

Synchronization Mechanisms for Integration of Distributed Manufacturing Simulation Systems

Susumu Fujii

Department of Computer and Systems Engineering
Kobe University,
Rokkodai, Nada, Kobe, Japan

Yasushi Kidani

NTT Human Interface Laboratories
Midori-cho, Musashino, Japan

Atsushi Ogita

Kansai Electric Power Co., Inc.
Shiokoji Karasuma-nishiiru, Shimogyo,
Kyoto, Japan

Toshiya Kaihara

Faculty of Information Science
University of Marketing and Distribution Sciences
Gakuen-nishi, Nishi, Kobe, Japan

In this paper, a virtual factory is proposed in the form of a distributed simulation model which provides a tool for performance evaluation of large and complex manufacturing systems. The virtual factory is constructed on a computer network developed for computer-integrated manufacturing (CIM). The relationship between the structure of a distributed simulation system and the model of a manufacturing system is described, and then basic requirements for the virtual factory are presented. Several Time Bucket methods, called Single-Phased Bucket, Double-Phased Bucket and Modified Double-Phased Bucket, are proposed as a synchronization mechanism to construct a distributed manufacturing simulation system, and their characteristics and effectiveness are discussed from the viewpoint of modeling a manufacturing system.

Keywords: Distributed manufacturing simulation, phased time bucket, synchronization, virtual factory, CIM

1. Introduction

As manufacturing systems became automated, so-called automation islands appeared in factories. Nowadays most manufacturing systems consist of several automation islands, such as direct numerical control systems (DNC), flexible manufacturing systems (FMS), automated cell systems, automated guided vehicle systems (AGV), etc. During this development, many simulation studies have been performed to obtain effective design of the automated systems. Some simulation systems developed for those studies were extended for use in operational decision making on the shop floor on which the new system was installed. This resulted in the advent of simulation islands, each corresponding to the real automation island in the factory.

Integrating automation islands has been attempted by connecting the computers in the islands via network information systems in order to integrate the information flow, and by connecting the storage in the islands by transportation systems to order to integrate the material flow. Presently, however, the simulation islands are left as they were. We consider the integration of simulation islands as a virtual manufacturing system or a virtual factory to utilize the potential effectiveness of simulation islands for overall improvement of the design and management of a factory [1, 2, 3].

This paper briefly reviews the necessary requirements to construct an integrated virtual factory, and

then considers constructing a virtual factory in a distributed computer network environment. We propose a synchronization mechanism for a distributed simulation system, called the Time Bucket mechanism, with some variations, such as the Single-Phased Bucket mechanism (SPB) and Double-Phased Bucket mechanism (DPB). Their properties and effectiveness are examined through experiments with a prototype simulator.

2. Requirements for a Virtual Factory

2.1 Virtual Factory Concept

In the design stage for a component in a new manufacturing system such as an FMS, system performance is evaluated during development. In a CIM environment, however, it is important to evaluate not only the performance of the component as a standalone system, but also its influence on neighboring systems and shops as a part of a total manufacturing system. The component will be utilized more effectively and efficiently as a subsystem of the total manufacturing system if appropriate adjustments and modifications pointed out by the overall simulation are made on neighboring subsystems and on the management system. Similarly, in making operational decisions on the shop floor, it is important to understand not only the local impact of the decision, but also its impact on neighboring shops [4, 5].

To cope with the above requirements in design and decision making, one of the most promising tools is a simulation system which can handle very large and complex manufacturing systems in some detail, much the same way as FMS's and AGV's have previously been modeled. A virtual manufacturing system may be considered as a software image of an integrated manufacturing system which can simulate both the design process of a product at the R&D level and the

manufacturing process at the shop floor. In this study, however, we confine ourselves to the simulation of the manufacturing process at the shop floor and refer to this as a virtual factory. In other words, we do not consider the simulation of the design and manufacturing process a particular product as is treated from the viewpoints of CAD/CAM/CAE.

2.2 Features of Virtual Factory

A virtual factory needs to satisfy the following features [6, 7, 8]:

1. The system at the factory level is a large-scale system in reality. The simulation system needs to model the total system in the same detail as it does subsystems.
2. The space covered by the system is spread widely over the factory or even over multiple factories. Each factory is made up of machining and assembly shops, warehouses, and transportation systems which are referred to as subsystems. A subsystem is in the ISO reference model.
3. The level of automation differs from shop to shop, and from machine to machine. The system needs to cope with such situations.
4. To the effort involved in model building, it is essential to provide the capability to extend the model gradually in its size and in its detail. Exchangeability of an old subsystem to a new one is also essential.

2.3 Simulation Development

To develop such a virtual factory, we can take two approaches: (1) to develop a completely new simulation system from scratch, or (2) to construct a system utilizing the existing simulation systems. The latter may be viewed as an approach to integrate the simulation islands in a factory. The second approach will be

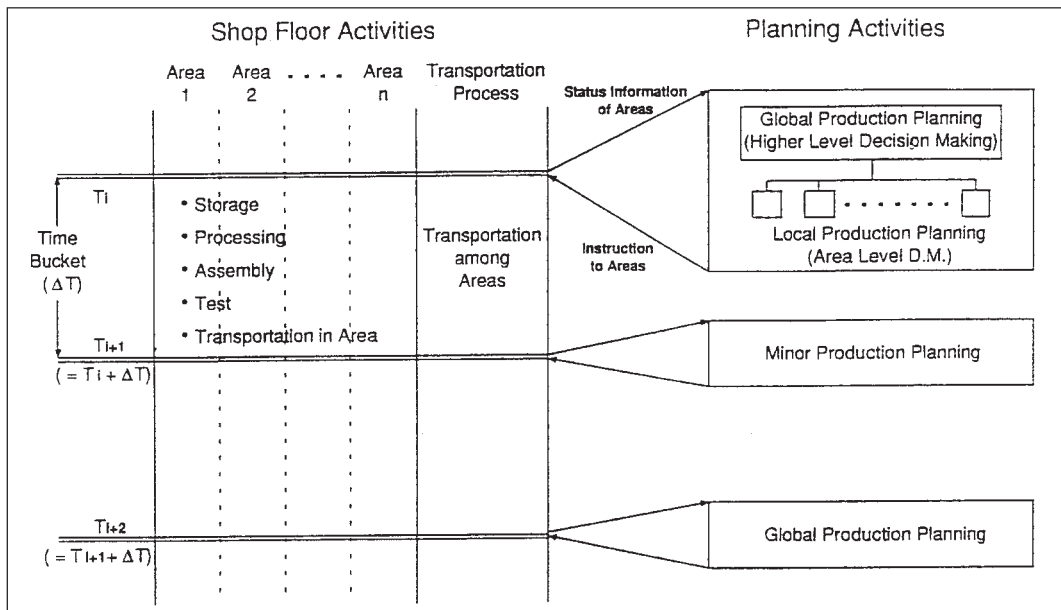


Figure 1. Shop floor activities

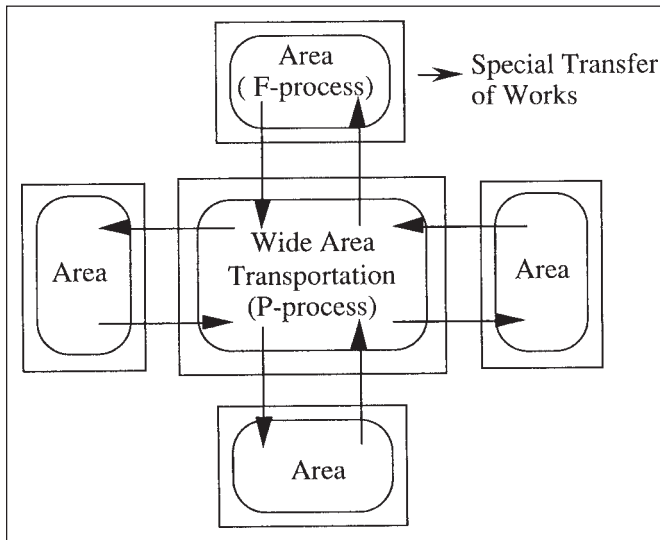


Figure 2. A factory with areas connected by a transportation system

effective in saving the effort of developing a total simulation model. In this paper, we concentrate the discussion on realizing the material flow or the physical distribution of products among subsystems. Other aspects such as information exchange for decision making, and the adjustments of statistics in the simulation are items for future research.

3. Structure of a Virtual Factory

3.1 Shop Floor Activities

Figure 1 shows the shop floor activities and planning activities. Areas in the figure could correspond to FMS's, machine shops, and storage areas. The transportation system is a system connecting the areas shown in Figure 2 that transports work and products from one area to the other. A global production plan is periodically generated based on the status information collected from areas. Local plans are made based on the global plan. The time interval for such periodic planning activities may correspond to the Time Bucket

in the MRP system. Minor planning activities, such as rescheduling, will also be done whenever necessary.

The structural features of a factory and the requirements for a factory-wide simulation system are mostly satisfied by a distributed simulation system when it is developed on a distributed computer system and installed as part of the CIM system infrastructure. Each subsystem at the area level can be modeled on one processor, and a transportation system connecting areas is modeled on another processor. Functions for information exchange among areas are also necessary to model the collection of data. One processor can also be allocated to modeling global decisions and collecting status reports of areas.

3.2 Constructing Steps of Virtual Factory

To construct a virtual factory, we consider the following procedure:

Step 1—Areas and transportation systems are modeled independently on computers (or processors) as simulation models of subsystems, Area i ($i = 1, 2 \dots$) and Transportation Processes, respectively. A Transportation Process may be abbreviated to T-Process in the following. In this step, the material flow within each subsystem is the primary focus of the model. The arrival of material from other subsystems and the departure of material from the subsystem to other subsystems are generally represented by the functions of source and sink nodes, respectively, in a simulation model as shown in Figure 3.

Step 2. Simulation models developed in Step 1 are connected by a computer network to be integrated into a total simulation model. In order to express the global material flow among subsystems, source and sink nodes in a simulation model of a subsystem are connected by specially designed messages transmitted through the computer network system to appropriate sink and source nodes of other models, as shown in Figure 4. In reality, any transportation among areas is

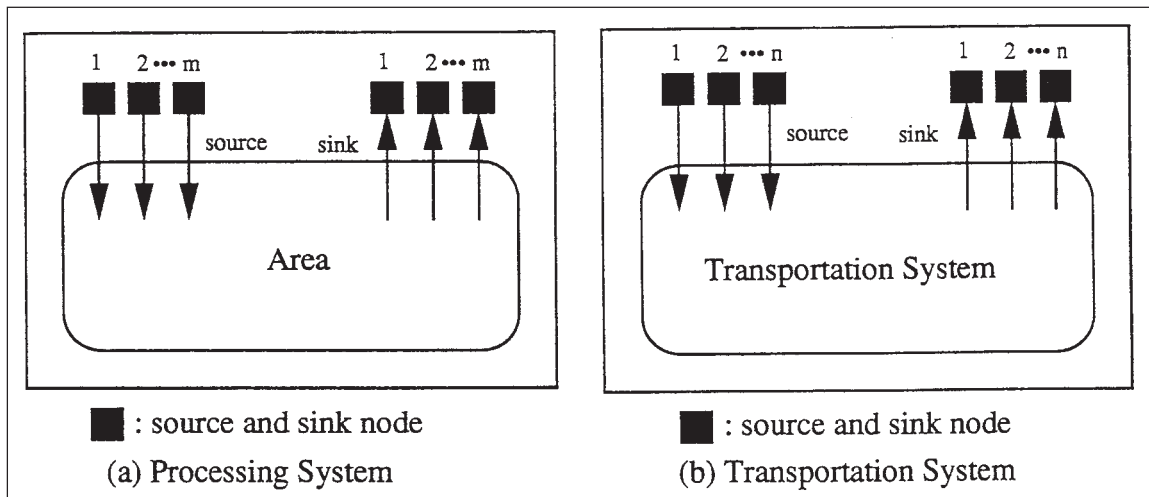


Figure 3. Simulation models of area and T-Process

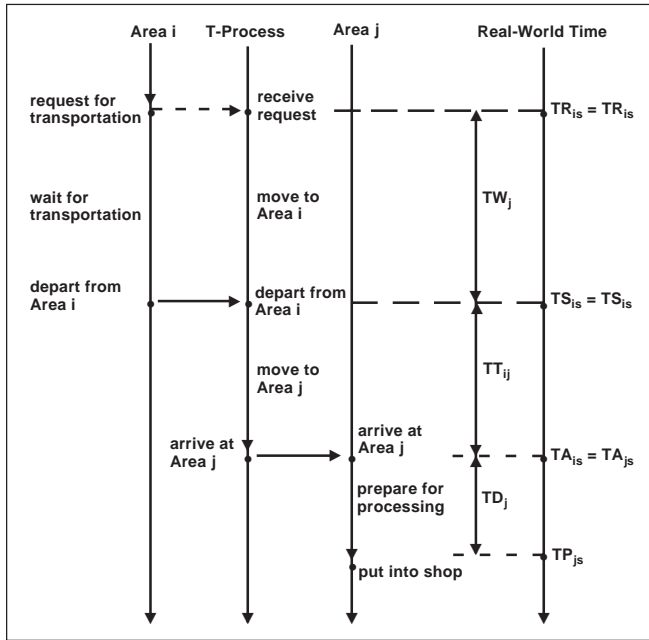


Figure 4. Global material flow from Area i to Area j

made by transportation systems, but in the simulation, the transportation process may not be modeled explicitly, as will be described later.

Step 3. In addition to exchanging messages for material flow, special capabilities are to be added that synchronize the processing of simulation models.

Step 1 is a stage to develop conventional simulation models which can be used for the simulation study of standalone systems. Steps 2 and 3 require modification to the models in Step 1 to effect a distributed simulation system. Therefore, the degree and/or the difficulty required for modification of models will be the key to obtaining the virtual factory.

4. Global Material Flow

4.1 Chronological Features of Material Flow

To construct a total simulation system, we first examine a global material flow in a factory as shown in Figure 2. Figure 4 shows a hypothetical flow of material, W_s , from Area i to Area j through the Transportation Process, with the following chronological features:

- $TR_{is} = TR_{ts}$
 W_s becomes ready to be sent to Area j at time TR_{is} and a request for transportation is transmitted to a T-Process. We assume the request is received by the T-Process at time TR_{ts} with no delay; hence, $TR_{is} = TR_{ts}$.
- $TS_{is} = TS_{ts}$
 A vehicle assigned to service the request moves to Area i and arrives at time TS_{ts} . The vehicle is loaded and starts to move to Area j at time TS_{is} . We assume that loading is instantaneous; hence, $TS_{ts} = TS_{is}$.

- $TA_{ts} = TA_{js}$
 The vehicle arrives at Area j and instantaneously unloads W_s , hence $TA_{ts} = TA_{js}$.
- TP_{js}
 W_s undergoes processing in the loading/unloading area of Area j and becomes ready to put into the shop at time TP_{js} .
- $TW_{is} = TS_{is} - TR_{is} = TS_{ts} - TR_{ts}$
 TW_{is} is the Waiting Time of W_s at Area i for transportation or equivalently, the time a vehicle requires to move to Area i from its position at TR_{ts} .
- $TT_{ij} = TA_{js} - TS_{is} = TA_{ts} - TS_{ts}$
 Time required for W_s (or the transporting vehicle) to move from Area i to Area j.
- $TD_{js} = TP_{js} - TA_{js}$
 Time Delay of W_s in the loading/unloading area of Area j.

In the conventional simulation on a single processor, all events are processed one by one in chronological order, resulting in an accurate representation of the above material flow. In a distributed simulation on multiple processors, however, a special device, named a synchronization mechanism, is necessary to synchronize the timing of sending/receiving operations of a request and loading/unloading operations of material, since processors simulating Area i, Area j and T-Process are different in their computations.

This observation indicates the importance of distinguishing three different times or clocks in the distributed simulation: real-world time on the clock in the real world, or RWT; local virtual time on a clock in the simulation of Area i (T-Process), or LVTi(.); and global virtual time on a clock of the total simulation, or GVT(.), where '.' stands for any type of time. In the following, we consider synchronization mechanisms to construct a total simulation system, i.e., a virtual factory, as a distributed simulation

4.2 Basic Assumptions for Modeling

Basic assumptions for modeling a virtual factory are described based on the discussion in the above and the observation of real factories and shop floor activities such as shown in Figure 1.

1. A shop, i.e., Area i, is mostly operated independently for a certain period, such as a shift, or a half of a shift in the morning or in the afternoon. In the period, materials in the shop at the beginning of the period are processed. The processing stops until new materials that arrived at the shop at TA_{is} become ready for processing at TP_{is} , if the materials are completely consumed in the period, if the number of materials at the beginning of the period is sufficiently large and if the length of the period is sufficient
2. The time required for transportation from one shop to the other, TT_{ij} , could be relatively long when the

factory is large and the size of a shop is set to be large, as assumed in item 1. This suggests that the following assumption is acceptable: the materials completed at Area *i* in a period will not be used in another shop, e.g., Area *j*, in the same period even if material becomes ready for processing.

3. The capacity of the loading/unloading area at a shop is large, and no blocking of material flow will occur due to the shortage of the space in which to keep the materials.

The period in item 1 is named as a Time Bucket in this study and following terms are defined:

- BT
Bucket Time (time length of a Time Bucket)
- TB_k
k-th Time Bucket in the simulation run
- STB_k, ETB_k
Times in GVT at the beginning of and at the end of *k*-th Time Bucket, TB_k , respectively. $STB_{k+1} = ETB_k = STB_k + BT$

4.3 Material Flow and Synchronization Mechanism

When a subsystem such as Area or T-Process is modeled in one simulation system, the arrival at and departure from the subsystem are represented by generating and absorbing nodes of transactions, often called Source and Sink nodes in simulation languages. In constructing a virtual factory as a distributed simulation system, these nodes in a model of a subsystem are connected, as in Figure 5. We need to notice the different nature of these nodes when subsystems are connected. Arrival of material will occur at any time;

therefore, an accurate realization of material flow depends on the method to represent the arrival of materials. In a distributed simulation, material flow may be realized by transmitting a message generating an arrival event in the subsystem of destination, and a synchronization mechanism which can respond to the material flow is essential to process the arrival event in a chronological order with no conflict in the subsystem. The conservative and Time Warp mechanisms are proposed for synchronization [9] and are widely used [10]. The Time Warp mechanism, however, requires state saving and exchanging functions for rollback operations. In simulation languages commercially available, a state-saving function is provided to restart a new simulation run from the state saved in other runs using a state exchanging function. This means that these functions in presently available commercial simulation languages are not applicable to the rollback operation, and that a distributed simulation system with a Time Warp mechanism has to be developed by a general programming language.

5. Time Bucket Mechanism

As suggested above, existing approaches for dealing with synchronization in distributed simulation (e.g., Time Warp) are not well suited to the needs of a CIM simulation system (i.e., virtual factory). The adaptation of existing programs for the development of the virtual factory has substantial advantages.

A Time Bucket mechanism was proposed by the authors [11, 12] as a mechanism for synchronization which circumvents the drawbacks described in the preceding section. The method is based primarily on observations of real manufacturing systems.

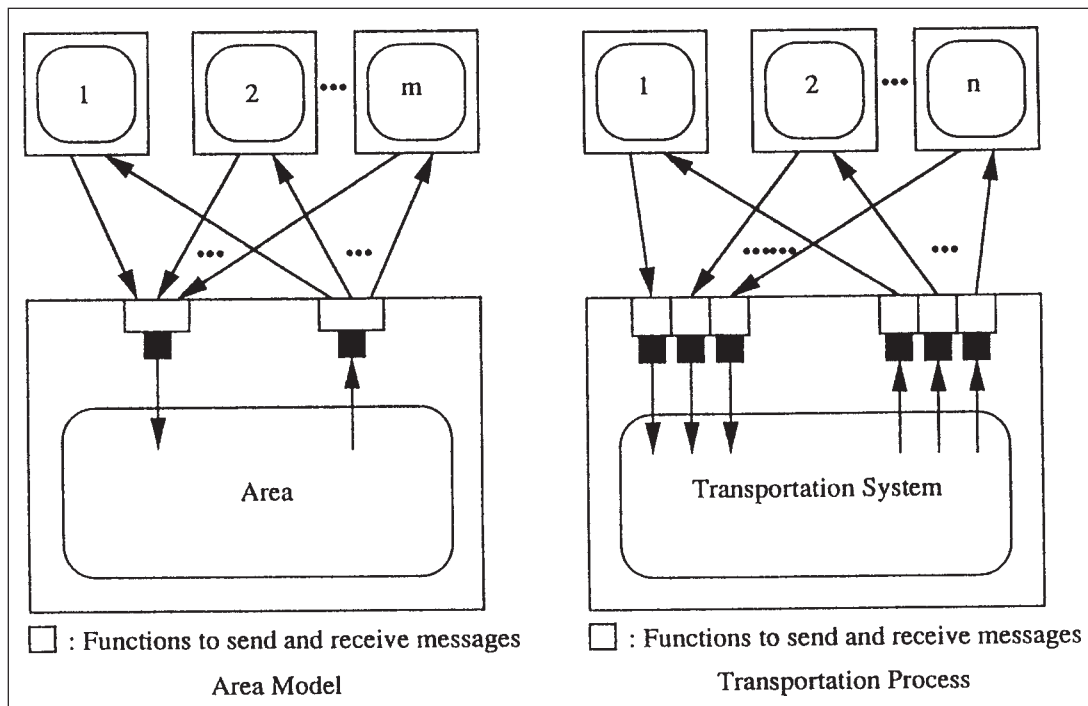


Figure 5. Connection of subsystems

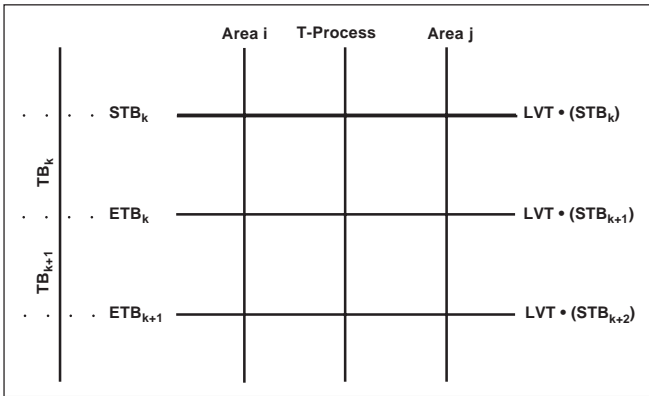


Figure 6. Basic concept of the Time Bucket mechanism

Steinman's Breathing Time Warp approach [13,14] shares common features with the Time Bucket method presented in this study. Event management in the Breathing Time Warp algorithm, however, is too strict to attain effective simulation for a virtual factory. The bucket size should be defined such that it is the minimum time interval between all the events. The algorithm might cause the overconsumption of memory and CPU for a virtual factory.

Furthermore, unlike Breathing Time Warp, our approach is specific to the virtual factory, which has known time delays for the transport of physical work between areas. We take advantage of this feature in our proposal of a new Time Bucket algorithm which is better tailored to the virtual factory.

5.1 Basic Concept

The Time Bucket mechanism in this study is schematically shown in Figure 6. Simulation runs of subsystems, i.e., Areas and T-Process, in k th Time Bucket, TB_k , start at $STB_k (= LVT.(STB_k))$ and stop at $LVT.(ETB_k)$. Since the simulation contents of subsystems are different from one another, the run lengths in Real-World Time (RWT) are different. When the latest run is completed, e.g., Area j with $RWT(LVT_j (ETB_k))$, $k + 1$ st Time Bucket, TB_{k+1} , starts. This means a processor stopped at $RWT(LVT.(ETB_k))$ will wait until

$RWT(ETB_k)$ before starting the new bucket.

The time chart is conceptually simple, but the material flow among subsystems requires some modifications. The modification depends on how the modeling of T-Process is carried out. This, in turn, affects the accuracy of the simulation results and the nature of the required modifications of existing simulation models of subsystems. In a virtual factory, the T-Process can be modeled either implicitly or explicitly according to the requirements of the simulation study. In the former case we refer to the Time Bucket mechanism as simple and in the latter as normal.

5.2 Simple Time Bucket Mechanism

When the T-Process is not explicitly modeled (e.g., as in Figure 7(a)), material flow in Figure 5 can be implicitly modeled by sending a message with a predetermined arrival time of the material at the destination area at time ETB_k , as in Figure 7(b). The arrival of material is realized by registering an arrival event in the event list of the area. Since the known arrival time TP_{js} is not communicated to Area j until ETB_k , it is necessary to assume that material W_s , which becomes ready for transportation from Area i at TS_{is} in TB_k , cannot be used in Area j even if it arrives there and is ready for processing at TP_{js} in TB_k . In other words, if both TR_{is} and TP_{js} are in the same Time Bucket, W_s can only be used in Area j in TB_{k+1} or later. The simplest case of the model corresponds to $TR_{is} = TP_{js}$, which neglects the transportation time.

5.3 Normal Time Bucket Mechanism

If the T-Process is explicitly modeled as shown in Figure 8, we need to devise a special mechanism to represent the transportation accurately. To simplify the presentation here, we assume that $TR_{is} = TS_{is}$, or equivalently, $TW_i = 0$, and $TA_{ts} = TA_{js} = TP_{js}$ or $TD_j = 0$. Possible mechanisms for representing the transportation accurately are: the conservative mechanism with Null Messages [9], the Time Wheel mechanism [15], the Time Warp mechanism [16], and the Time Bucket with Time Warp [3] (see Figure 9). With

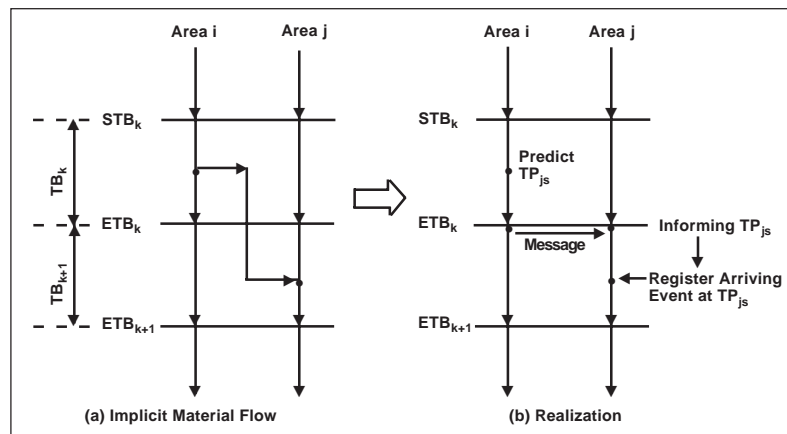


Figure 7. Simple Time Bucket mechanism

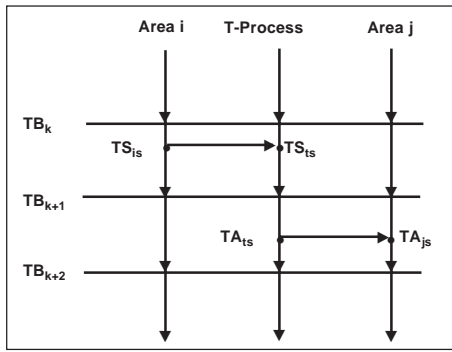


Figure 8. Normal Time Bucket Mechanism

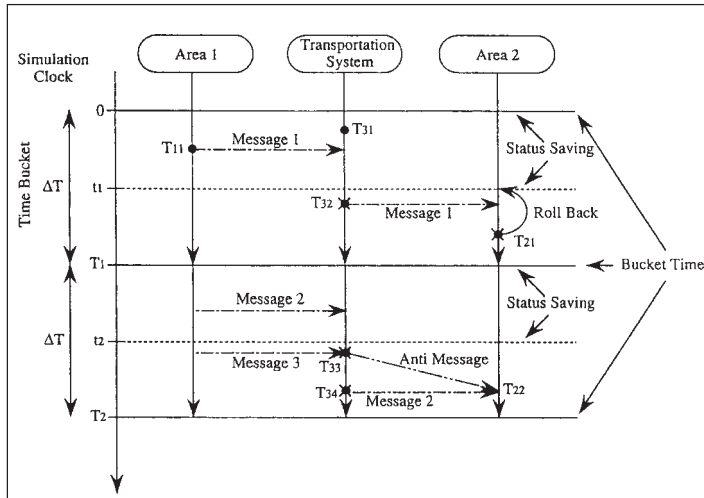


Figure 9. Time Bucket Mechanism with Time Warp

the Time bucket with Time Warp approach, the simulation run stops at the end of a Time Bucket, and other operations are the same as with the Time Warp mechanism. As previously noted, the Time Warp mechanism is generally not applicable to simulation models developed in commercial simulation languages. Other mechanisms may also be applicable, but may not be computationally efficient. In the following section we propose some variations of the normal Time Bucket mechanism which can be applied in the situation under study. These may, however, introduce small inaccuracies in the simulation results.

5.4 Phased Time Bucket Mechanism

The Time Bucket Mechanism has shortcomings caused by the concurrent processing of all simulation sub-models, including T-Processes. In Figure 4, Area i sends a transportation request message to a T-Process. In reality, TR_{is} should equal TR_{ts} , but the Time Bucket Mechanism does not permit the T-Process to receive the appropriate message, because a T-Process can receive messages only at the end of a Time Bucket. As a result, the erroneous interval between TR_{is} and TR_{ts} , ΔT , can influence the simulation results.

To deal with this problem, we propose the Phased Bucket Mechanism (PB), i.e., a modified Time Bucket

Mechanism which models the transportation system as a T-Process. We propose two types of PB mechanisms. The first is the Single-Phased Bucket Mechanism (SPB). In the SPB algorithm, simulation processing during a Time Bucket is divided into two phases: namely, an Area simulation processing phase, and a transportation simulation phase. These are executed alternately. Since a T-Process starts its processing of one Time Bucket after all Area simulators come to the end of the Time Bucket, it can receive transportation request messages appropriately, as shown in Figure 10.

Step 1: Area simulators execute their simulation processing for TB_k independently and stop their processing at the end of TB_k . They then send transportation messages to a T-Process.

Step 2: After the T-Process receives messages from all Area simulators and registers all times for transportation requests in its event list, it starts execution of its simulation processing for TB_k .

Step 3: When the T-Process stops its processing at the end of TB_k , it sends work arrival messages to Area simulators.

Step 4: When the Area simulators receive messages, they modify their status and statistical data in TB_k .

Step 5: Loop to Step 1.

Via the SPB, material which arrives at Areas during TB_k is reported at the end of TB_k , and assumption (3) is required. This assumption can be avoided by using the alternate Double-Phased Bucket Mechanism (DPB).

The DPB requires the state-saving function and loading function in a T-Process for rollback operation. The difference between SPB and DPB is the transportation simulation phase; that is, the performance of T-Process. In the DPB algorithm, the T-Process which received messages from Area simulators executes its TB_k processing for two Time Bucket periods, TB_k and TB_{k+1} , and then stops. After message exchange among simulators, the T-Process rolls back to its status at the end of TB_k . That is, before Area simulators start their

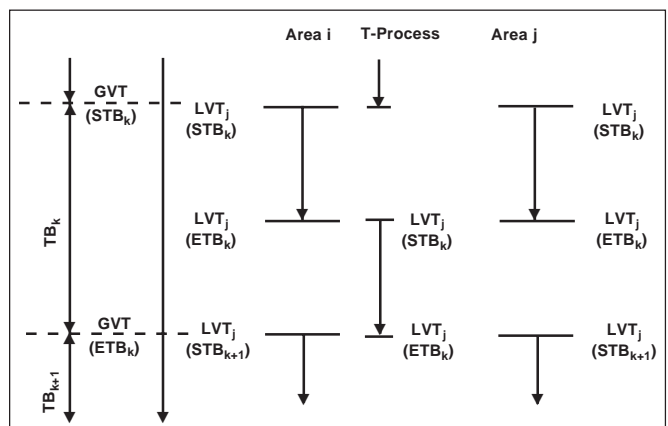


Figure 10. Single-Phased Bucket Mechanism

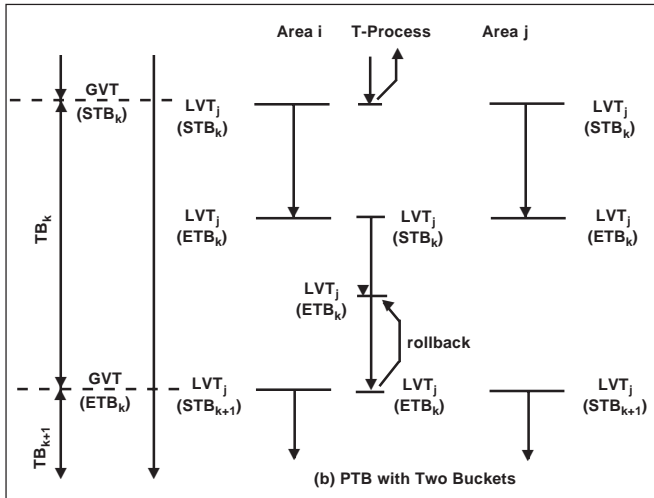


Figure 11. Double-Phased Bucket Mechanism

simulation processing for TB_{k+1} , the T-Process predicts the time of all work arrivals to Area simulators so that Area simulators can register the time of arrival events.

Note that an assumption for DPB is that the duties assigned to vehicles in a T-Process are not changed until the duty is completed.

Step 1: Each Area simulator executes the simulation processing for TB_k independently and stops its processing at the end of TB_k . They then send transportation messages to a T-Process.

Step 2: After the T-Process receives messages from all Area simulators, it starts execution of simulation processing for TB_k and TB_{k+1} . At the end of TB_k , the T-Process saves its state for rollback operation and restarts its execution in TB_{k+1} . When it stops its processing at the end of TB_{k+1} , arrival messages of work in TB_{k+1} are sent to Area simulators.

Step 3: The T-Process rolls back to the end of TB_k with the state loading function.

Step 4: Loop to Step 1.

6. Evaluation of SPB/DPB

6.1 Prototype System

In order to obtain some practical insight into the effectiveness of the concepts outlined above, we constructed a prototype distributed manufacturing simulator. Its general configuration is shown in Figure 12. The following should be noted: the overall simulation model consists of several Area simulators and a T-Process. The T-Process contains the synchronization control function called the master process.

Some Area simulators are built with SLAM II [17], a Fortran-based general-purpose simulation language, while the T-Process and the remaining Area simulators are built with SMPL [18], which is a C-based discrete-event simulation language.

All simulators include an interface program for message exchange and synchronization.

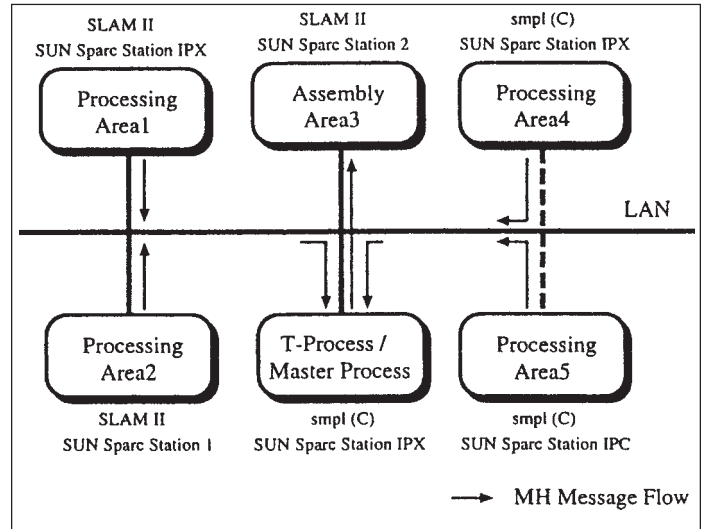


Figure 12. The prototype system

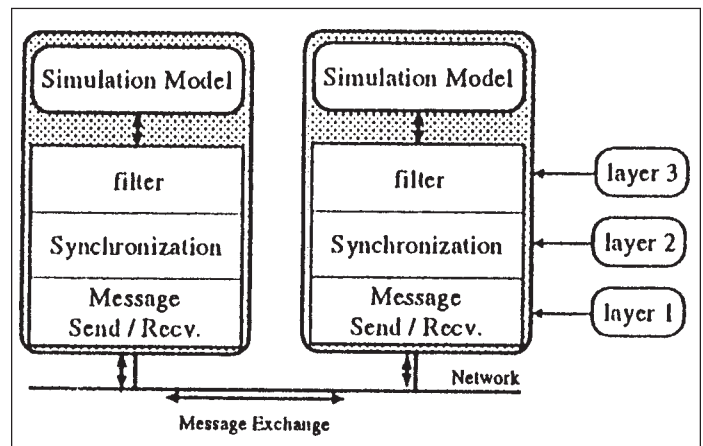


Figure 13. Interface program structure

Each simulation sub-model was first developed as a standalone simulator and was then connected into the overall model with the interface program as shown in Figure 13. Layer 1 in Figure 13 has communication functions for sending and receiving messages and the synchronization mechanism is in Layer 2. Layer 3 provides a filter, which exchanges data formats between messages and the simulators.

6.2 Experiments

Various experiments were carried out to confirm the validity of the SPB/DPB approaches and to evaluate their efficiency. The conditions for these experiments are summarized in Table 1.

Figure 14 shows the execution times for SPB and DPB for the case where the number of Areas is 5. This

Table 1. The experimental conditions

Time	10000 (Sim. Clock)
Bucket Size	2-100
Number of Areas	3-5

figure clearly indicates that the required processing time decreases as the bucket size becomes larger.

Various relevant times are given as follows:

$$\begin{aligned} \text{SPT} &= \text{AT} + \text{STT} + \text{MT} \\ \text{DPT} &= \text{AT} + \text{DTT} + \text{RBT} * \text{RN} + \text{MT} \\ \text{STT} &= \text{EN} * \text{ET} \\ \text{DTT} &= (\text{EN} + \text{REN}) * \text{ET} \end{aligned}$$

where:

- SPT,DPT processing time in SPB, DPB
- AT processing time of Area simulators
- STT,DTT processing time of T-Process in SPB, DPB
- EN number of events
- ET average processing time per an event
- RBT processing time for one rollback operation
- REN number of events caused by rollback
- RN number of rollbacks
- BN number of Time Buckets
- MT time required for message exchange

The values for EN and REN as obtained from the experiments are shown in Table 2, and the measured value of RBT was 132.1 milliseconds.

Figure 14 shows that DPB requires about 10 times the processing time of SPB. This is because of the number of rollback operations, each of which requires a

Table 2. The number of events

Area	3	4	5
EN	1007	2205	3402
REN	690	3847	5986

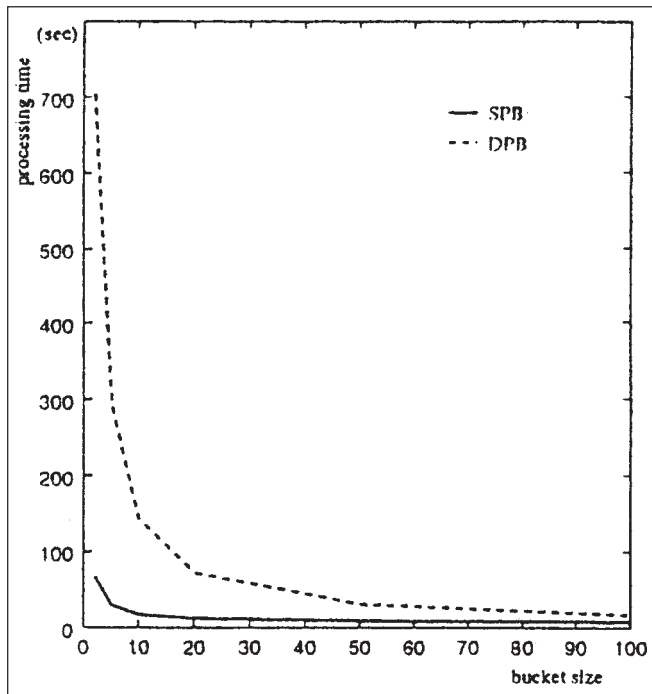


Figure 14. Processing time of SPB/DPB

time of RBT. Furthermore, the number of rollback operations increases as the bucket size decreases, and consequently, with DPB, shorter processing time can be achieved by choosing larger bucket sizes.

6.3 Modified DPB

With the DPB algorithm, a rollback operation is always required at the end of each Time Bucket. But if no arrival event occurs in the second phase of the T-Process, then a rollback operation of the T-Process is not necessary because the T-Process does not send any arrival message to Area simulators. In such a case, rollback operations can be avoided, thereby reducing processing time.

The Modified DPB (MDPB) algorithm is based on the above observation. It can be formulated from the DPB algorithm by deleting Step 3 and replacing Step 2 with:

Step 2: If T-Process receives messages from Area simulators, T-Process rolls back and starts execution of a simulation processing of TB_k and TB_{k+1} . If T-Process receives no message, T-Process does not roll back and start its simulation processing of TB_k .

The processing time of MDPB compared with SPB/DPB is shown in Figure 15. Figure 15 shows that MDPB provides shorter processing time in the case where Bucket Size (BS) is relatively small (i.e., the number of buckets is large). That is because there are fewer roll backs in the MDPB algorithm, while the number of roll backs in the DPB algorithm is equal to the number of Time Buckets. The relation between modeling and some extended Time Bucket Mechanisms is summarized in Table 3.

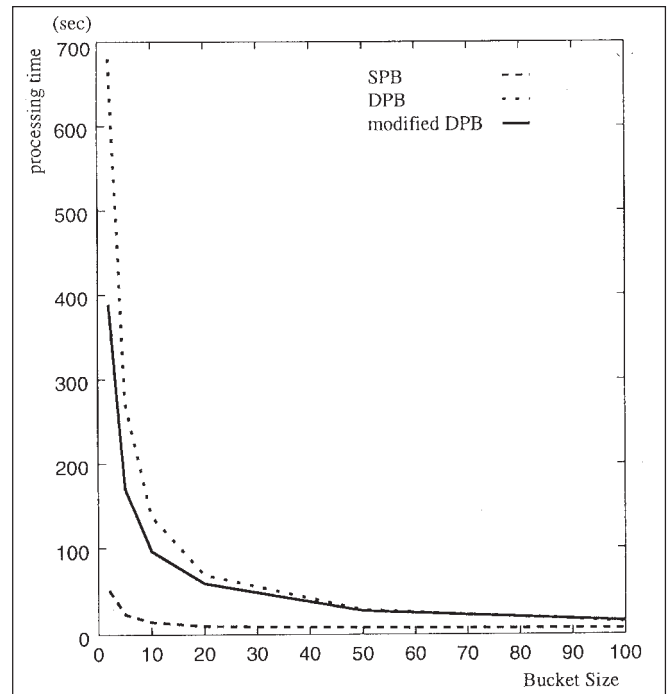


Figure 15. Processing time of MDPB

Table 3. The relation between modeling and various TB mechanisms

Type	Assumptions			T-Process	BS
	(1)	(2)	(3)		
Simple TB	○	○	○	×	-
SPB	○	○	○	○	-
DPB	○	×	×	○	large
MDPB	○	×	×	○	small

7. Conclusion

In this study a distributed simulation model for virtual manufacturing has been proposed. Such a model has substantial value in performance evaluation of a proposed new system and/or in operational decision making at the area level.

Several synchronization mechanisms for a distributed simulation have been proposed. These are all variations on a Time Bucket mechanism; namely, Single/Double/Modified Double-Phased Time Bucket. They have been compared from the point of view of their effectiveness in modeling the global material flow. This was achieved by examining the behavior of a prototype simulator which was constructed to study the effectiveness of the synchronization mechanisms proposed in this study.

Our results indicate that when a comprehensive simulation model is constructed by integrating simulation sub-models, each developed in a general-purpose simulation language, the Phased-Time Bucket Mechanisms can be effectively used to represent the distributed material flow. Care must be taken, however, in appropriately selecting the bucket time.

8. References

- [1] Fujii, S., et al. "A Study on Distributed Simulation for Flexible Manufacturing Systems, Information Control Problems in Manufacturing Technology." Proceedings of the 6th IFAC/IFORS/IMACS Symposium, pp 27-32, Pergamon Press, 1989.
- [2] Fujii, S., Hirano, M. "A Basic Study on Distributed Simulation Model of Virtual Manufacturing System under CIM Environment." Proceedings of the International Conference on Production Research 1993, pp 281-282, Elsevier Science Publ., 1993.
- [3] Fujii, S., et al. "A Basic Study on a Distributed Simulation for Virtual Manufacturing." Proceedings of New Directions in Simulation for Manufacturing and Communications (SIM 94), pp 277-281, 1994.
- [4] Fujii, S., et al. "A Study on Distributed Simulation System for Design and Operation of a Large Manufacturing System." Proceedings of the 1994 Japan-USA Symposium on Flexible Automation, pp 769-772, 1994.
- [5] Fujii, S., Sugimura, N., et al. "A Study on Distributed Simulator for Large-scale Discrete Manufacturing Systems." Proceedings of the 1994 Japan-USA Symposium on Flexible Automation, pp 1351-1354, 1994.
- [6] Kaihara, T., Besant, C.B. "Object-Oriented Flexible and Integrated Manufacturing System Modelling." 1993 CompEuro Proceedings, pp 312-319, 1993.
- [7] Kaihara, T., Besant, C.B. "Manufacturing System Modelling Structure based on Object Oriented Paradigm." Proceedings of the 1993 Summer Simulation Conference, pp 831-836, 1993.
- [8] Fujii S., et al. "Distributed Simulation Model for Computer Integrated Manufacturing." Proceedings of the 1994 Winter Simulation Conference, pp 946-953, 1994.
- [9] Chandy, K.M., Misra, J. "Distributed Simulation: A Case Study in Design and Verification of Distributed Programs." IEEE Transactions on Software Engineering, Vol. 5, pp 440-452, 1979.
- [10] Fujimoto, R.M. "Parallel Discrete Event Simulation." Communications of the ACM, Vol. 33, pp 31-53, 1990.
- [11] Fujii S., et al. "Synchronization Mechanisms for Integration of Distributed Manufacturing Simulation Systems under CIM Environment." Proceedings of Advances in Intelligent Computer Integrated Manufacturing System, pp 268-274, 1994.
- [12] Fujii, S., et al. "A Study on a Distributed Simulation for the Evaluation of Large Manufacturing Systems." S.M.Wu Symposium, Vol. 2, pp 7-12, 1996.
- [13] Steinman, J.S. "SPEEDES: A Unified Approach To Parallel Simulation." Proceedings of the 1992 Workshop on Parallel and Distributed Simulation, pp 75-83, 1992.
- [14] Steinman, J.S. "Breathing Time Warp." Proceedings of the 1993 Workshop on Parallel and Distributed Simulation, pp 109-117, 1993.
- [15] Agrawal, P. "Concurrency and Communication on Hardware Simulators." IEEE Transactions on Computer-Aided Design, CAD-5, pp 617-623, 1986.
- [16] Jefferson, D.R. "Virtual Time." ACM Transactions on Programming Languages and Systems, Vol. 7, pp 404-424, 1985.
- [17] Pritsker, A.B., et al. SLAM II Network Models for Decision Support, Prentice-Hall, Inc., 1989.
- [18] MacDougall, M.H. Simulating Computer Systems: Techniques and Tools, The Massachusetts Institute of Technology, 1987.



Susumu Fujii is a Professor of Computer and Systems Engineering at Kobe University, Kobe, Japan. He received a BS in Mechanical Engineering and an MS in Precision Engineering from Kyoto University, and a PhD in Mechanical Engineering from the University of Wisconsin. His research interests include modeling and analysis of computer integrated manufacturing systems, production planning and scheduling, and manufacturing system simulation.

Dr. Fujii is a member of various academic societies such as JSPE, JSME, ORSJ, ISCIE, SICE, INFORMS and others.



Yasushi Kidani received his BE from the Department of Computer and Systems Engineering at Kobe University, Kobe, Japan, in 1994, and his ME from the University's Graduate School in 1996. He worked for Nippon Telegraph and Telephone Corporation (NTT), and has since moved to the company's Media Technology Factory in NTT-Long Distance and Global Provisional Headquarters. Kidani's research areas include video communications systems using animation and multi-point communication systems.

communications systems using animation and multi-point communication systems.



Atsushi Ogita received his Bachelors of Engineering in 1994 from the Department of Computer and Systems Engineering of Kobe University, Kobe, Japan. He earned his Masters of Engineering at the Graduate School of Kobe University in 1996. He currently works as a Supporting Engineer of Information System Development and Introduction at Kansai Electric Power Co., Inc.



Toshiya Kaihara received his BE and ME degrees in Precision Engineering from Kyoto University, Kyoto, Japan, in 1983 and 1985, respectively. He joined Mitsubishi Electric Corporation in 1985, where he worked in manufacturing system development. He received his PhD in Mechanical Engineering from Imperial College, University of London, United Kingdom, in 1994. He joined the University of Marketing and Distribution Sciences as an Associate Professor in 1996. Dr. Kaihara is a member of IEEE, ISCIE, ORSJ, JSPE, JSME, and other organizations.

Dr. Kaihara is a member of IEEE, ISCIE, ORSJ, JSPE, JSME, and other organizations.