Proceedings of the 2004 Winter Simulation Conference
R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds.

A RESOURCE RECONCILIATION MECHANISM FOR A MANUFACURING FEDERATION COORDINATED USING AN MRP/ERP SYSTEM

Seungyub Lee
Richard A. Wysk

Department of Industrial and Manufacturing Engineering
Pennsylvania State University
University Park, PA 16802, U.S.A.

ABSTRACT

This paper presents a top-down mechanism for coordinating Distributed Discrete Event Simulation (DDES) models using an MRP/ERP system as the federation coordinator. The same MRP/ERP system, which is typically used as a coordination tool for interactions between complex highly variable manufacturing systems, serves to coordinate and synchronize complex highly variable simulation models of these same systems. This research focuses on enabling each system entity modeled by DDES models to constantly correct its performance with respect to reference trajectories which consist of planned orders and the size of a time bucket generated by an MRP/ERP system, and trigger a global coordinator which consists of the MRP/ERP system and adapter if necessitated by any discrepancies observed by the entity through simulation models. A global coordinator can synchronize timing of DDES models and provide adaptive time buckets using the cost-based mathematical model and corrected plans using the updated time bucket.

1 INTRODUCTION

Distributed simulations with a high degree of fidelity are necessary to model "Systems of Systems" for the large and complex systems such as manufacturing enterprises, supply chain systems, military systems, and biologic systems since time and model fidelity become critical issues for these systems (Lee et al. 2002). Before simulation developers map the real systems into simulated systems, it is preferential to construct the general framework or formal ontological model to define resources, behaviors, information, granularities for the manageability of the target systems and integration of different software systems interacting with each other. For the most straightforward approach, the target system is broken up into general sub components, which correspond to target granularities of autonomous distributed units in the real world system and in turn can be specified with increasing detail. A single conceptual model can be used as an information or reference model with aggregated information for building simulation models. This top-down modeling can be a powerful technique to develop DDES models of complex systems by breaking a large system into a number of smaller, more manageable subsystems (Daum and Sargent 1999).

Even though these distributed simulation models can be successfully constructed using a formal modeling mechanism, they sometimes violate causal relations among the entities if they run independently. Therefore, it is necessary to coordinate timing and events among all models, using a suitable synchronization and coordination method while also considering the realistic granularity of times and interactions of the simulated systems. For these reasons, many researchers have developed various synchronization methods such as master event calendar, null message, barrier synchronization, simple time bucket, time warp mechanism, double phased time bucket, breathing time bucket, adaptive time synchronization and so on to prevent violation of causal relations among the models under the conservative, optimistic, and variant approaches (Righter and Walrand 1989, Fujii et al. 2000, Fujimoto 2000, Lee et al. 2002).

Recently, several distributed simulation-based architectures such as the High Level Architecture (HLA)/Run Time Infrastructure (RTI), Generic Runtime Infrastructure for Distributed Simulation (GRIDS), and so on have been developed to foster the interoperation of simulations, time management of distributed models and the re-use of simulation components (Kuhl 1999, Taylor et al. 2002). There is always a trade-off between fidelity of event interactions and efficient operation of distributed simulation models. In other words, there is no perfect solution to both maximize the efficiency or speed and minimize the errors. For example, a typical conservative approach may eliminate any advantages of parallelism. On the flip side, the optimistic approach requires a great deal of computer memory for state saving for rollback. Most commercial simulation packages do not support typical rollback mechanisms (Fujii et al. 2000). Some of distributed simulation based architectures such as HLA/RTI enable different distributed simulation models (federation) to connect with each other via RTI. However, each simulation model requires a great deal of programming effort for the wrapper or interface to this architecture (Mertins et al. 2000). Additionally, the most critical observation is that no method or architecture provides a coordination mechanism considering varying levels of fidelity and granularity of the real system. For example, in the manufacturing systems domain, each workstation or shop should coordinate with each other by letting it operate independently for a time bucket (usually daily) and fixing errors between a predicted and actual scenario in order to ensure fidelity of transactions and meet enterprise's goals such as cost effective and on-time production of its products.

As shown in Figure 1, the behavior of the actual system, activities of a federation of simulation models, and transactions of sub entities in the MRP/ERP system are very similar (almost identical), and they can be identified through behavior formalism and commonalities among different systems using a formal ontology. A mapping mechanism should be required for different systems that have different granularities to fit together into a target domain and make an entire federation functional. Since there exist different component applications with different granularities for time and interactions among sub entities, the federation interacting with an external coordinator (i.e., MRP/ERP systems) should generate consistent results regardless of model's granularity or domain level.

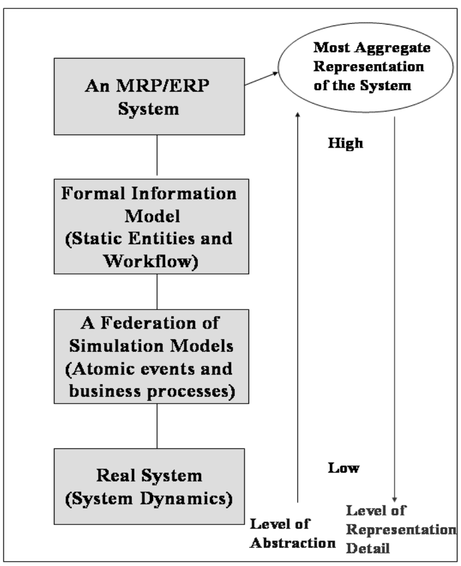

Figure 1: Relationship among Different Systems

Therefore, a formal enterprise information model using set theoretic symbology, system objects (object-oriented approach) and formal business process (i.e., Supply Chain Operations Reference (SCOR) Model) are proposed to map federate object interactions, resolve granularity issues for a federation, and verify consistency of formal business behavior among distributed objects and adequacy of mapping under the different operating policies. The framework developed can significantly increase efficiency of model development and management in terms of overheads dealing with component systems with different granularities by standardizing data interfaces. For this paper, each "as-fast-as-possible" simulation model is used to evaluate rough schedules generated either the MRP/ERP system (planner) or external finite capacity scheduler.

2 OVERVIEW OF METHODOLOGY

We propose a method called *resource reconciliation* to provide maximum parallelism among simulation federates. This method is a variant of the traditional synchronization approach, where each federate executes on a single processor as a local activity area model. Each federate moves forward simultaneously in time but never rolls back in time (unlike the traditional optimistic methods). All federates execute in parallel advancing time independently for a time bucket. The end of each time bucket is the only point in time where federates interact – within buckets all federates run at full speed. All federates wait at the completion of a time bucket until all other federates have reached this point in time. When all time buckets have been completed, each federate may send and receive messages and all resource levels and values are reconciled. Once this exchange is completed, federates again move independently forward at full speed until they arrive to the next check point. They move forward in this fashion – from time bucket to time bucket until a terminal time is reached. Resource reconciliation provides for maximum parallelism, but introduces the problem of what levels and states are accurate, and how does one account for exogenous events that might cause unforeseen federate interactions.

The key hypothesis of this research is, "MRP/ERP systems are used to coordinate and synchronize interactions between complex highly variable manufacturing systems. These same MRP/ERP software entities can also serve to coordinate and synchronize complex highly variable simulation models (using commercial simulation packages) of these same systems." The main goal of this research is to develop a methodology to coordinate fast forward distributed manufacturing simulation models representing loosely coupled distributed entities for a manufacturing enterprise using the MRP/ERP system. To achieve the stated goal and make the proposed methodology functional, it first attempts to develop a formal information model of resources, activities and information for components in a manufacturing enterprise that can be used for providing a modeling paradigm for specifying component simulation models at different levels of abstraction to coordinate activities in the federation. An ontological model (i.e., object-oriented approach) is used to formalize and encapsulate manufacturing system entities' interactions and operations. This formal model is used for reusing generalized model components for the system entities. Based on the formal model, we describe how the model maps to components in a manufacturing system. In order to map components that have different granularities, commonalities in the component systems are identified in the formal information model and simulation modeling logic for order release, time granularity, and attributes manipulation is specified. In order to test the hypothesis and effect of the new coordination mechanism, it validates adequacy of mapping mechanism through experiments under the various circumstances.

The proposed coordination mechanism shown in Figure 2 focuses on enabling each system entity modeled by DDES models to constantly correct its performance with respect to actual parameters which consist of planned orders and the size of a time bucket generated by an MRP/ERP system, and trigger a global coordinator which consists of the MRP/ERP and Adapter systems if necessitated by any discrepancies observed by the entity through simulation models.

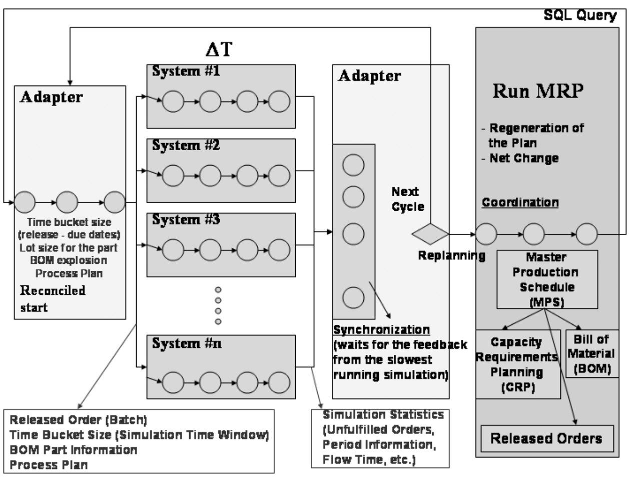

Figure 2: Overview of the Proposed Coordination Mechanism

The main classes in this architecture are a federation of simulation models, an adapter (a TCP/IP-based Windows-messaging mediator between an application program and a network, a repository for intermediate data necessary for each manufacturing federate, a synchronizer for simulation models, a generator for adaptive parameters as a decision supporter and a mapping tool between two systems that have different granularities), and the MRP/ERP system. Each component of the software system also has the ability to invoke a neighborhood application or deliver necessary information to others using messages and interfaces such as a Remote Procedure Call (RPC), Dynamic Data Exchange (DDE), Dynamic Link Library (DLL), Object Database Connectivity (ODBC) and so on. In the proposed architecture, closed-loop coordination is implemented.

3 FORMAL INFORMATION MODEL

A formal language is very useful to describe a context free information model for complete description of the systems which have different level of data abstraction. Therefore, the architecture for the target system can be modeled using several descriptive modeling techniques (Computer Aided Software Engineering (CASE) tools), such as object-oriented process modeling, Unified Modeling Language (UML), Integrated Computer-Aided Manufacturing (ICAM) DEFinition (IDEF) ontologies, Computer-Integrated Manufacturing – Open System Architecture (CIM-OSA), SCOR Model, Process calculus (i.e., Business Process Management Language (BPML) and so on) or combined techniques. This architecture is necessary to represent the structure, activities, processes, information, resources, behavior, goals and constraints of a business and overcome challenges such as communication between incompatible software applications, knowledge exchanging and sharing, integration of heterogeneous entities, ontological consistency and so on. Hence, communication can be established between these different systems by mapping different granularities for them and creating software mechanism that manipulate instances of the elements formally established in the different systems. By storing data required for various manufacturing domains modeled or adapted using the desired granularity, different software packages can then be created that reason about the resources by using the same object-oriented structure with different class behaviors. Based on the formal information model, an intermediate database model can then be generated and used for simulation model generation along with the algorithmic mapping mechanism (Son et al. 2003). This architecture facilitates sharing of resource data across the distributed enterprise and the development of software to implement engineering functions (Steele et al. 2001).

Figure 3 represents the general steps for developing the proposed methodology. Using the conceptual information model as a reference system, both system developers and users can develop, test, use, and maintain the various systems with less effort and time. For example, if the planned orders and other parameters in the MRP/ERP system are generated based on a workstation-based perspective, simulation models that have different granularities (i.e., shop level) with the MRP/ERP system should have the mechanism to manage and convert the parameters given by the MRP/ERP system to attributes of simulation models at different levels of resolution in order to process a batch of orders in a consistent manner. For resolving overheads associated with model development and management, direct mapping using the mapping tool (i.e., Adapter (Visual Basic Application)) has an important implication in the model development phase in order to maintain consistency, modularity, and reuse of the model generated.

A formal model representing abstraction of the target domain can be firstly generated using domain knowledge.

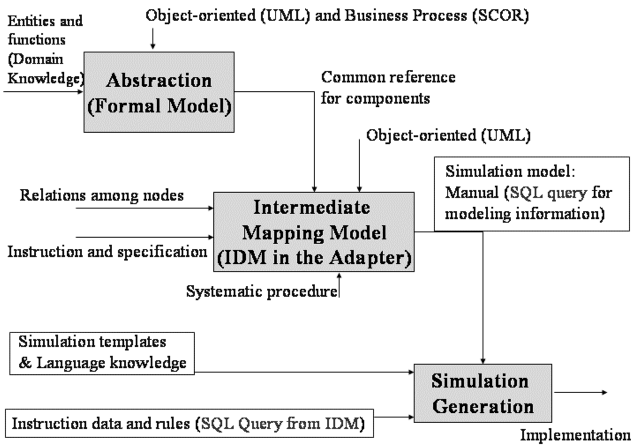

Figure 3: Mapping Procedure for Simulation Generation using the Information Model

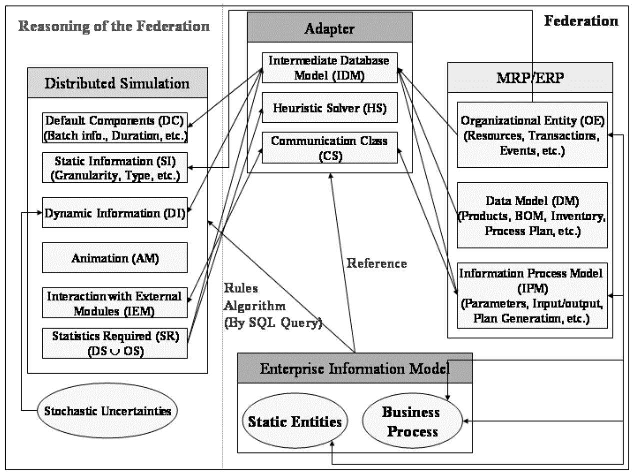

Figure 4: Correspondence among Classes in the System

And then Intermediate mapping model is developed using common data from the formal model, instructions and specifications. Finally, simulation models (i.e., using commercial simulation packages such as Arena) can be generated either manually or automatically from the mapping procedure, modular template information, and specifications of attributes. In this paper, we consider only manual generation of simulation model. For example, if modelers want to develop modular simulation models with the required granularities, they can obtain the information as a form of the report for necessary information such as attributes, statistics, and simulation blocks in order to generate models. Related to the information model we propose here, it is noted that some works have been developed at National Institute of Standards and Technology (NIST) for standardization of neutral data interfaces for integrating machine shop software application with simulation models using the Unified Modeling Language (UML) and Extensible Markup Language (XML) (Lee et al. 2003).

Figure 4 describes a mapping using a system morphism that establishes a correspondence between two systems at different levels of resolution and depicts an example of resources and manufacturing activities in them. For example, the MRP/ERP system can have different sets of sub classes as a form of a relational data model. An Organization Entity (OE) can be unions of factories, transporters, warehouse, and so on. Using the aggregated information in OE, the detailed static resource information can be created in the information model. And each simulation model can be a model for representing the class which is the element of these static entities. Between these two systems, rules and interfaces should be identified in order to guarantee ontological matching for these systems, communicate with each other properly and represent the behavior of the real system accurately. In addition, mechanism for obtaining order and parts information for simulation models should be also developed in the mapping mechanism.

Therefore, Object-Oriented (OO) approach based on “top-down” event-based modeling is used in this paper to model, map, and simulate resources and their interactions. The proposed system architecture using OO approach can also be defined as the formal resource model in terms of a set of standard object-oriented classes with the properties of inheritance, ownership, data hiding, automatic class initialization, and polymorphism in order to integrate the different functions and domains in the manufacturing enterprise (Booch 1986, Steele et al. 2001). A resource model, an enterprise information model in a more generic way, contains a set of definitions and symbolic descriptions that are required to describe all of the individual resources in a facility as well as the necessary interactions between these resources. It includes both the physical and logical information in the facility directed toward the manufacture of the products (Wysk et al. 1995, Son and Wysk 2001, Steele et al. 2001). Formal definition of the manufacturing resource model is developed and presented by Wysk et al. (1995). Steele (2001) presented a modified resource model for the manufacturing, especially shop, workstation and equipment levels, domain. Additionally, Son (2001) developed a method to automatically generate a simulation model from a formal resource model discussed in Wysk at al. (1995) and Steele et al. (2001). Even though this resource model can model object interactions by means of a connectivity graph, sequence charts and interaction diagram in the UML, it might be required to model a high-level business process that facilitates integration across the supply chain in order to take a full advantage of using Enterprise Applications (EA) such as an ERP system. The Supply Chain Council has established a standard way to examine and analyze supply chains with SCOR model. Based on the e-SCOR discussed in Barnett and Miller (2000), an object-oriented resource and process models can be developed to construct simulation models. Table 1 describes four levels of top-down process hierarchy in the SCOR model. Blocks at the lowest level map conveniently into elements in the UML for activity diagrams or simulation blocks such as

Arena. Hence, entities in both MRP/ERP and simulation systems can be identified and mapped through this model and object-oriented structure in this paper.

Table 1: Four Levels of Process Detail in the SCOR Model (Barnett and Miller 2000, Supply Chain Council 2004)

Level	Description	Comments
1	Top Level (Process Types)	Plan, Source, Make, Deliver, Return
2	Configuration Level (Process Categories)	30 pre-defined categories according to company's operation
3	Process Element Level (Decompose Process)	Process element inputs and outputs
4	Implementation Level (Decompose Process Element)	"Atomic" process blocks (represented using simulation blocks)

4 DOUBLE-PHASE RESOURCE RECONCILIATION MECHANISM

For an MRP-type coordination mechanism, a global plan is periodically generated or regenerated based on the status information collected from the system entities. Local plans or short-term schedules are made based on the global plan. The time interval for such periodic planning activities corresponds to the size of time bucket in the MRP/ERP system for the manufacturing domain. Since the performance of the system and execution speed of a federation also depend on the level of interactions or frequency of synchronization among the distributed models, appropriate selection of a simulation time window (ΔT) or size of time bucket is very critical. Larger time windows imply a larger degree of decoupling among federates and can decrease the frequency of synchronization among them (Brandimarte et al. 2000). However, a larger time window also implies deterioration of manufacturing cycle time since it can increase lead time for the manufacturing systems (Riezebos 2001). There is a trade-off between communication overheads to and from federates and manufacturing cycle time reduction. Hence, the time window may be selected or adjusted according to the nature of the systems, while considering all the constraints in the systems. The main objective of resource reconciliation associated with the robust size of time bucket can be represented using two extreme cases as shown in Figure 5. In this paper, the initial time interval for periodic activities in the actual and simulated systems is provided by the MRP/ERP system for its coordination cycle. If there is consensus of a natural checkpoint for all participants or federates (the desired granularity might totally depend on the problem domains) in the target domain (i.e., Supply Chain System, Enterprise, Military), it can be used as a sort of standard (obtained from experience and historical data). Otherwise, it is another research subject to determine the initial synchronization interval or checking point. Hence, experience-based method that observe all the events in the federates and decide a safe-enough window (providing some tolerances for planning) to ensure all federates do not violate causal relationship and successfully produce the assigned lot within the time bucket.

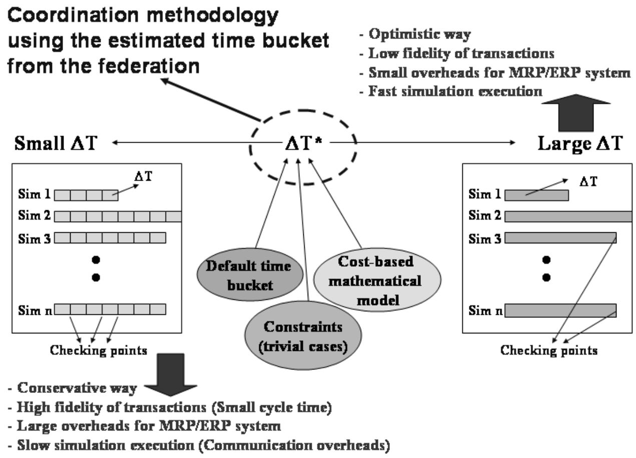

Figure 5: A Methodology to Find a Suitable ΔT

To use a small time bucket is a more or less conservative way. It has both a large frequency of synchronization request and high fidelity of transactions. Hence, it can improve manufacturing cycle time but cause large overheads for the MRP/ERP and adapter systems due to frequent requests from the federates and, therefore, deteriorates speed of simulation execution. On the other hand, to use a large time bucket is a more or less optimistic way. It has a small frequency of synchronization request and low fidelity of transactions. Hence, it can deteriorate manufacturing cycle time but cause relatively small overheads for the MRP system and adapter due to less frequent requests from the federates, and consequently improve speed of simulation execution. Using the default time bucket provided by the MRP/ERP system, cost-based mathematical model, and constraints, the goal of the proposed mechanism is to find out appropriate time bucket size and provide it with simulation models.

Figure 6 represents a double-phase mechanism for synchronization and coordination for the resource reconciliation mechanism. Even though two terminologies such as "Synchronization" and "Coordination" are treated as having analogous meaning in the literatures, their usages are discriminated in this paper. "Synchronization" means periodical matching of all local virtual clocks to a global virtual clock. Synchronization is accomplished by the Adapter while providing timing control for distributed simulation models and maintaining the size of a time bucket. This synchronization method is very similar to a "Conservative" time bucket method and barrier synchronization method (Fujii et al 2000, Fujimoto 2000). On the other hand, "Coordination" implies correcting action for discrepancies observed in the system entities to guarantee consistent views of the system. Coordination is performed by the MRP/ERP system if it is necessary. This coordination is a sort of an optimistic approach for recovering production errors rather than causality errors since these causality errors are already prevented by "Conservative" synchronization. Therefore, frequency of re-planning in the MRP/ERP system for coordinating activities and tuning errors in the federates are controlled and maintained by this mechanism. Although it is not considering fidelity of atomic time (i.e., transit time), it can increase efficiency of the entire federation in terms of fidelity in order to achieve a global goal such as providing on-time production scenarios.

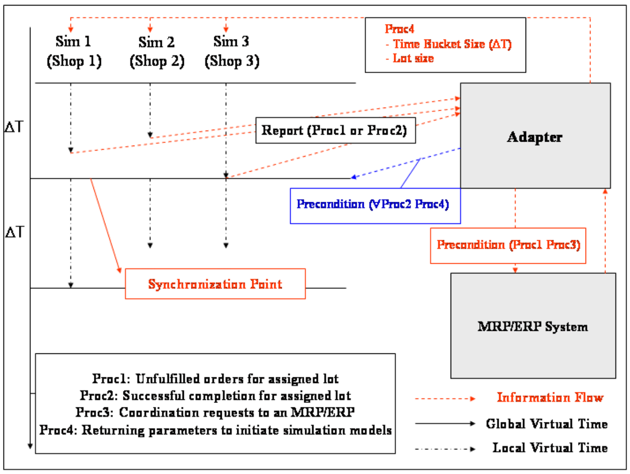

Figure 6: Synchronization and Coordination Mechanism

As shown in Figure 6, all federates are initiated together and are due to be completed together at the end of time bucket ΔT (specified and controlled by Global Virtual Time (GVT)) since single phase order release scheme is used in this research. However, they might be completed at different time due to difference of each simulation's Local Virtual Time (LVT) affected by computer performance. Upon completion of its process for a time window (time bucket ΔT), each simulation model reports its preconditions determined by statistics such as the number of parts produced, flow time, cost estimation, etc. to the Adapter and waits for decision of the next manufacturing cycle. Once the Adapter collects all the reports including statistics and preconditions from all federates, it decides whether the entire federation has to be coordinated by the MRP/ERP system using re-planning such as "regeneration" or "net changing" or it should be re-initiated using the updated time window ΔT and the next assigned lots. As shown in Figure 6, general predicates for preconditions are further explained as follows.

- **(precondition proc1 proc2)**: proc2 can exist or be true only if proc1 exists or is true.
- **(precondition ∀proc1 proc2)**: proc2 can exist or be true only if proc1 exists and is true for all the instances.

- If proc1 is a precondition of proc2,
- then if there exists inst2 so that inst2 is an instance of proc2,
- then there exists inst1 so that inst1 is an instance of proc1.
- **Proc1**: unfulfilled orders for the assigned lot
- **Proc2**: successful completion of the assigned lot
- **Proc3**: coordination request to the MRP/ERP
- **Proc4**: returning parameters to re-initiate simulation models

The pseudo code for the synchronization and coordination mechanism in the coordinator and order lease logic for each federates is as follows:

```
While (A Planning horizon is not over)
In the Adapter system
    If # of messages collected in the counter = #
    of federates
        CalculateTimeBucket ( )
            if there is a federate that has Proc1
                InvokeMRP ( )
                ObtainNewPlan ( )
                Insert new plan into intermediate
                database
            Else
                Update the row in the intermediate
                database
        Return the Updated time bucket
        Send feedback to federates
        Reset the counter
    Else
        Wait for the last message
In each federate
    If a federate's TNOW = the end of time bucket
        Collect statistics and send message to the
        Adapter

    If a federate receives the feedback from the
    Adapter
        Invoke and obtain new data from the inter-
        mediate database
        if TNOW = Release date
            release orders and obtain data
        else if TNOW < Release date
            Delay for Release date - TNOW
    Else
        Wait for the feedback and holding the en-
        tity in the block
```

5 TIME BUCKET AND COST BASED HEURISTIC MODEL

Considerable research has been conducted for the importance of the time bucket on the performance and cost for operating the production system. For example, Brandimarte et al. (2000) proposed a modular approach to devise and assemble local schedulers and a way to link predictive and real time scheduling using a scheduling architecture based on the Shifting Bottleneck (SB) method. They implied that the operation time windows in the real time scheduler (i.e., a simulation model) are somehow reminiscent of MRP operation lead times and presented the method to update time windows based on the flow time of the work center. Using the time bucket given, the simulation model verifies for both lateness and earliness penalties. Since it is not sufficient to know whether a batch has been completed on time, but if it is early or late (within a pre-defined "tolerance" interval), the model verifies when each batch is completed. This information is used to ensure that the required production rate is being maintained. This can be achieved by monitoring the number of batches being processed and crosschecking it with the production schedule for that entity. Each local simulation model is trying to achieve a local optimization such as minimizing the cost for local activities associated with parts. However, a collection of individual local optimizations does not mean a global optimization as a whole. Therefore, a cost-based global optimization model should provide an optimal variable for a federation as a whole while reflecting the feasibility of the local system. Each federate reports their activities converted into the estimated cost to the adapter. Inside this system, a mathematical model is implemented to update the time bucket as shown in Figure 7.

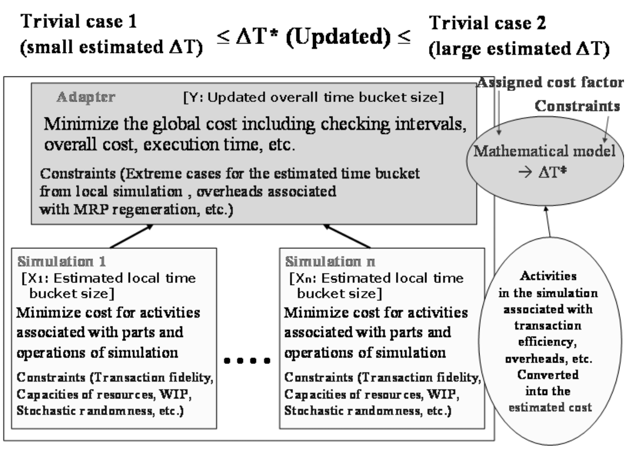

Figure 7: The Proposed Scheme for Updating the Time Bucket

The basic idea is to increase the time bucket size if the dominant number of system entities has unfulfilled orders, which implies lateness, and to decrease it otherwise. The partial mathematical model to calculate the update time bucket is provided in Equation (1). Initial time bucket size might be set using the standard time bucket in the typical MRP/ERP. However, since initial time bucket size can be very critical for system performance, some other methods such as the ratio of the work center load with respect to the overall load are also proposed (Brandimarte et al. 2000).

$$\Delta T_{new} = \frac{\sum_i (\hat{F}_i \times \breve{C}_i) + \sum_j \left(\tilde{\nabla} F_j \times \breve{C}_j\right)}{\sum_i \sum_j \left(\breve{C}_i + \breve{C}_j\right)} \qquad (1)$$

$$s.t. \quad Min\left[\hat{F}_i\right] \le \Delta T_{new} \le Max[\tilde{\nabla} F_j]$$

Where

$\breve{C}_i = C^H_{k,i} \times (\Delta T_{now} - \overline{F}_i) + C^A_i$

$\breve{C}_j = U_j \times C^P_k + C^A_j$

$i \in S_F$: Index of a system entity (modeled by a federate)

$j \in S_{NF}$: Index of a system entity

S_F : Set of entities that have earliness

S_{NF} : Set of entities that have lateness

k : Index of a part

L_i, L_j : Lot size for an entity *i* and *j*

ΔT_{now} : The current size of a time bucket

ΔT_{new} : The updated length of a time bucket

$\hat{F}_i$ ($\hat{F}_i \le \Delta T_{now}$ and $\hat{F}_i = MAX\{\overline{F}_i\}$): Local maximum flow time for an entity *i*

$\overline{F}_i (= \sum_{p=1}^{L_i} F_p / L_i)$: Local average flow time for an entity *i*

P_j : Total number of parts produced at an entity *j*

$U_j = L_j - P_j$: Unfulfilled parts at an entity *j*

$\tilde{\nabla} F_j = \Delta T_{now} + \left(\frac{\Delta T_{now}}{P_j}\right) \times U_j$: Estimated local maximum flow time for an entity *j*

C^A_i, C^A_j : Total activity cost of an entity *i* or *j*

C^P_k : Penalty cost for unfulfilled parts *k* (Lateness)

$C^H_{k,i}$: Cost to hold a part *k* at the entity *i* or *j* (Earliness)

C^T_k : Cost to transport a part *k*

$\breve{C}_i$: Total cost factor for entities $\in S_F$

$\breve{C}_j$: Total cost factor for entities $\in S_{NF}$.

It is assumed that the total activity cost for a federate is provided by each simulation model since some simulation packages such as Arena can provide cost estimation based on the activities occurring in simulation run. The speed of system convergence as well as the performance can be varied with respect to the degrees of cost values such as holding cost and penalty cost. For example, if the unfulfilled orders are more critical than the holding the finished products in the factory (WIP), a large value might be assigned to this penalty cost. In this case, time bucket is updated and converge to the value towards the bigger size. A detailed discussion on the elaborated mathematical model and relationship between transition of the time bucket size and system performance will be provided in the subsequent paper.

6 VALIDATION AND IMPLEMENTATION

All simulation models have been built in Arena 7.0 for a target manufacturing network (an enterprise and associated supply chains using an MRP/ERP system) for this paper. Various instances of systems such as *m* factories including *w* workstations, *t* transporters (trucks), *s* suppliers and so on are implemented to illustrate Printed Circuit Board (PCB) assembly houses in this research. This research provides a common structure and framework that can be applied to various systems in the target domain. It implies scalability of the target problem domain can be a factor for the experimental framework in order to test validity of the proposed methodology under the various scenarios of the system dynamics. The experimental design and related simulation results are provided with respect to levels of product complexity, a scale of the target domain, variability of the stochastic uncertainties, cost factor, lot sizing rules, control policies and so on for checking variability and causality of the system. For example, a factor for variability of the stochastic uncertainties can have three levels such as deterministic system (no uncertainty), internal uncertainties only (i.e., variable setup times, machine breakdown, etc.), and internal and external uncertainties (i.e., demand fluctuation). A cost factor may have two levels such as (1) High C_k^P Low $C_{k,i}^H$ and (2) Low C_k^P High $C_{k,i}^H$.

Table 2 summarizes experimental framework for this paper. The objective of the experiment is to check the effectiveness of the proposed methodology for system variability on the performance criteria. In general, five major performance criteria have shown in the literature: completion time-based, due-date-based, flow time-based, machine-based, and throughput-based. In this research, these general performance criteria as well as criteria related to efficient operation of software systems such as simulation execution time and the total cost associated with activities in the manufacturing system are investigated. The speed and traffic of a network and performance of computers considerably affect overall system's performance since the proposed system depends on communication process and message parsing for synchronization among simulation models. Moreover, it would be very interesting to investigate whether overheads associated with re-planning of the MRP/ERP system offset the gains for computing speed from parallel execution. Since Total Enterprise Application Manager (TEAM) developed by Georgia tech (relatively small ERP system) is employed for implementation in this paper, the overheads for running the MRP/ERP system is quite small compared to execution time of a federation.

Table 2: Experimental Study for This Paper

Product Structure			Target Domain		Level of Uncertainties			Cost Factor		Lot sizing
1	2	3	4	5	6	7	8	9	10	11-13

7 CONCLUSION

In this paper, we have introduced a new coordination mechanism for a federation consisting of distributed simulation models by benchmarking the coordination cycle of typical MRP-type systems. It can be argued that the results from the proposed architecture can be no worse than the traditional methods while at the same time guarantees the ability of an entity (and the entire system) to respond to unplanned events in real time. In order to test the utility of the proposed methodology not just instance-specific, the various example scenarios are demonstrated using a set of experiments for system variability on the basis of performance criteria. Symbolic representation of the formal information model of resources and component systems for the target domain forms the basis for software development and maintenance. Without an embedded complex synchronization algorithm or state saving function for underlying simulation programming, development and implementation of distributed simulation models for the large system (i.e., manufacturing system in this paper) can be easily conducted. A case study based on manufacturing supply chain is being implemented at Penn State and will be used to compare the performance of the proposed methodology with traditional methods.

REFERENCES

Barnett, M. W., and C. J. Miller. 2000. Analysis of the virtual enterprise using distributed supply chain modeling and simulation: an application of e-SCOR. In *Proceedings of the 2000 Winter Simulation Conference,* ed. J. A. Jones, R. R. Barton, K. Kang, and P. A. Fishwick, 352-355. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Booch, G. 1986. Object-oriented development. *IEEE Transactions on Software Engineering* SE 12(2): 211-221.

Brandimarte, P., Rigodanza, M., and L. Roero. 2000. Conceptual modeling of an object-oriented scheduling architecture based on the shifting bottleneck procedure. *IIE Transactions (Institute of Industrial Engineers)* 32(10): 921-929.

Daum, T., and R. G. Sargent. 1999. Scaling, hierarchical modeling, and reuse in an object-oriented modeling and simulation system. In *Proceedings of the 1999 Winter Simulation Conference*, ed. D. T. Sturrock, G. W. Evans, P. A. Farrington, and H. B. Nemhard, 1470-1477. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Fujii, S., Kaihara, T., and H. Morita. 2000. A distributed virtual factory in agile manufacturing environment. *International Journal of Production Research* 38(17 SPEC): 4113-4128.

Fujimoto, R. M. 2000. *Parallel and distributed simulation systems*. New York, NY: Wiley.

Kuhl, F., Weatherly, R. and J. Dahmann. 1999. *Creating computer simulations: An Introduction to the High Level Architecture*. Upper Saddle River, NJ: Prentice-Hall.

Lee, S., Ramakrishnan, S., and R. A. Wysk. 2002. A federation object coordinator for simulation based control and analysis. In *Proceedings of the 2002 Winter Simulation Conference,* ed. E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, 1986-1994. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Lee, Y. T., McLean, C., and G. Shao. 2003. A Neutral information model for simulating machine shop operations." In *Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, 1296-1304. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Mertins, K., Rabe, M., and F. Jaekel. 2000. Neutral template libraries for efficient distributed simulation within a manufacturing system engineering platform. In *Proceedings of the 2000 Winter Simulation Conference*, ed. J. A. Jones, R. R. Barton, K. Kang, and P. A. Fishwick, 1549-1557. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Righter, R., and J. C. Walrand. 1989. Distributed simulation of discrete event systems. *Proceedings of the IEEE* 77(1): 99-113.

Riezebos, J. 2001. Design of a Period Batch Control Planning System for Cellular Manufacturing. Doctoral Dissertation, Management and Organization, University of Groningen, Groningen, Netherlands.

Son, Y. J., and R. A. Wysk. 2001. Automatic simulation model generation for simulation-based, real-time shop floor control. *Computers in Industry* 45(3): 291-308.

Son, Y., Jones, A. T., and R. A. Wysk. 2003. Component based simulation modeling from neutral component libraries. *Computers and Industrial Engineering* 45: 141-165.

Steele, J., Son, Y.-J., and R. A. Wysk. 2001. Resource modeling for the integration of the manufacturing enterprise. *Journal of Manufacturing Systems* 19(6): 407.

Supply-Chain Council. 2004. *Supply-chain operations reference model: an overview of SCOR version 6.0*, Available online via `http://www.supply-chain.org` [accessed February, 2004].

Taylor, S. J. E., Sudra, R., Janahan, T., Tan, G., and J. Ladbrook. 2002. GRIDS-SCF: An infrastructure for distributed supply chain simulation. *Simulation* 78(5 SPEC): 312-320.

Wysk, R. A., Brett, P., and J. S. Smith. 1995. A Formal Process Planning Schema for Shop Floor Control. *Engineering Design and Automation* 1(1): 3-10.

AUTHOR BIOGRAPHIES

SEUNGYUB LEE is a doctoral candidate in the Department of Industrial and Manufacturing Engineering at Pennsylvania State University, University Park. He received his B.S. degree in Industrial Systems Engineering from Yonsei University, Seoul, Korea and his M.S. in Industrial and Manufacturing Engineering from Pennsylvania State University. His research interests are distributed simulation, simulation-based control, and enterprise information modeling and integration. His email address is <SXL287@psu.edu >.

RICHARD A. WYSK is the Leonhard Chair in Engineering and a Professor of Industrial Engineering at Pennsylvania State University, University Park. Dr. Wysk has coauthored six books including *Computer-Aided Manufacturing*, with T.C. Chang and H.P. Wang -- the 1991 IIE Book of the Year and the 1991 SME Eugene Merchant Book of the Year. He has also published more than a hundred and fifty technical papers in the open-literature in journals including the *Transactions of ASME*, the *Transactions of IEEE* and the *IIE Transactions*. He is an Associate Editor and/or a member of the Editorial Board for five technical journals. Dr. Wysk is an IIE Fellow, an SME Fellow, a member of Sigma Xi, and a member of Alpha Pi Mu and Tau Beta Pi. He is the recipient of the IIE Region III Award for Excellence, the SME Outstanding Young Manufacturing Engineer Award and the IIE David F. Baker Distinguished Research Award. He has held engineering positions with General Electric and Caterpillar Tractor Company. He received his Ph.D. from Purdue University in 1977. He has also served on the faculties of Virginia Polytechnic Institute and State University and Texas A&M University where he held the Royce Wisenbaker Chair in Innovation. His email and web addresses are <rwysk@psu.edu> and <http://www.engr.psu.edu/cim/wysk.htm>.