

Rule Based Simulation Model for the Just-In-Time Production System

Abdelaziz ARAAR *

Abstract

We present a knowledge base simulation model (KBSM) to optimize and validate a just-in-time (JIT) production system. Answering "what if" questions and generating scenarios can be implemented in the "Kanban" production system. Also we present an automatic test based constraints to validate the results of simulation.

Key words: KBSM, JIT and Kanban systems, automatic test based constraints, and validation

1. INTRODUCTION

Simulations have been extensively used in many fields related to engineering and management science. According to the methodologies how the simulation models are build, we can classify them into physical, digital, analog simulations, or graphics animations. The majority of simulation models are build in digital computer simulations. Discrete event simulation is usually time consuming. Recently, there has been a great deal of interest in using hybrid models in the simulation process. Reddy, Fox and Husain (1986) presented a knowledge base simulation system to the distribution system. Manniyannan and Pegden (1990) developed a simulator bases-rule for medeling JIT production system (JITSAT). However, their system does not have automatique validation and cannot generate the best scenario. Nadoli and Biegel (1993) presented an Intelligent Manufacturing-Simulation Agents Tools (IMSAT) to model a manufacturing process organized with multiple levels. They introduced the artificil intelligence distributed. The system consists of multiple organization levels. The decision-making functions residing at different levels interact with each other during the operation of the system. Traditional simulation does not support the notion of hierarchical modeling. There is a lack of constructs that can encapsulate and model the decision-making functions. In this paper, we present knowledge base simulation model that can generate scenrios and to answer 'what if' questions in a production system using the just-in-time inventory policy. We try to introduce an automatic test based constraints for the validation.

2. SYSME JUST-IN-TIME PRODUCTION

Just-in-time production was initially developed and implemented by Japan's Toyota Motor Company in the 70's. Subsequently, the philosophy was adopted by other Japanese manufacturers, as well as companies in North America. To achieve a "plus-or-minus zero" performance-to-schedule variance, JIT requires the production of precisely the necessary units, in the necessary quantities, at the necessary time. JIT purports that time, efforts, and materials spent for something not needed now cannot not be utilized now, and therefore anything over the minimum amount necessary is wasteful. Producing an extra piece is just as bad as being one piece short. Ideally, the number of parts, components, etc., produced in a plant at any one time should be just enough to assemble one final unit. For example, in an automobile assembly operation, the right sub-assemblies should arrive at the assembly-line at the right station, at the time needed, and in the correct quantities.

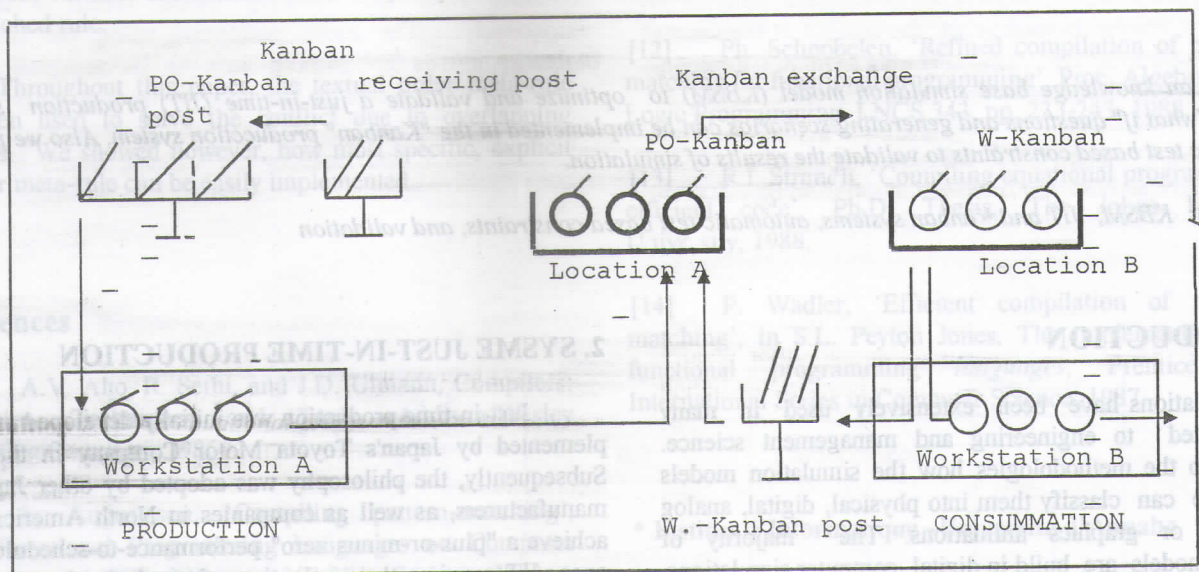
The word 'Kanban' in japanese means 'card'

There are three facets of Toyota's JIT system information system, production methods, and defects control (Autonomation). The information system is usually facilitated by the kanban system. Production methods that are adaptable to the JIT system involves three basic elements: the smoothing of production, the rationaldesign of processes, and the appropriat assignment of jobs. Autonomation refers to the autonomous check for defects. It is a built-in quality control mechanism to prevent mass production of defective work.

Kanban - An Information System

Although Toyota's JIT production system is commonly labeled the "kanban system", the kanban system is in fact only a sub-system -- the information system -- of JIT. There are two types of kanbans used: the withdrawal kanban and the production-ordering kanban. To illustrate the use of each type of kanbans, consider two adjacent (in the sense of job flow)

work stations: a preceding station A and a succeeding station B. A withdrawal kanban specifies the quantity required at station B (which are to be withdrawn from station A); while a production-ordering kanban orders the station A to replenish the parts withdrawn. Figure 1 illustrates the flow of parts and kanban in a JIT production system. [3],[4],[6],[8],[10],[13], et [18]



PO -- Production Ordering, W -- Withdraw, -- Kanban, and / -- Container ○

Figure 1: Le système Kanban

Kanban cards are attached to containers. All containers have the same size. When the accumulated number of withdrawal cards in the post of reception reaches a threshold then an exchange will occur. However, when the number of accumulated number production cards in the post of reception reaches another threshold then a production is set up. Usually we use the dynamic programming to optimize the two thresholds.

Because of changes in values and parameters, we need to construct a model which is able to answer questions and generating scenarios.

Changing thresholds changing scenarios

3. KNOWLEDGE BASE SIMULATION MODEL

The knowledge base simulation uses the process oriented concepts. Knowledge base simulation model (KBSM) is a collection of schema which are represented by a language. The language represents abstract and physical entities of the system [15]. The schema is the basic unit that represent objects, processes, ideas, etc. Each slot in schema acts as relation tying schema to the others. The schema may inherit slots and their values along these relations. Developing model in KBSM is the creation of schema that represent model's entities, including specifications for their event behavior and interconnection with other schema. The window-based schema editor SEDIT offers a facility to create or alter schema.

Model validation

A consistency constraint relating workstations may be specified as a Lisp expression. KBSM evaluates each constraint and reports whether it was satisfied. In case of constraint failure, the interpreter's trace facility helps determine the source of failure. We use base a specifications to help the validation process.

Model reduction

Complex simulation models can be simplified or reduced by automatic alterations if the model builder has created a knowledge framework for the task. The simplification makes models run faster, increases model understanding, simplifies analysis, and eliminates unnecessary details. Two techniques for simplifying a model, are static and dynamic.

Static model structure and parameters can be altered but not event behaviors

Dynamic model's dynamic process can be altered
redefining some of event behaviors
equivalent node aggregation

Static abstraction {
data class aggregation
analyzing stimulus event behavior
Dynamic abstraction {
statistics on event behavior

Model dynamics

The KBSM kernel interprets the model once it has been defined. KBSM uses the discrete event simulation (DES) approach which manages a calendar of events. A calendar is a set event notices ordered by execution time. In DES, the system changes state at discrete points in time. Running the model simply means executing the nodes which make queues in the calendar. The execution terminates when there are not events or until some criteria are encountered. For exemple, the event receive order is to occur on April 17, 1996 at 11 am focused : cund workstation B. To execut this event the simulation : lock is advanced by the system to 11 am on April 17 1996, the value of the slot receive order event of the schema workstation B is extracted, and each item found in the slot of the values is interpreted. The items in the list of values in the event slot may be representing a Lisp function. A rule or an instrument signed to collect data display some text or produce graphic mode effects.

-Events

Event behavior may be expressed as rules to be executed when the event occur. When interpreted, this rule implement the policy "If there is sufficient inventory to process this order, then schedule a transportation event and remove withdrawal cards". The order and the object are specified in the rule's parameters.

```
{{ receive-order-rule
```

```
INSTANCE: event-rule
```

```
IF: (something-in-inventory)
```

```
THEN: (schedule-transport)(remove withdrawal  
cards from containers and put them in the  
post)}}}
```

Simulation execution

At the end of each simulation run, we may fire a rule base that will suggest changes to the model for the next run. After running many of the goal-directed experiments, we hope to get scenarios that come close to verifying simulations goals. The KBSM experiment schema describes individual experiment. The run model changes is filled with a set of change specifications that describe the changes that need to be performed on the model before the experiment begins. There are also slots whose values reflect the experiment's status.

Runtime model management

KBSM supports the ability to conduct a series of experiments without humain intervention. The improvements on past experiment are automatically achieved by rules bases detailing the diagnosis and correction heuristics. To work correctly, the rules must allow for reasoning between several model scenarios. One can manage alternate scenarios by creating a context tree that devotes each context to scenario. Context in the schema representation language are hierarchically arranged so that if a schema being accessed is not found in the specified context, the context's ancestors are searched until the schema is found or the search fails. A shcema may only be created within some context, and the name of the schema must be unique within that context. Schema with the same name exist within different contexts and contain conflicting information.

Automatic analysis

One important task for a simulation analyst is to fine-tune the input parameters of a model to bring the values output variables within desired range. Rating a scenario measures the

goodness or the badness of simulation results. For example, if the goal is to eliminate shortages and if in a given simulation the average shortages is 10% of total orders, we would like better answer "NO" when asked, "have we reached the goal".

To rate a scenario smoothly, we choose a continuous scale of rating from -1 to +1. -1 means the results are far from the goal and +1 means the goal has been completely satisfied. Since goal are often complex and may consist of conflicting subgoals, we approach the specification of simulation goals as a composite set of constraints on the performance of various entities of the system modeled.

The steps in the construction and evaluation of goals are:

- Representing each organizational goal as a set of constraints
- Selecting and attaching instruments to gather data
- Specifying procedures for computing performance measures from raw data
- Executing the simulation for a given scenario
- Evaluating each constraint by computing a coefficient of constraint satisfaction, which may be positive to indicate the goal has been met or negative to indicate the goal has not been met
- Evaluating of scenario by computing a coefficient of goal satisfaction as a weighted average of constraint satisfaction coefficients

Consider a composite organizational goal to increase customer satisfaction while keeping the production rate low. This goal may be broken down into two subgoals.

Consider a organisational goal composed to increase the satisfaction in workstations B (WB) while the production in workstations A (WA) is low. This goal is divided in two sub-goals:

- (1) The shortage does not exceed 5% of demand of WA.
- (2) The cost of inventory and distribution per container does not exceed 10% of the cost of the production of this container.

The actual goal evaluation is an event scheduled to occur at some future date according to evaluation-schedule at which time is evaluated. A goal evaluation function will retrieve the contributing-constraints of the organizational goal. Each constraint is evaluated to compute a rating. This rating contributes to the overall rating of the organizational goal.

The example is not an actual rule. Currently knowledge base systems simulation diagnosis and correction rules are written in OPS5. The working memory on which these rules operate is created from schema representation language.

We consider an example where the satisfaction of WB is bad because it is estimated negatively (-0.75), but inventory and distribution cost is economic and estimated positively to (+0.80).

However, the calculation shows the combination of estimations which is negative and therefore unsatisfactory.

$$\text{Goal rating} = \text{weight average of individual constraint ratings} \\ = (0.3 * 0.8 + 0.7 * -0.75) / (0.3+0.7) = -0.285.$$

The constraints and goals are built manually but the collection of data and records are made automatically. When goals are complicated, they may be viewed graphically with KIVIAT chart. In KIVIAT chart, all parameters of performance are 'GOOD' when they assume large values are placed above the x-axis while the performance parameters that are 'BAD' when they are large are plotted below the x-axis.

Causal path analysis

The output from simulations is often analyzed by traditional methods as regression and correlation between variables in the model. However, we must determine the underlying causal structure before we can undertake the analysis.

The objective of path analysis is to detect causal relations embedded in a simulation model and exploit that knowledge to generate scenarios that realize organizational goals. Causal relationships between variables can be detected by postprocessing knowledge learned in introspecting simulation. Path analysis research handles both the qualitative and quantitative aspects of a rule based analysis of simulations.

A simple rule based analysis without path analysis is nothing more than automating the ad hoc approach to model simulation. Typical steps involved in the refinement of rules using path analysis are:

- A rule is proposed by the domain expert
- The causal assumptions the rule is based on are validated against the running model.
- Validation implies detection of causal chains in KBSM. The rule may need to be modified if the initial causal assumptions were either erroneous or insufficient.

Alternate causal structures are then proposed and automatically analyzed. The most appropriate causal structure is chosen and the results are summarized to yield equations reflecting sensitivity of the output parameters with respect to controllable input parameters. Thus, path analysis finds the degree to which each particular cause in the modeled system determines the variation of given effects.

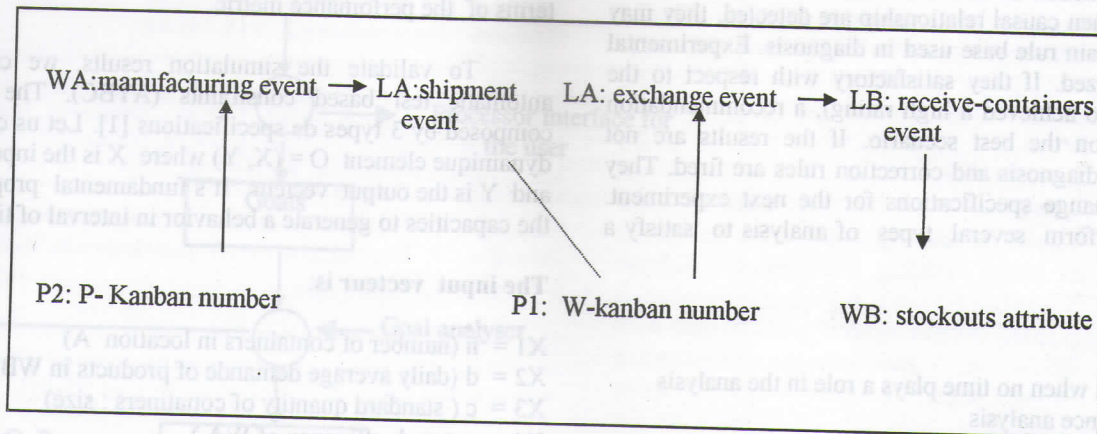
-The sensitivity information is used to quantitatively refine the rules.

As the example of the Kanban system.

IF the goal is to minimize shortages
and the average number of containers in location **B** is low
and shortages are high
and the threshold can be monitored
THEN decrease the threshold

To be specific, the rule must refer to actual instances of objects in the model. In the proposed model of the JIT system using Kanban in Figure 1. We concentrate in workstation B (WB), and the post of reception of W-Kanban (P1), the rule becomes:

IF the goal is to minimize WB : stockouts
and LB: average number of container is low
and WB: stockouts are high
and P1: the threshold can be altered
THEN decrease P1: threshold



Causal chain

WA:manufacturing event accesses attribute P2: P-kanban number. WA:manufacturing event causes shipment event. LA: exchange event accesses attribute P1: W-Kanban number. LA: exchange event causes receive containers event. If the number of containers is insufficient for that order, LB:receive

containers event records a stockout which affects the attribute WB:stockouts. Therefore, we assume that LB: receive containers event is causally connected to WB:stockouts. After series of experiments, we will know which hypothesis will be dropped. Possible refinement may be.

IF the goal is to keep WB:stockouts below 40%
and average number of containers < 30
and stockouts are in the range 40-50 %
and P1: threshold can be changed
THEN
$$P1: \text{threshold} = P1: \text{threshold} - C * (WB: \text{stockouts} - 40)$$

4 RULE BASE SIMULATION MODEL ARCHITECTURE

Simulation systems are essentially descriptive tools because they only describe the performance of a given scenario. They do not provide any clues or to what change should be proposed so that the simulation goals may be realized. The user interface processor receives a request from the user. If it is a simple request for information, the model database is accessed to retrieve the appropriate information. If it is a request for prescription (goal-oriented request), the request is analyzed by a rule based goal analyzer. This analysis may invoke an operations research tool, a specification to conduct a series of experiments or a specification to execute the simulation model in the learn mode to detect causal relationships. When causal relationships are detected, they may enhance the domain rule base used in diagnosis. Experimental results are analyzed. If they are satisfactory with respect to the goal (the scenario achieved a high rating), a recommendation is made based on the best scenario. If the results are not satisfactory, the diagnosis and correction rules are fired. They may generate change specifications for the next experiment. KBSM can perform several types of analysis to satisfy a specified goal.

-Static analysis

It is used when no time plays a role in the analysis

-Performance analysis

It is used when of the following is requested:

- * scenario rating
- * diagnosis
- * scenario generation
- * trade off

-Learning

If the objective of executing the simulation model is to detect embedded causal relationships, the model is the causal analysis module.

Mathematical analysis

If the goal analyzer determines that the current request can be satisfied by a mathematical analysis rather than by a simulation, it will invoke an appropriate tool. Analyses that may be performed by a system such as the one shown in figure

- Predictive analysis

This analysis deals with the problem of type « What is the effect of adding more containers in location A ». The solution is to change model parameters and structure, select

performance parameters, execute the model and report conclusions.

Diagnostic analysis-

This analysis deals with problem such as « What is the cause for delays the exchange of containers ». The solution is to use causal path analysis and response analysis.

Trade-off analysis

These address problem such as « To reduce the exchange of containers, what is the trade off of ». The solution is to generate 2 scenarios, evaluate a performance metric and print the trade off between the 2 alternatives in terms of the performance metric.

To validate the simulation results, we consider an automatic test based constraints (ATBC). The ATBC is composed by 3 types de specifications [1]. Let us consider the dynamique element $O = (X, Y)$ where X is the input vecteur and Y is the output vecteur. It's fundamental properties have the capacities to generate a behavior in interval of time

The input vecteur is:

- $X1 = n$ (number of containers in location A)
- $X2 = d$ (daily average demande of products in WB)
- $X3 = c$ (standard quantity of containers : size)
- $X4 = \alpha$ (work efficiency of WA)
- $X5 = ms$ (mean service time of containers in WA)
- $X6 = nr$ (threshold of the post reception of W-Kanban for the exchange)
- $X7 = np$ (threshold of the post reception of P-Kanban for setting up a production)
- $X8 = N$ (nombre de containers consumed by WB during the simulation)

The output vecteur is:

- $Y1 =$ waiting time of containers in WA
- $Y2 =$ number of exchanges of full and empty containers
- $Y3 =$ number maximal of stockouts
- $Y4 = p$ (average stockouts)
- $Y5 =$ number of times the production is set up
- $Y6 = td$ (simulation time)

The automatic test based constraints is composed of 3 types of specifications. The formal specifications are made for relations among the composants of X and Y . The qualitative specifications are made for causal relations among valeurs acqueried by composants of X and Y . However, the observable specifications require real observations of the systeme.

- $X8 = N \longrightarrow Y6 \geq N * X5$
- $X1 = 0 \longrightarrow Y2 = 0$
- $X7 = 1 \longrightarrow (Y3 = 0) \cap (Y4 = 0) \cap (Y5 \geq N)$
- $X5 = \varepsilon \longrightarrow Y1 = 0$

b) Change-in-values (increasing values from an experiment to another)

- $X1^{(2)} > X1^{(1)} \longrightarrow (Y3^{(2)} < Y3^{(1)}) \cap (Y4^{(2)} < Y4^{(1)})$
- $X6^{(2)} > X6^{(1)} \longrightarrow (Y4^{(2)} > Y4^{(1)}) \cap (Y3^{(2)} > Y3^{(1)}) \cap (Y2^{(2)} < Y2^{(1)})$
- $X7^{(2)} > X7^{(1)} \longrightarrow (Y5^{(2)} < Y5^{(1)}) \cap (Y2^{(2)} < Y2^{(1)}) \cap (Y1^{(2)} > Y1^{(1)})$

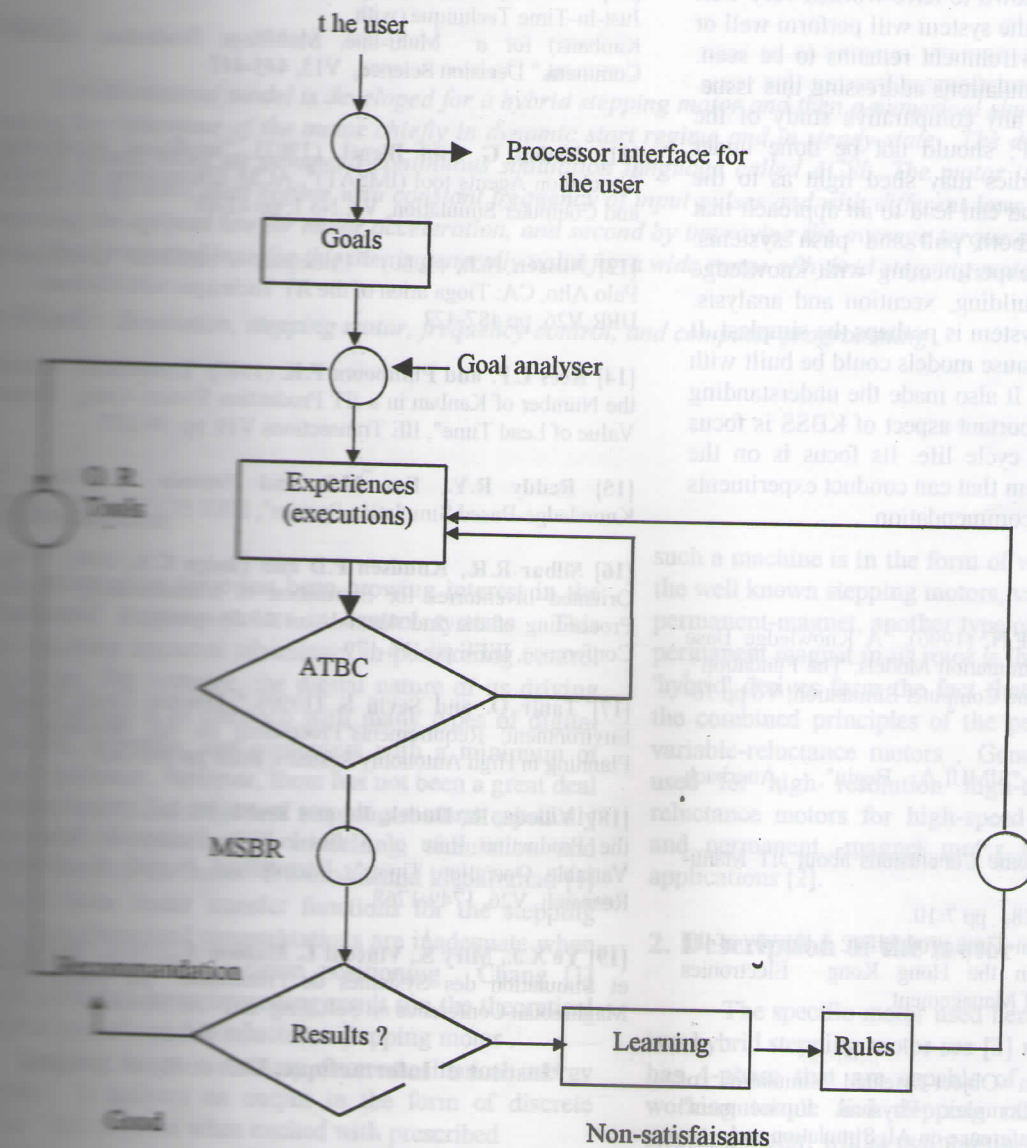


Figure 4 : Architecture KBSS

5. DESIGN AND RESULTS

Reddy, Fox and Husain (1986) presented a knowledge base simulation system to a distribution inventory system. They developed INET-pc system implemented in knowledge Craft. Mannivannan and Pegden (1990) developed a rule-based simulator for modelling just-in-time manufacturing systems (JITSAI). However, in their works answering what if questions are not implemented. We design a model for the Kanban system using the same approach de Reddy [14].

6. CONCLUSION

The kanban system is known to have worked very well for Toyota. However, whether the system will perform well or not in different culture and environment remains to be seen. There are various studies and simulations addressing this issue. However, it is our belief that any comparative study of the kanban systems vis-a-vis MRP, should not be done under identical settings. Further studies may shed light as to the applicability of each system, and can lead to an approach that combines the advantages of both pull and push systems. KBSS is used as testbed for experimenting with knowledge based techniques for model building, execution and analysis. The use of KBSS for kanban system is perhaps the simplest. It made model creation easier because models could be built with objects specific to the domain. It also made the understanding the model easier. The most important aspect of KBSS is focus on automating the simulation cycle life. Its focus is on the automated analysis expert system that can conduct experiments and rate scenarios to make a recommendation.

REFERENCES

- [1] Birta G.L., and Ozmirzak F.N, (1996), "A Knowledge Base Approach for the Validation of Simulation Models: The Foundation", ACM Transactions on Modeling and Computer Simulation, V6 pp 76-98
- [2] Birtwistle G. M., (1973), "SIMULA Begin", Auerbach Publishers Inc., Philadelphia, PA.
- [3] Burnham, J. M., (1987), "Some Conclusions about JIT Manufacturing", Production and Inventory Management, Vol. 28, pp 7-10.
- [4] Cheng, T., (1987), "The Just-In-Time production: A Survey of the Development and Perception in the Hong Kong Electronics Industry. International Journal of Management Science, V16, 25-32.
- [5] Craig D.B. (1991) " An Object-Oriented Simulation of Autonomous Agents in a Complex Physical Environment" Proceeding of the 2nd Annual Conference on AI, Simulation and Planning in High Autonomy Systems, IEEE pp 31-38
- [6] Davis, W. J., and Stubitz, S. J., (1987), "Configuring a Kanban System Using a Discrete Optimization of Multiple Stochastic Responses", International Journal of Production Research, V25, 721-740.
- [7] Fishman, G.S., (1978), "Principles of Discrete Event Simulation", Wiley.
- [8] Huang, P. Y., Rees, L. P., and Taylor III, B. W., (1983), "A Simulation Analysis of the Japanese Just-In-Time Technique (with Kanbans) for a Multi-line, Multistage Production System", Decision Science, V14, 326-343.
- [9] Mannivannan S. and Pedgen C. (1990), "A Rule-Based Simulator for Modelling Just-In-Time Manufacturing Systems (JITSAI)", Simulation V55, 2, pp 109-117
- [10] Monden, Y., (1984), "A simulation analysis of the Japanese Just-In-Time Technique (with Kanbans) for a Multi-line, Multistage Production System: A Comment." Decision Science, V15, 445-447.
- [11] Nadoli G. and Biegel, (1993), "Intelligent Manufacturing Simulation Agents tool (IMSAT)", ACM Transactions on Modeling and Computer Simulation, V3, No 1, pp 42-65
- [12] Nilsson N.J., (1980) " Principles of Artificial Intelligence " Palo Alto, CA: Tioga ation of the JIT Technique with Kanbans", IJPR V26, pp 487-472
- [14] Rees L.P. and Philipoom P.R. (1987) "Dynamically Adjusting the Number of Kanban in a JIT Production System Using Estimated Value of Lead Time", IIE Transactions V19, pp 99-207
- [15] Reddy R.Y., Fox M.S. and Hussain N. (1986), "The Knowledge-Based Simulation System", IEEE SOFTWARE, pp 27-37
- [16] Silbar R.R., Knudsen P.D and Hodge C.A.(1990) " Object-Oriented Inventories for Simulation of Manufacturing Processes" Proceeding of the 5nd Annual on AI System in Government Conference, IEEE pp 176-179
- [17] Tanir O. and Sevin S. (1990) " Discrete Event Simulation Environment: Requirements"Proceeding on AI, Simulation and Planning in High Autonomy Systems, IEEE pp 160-166
- [18] Villeda, R., Dudek, R., and Smith, M. L., (1988), "Increasing the Production Rate of a Just-In-Time Production System with Variable Operation Times", International Journal of Production Research, V26, 1749-1768.
- [19] Ye X.J., Miry S., Vincent L. Mathon A (1993), "Modelisation et Simulation des Systemes de Production par Objets s", 3rd Maghrebian Conference on Soft. Eng. and AI, Morocco

*Institut d' Informatique, University of Annaba,