N90-27279

# DEVS-based Intelligent Control of Space Adapted Fluid Mixing

Sung-Do Chi and Bernard P. Zeigler
AI-Simulation Group
Department of Electrical and Computer Engineering
University of Arizona, Tucson, AZ 85721

## Abstract

This paper describes the development of event-based intelligent control system for a space-adapted mixing process by employing the DEVS (Discrete Event System Specification) formalism. In this control paradigm, the controller expects to receive confirming sensor responses to its control commands within definite time windows determined by its DEVS model of the system under control. We apply the DEVS-based intelligent control paradigm in a space-adapted mixing system capable of supporting the laboratory automation aboard a Space Station.

## I. Introduction

"Intelligent control", the intersection of artificial intelligence, conventional automatic control, and operations research approaches, is receiving increasing attention in both theory and application(3). An intelligent control system often employs a hierarchical control structure in which a higher-level intelligent controller supervises a lower-level conventional controller. The event-based control paradigm, introduced by Zeigler(4), realizes such intelligent control by employing a discrete eventistic form of control logic represented by the DEVS formalism.

In this control paradigm, the controller expects to receive confirming sensor responses to its control commands within definite time windows determined by its DEVS model of the system under control. Since the DEVS formalism is at heart of event-based control system design, such controllers can be readily checked by computer simulation prior to implementation. Thus the DEVS formalism plays the same role to event-based control that differential and difference equation formalism play to conventional control(4). An advantage of an event-based control system using DEVS models includes its fault diagnostic capability supported by DEVS-Scheme, a LISP environment implementing of DEVS formalism(2,7,8).

This paper describes the development of DEVS-based intelligent control system for a space-adapted mixing process. The paper first reviews the concept of DEVS formalism, then uses it to formalize the dynamics of a mixing process. It shows the hierarchical architecture of the realized intelligent control system for mixing. Several simulation runs illustrate the technique.

## II. DEVS Concept for Event-based Control

The Discrete Event System Specification (DEVS) formalism introduced by Zeigler(5) provides a means of specifying a mathematical object called a system. Basically, a system has a time base, inputs, states, and outputs, and functions for determining next states and outputs, given current states and inputs(6). In the DEVS formalism, one must specify 1) basic models from which larger ones are built, and 2) how these models are connected together in hierarchical fashion. Detail descriptions of how a basic model, called an *atomic model* and the second form of model, called a *coupled model*, are specified are found in (1,7).

DEVS-Scheme is an implementation of the DEVS formalism in SCOOPS, an object-oriented superset of Scheme (a Lisp dialect), which enables the modeler to specify models

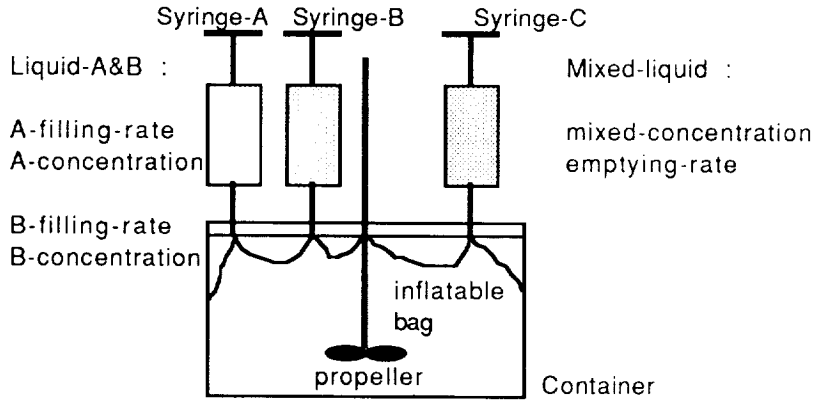



Figure 1: Space Adapted Mixing System

directly in DEVS terms. DEVS-Scheme supports building models in a hierarchical, modular manner (successively putting systems together to form larger ones), a systems-oriented approach not possible in conventional languages. Detail description of DEVS-Scheme including its class hierarchy is available in (1,7).

The DEVS formalism is more than just a means of constructing simulation models. It provides a formal representation of discrete event systems capable of mathematical manipulation, just as differential equations serve this role for continuous systems. We illustrate how mixing systems specified for the space environment may be advantageously mapped into DEVS representations. Suitably operating on the structure of such DEVS models provides a basis for the design of an event-based controller.

### III. Space Adapted Mixing Process

The design of a space-adapted container would have an aluminium bottle containing an inflatable bag, which is the actual liquid container; liquid is injected/extracted by means of syringes; air pressure between the outside of the bag and the inside of the bottle wall ensures that the bag remains "full" at all times. We treats a system with space adapted container stirred by rotating propeller as shown in Figure 1. The mixing process is batch - some quantity of fluid with a given concentration is added to a container already filled with a liquid of another concentration. The container is fed with an incoming liquid-A from the syringe-A with a flow rate $r_a$ by the control command, FILL-A. Once the level of liquid-A reaches to its pre-specified level, the flow of liquid-A might be stopped by control command, STOP. Then, another liquid-B from the syringe-B is added into the liquid-A in the container with a flow rate $r_b$ until it receives the control command, STOP. Both feeds contain dissolved material with constant concentrations, $C_a$ and $C_b$, respectively. Assume that the propeller starts either at the same time as liquid-B starts to be filled or after filling liquid-B is complete. The propeller should cease its operation by the control command, STOP. When the propeller stops, the concentration of both liquids should reach to the equilibrium value. The outgoing flow to the syringe-C has a flow rate $r_c$.

Figure 2 illustrates the relationship between dynamic characteristics and times. Figure 2(a) represents the various filling rates : A-filling-rate ($r_a$), B- filling-rate ($r_b$), and emptying-rate ($r_c$). Figure 2(b) shows the normal and fault cases of mixing effect ($\alpha$) characteristics. In the fault case, broken propeller, $\alpha$ might be slowly decreased (propeller speed is reduced). Figure 2(c) shows the volume characteristics : liquid-A volume($V_a$), liquid-B volume($V_b$), and total volume($V_c$). The concentration characteristics of normal and fault cases are shown in Figure 2(d). The concentrations of both liquids are exponentially changed toward the equilibrium value. Here we adopted the 2% steady value

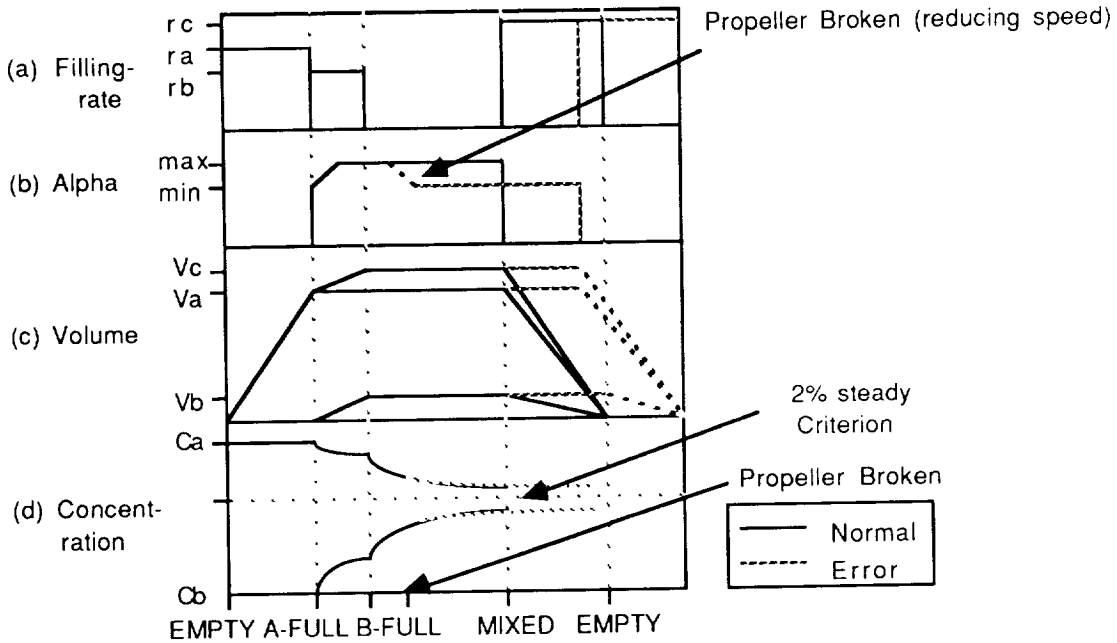



Figure 2: Dynamic Characteristics of Mixing System

criterion for reaching equilibrium level. Later, we will show how this system can be successfully controlled and how the error can be diagnosed.

## IV. DEVS Representation of Mixing Process

In the DEVS representation of event-based control, a DEVS model moves through its checkstates in concert with external inputs, as long as those inputs arrive within the expected time windows. Associated with each checkstate are a minimum time and a window. In contrast to conventional sampled data systems, event-based logic does not require sensor output precision. Sensors can have threshold-like characteristics. Only two output states, for example, on/off, are needed although more may be employed. However, to generate the time windows the output states of the sensor must be accurately and reliably correlated with values of significant process variables. Figure 3(a) shows possible threshold-type sensors. A visual sensor in Figure 3(b) can provide more precise information for fault diagnosis. Figure 4 illustrates various data types of sensory inputs. The indicator used by controller keeps track of the container state estimated by level-sensors. However, the backup sensor, tube sensor and vision sensor provide more accurate container state. Therefore, by checking these various sensory sources, the diagnoser can do the fault diagnosis.

The control task is performed as the DEVS model of control system changes its state from an initial position on a given threshold sensor boundary to a succession, possibly cyclic, of boundaries. More concretely, this means we want the system to go through a predetermined sequence of states as reported by sensor readings. Our control logic will, as each boundary crossing is achieved, issue a control action, i.e., send an appropriate input to the system, in order to move toward the next desired boundary. The controller has a time window in which it expects the appropriate sensor states to change to confirm the expected boundary crossing. The time windows are derived from the DEVS external model of the system. Figure 5 presents the phase categorization in terms of the boundary conditions of the dynamic chacteristics and their transition cycle with control commands. The transitions labelled by () are expected to take place within given time windows, as illustrated in the next section.

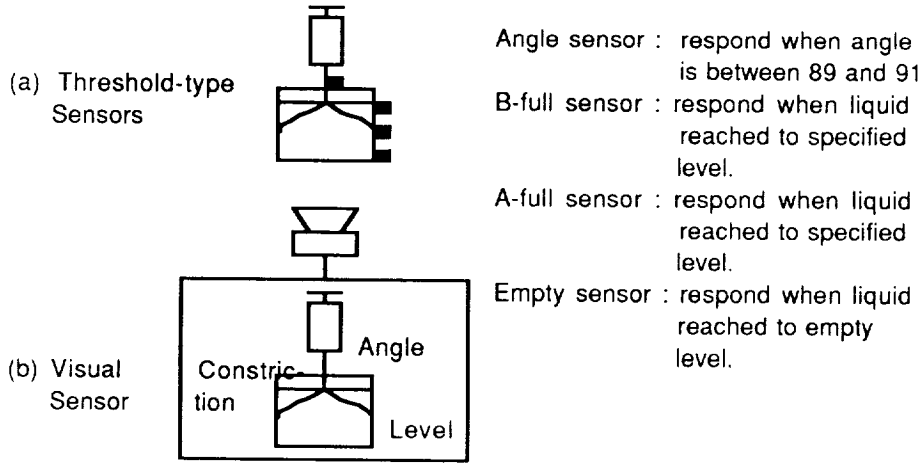



Figure 3: Sensory Inputs of Space Adapted Mixing System

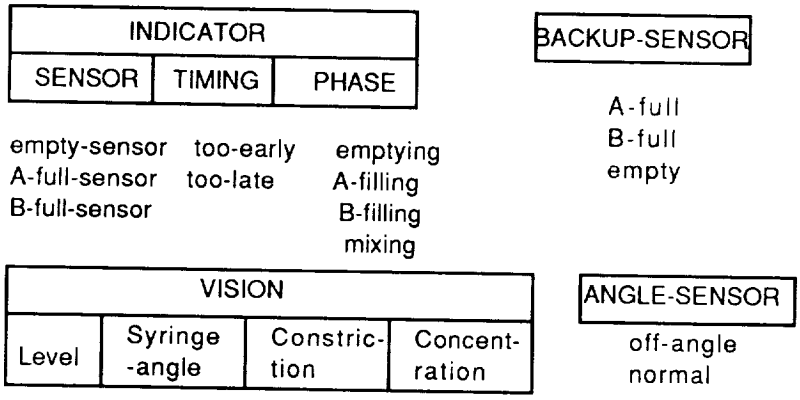



Figure 4: Data Types of Sensory Inputs

## V. Simulation Approach of Mixing Process

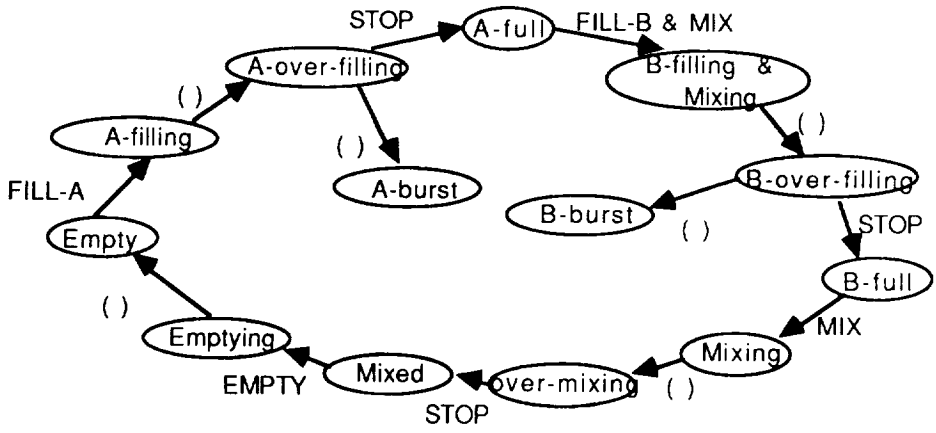
Figure 6 shows a simulation structure for the mixing process control. The simulation test is decomposed into a model representing the real bottle, MIX-E, and control-unit. The control-unit is decomposed into an operator for filling, mixing and emptying a container and a diagnoser for discovering the causes of any operational faults. There are three sub-models in the simulation :

MIX-E : Model of a space adapted mixing system. MIX-E is able to respond to both operational commands and diagnostic probes. It is external to the controller.

MIX-O : an operational model of the mixing system used by a controller, CONTRL, to generate its commands and verify the received sensor responses. Table 1 presents the operation table used in the MIX-O model. For example, the first column states that if the current state is EMPTY and the input is FILL-A then the next-state is A-FILLING; also the output in the current state is nil and its time-advance is infinity (ta = inf). The time window of the next state is given by two fields : next-ta(20) gives its lower bound, and next-wind(6) gives its width (so its upper bound is 26).

MIX-D : a classification, or expert-system-like model, employed by the diagnoser inference engine, DIAGN, to determine the probable source of breakdown. As an example, an informal presentation of some of the diagnostic rules is given below :

R1 : If backup-sensor is not A-full and A-full-sensor is true, then "A-full-sensor is bad".




| | |
|---|---|
| Empty | : empty sensor on |
| A-filling | : level < A-full level |
| A-over-filling | : A-full-sensor on. level > A-full level |
| A-burst | : Liquid-A overflow. level = TOP |
| A-full | : A-full-sensor on |
| B-filling & mixing | : A-full level < level < B-full-level. propeller sensor on. |
| B-over-filling | : B-full-sensor on. level > B-full level |
| B-burst | : Liquid overflow. level = TOP |
| B-full | : B-full-sensor on |
| Mixing | : Concentration > 2% criterion |
| Over-mixing | : Concentration < 2% criterion. Propeller on |
| Mixed | : Concentration < 2% criterion. Propeller off |
| Emptying | : empty level < level < B-full level |

Figure 5: Phase Categorization and Phase Transition Diagram

R2 : If timing is too-late and phase is A-filling and level is less than 100, then "Tube is off-angle or leaky".

R3 : If timing is too-late and phase is mixing and concentration-level is greater than 31, then "Propeller is broken".

Figure 7 shows the hierarchical structure of the mixing control system. Level III is the lowest level, where the real system under control exists. The external model, MIX-E, receives input messages such as control commands and read-sensor commands from the next higher level, Level II. It sends the sensor readings to the next higher level. Level II corresponds to the control unit shown in Figure 6. It has an operational model, MIX-O, and a rule-based model, MIX-D, to provide necessary information for the controller and diagnoser, respectively. It also sends a result message to its next higher level, Level I, which contains the goal agent, the highest unit of the control system. The agent may represent a robotic or other autonomous decision maker.

## VI. Simulation Results

To test the control logic for a mixing process, we have run several simulation experiments of a possible mixing process. The simulation experiments concern two cases : normal case and fault case. Initial values for an external model, MIX-E, under normal operation are given as follows ; $r_a = 5$, $r_b = 6.67$, $r_c = 8.3$, $V_a = 100$, $V_b = 66.6$, $C_a(0) = 50$, $C_b(0) = 0$, angle = 90, constriction effect = 5, leakage rate = 0.001, and $\alpha = 0.005$. We also assume several time delays of sensor readings, for example, 1 sec for backup-sensor

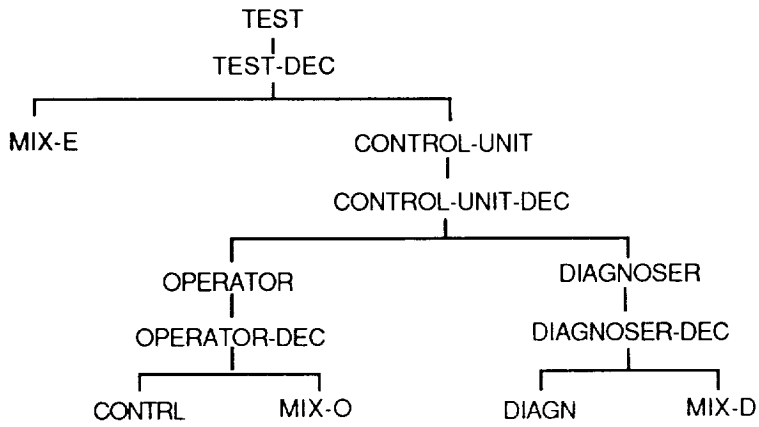



Figure 6: Hierarchical Simulation Structure for a Mixing System

| state | input | next-state | output | ta | next-ta | next-wind |
|---|---|---|---|---|---|---|
| empty | fill-A | A-filling | () | inf | 20 | 6 |
| A-filling | () | A-over-filling | A-full-sensor | 20 | 1000 | 20 |
| A-over-filling | () | A-burst | () | 1000 | inf | () |
| A-over-filling | stop | A-full | () | 1000 | inf | () |
| A-full | fill-B&mix | B-filling | () | inf | 10 | 4 |
| B-filling | () | B-over-filling | B-full-sensor | 10 | 1000 | 20 |
| B-over-filling | stop | B-full | () | 1000 | inf | () |
| B-full | mix | mixing | () | inf | 45 | 7 |
| mixing | () | over-mixing | () | 45 | 1000 | 20 |
| over-mixing | () | inf-mixing | () | 1000 | inf | () |
| over-mixing | stop | mixed | () | 1000 | inf | () |
| mixed | empty | emptying | () | inf | 20 | 6 |
| emptying | () | empty | empty-sensor | 20 | inf | () |

Table 1: Table Specification of an internal model,MIX-O

reading and 100 sec for visual-sensor reading. In the future, these time delays could be taken into account when deciding on sensors to interrogate.

The partial simulation results of normal and fault cases for a goal plan from B-full to MIXED are illustrated in Table 2(a) and (b), respectively. In the normal case, the controller, CONTRL, issues the control command, MIX, to the external model, MIX-E, and also to the internal model, MIX-O, and then waits for the sensory response during the scheduled time window (7 sec). If the sensory response arrives within the time window, the controller generates the next command and so on, till the MIX-E reaches to its goal state. But, in the fault case, where the propeller is broken during mixing (at clock time 35), the mixing effect, $\alpha$, decreases from 0.08 to 0.05 (the base level) with decreasing rate 0.005 at each time step (see Figure 2(b)). In this case, Table 2(b) shows that there is no response from MIX-E before time step 86.3, the upper bound of the time window given by MIX-O. Therefore, the controller generates the error command, ERROR, to the diagnoser. The diagnoser, DIAGN, checks the data associated with the discrepancy, such as phase in which it occurred, and its timing. It also gets sensor data from MIX-E. The expert-system-like model, MIX-D, concludes "the propeller is broken" by using the data from DIAGN. This is because the indicator shows that its timing is TOO-LATE and current phase is MIXING but the visual sensor shows that the concentration has not reached to

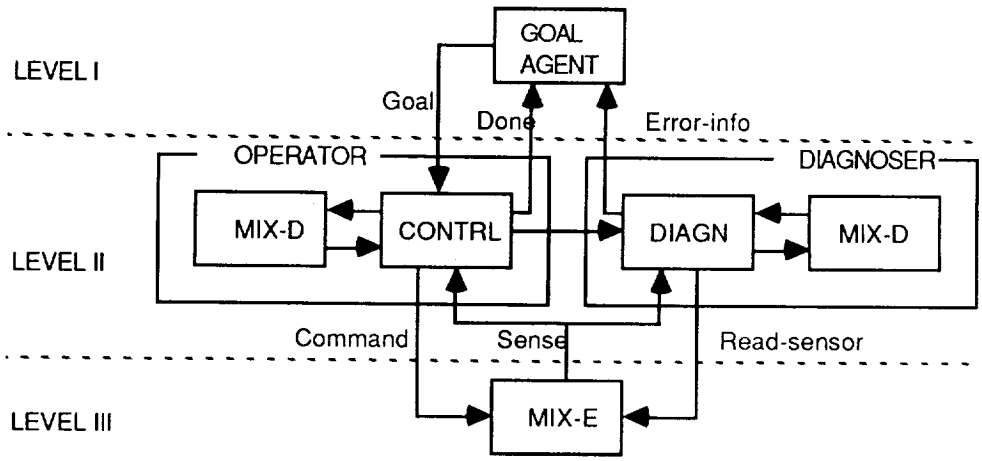



Figure 7: Block Realization of the Mixing Control System

| clock | MIX-E | | MIX-O | | CONTRL | |
|---|---|---|---|---|---|---|
| | state | output | state | output | state | output |
| 34.3 | (mixing 45) | () | (mixing) | (mix 45 7) | () | (mix) |
| 79.3 | () | () | () | () | (window 7) | () |
| 83.2 | (over-mixing 200) | () | (mixed) | (stop) | (check 1) | () |
| 84.2 | (mixed) | () | () | () | () | (stop) |

(a) Goal Plan : B-FULL -> MIXED (normal case)

| clock | MIX-E | | MIX-O | | CONTRL | | MIX-D | | DIAGN | |
|---|---|---|---|---|---|---|---|---|---|---|
| | state | output | state | output | state | output | state | output | state | output |
| 34.3 | (mixing 45) | () | (mixing) | (mix 45 7) | () | (mix) | () | () | () | () |
| 79.3 | () | () | () | () | (wind 7) | () | () | () | () | () |
| 86.3 | () | () | () | () | (error) | (too-late) | () | () | (start 1) | () |
| 87.3 | (reading 1) | () | () | () | () | () | () | () | (wait-sensor) | () |
| 88.3 | (reading 10) | (B-full) | () | () | () | () | () | () | (wait-sensor) | () |
| 98.3 | (visual 100) | (normal) | () | () | () | () | () | () | (wait-sensor) | () |
| 198.3 | (finish-read) | (vision-info) | () | () | () | () | () | () | (wait-sensor) | () |
| 199.3 | (passive) | (done) | () | () | () | () | () | () | (start-diagn) | (sensor-data) |
| 200.3 | () | () | () | () | () | () | (passive) | (propeller -broken) | (passive) | (propeller -broken) |

(b) Goal Plan : B-FULL -> MIXED ( error case )

Table 2: Simulation Results (partially shown)

its equibrium value.

## VII. Conclusions

This paper has shown how the space-adapted mixing control system is advantageously represented as discrete event models by employing techniques based on the DEVS formalism. Several fluid handling models have been successfully testing in the DEVS-Scheme environment. Suitably operating on the structure of such DEVS models provides a basis for design of event-based logic control. Since the DEVS formalism is at the heart of event-based control system design, such controllers can be readily checked by computer simulation prior to implementation. Thus the DEVS formalism plays the same role with respect to event-based control that differential and difference equation formalisms play to conventional control. This principle and the applicability of the DEVS-based control paradigm was illustrated here in the design of a fluid mixing system capable of supporting laboratory automation aboard a Space Station. The inclusion of event-based control units within robotic agents is discussed in (9, 10).

## Acknowledgement

This research was supported by NASA-Ames cooperative agreement No. NCC 2-525, "A Simulation Environment for Laboratory Management by Robot Organizations".

## References

(1) Kim, T. G., "A Knowledge-Based Environment for Hierarchical Modelling and Simulation," Ph.D. Thesis, Dept. of ECE, University of Arizona, May, 1988.

(2) Oren, T. I., "Taxonomy of simulation model processing," in Encyclopedia of Systems and Control, M. Signh, Eds. New York, NY : Pergamon Press.

(3) Saridis, G. N., "Knowledge Implementation: Structures of Intelligent Control System," in *Proc. IEEE International Symposium on Intelligent Control*, 1987, pp. 9 - 17.

(4) Zeigler, B. P., "DEVS Representation of Dynamical Systems : Event-based Intelligent Control," *IEEE proc.* Vol. 77, no. 1, Jan. 1989, pp. 72 - 80.

(5) Zeigler, B. P., Theory of Modelling and Simulation, New York, NY : Wiley, 1976 (reissued by Krieger Pub. Co., Malabar, FL, 1985).

(6) Zeigler, B. P., "System-theoretic representation of simulation models," *IIE Trans.*, Mar. 1984, pp. 19 - 34.

(7) Zeigler, B. P., "Hierarchical, modular discrete event modelling in an object oriented environment," *Simulation*, Vol. 49, no. 5, 1987, pp. 219 - 230.

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

(9) Zeigler, B. P., *Object-oriented Simulation with Hierarchical, Modular Models : Intelligent Agents and Endomorphic Systems*, Academic Press, 1990.

(10) Zeigler, B. P., Cellier, F. E., and Rozenblit, J. W., "Design of a simulation environment for laboratory management by robot organizations," *J. Intelligent and Robotic Systems*, Vol. 1, 1988, pp. 299 - 309.