERIA

Each design alternative will be evaluated under the system- generated experimental frame. Since design evaluations proceed under multiple, often conflicting, objectives, it is necessary to define methods that handle such situations adequately. In order to determine the best design from a number of design alternatives, the Trade-Off Frame (TOF) is designed to perform the selection purpose. A trade-off frame is a coupling of individual experimental frames (each corresponding to a single performance objective) with trade-off orderings defined over the set of output variables of each frame. The Multiple Criteria Decision Making (MCDM) methods will be integrated in the trade- off frame to make the best decision for the user. It is hard to say which MCDM method is best. Furthermore, no single MCDM method can be applied to solve all different types of decision making problems.

Typical weighting schemes of MCDM problems are:

- Unknown weighting: The unknown weighting is used when the user is unable to give any preference information about design criteria. Under this situation, all criteria are regarded as having equal importance.
- Complete weighting: The complete weighting is used when the user is able to specify exact preference for each criterion. An example of this category is to assign each criterion a positive weight and let the sum of total weights be equal to 1.
- Ranked weighting: The ranked weighting is used whenever partial preference information is available but is not comprehensive enough to give exact preference values of criteria. A typical example in this category is to list the trade-off order of criteria.
- Fuzzy weighting: The fuzzy weighting is used whenever the user is able to define the preference range (indicated by the lower-bound and upper-bound) of criteria.

To assure a reliable solution, the most appropriate MCDM method will be selected by a rule-based system according to the weighting scheme specified in the design specification. For example, if the rank weighting is used to indicate the preference of criteria, simple application of MCDM methods, such as Maximin, Maximax, Minimax, or Bayes-Laplace (Kmietowicz 1981; Osyczka 1984) may cause incorrect selection of the best design (Hu and Rozenblit 1988). The schematic representation of a TOF evaluation is explained in Figure 9.

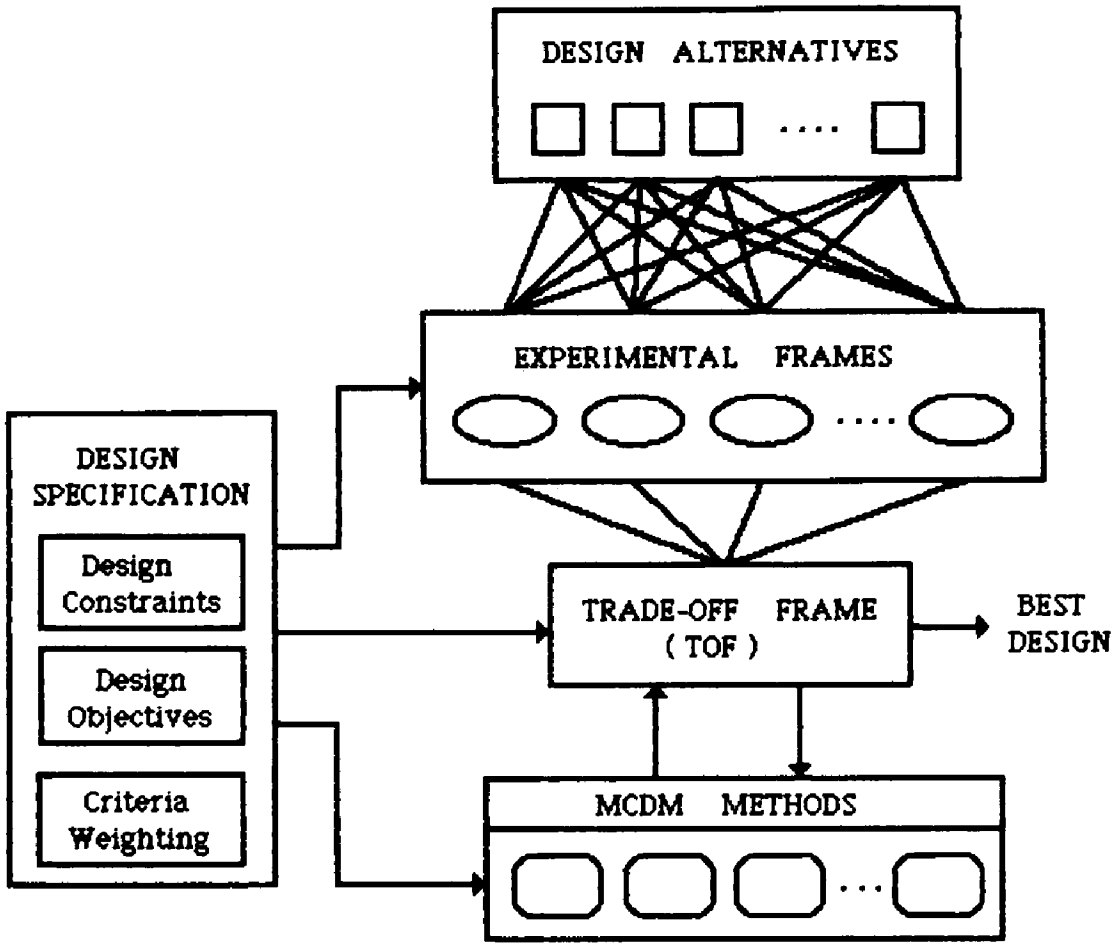

Figure 9. Evaluation of Design Model Under Multiple Criteria

5. DIRECTIONS OF RESEARCH AND CONCLUSIONS

This chapter has presented our work concerned with the development of procedures for automatic experimental frame generation in a knowledge-based system design and simulation environment. Perceived contributions of our approach are:

- Complicated engineering design process becomes transparent to the system designer.
- Design cycle is reduced by automating the design process. The automation includes generating design configurations, validating the design alternatives by comparing the simulation results with the design constraints, and selecting the best design configuration by considering all conflicting design objectives.
- The concept of associating the semantic structural trees to the behavior model/frame description facilitates the knowledge manipulation in the model-based expert design support system.

There are a number of issues that need to be addressed as our reserch progresses. We are currently developing methods for parsing compound perfromance indices and decomposing them into performance index trees. We are also developing semantic trees for aggregation of acceptors and generators. The results will realized in an object oriented programming shell and embedded in the DEVS-Scheme simulation environment.

6. REFERENCES

Duh, Jang (1987). Realization of Distributed Experimental Frames in DEVS Scheme Environment, Master Thesis, University of Arizona, Tucson.

Hu, Jhyfang and J. W. Rozenblit (1988). Towards Automatic Generation of Expeprimental Frames in Simulation-Based System Design, In: Proceedings of 1988 Eastern Simulation Conference, The Society for Computer Simulation, Orlando, Florida.

Kmietowicz, Z. W. (1981). Decision Theory and Imcomplete Knowledge, Gower Publishing Company Ltd., England.

Osyczka, Andrzej. (1984). Multicriterion Optimization in Engineering, Ellis Horwood Ltd., England.

Rozenblit, J. W. (1986). A Conceptual Basis for Integrated, Model-Based System Design, Technical Report, Department of Electrical and Computer Engineering, University of Arizona, Tucson, Arizona.

Rozenblit, J. W. and B. P. Zeigler. (1985). Concepts for Knowledge-based System Design Environments, In: Proceedings of the 1985 Winter Simulation Conference, San Francisco, California, December.

Rozenblit, J. W. and B. P. Zeigler. (1986). Modelling and Simulation in the Artificial Intelligence Era, North Holland, Amsterdam, 79 p.

Rozenblit, J. W. and Y. M. Huang. (1987). Constraint-Driven Generation of Model

structures, In: Proceedings of the 1987 Winter Simulation Conference, Atlanta, Georgia, December.

Rozenblit, J. W. and B. P. Zeigler. (1988). International Encyclopedia of Robotics Applications and Automation, John Wiley and Sons, Inc., New York, 308 p.

Winston, P. H. (1984). Artifical Intelligence, 2nd Ed., Addison Wesley Publishing Company, Reading, Massachusetts.

Zeigler, B. P. (1984). Multifaceted Modelling and Discrete Event Simulation, Academic Press, London.

Zeigler, B.P. (1986). DEVS-Scheme: A LISP-based Environment for Hierarchical, Modular Discrete Event Models, Technical Report AIS-2, Dept. of Electrical and Computer Engr., The University of Arizona, Tucson, AZ.

Zeigler, B.P. (1987). Hierarchical, Modular Discrete Event Modelling in an Object Oriented Environment, Simulation, Vol.49, nr. 5, pp. 219-230.

ELSEVIER SCIENCE PUBLISHERS B.V.
Sara Burgerhartstraat 25
P.O. Box 211, 1000 AE Amsterdam, The Netherlands

Distributors for the United States and Canada:

ELSEVIER SCIENCE PUBLISHING COMPANY INC.
655 Avenue of the Americas
New York, N.Y. 10010, U.S.A

ISBN: 0 444 88044 5

Printed in The Netherlands