Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

A FAMILY OF MARKET-BASED SHIPMENT METHODOLOGIES FOR DELIVERY SUPPLY CHAIN

Seog-Chan Oh
Soundar R. T. Kumara

Industrial & Manufacturing Engineering Department
The Pennsylvania State University
University Park, PA 16802, U.S.A.

Shang-Tae Yee
Jeffrey D.Tew

Manufacturing Systems Research Laboratory
General Motors Research and Development Center
Warren, MI 48088, U.S.A.

ABSTRACT

Load building is an important step to make the delivery supply chain efficient. We present a family of load makeup algorithms using market control-based strategy, named ***LoadMarket,*** in order to build efficient loads where each load consists of a certain number of finished products having destinations. LoadMarket adopts minimum spanning tree graph algorithm for generating initial endowment for *Load Traders* who cooperate to minimize either total travel distance or the variance with respect to the travel distances of loads through a spot market or double-sided auction market mechanism. For the simulated load shipment market, the efficiency of the LoadMarket algorithms is analyzed using simulation experiments.

1 INTRODUCTION

Load building is a process of assigning a set of products to a number of transportation carriers like truck or railcar. Companies have made an effort to achieve more efficient shipment planning and execution of finished products on their delivery supply chain. The efficiency in shipment planning and execution can improve customer fulfillment with fast delivery and reduce total transportation cost. We present LoadMarket, a family of load assignment methods using market-based control mechanism.

Let us consider an example scenario where the optimization of load assignment is crucial issue. Suppose an automotive company produces cars and transports them to the customers nationwide (Yee, 2002). As shown in Figure 1(a), whenever an ordered car is produced in the plant, it moves to the yard where cars wait to be assigned to a truck, the transportation carrier. Maintaining cars in the yard results in costs. Each truck has a limitation for the maximum number of vehicles to carry and after containing one load of cars, it starts to visit each city to deliver cars. The transportation cost of each truck is proportional to the total travel distance. In summary, the daily cost model for this scenario has the following:

Cost model = (average # of cars in the yard × maintenance fee per a day) + (daily # of trucks departing from the yard × average travel distance of trucks)

The above cost model implies that the main costs saving problems are: (1) fast shipping out cars waiting in the yard and (2) finding a shortest average travel distance. In this paper, we consider the second problem that is, finding a shortest average travel distance.

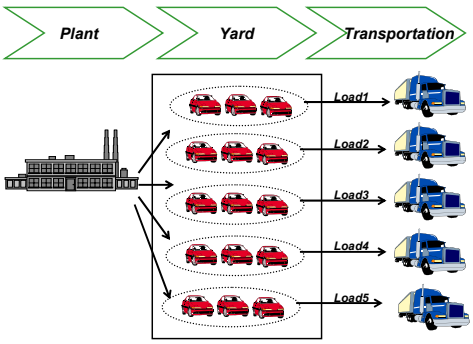

Figure 1(a): A scenario of the load assignment

If there is a large truck with the ability to carry all daily produced cars and cover all destination cities, this problem is reduced to a traveling salesman problem (TSP) (i.e., finding a path through a weighted graph which starts and ends at the same vertex) (Flood, 1956). However, in the real world, a truck can carry at most ten cars at one shipment and can travel at most over thirty neighboring cities. Thus, the problem needs to employ multiple trucks in order to deliver many daily produced cars, which leads to the capacitated vehicle routing problem (CVRP) (Clark and Wright, 1964). For solving CVRP, as shown in Figure 1(b), we can use the *k*-tree covering algorithm which is NP-complete proven by reduction from the Bin-Packing Problem (Even et al., 2003).

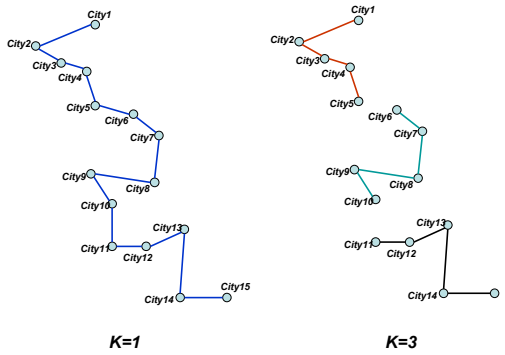

Figure 1(b): *k-tree covering*

The presented scenario often occurs in many other industries that aim to *Order to Delivery* (OTD) service. However, identifying minimized cost value when the number of cities to visit and size of items to deliver increase, is a challenging task, since in real application, such numbers can be many and non-trivial. Therefore, there is an imminent need for the method that systematically and mechanically helps to plan a load assignment such that average travel distance for delivering the loads is minimized or load balancing between the loads is satisfied.

Market-based automated negotiation, or more commonly *market-based control*, is a paradigm for controlling complex system that would otherwise be very difficult to handle, by taking advantage of some desirable features of a market (especially a free market) including decentralization, interacting agents, and some notion of resources that need to be allocated (Clearwater, 1996). This approach has been applied to a wide range of fields such as supply chain management (Hinkkanen et al., 1997; Sauter et al., 1999), vehicle routing (Sandholm, 1993), manufacturing scheduling (Tilley, 1996; Baker, 1996), and process control (Jose and Ungar, 1998). The load assignment problem domain can be classified into the case of distributed resource allocation having a team objective.

In this paper, we consider two market control mechanisms (Lee, 2002) that are designed as follows:

- Double Auction (DA): Within the control window, multiple sellers and multiple buyers place or ask bids for the exchange of a designated commodity in a virtual market which matches buyers and sellers immediately on detection of compatible bids;
- SpotMarket: A market is established for a seller and a buyer within the control window, meaning that the seller sells any commodity that is most benefit to the buyer.

Based on this setting, we can define our load problem in detail. In general, when considering the load assignment problem, the following definitions hold:

Definition 1 *Let* ***Load*** *denote the graph* $<V,E>$ *which contains* V *as a set of products intended to be delivered to their destinations and* E *as a set of distances between destinations,* $v_i(\in V)$ *and* $v_j(\in V)$.

Definition 2 *Let* $\boldsymbol{TD(Load_i)}$ *denote the length of the shortest path of* $Load_i$ $i=1,\ldots,k$.

Definition 3 ***LoadMakeup*** *is a set of* $Load_i$ $i=1,\ldots,k$. *Let* W *products intended to be delivered from the yard to customers. Then,* $\bigcup_{i=1}^{k} V(Load_i) = W$. *When a LoadMakeup is obtained,* $\sum_{i=1}^{k} TD(Load_i)$ *is the total distance required to deliver all the products of* W.

Definition 4 ***Load Trader*** *is an agent that processes a Load and participates in DA or SpotMarket in order to minimize: (1)* $TD(Load)$ *or (2)* $\sum_{i=1}^{k} TD(Load_i)$ *by cooperation with other Load Traders*.

In this paper, we aim at solving the following problems.

Given a set of products, W, *effectively find a LoadMakeup such that for* $\forall Load_i \in LoadMakeup$ *(1)* $\sum_{i=1}^{k} TD(Load_i)$ *is minimized; or (2) variance w.r.t* $TD(Load_i)$ *is minimized.*

In particular, the second problem of finding a LoadMakeup with minimized variance over TD(Load) is what we refer to as the *Load Balancing* problem, and can be essentially formulated by balance spanning tree (BST). A spanning tree T of a graph G is called a BST if it minimizes the difference between the most costly arc and the least costly arc selected (that is, from among all spanning trees of G, the difference between the maximum arc cost in T and the minimum arc cost in T is as small as possible). Obviously, one very inefficient (if not intractable) method of finding a BST is to enumerate all spanning trees of G and then select the balanced spanning tree from among these. Since we aim at obtaining a set of Loads rather than discovering just a single balanced shortest path, such existing BST solution is, however, not feasible.

In the following section, we investigate alternative, non-exhaustive, but approximate solution based on both Prim's MST algorithm (Prim, 1957) and aforementioned DA or SpotMarket.

2 OVERVIEW OF LOADMARKET

In handling our problems with market-based control, two issues are critical: (1) the initial Loads to be assigned to Load Traders for their endowment; and (2) the market mechanism to control Load Traders. To address the issues, we propose techniques – generating initial load assignment using the tree covering technique adopting Prim's MST and two market control-based algorithms.

2.1 LOADMARKET^{MST}

Input: W : all products, E : distance matrix, and
$\#$: one load size
Output: L : a set of loads
$l \leftarrow \phi$, $L \leftarrow \phi$ and $\delta \leftarrow W$;
while $\delta \neq \phi$ **do**
  $l \leftarrow Matching^{Prim}(\delta, E, \#)$;
  $\delta \leftarrow \delta \setminus l$
  print l , " $\Rightarrow$ ";
end
print L ;

Algorithm: LOADMARKET^{MST}

LOADMARKET^{MST} uses a naïve tree covering algorithm for obtaining initial assignment. δ denotes the set of products waiting their assignment to some Load and l denotes a Load which product size is $\#$. $Matching^{Prim}$ adopts Prim's MST algorithm to cut out one feasible Load l with a size of $\#$ from δ . It is fixed point algorithm because when $|W|=\#$, it has a computational complexity $O(\#^2)$ which is the same as that of the partially-ordered set problem (i.e., lattice).

Input: V , E , and $\#$
Output: $U(\subseteq V)$
$T \leftarrow \phi$ and $U = \{1\}$;
while $|U| \neq \#$ **do**
  $\delta \leftarrow \{(u,v) \mid u \in U, v \in V \setminus U\}$;
  $(u,v)^{\min} \leftarrow (u,v)(\in \delta)$ with MIN$(d(u,v)(\in E))$;
  $T \leftarrow T \bigcup \{(u,v)\}$;
  $U \leftarrow U \bigcup \{v\}$;
end
print U ;

Algorithm: $Matching^{Prim}$

2.2 LOADMARKET^{DA}

The main idea of this algorithm lies on the post process to run DA (i.e., double-sided market between Load Traders) to enhance a initial assignment generated by LOADMARKET^{MST} such that each Load can make their TD(Load) shorter.

Input: W : all products, E : distance matrix, and
$\#$: one load size
Output: L : a set of loads
$L \leftarrow \phi$;
$L' \leftarrow LOADMARKET^{MST}(W, E, \#)$
while *Converged* $\neq$ *true* **do**
  $L' \leftarrow DA(L')$;
end
print $L(\leftarrow L')$;

Algorithm: LOADMARKET^{DA}

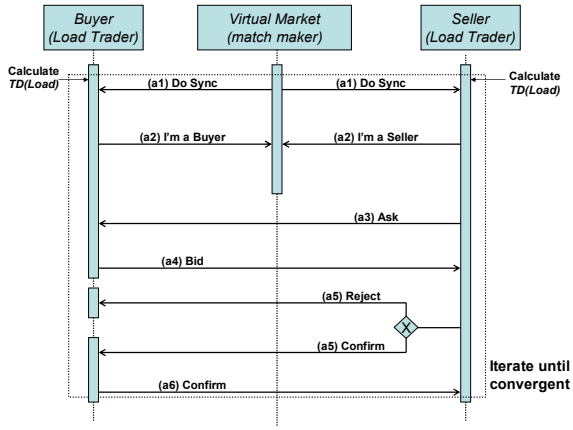

Figure 2(a): DA Protocol

As shown in Figure 2(a), when a new iteration of DA begins, a Virtual Market sends out "Do Sync" signals to all Load Traders. Next, since Load Traders have an option to choose one role from either a buyer or a seller, they pick up one role and respond to the Virtual market. Subsequently, the Virtual market matches buyers and sellers. For matched buyer and seller, they start a negotiation to decide on exchanging their products. If exchanging activity is beneficial to both of them, they immediately conduct the exchanging activity.

All traders have a common policy, that is, *minimize TD(Load)*. With this policy, the DA protocol exchanging process is iterated until the market converges or a pre-defined iteration is satisfied. The iterating DA protocol process is expected to reach emergent behavior that is, a more efficient load assignment (i.e., more reduced travel distance).

2.3 LOADMARKET^{SpotMarket}

Like LOADMARKET^{DA}, it also applies a post process to the initial assignment created by the MST based tree covering algorithm. One thing different from LOADMARKET^{DA} is that it runs afore-defined SpotMarket mechanism which allows only two Load Traders to trade each other. In this context, one Load Trader with the most costly TD(Load) becomes a buyer while the other Load Trader with the least costly TD(Load) becomes a seller. The seller's role is focused on supporting the buyer by allowing the buyer to exchange any product which the buyer wants as long as her loss is not greater

than some threshold. In other words, the seller's policy is to *minimize* $\max_i TD(Load_i)$ such that load balancing on the corresponding LoadMakeup is obtained even though she can take some loss. The definition of LoadMakeup is found in the previous section.

Input: W : all products, E : distance matrix, and
$\#$: one load size
Output: L : a set of loads
$L \leftarrow \phi$;
$L' \leftarrow LOADMARKET^{MST}(W, E, \#)$
while $Converged \neq true$ **do**
  $L' \leftarrow Auction(L')$;
end
print $L(\leftarrow L')$;

Algorithm: LOADMARKET^{SpotMarket}

Compared with LOADMARKET^{DA} which invokes the Load Traders to seek their immediate increased benefit (i.e., greedy behavior), LOADMARKET^{SpotMarket} drives the Load Traders to cooperate each other. Figure 2(b) shows the protocol between the buyer and seller. This SpotMartket protocol exchanging process is iterated until the market converges or a pre-defined iteration is satisfied.

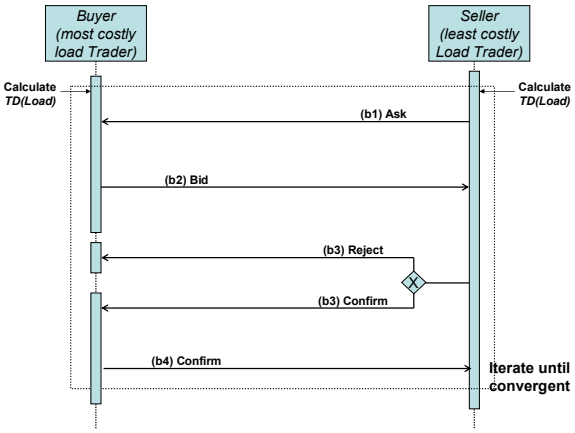

Figure 2(b): SpotMarket Protocol

3 EXPERIMENTAL VALIDATION

3.1 Set-up

To validate the efficacy of our proposals, we conducted experiments using simulation. First, we built a simulator whose scenario is identical to the example scenario in the Introduction section. In other words, the simulator mainly consists of three parts: (1) a plant, (2) a yard and (3) a transportation system (i.e., a set of trucks). Every day, the plant produces 50 cars that move into yard. Each produced car has its delivery destination indexed by city *i*. We assume that the daily available number of trucks is five so that a LoadMackup can be generated daily meaning that ten cars are wrapped in one Load together and contained in one truck. Each truck carries ten cars and delivers them to destination cities.

In order to analyze the effect of the number of cities to visit, we simulated the aforementioned scenario for 3 cases where the number of cities is 50, 100, and 200 respectively. For example, in the case of 50 cities, each city is assigned with unique index between 1 and 50, and the distance between any two cities, *i* and *j* are defined by |*i*-*j*|. For example, city 2 and city 28 has the distance of 26.

At the end, overall, we experimented a combination of 3 different numbers of cities × 3 different algorithms = 9 combinations. Each combination has five replications. Furthermore, we assume that the protocol exchange iteration per a trading is 100. Finally, we note that each Load is assumed as an un-rooted tree for the convenience of analysis. That is, in TD(Load), the distance from the yard to the first and last visiting city is ignored. In addition, a shortest travel path within a Load starts from a city with the smallest index and goes along cities with increasing number of index and ends at a city with the largest index because all cities are assumed to be connected.

3.2 Results

Figure 4(a) and Table 1(a) summarize the average travel distance time of LoadMakeup w.r.t three algorithms over three different numbers of cities, where all algorithms' performance get worse reasonably as the number of cities increases. When it comes to the sample size of the simulation, we generate 18,250 data (i.e., 50 cars per a day × 365 days) with 5 replications using the simulator mentioned in the previous section. Assigning a city index to a generated data (i.e., a car) follows the uniform distribution which is specified by the number of cities.

The distribution LOADMARKET^{DA} slightly outperforms LOADMARKET^{MST} while LOADMARKET^{SpotMarket} performs worse than others. Figure 4(b) and Table 1(b) summarize the standard deviation (STD) with respect to the travel distance between loads within the same LoadMakeup, where all algorithms' performance get worse as the number of cities increases.

Table 1(a): Average Travel Distances

Algorithm	Number of cities		
	50	100	200
LOADMARKET^{DA}	38.35	79.92	162.09
LOADMARKET^{MST}	40.13	81.35	163.72
LOADMARKET^{SpotMarket}	41.23	83.05	166.55

Table 1(b): STD w.r.t Travel Distances

Algorithm	Number of cities		
	50	100	200
LOADMARKET^{DA}	8.24	13.42	23.75
LOADMARKET^{MST}	5.66	10.75	20.26
LOADMARKET^{SpotMarket}	3.43	5.71	10.88

Interestingly, LOADMARKET^{SpotMarket} outperforms other algorithms with a big difference. When the number of city is 200, the difference of STD between LOADMARKET^{SpotMarket} and LOADMARKET^{DA} reaches around 13. This is somewhat intuitive since LOADMARKET^{DA} stimulates Load Traders to act a greedy behavior seeking their immediate increased benefit while LOADMARKET^{SpotMarket} drives the least costly Load Trader to yield the most costly Load Trader its benefit.

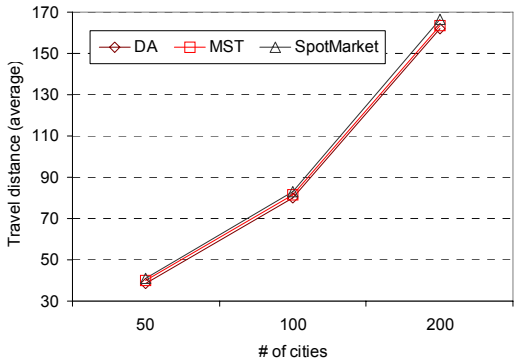

Figure 3(a): Comparison of Travel Distance for Different # of Cities

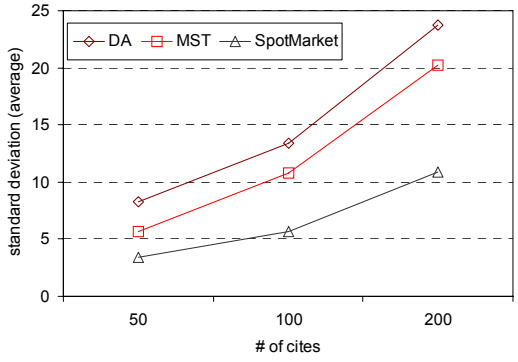

Figure 3(a): Comparison of STD w.r.t Travel Distance for Different # of Cities

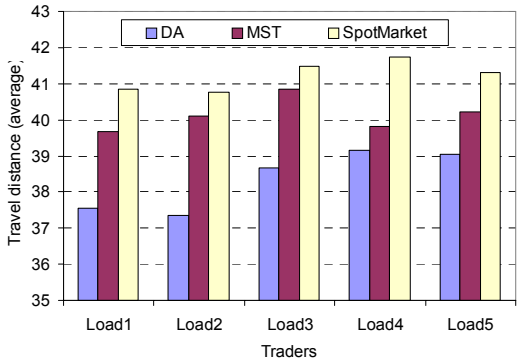

Figure 4(a): Comparison of Average Travel Distance w.r.t Load Traders (# of Cities = 50)

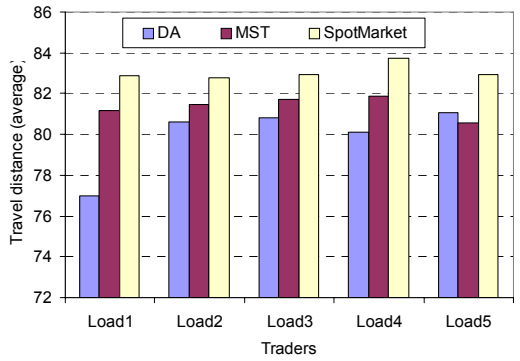

Figure 4(b): Comparison of Average Travel Distance w.r.t Load Traders (# of Cities = 100)

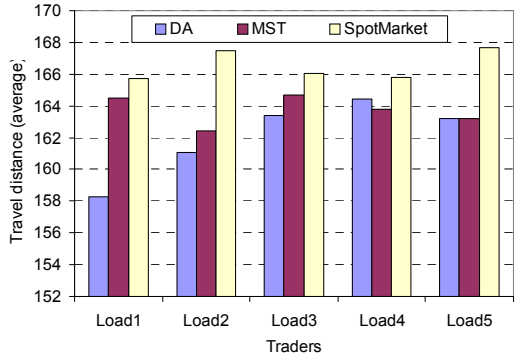

Figure 4(c): Comparison of Average Travel Distance w.r.t Load Traders (# of Cities = 200)

Figure 4(a)-(c) illustrates the average total distance w.r.t each Load Trader on different # of cities. Consistently, LOADMARKET^{DA} outperforms other algorithms but as far as the *Load Balancing* problem is concerned, it performs worse. For example, in the case of 200 cities, the unbalance of travel distance between Load 5 and Load 1 exceeds 6.

4 CONCLUSION

In this paper, we have considered the problem of finding an efficient load assignment, and suggested a solution using (1) Prim's minimum spanning tree algorithm, and (2) market controls such as DA and SpotMarket. Despite its worse performance of average travel distance, LOADMARKET^{SpotMarket} algorithm showed the best performance as far as the balanced load assignment is concerned.

Several directions exist for further research. Rather than the one-shot assignment we dealt with, dynamic load assignment can be considered next. For example, whenever a product is produced from the plant, our proposals can be applied and the load assignment is updated continuously. Also, we can address a problem to find an optimal iteration number of protocol exchange per a trading.

ACKNOWLEDGMENTS

This research has been done with support from General Motors Research and Development Center.

REFERENCES

Clark, G., and Wright, J. W. 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research* 12: 568-581.

Clearwater, S. H., Costanza, R., Dixon, M., and Schroeder, B. 1996. Saving energy using market-based control. In *Market-Based Control - A Paradigm for Distributed Resource Allocation*. World Scientific, Singapore.

Even, G., Garg N., Konemann, J., Ravi, R., and Sinha, A. 2003. Covering graphs using trees and stars. In *Proceedings of RANDOM-APPROX*.

Flood, M. M. 1956. The Traveling Salesman Problem. *Operations Research* 4: 61-75.

Hinkkanen, A., Kalakota, R., Saengcharoenrat, P., Stallaert, J., and Whinston, A. B. 1997. Distributed decision support systems for real-time supply chain management using agent technologies. *Readings in Electronic Commerce*. AW Computer and Engineering Publishing Group.

Jose, R. A., and Ungar, L. H. 1998. Auction-driven coordination for plantwide optimization. In *Proceedings of Foundations of Computer-Aided Process Operation (FOCAPO)*, Snowbird, UT, USA.

Lee, Y.-H. 2002. Market-based dynamic resource control of distributed multiple projects. *Ph.D. thesis, Department of Industrial Engineering. The Pennsylvania State University*, PA, USA.

Prim, R.C. 1957. Shortest connection networks and some generalizations. *Bell System Technical Journal* 36: 1389-1400.

Sauter, J. A., Parunak, H., Van, D., and Goic, J. 1999. ANTS in the supply chain. In *Workshop on Agent for Electronic Commerce at Agents '99*, Seattle, WA, USA.

Sandholm, T. 1993. An implementation of Contract Net Protocol based on marginal cost calculations. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, Washington DC, USA.

Tilley, K. J. 1996. Machining task allocation in discrete manufacturing systems. In *Market-Based Control - A Paradigm for Distributed Resource Allocation*. World Scientific, Singapore.

Yee, S.-T. 2002. Simulation applications in the automotive industry: establishment of product offering and production leveling principles via supply chain simulation under order-to-delivery environment. In Proceedings of the 2002 Winter Simulation Conference, ed. E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, 1260-1268. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

AUTHOR BIOGRAPHIES

SEOG-CHAN OH is a Ph.D. candidate in Industrial and Manufacturing Engineering, since 2002. Before Ph.D. student, he worked as an IT consultant for seven years at Daewoo Information Systems. His research interests include Artificial Intelligence, Multi Agent System and Web Intelligence. He has a Certified Professional Engineer of Information Management from the Korean government. His e-mail address is sxo160@psu.edu.

SHANG-TAE YEE is Staff Research Engineer of Manufacturing Systems Research Laboratory at the General Motor's Research and Development Center in Warren, MI. He has been leading a research program at General Motors that develops and implements real-time enterprise decision systems framework using wireless, advanced software, and decision technology. He received his Ph.D. in Industrial Engineering from the Pennsylvania State University in 1998. He is a member of INFORMS and IIE. His email address is mshang-tae.yee@gm.com.

SOUNDAR R. T. KUMARA is the Distinguished Professor of Industrial Engineering, with joint appointments in the department of Computer Science and Engineering, and School of Information Systems and Technology. His research interests are in: Intelligent systems research with emphasis on sensor based equipment monitoring and diagnosis, Software Agents, and Complexity in Supply Chains. He has over 150 publications. He has won several awards and is an elected member of The International Institution of Production Research (CIRP) skumara@psu.edu.

JEFFREY D. TEW is Group Manager of the Manufacturing Enterprise Modeling group in the Manufacturing Systems Research Laboratory at General Motor's Research and Development Center in Warren, MI. He received his Ph.D. in Industrial Engineering from Purdue University in 1986. He is a member of Alpha Pi Mu, The Association for Computing Machinery, The American Statistical Association, The Institute of Industrial Engineers, The Institute for Mathematical Statistics, INFORMS, The Society of Computer Simulation, and Sigma Xi. His email address is jeffrey.tew@gm.com.