IRis

Northeastern University

Department of Mechanical and Industrial Engineering

January 01, 2004

Modeling operational behavior of a disassembly line

Surendra M. Gupta
Northeastern University

Recommended Citation

Gupta, Surendra M., "Modeling operational behavior of a disassembly line" (2004). . Paper 71. http://hdl.handle.net/2047/d10003381

This work is available open access, hosted by Northeastern University.

Laboratory for Responsible Manufacturing

Bibliographic Information

Kizilkaya, E. A. and Gupta, S. M., "Modeling Operational Behavior of a Disassembly Line", ***Proceedings of the SPIE International Conference on Environmentally Conscious Manufacturing IV***, Philadelphia, Pennsylvania, pp. 79-93, October 26-27, 2004.

Copyright Information

Copyright 2004, Society of Photo-Optical Instrumentation Engineers.
This paper was published in Proceedings of SPIE (Volume 5583) and is made available as an electronic reprint with permission of SPIE. One print or electronic copy may be made for personal use only. Systematic or multiple reproduction, distribution to multiple locations via electronic or other means, duplication of any material in this paper for a fee or for commercial purposes, or modification of the content of the paper are prohibited.

Contact Information

Dr. Surendra M. Gupta, P.E.
Professor of Mechanical and Industrial Engineering and
Director of Laboratory for Responsible Manufacturing
334 SN, Department of MIE
Northeastern University
360 Huntington Avenue
Boston, MA 02115, U.S.A.

(617)-373-4846 ***Phone***
(617)-373-2921 ***Fax***
gupta@neu.edu ***e-mail address***

http://www.coe.neu.edu/~smgupta/ ***Home Page***

Modeling Operational Behavior of a Disassembly Line

Elif A. Kizilkaya and **Surendra M. Gupta***
(*Corresponding Author)

Laboratory for Responsible Manufacturing
334 SN, Department of MIE
Northeastern University
360 Huntington Avenue
Boston, MA 02115, USA.

Tel.: (617)-373-4846
Fax: (617)-373-2921
e-mail: gupta@neu.edu

ABSTRACT

In this paper we present a dynamic kanban (pull) system specifically developed for disassembly lines. This type of kanban system is much more complex than the traditional kanban system used in assembly lines. For instance, unlike the assembly line where the external demand occurs only at the last station, the demands in the disassembly case also occur at any of the intermittent stations. The reason is that as a product moves on the disassembly line, various parts are disassembled at every station and accumulated at that station. Therefore, there are as many demand sources as there are number of parts. We consider a case example involving the end-of-life products. Based on the precedence relationships and other criteria such as hazardous properties of the parts, we balance the disassembly line. The results of the disassembly line-balancing problem (DLBP) are used as input to the proposed dynamic kanban system for disassembly line (DKSDL). We compare the performance of the DKSDL to the modified kanban system for disassembly line (MKSDL), which was previously introduced by the authors. We show, via simulation, that the DKSDL is far superior to MKSDL considered.

Keywords: Disassembly line; DKSDL; JIT; Kanban; MKSDL; pull system; simulation**.**

1. INTRODUCTION

In recent years, the continuous growth in consumer waste has seriously threatened the environment. Realizing this, many countries are contemplating regulations that force manufacturers to take back used products from consumers so that the parts and materials retrieved from the products may be reused and/or recycled. For example, Germany has passed a regulation that requires companies to remanufacture products until the product is obsolete. Japan has passed similar legislation requiring design and assembly methodologies that facilitate recycling of durable goods [2]. Comparable regulations are also being considered in the United States. The two legislative acts that are expected to gain traction are the Automotive Waste Management Act (which will enforce the complete reclamation of automobiles) and Polymers and Plastic Control Act (which will enforce the complete reclamation of polymers and plastics). Also, Massachusetts has recently banned the disposal of computer monitors and TVs into landfills, which bring the end-of-life issues of these manufactured products to the attention of manufacturers and consumers.

The first step in retrieving parts and materials (for reuse, recycling and remanufacturing) from consumer products is disassembly. *Disassembly* is the process of systematic removal of desirable constituents from the original assembly so that there is no impairment to any useful part. Disassembly can be *partial* (product not fully disassembled) or *complete* (product fully disassembled) and may use a methodology that is *destructive* (focusing on materials rather than parts recovery) or *non-destructive* (focusing on parts rather than materials recovery). In this paper, we limit ourselves to complete and non-destructive disassembly.

Environmentally Conscious Manufacturing IV, edited by Surendra M. Gupta,
Proceedings of SPIE Vol. 5583 (SPIE, Bellingham, WA, 2004)
0277-786X/04/$15 · doi: 10.1117/12.580419

Our main focus is in the operational aspect of disassembly and, in particular, scheduling disassembly [1,3,4,5,6,7,16,17,18,19]. Although, the scheduling problems of assembly and disassembly share many characteristics (e.g., the dependent demand concept in discrete parts production systems), they also have their differences, and often the approaches to solve these two kinds of problems are very different [1]. Perhaps the most important difference is in the number of demand sources. In an assembly setting, the parts tend to converge to a single demand source (final product) as they move on the production floor. The governing principles are constrained by this "convergence" property. Under the disassembly setting, as the parts start moving away from their source of origin, they tend to diverge from each other leading to the "divergence" property. In addition to the "divergence property", each part item constitutes a source of demand, and fulfilling the demand of those separate part items cannot be done in an independent manner, since many of these part items share the same procurement source [7,18,19].

In the last two decades, the Just-In-Time (JIT) technique has become very popular and has even spread to small companies [5]. The JIT philosophy evolved from a number of principles such as the elimination of waste, reduction of production cost, total quality control and recognition of employees' abilities. Some advantages of JIT include its simplicity in production scheduling, reduced burden on operators, ease of identification of parts by the Kanbans (Kanban is a Japanese word that means "visible sign" or "card" and is used to control production) attached to the containers and substantial reduction in paper work. The best implementations of JIT happen for companies with high volume and repetitive manufacturing. The JIT manufacturing requires a stable environment, which means that there are no disturbances like sudden increase or decrease in demand, sudden machine breakdowns and late raw material deliveries. In JIT the "pull system" of material flow control occurs which means a work center or station is only authorized to produce when there is a need (demand) for that product in the following operations. Therefore a workstation cannot keep on producing when there is no demand, just to keep busy. Thus, unless a signal is received from the following processes, the work center does not push down new parts. The signaling is done via the Kanban system, which is a control mechanism that coordinates the information and material flow between operations and workstations.

In Traditional Kanban system the number of production and withdrawal Kanbans are kept constant even though it is a known fact that the production control managers at times increase the number of Kanbans in the system on an *ad hoc* basis. The JIT system and the Kanban control mechanism are applicable to stable environments while the classical manufacturing firm has a very unstable environment. In order to deal with these uncertainties of the manufacturing environment, Gupta and Al-Turki proposed a Flexible Kanban system where they systematically increase and decrease the number of production and withdrawal kanbans in the system [8,9]. Following Gupta and Al-Turki's work, other researchers such as Tardif and Maaseidvaag [21], and Takahashi and Nakamura [20] also proposed techniques and showed that manipulating the number of kanbans in unstable manufacturing environments is superior to the traditional kanban system with respect to the backorders and work in process inventory levels. Since the disassembly system is a highly uncertain system due to (1) demand fluctuation, (2) returned products stream from customers (which is the input to the system), (3) parts commonality, (4) condition of parts, (5) different number of the same part in an assembly and (6) other inherent uncertainties, we propose the concept of Dynamic Kanban System for Disassembly Line (DKSDL) to overcome the effect of them.

DKSDL is much more complex than the traditional Kanban system used in production systems. For example, unlike the production system where the external demand normally occurs only at the last station, the demand in the disassembly case can occur at any of the intermittent stations. The reason is that as a product moves on the disassembly line, various parts are disassembled at every station and accumulated at that station. Thus, there are as many demand sources as there are number of parts. This necessitates the use of at least two types of production Kanbans at every station: one type triggered by the individual part demand at the station and the other type triggered by the downstream stations' requirements. This process creates unique and intriguing conditions. We identify the conditions and scenarios unique to such disassembly environment and describe the technique to cope with them. We apply the technique to a case study and describe in detail the use of the Dynamic Kanban System for Disassembly Line. In order to assure the most efficient design we first start with balancing the proposed disassembly line for the case study. We apply the 2-Opt Heuristic for the Disassembly Line Balancing problem (DLBP) introduced by McGovern and Gupta [15] to our proposed end of life product. This way we make sure that we start with a balanced disassembly line for our study. Simulation is used to retrieve results from the application of DKSDL to the case study. We also discuss the advantages and disadvantages of applying the Dynamic Kanban System to the disassembly line.

2. OBJECTIVES

The general objective of this paper is to introduce a newly developed Dynamic Kanban system for disassembly. Specifically, we

- Present a Dynamic Kanban System for Disassembly Line (DKSDL) where following inherent challenges of the disassembly line exist [4]:
  - Changing characteristics of the products disassembled
  - Disassembly Line configurations such as layout, **paced** versus unpaced lines
  - Part Considerations such as; **quality** and quantity of parts in incoming products, early leaving work pieces, self skipping work pieces, skipping work pieces, disappearing work pieces, revisiting work pieces and exploding work pieces
  - Operational challenges such as; stochastic disassembly task times,
  - Variable Demand of multiple parts at various disassembly work stations
  - Assignment Challenges due to precedence relationships, hazardous parts, tasks requiring similar working conditions, tasks requiring special tooling and tasks requiring directional changes.
- Consider a case study where first we balance the proposed disassembly line using the 2-Opt Heuristic for Disassembly and then we introduce the Dynamic Kanban System for Disassembly Line for this proposed case study.
- Compare DKSDL to MKSDL (Modified Kanban System for Disassembly Line [13]).
- Critique the advantages and disadvantages of the proposed DKSDL model

3. DYNAMIC KANBAN SYSTEM FOR DISASSEMBLY LINE (DKSDL)

In a classical push system each station is pushed to disassemble at maximum capacity to fill the parts' inventory buffers whether there is immediate demand for these parts or not. On the other hand, in DKSDL, the disassembled part inventories are kept at a minimum. They are limited by the number of the disassembly Kanbans (DKs), employed in the system at a given time. In DKSDL the number of disassembly Kanbans are varied based on the variable demand rate between a previously calculated base number of DKs and a maximum number of DKs. Products are disassembled in response to the parts' individual demands.

Figure 1 presents a Disassembly Station of a balanced DKSDL composed of *N* disassembly stations in series. Each Disassembly Station (DS) has one processing machine or operator, an input buffer and two types of output buffers. Depending on the end-of life product disassembled and applied Disassembly Line Balancing (DLB) algorithm or heuristic there can be one or more types of parts disassembled at each station. The material flow is controlled via the use of disassembly withdrawal and disassembly production Kanbans. The production Kanban of a DS always circulates within that station, controlling disassembly at that station. The withdrawal Kanbans of a DS always circulate between input buffer of that station and the partially disassembled product buffer of its preceding station.

Disassembly station in DKSDL has two types of production Kanbans, viz. a disassembly production Kanban for partial disassembled unit (DPKAN) due to demand occurring at the succeeding stations and a disassembly production Kanban for the disassembled parts (DPKAND) to satisfy the demand for individual parts at that station. Station *N* has only DPKAND to satisfy the demand for the disassembled parts at that station. The number of production Kanbans at each station limits the number of disassembled parts.

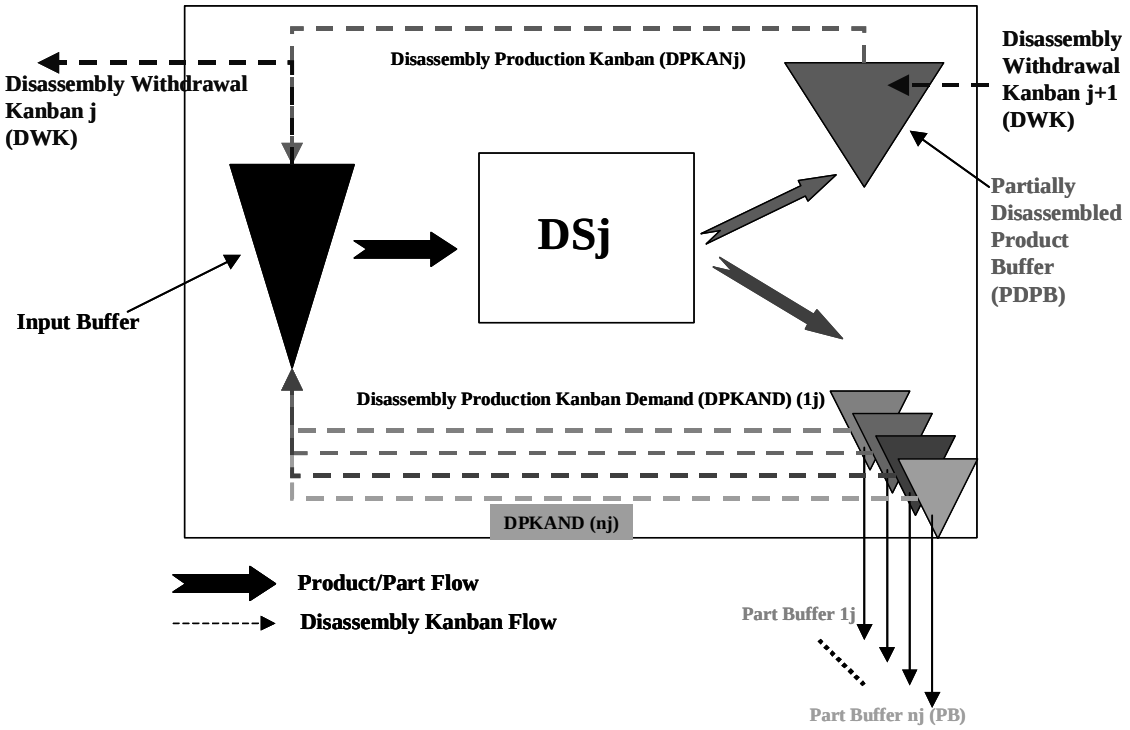

Figure 1. Disassembly Station in DKSDL

There are two types of work-in-process (WIP) (types I and II) in the disassembly system. Type I WIP represents the partially disassembled product anywhere in the system and type II WIP represents the disassembled parts at any station's parts buffers. Parts buffers at station *j* are represented by 1*j*, 2*j*, 3*j*, …., *nj* (See Figure 1).

In the DKSDL, the arrival of demand at any DS for any one of the part(s) triggers a pull action at the preceding DS. Therefore whether there is a need for the preceding DS's part(s) or not, one more product will be disassembled, possibly causing the disassembled part(s) buffer(s) (PB) for those other parts disassembled at that station and disassembled part(s) buffer(s) of the preceding DS to exceed their capacity. On the other hand, if there is no demand for the succeeding DS's parts, the partially disassembled product buffer (PDPB) for the current DS may exceed its capacity. In order to eliminate excess inventory at these buffers, DKSDL disposes additional products and/or part(s) (at some cost) when disassembly Kanban (DK) capacities are exceeded. Figure 1 shows in detail the inner workings of a Disassembly station *j* in DKSDL, in addition to depicting the newly introduced kanban concepts for the disassembly line.

In order to model the disassembly environment more realistically we also had to take into account factors such as quality and quantity of parts in incoming products, stochastic disassembly task times, early leaving work pieces, self skipping work pieces, skipping work pieces, disappearing work pieces revisiting work pieces, exploding work pieces, variable demand of multiple parts at various disassembly work stations as well as the precedence relationships, hazardous parts, tasks requiring similar working conditions, tasks requiring special tooling and tasks requiring directional changes.

Our approach to manipulating the number of production kanbans at each DS has been motivated by Gupta and Al-Turki's studies of an adaptive kanban control system for a production environment where the number of kanbans are varied with respect to the demand and the capacity of the system [9,10,11]. The number of kanbans in their model never reaches below a base number of kanbans and the adjustments to the number of kanbans in the system are done once a period. The limitations of their model due to this inflexibility of not being able to modify the number of kanbans more than once in a period are, potential increase in WIP levels or backorders. Tardiff and Maaseidvaag's study of an adaptive

kanban system allows the numbers of kanbans to be varied based on the current inventory levels and backorders in the system [21]. They assume to have extra kanbans available in the system, which they introduce to the system or take them back, based on pre-calculated capture and release inventory level thresholds. This way they are able to adapt to changing demand levels within a planning period. However their model is limited to a single-stage, single-product environment.

Given that the disassembly environment is a multi-stage and multi-product (part) system, the Dynamic Kanban System for Disassembly Line changes the number of kanbans at each DS with respect to the following developed logic:

A base and a ceiling number of kanbans are calculated for each type of production kanban, namely DPKAN and DPKAND, at every DS *i* and for every part *j*. The number of withdrawal kanbans is set to 1. They are the type of kanbans that trigger the disassembly process between the disassembly stations. As long as there is a production kanban available, the withdrawal kanban can be freed immediately. Because in a disassembly environment, each DS acts independently with its own processing, finished parts queues (PB) and individual demand source points, we need to calculate individual release and capture inventory level thresholds for the two types of production kanbans at each disassembly station. The overall numbers of DPKAN and DPKAND (for each of the parts) at each DS never go above their respective ceiling levels or below the base number of kanban levels.

In our model we not only change the number of production kanbans in the system but also the capacity of the PBs based on the same logic. If the number of DPKAND for a part is increased in a DS, the queue capacity is also increased by the number of parts disassembled at that station. In Tardiff's research the finished queue capacities are assumed to be infinite, which obviously is not be a desirable scenario as it would cause the queues to grow indefinitely for parts and partially disassembled products that are not demanded or have a low demand rate. This is why the parts in our model are disposed of at a disposal cost, after they exceed certain limit.

<u>Calculating the base number of Production Kanbans at each DS:</u>

Base Number of $DPKAND_{ij}$:

The base number of disassembly production kanbans due to demand at the disassembly station is calculated based on the demand rate for each part demanded at that disassembly station, the production rate to retrieve those parts demanded at that DS, cost for processing return for those parts disassembled and finally the holding cost per part retrieved from each product disassembled.

$$N_{DPKAND_{ij}} = \sqrt{\frac{2 \times d_{ij} \times R_i \times P_{ij}}{H_i \times \left| P_{ij} - d_{ij} \right|}}$$

where;

d_{ij}: Demand rate for part type *i* (*i*=1… *k*) in Disassembly station *j=1…N*

P_{ij}: Production rate at station *j* to disassemble part *i* *for i=1…k, j=1…N*

N: Number of disassembly stations

R_i: Return cost of part type *i*

H_i: Holding cost of part type *i*

Base Number of $DPKAN_j$:

In contrast to DPKAND, DPKAN is calculated differently since DPKAN is triggered by the existence of demand in the succeeding workstations. Therefore we had to extend the base kanban formula out to all the succeeding disassembly stations and minus the current stations value from the total so that we do not inflate the base number of kanbans in this station. The current station's DPKAND already account for the existence of demand and inventory levels at that station when we did the calculations previously. There will be no DPKAND for the last DS.

$$N_{DPKAN_j} = N_{DPKAND_{ij+1}} + \ldots + N_{DPKAND_{kN}} - N_{DPKAND_{ij}}$$

where;

N : Number of disassembly stations *where j=1..N*

k: Number of different part types in product

Ceiling of DPKAN$_i$ and DPKAND$_i$:

The ceiling is calculated based on multiple simulation runs and considering that we do not want the model to keep increasing the number of kanbans without control. We want to keep the WIP levels low. A constant factor based on the system's historical data or capacity of the facility can be calculated. Given the constant is calculated then the ceiling levels for the number of production kanbans at each DS becomes:

$$Ceiling_{DPKAN_j} = N_{DPKAN_j} \times constant$$

$$Ceiling_{DPKAND_{ij}} = N_{DPKAND_{ij}} \times constan$$

Disassembled Part Buffer Size is set equal to the kanban capacity:

$$PB_{ij} = N_{DPKAND_{ij}} \times C_i \qquad \text{for } i=1\ldots k \ \& \ \text{for } j=1..\ N$$

where *C* is a constant

<u>Release and Capture Thresholds:</u>

According to Tardiff and Maaseidvaag's study the capture threshold should be less than or equal to the base number of kanbans plus 1. And the Release threshold should be strictly less than the capture threshold. Therefore in our calculations we based the capture and release thresholds to the following.

Capture Thresholds:

$$C_{DPKAN_j} \le N_{DPKAN_j} + 1 \qquad \text{for } i=1\ldots k \ \& \ \text{for } j=1..\ N$$

$$C_{DPKAND_{ij}} \le N_{DPKAND_{ij}} + 1 \qquad \text{for } i=1\ldots k \ \& \ \text{for } j=1..\ N$$

Release Thresholds:

$$R_{DPKAN_j} = C_{DPKAN_j} - n_i \qquad \text{for } i=1\ldots k \ \& \ \text{for } j=1..\ N$$

$$R_{DPKAND_{ij}} = C_{DPKAND_{ij}} - n_i \qquad \text{for } i=1\ldots k \ \& \ \text{for } j=1..\ N$$

where n_i is a constant greater than 0

<u>Capture Logic for production kanbans of a DS:</u>

The proposed capture logic for the production kanbans looks at the inventory levels at the respective PBs of each disassembly station. A variable *N(t)* is calculated for each production kanban type in their respective DSs and the resulting value of this variable is compared to the capture and release thresholds of those production kanbans in that disassembly station. Depending on the results of the comparison either an additional production kanban of that type is introduced to that DS or removed.

Capture Logic DPKAN$_j$

Let us recall that *DPKAN* of an individual DS is the production kanban due to demand generated from the succeeding disassembly stations. Therefore the calculation of *N(t)* will be primarily based on the inventory levels of the succeeding stations and the backorders at those stations. A capture of one of these types of production kanbans will directly affect the succeeding stations inventory levels. In the below formula D_i represents the demand queue at each disassembly station *j* and PB is the buffer where the disassembled parts are stored to satisfy incoming demand. Therefore *N(t)* is the left over amount of parts in PBs after demand is satisfied at a given time *t*. In order to account for the effect of the *DPKAND* we remove the current stations inventory levels from the calculation.

$N_{ij}(t) = (PB_{ij+1} - D_{ij+1}) + ... + (PB_{kN} - D_{kN}) - (PB_{ij} - D_{ij})$ Where *i=1…k and j=1…N*

If $N_{ij}(t) > C_{DPKAN_j} \cap NR(DPKAN_j) < MR(DPKAN_j) \cap MR(DPKAN_j) > N_{DPKAN_j}$

then Decrease $MR(DPKAN_j)$ by *–1*

where;

$NR(DPKAN_j)$: Number of $DPKAN_j$ currently occupied *for j=1…N*

$MR(DPKAN_j)$: Total Number of $DPKAN_j$ at given time in the system *for j=1…N*

C_{DPKAN_j} : Capture threshold for $DPKAN_j$

N_{DPKAN_j} : Base number of $DPKAN_j$

Capture Logic DPKAND$_{ij}$:

Let us recall that *DPKAND* of an individual DS is the production kanban due to demand generated at that disassembly station. Therefore the calculation of *N(t)* is based on the inventory levels at that DS only. We calculate *N(t)* as being the difference of number of finished parts in PB minus the number of demand at the demand queue at any given point in time for each demanded part.

$N_{ij}(t) = (PB_{ij} - D_{ij})$ Where *i=1...k and j=1…N*

If $N_{ij}(t) > C_{DPKAND_{ij}} \cap NR(DPKAND_{ij}) < MR(DPKAND_{ij}) \cap MR(DPKAND_{ij}) > N_{DPKAND_{ij}}$

Then Decrease $MR(DPKAND_{ij})$ by *–1*

where;

$NR(DPKAND_{ij})$: Number of $DPKAND_{ij}$ currently occupied *for i=1…k & for j=1.. N*

$MR(DPKAND_{ij})$: Total Number of $DPKAND_{ij}$ at given time in the system *for i=1…k & for j=1.. N*

$C_{DPKAND_{ij}}$: Capture threshold for $DPKAND_{ij}$

$N_{DPKAND_{ij}}$: Base number of $DPKAND_{ij}$

<u>Release Logic for production kanbans of a DS:</u>

The proposed logic for the increase of the number of production kanbans in a disassembly station is based on the similar variables and criteria for the capture logic but the opposite is true for the conditional statements. Here once the inventory levels fall below the release threshold we want to introduce more production kanbans to the system to be able to satisfy the increase in demand.

Release Logic DPKAN$_j$:

In order to decide to increase the number of $DPKAN_j$ of a DS_j we are interested in finding out when do we need to introduce a new DPKAN to the DS in order to better deal with the increase in demand. To do this the formulation is basically the same with the capture logic. However here we compare N(t) to the release threshold and the ceiling values set to give an upper bound to the number of production kanbans in the system at a given time.

$N_{ij}(t) = (PB_{ij+1} - D_{ij+1}) + ... + (PB_{kN} - D_{kN}) - (PB_{ij} - D_{ij})$ Where *i=1..k and j=1..N*

If $N_{ij}(t) \le R_{DPKAN_j} \cap MR(DPKAN_j) < Ceiling_{DPKAN_j}$

then Increase $MR(DPKAN_j)$ by *+1*

where;

$MR(DPKAN_j)$: Total Number of $DPKAN_j$ at given time in the system *for j=1…N*

R_{DPKAN_j} : Release threshold for $DPKAN_j$

$Ceiling_{DPKAN_j}$: Upper bound for the number of $DPKAN_j$

Release Logic $DPKAND_{ij}$:

For the $DPKAND_{ij}$ of a DS_j we are using the inventory levels with respect to the demand levels at each demand queue and deciding when to increase the number of disassembly production kanbans due to demand at that station. The conditional formula again checks the inventory levels with respect to the release threshold set for that station for the $DPKAND_{ij}$ and the upper bound for that $DPKAND_{ij}$ is used to control the system.

$N_{ij}(t) = (PB_{ij} - D_{ij})$ Where $i=1..k$ *and* $j=1..N$

If $N_{ij}(t) \le R_{DPKAND_{ij}} \cap MR(DPKAND_{ij}) < Ceiling_{DPKAND_{ij}}$

then Increase $MR(DPKAND_{ij})$ by $+1$

where;

$MR(DPKAND_{ij})$: Total Number of $DPKAND_{ij}$ at given time in the system *for i=1…k & for j=1.. N*

$R_{DPKAND_{ij}}$: Release threshold for $DPKAND_{ij}$

$Ceiling_{DPKAND_{ij}}$: Upper bound for the number of $DPKAND_{ij}$

4. PARTS CONSIDERATIONS FOR DISASSEMBLY LINE

One of our objectives in this paper is to study the effects of the separate part issues' impact on the operations of the disassembly line. The following parts considerations exist in a given disassembly line:

4.1 Quality of return product & parts:

When a product is received in a returns warehouse the condition of the returned product and its parts is usually not known. Invariably there are physical or functional defects, which can make the parts not usable. In order to model the quality issues in our model we have introduced a scrap rate for each part disassembled.

SR_i *for i=1..k*

4.2 Quantity of Parts in Returned product:

Today many consumers after purchasing the product make changes or improvements to the product during their usage period. One example of this is the computer RAMs. In order to increase memory size of our PCs every one of us at one point in time have purchased additional RAM and installed it to our PCs. Therefore, even though the expected number of 32 MB RAM is 1 in a given 486 machine, there is a high probability that we may find 2 or more up to available slots in that PC. There is also a chance of finding less than expected number of parts as a consumer may remove the RAM before returning the product. In order to model this phenomenon we have assigned two probabilities to each part in the returned product. Probability of having 1 or more number of parts in the returned product, X% and probability of missing 1 or more of that part in the returned product, Y%.

$$X_i = 1..0$$

$$Y_i = 1..0$$

$$where$$

$$i = 1..k$$

4.3 Skipping Work Pieces (SWP):

In this phenomenon the work piece (subassembly) being worked on does not move down to the disassembly station it should be worked on next but rather moves down to any one of the following disassembly stations. The number of disassembly stations a work piece skips is known as the strength of skipping.

In our case study the following is used to model this phenomenon:

With a probability of;

P_i, WP_j, *goes to* $PDPB_j$ *or* $PDPB_{j+1}$ *or* …$PDPB_{j+N-1}$ *where* $i=1\ldots k$ & $j=1\ldots N$

Disassembly Station 1, Work Piece 1

P_1 probability that WP1 goes to PDPB1
P_2 probability that WP1 goes to PDPB2
P_3 probability that WP1 goes to PDPB3
P_4 probability that WP1 goes to PDPB4

Disassembly Station 2, Work Piece 2

P_1 probability that WP2 goes to PDPB2
P_2 probability that WP2 goes to PDPB3
P_3 probability that WP2 goes to PDPB4

Disassembly Station 3, Work Piece 3

P_1 probability that WP3 goes to PDPB3
P_2 probability that WP3 goes to PDPB4

Disassembly Station 4, Work Piece 4

P_1 probability that WP4 goes to PDPB4
P_2 probability that WP4 goes to A, B, and F Part Buffers

4.4 Disappearing Workpieces (DWP):

Due to one reason or another, some major defect on the rest of the product, a work piece may be taken off the disassembly line permanently and disposed off. This is called the Disappearing Work Piece phenomenon and it will be modeled similar to poor quality parts. We will introduce a scrap rate for each work piece generated at each disassembly station.

$$SR_j \text{ for } j=1..N$$

5. DKSDL CASE STUDY

We selected a smaller scale problem to apply the DKSDL system and talk about the intricacies of the disassembly environment. Even with a problem this size we will show the complexity of the system and reader can see how further more complicated can this problem get as we increase the number of products, number of parts, etc.

Our Case study is based on a disassembly facility, whose sole operation is to disassemble incoming products and to retrieve those valuable parts and satisfy individual part demands. Demand source for these parts may be a manufacturing facility using these parts as raw material in its manufacturing process, or other manufacturers who demand these parts, or other customers who demand these parts as replacement parts.

There is only one product that is returned to this facility, which consists of 10 different parts. The product structure is shown in figure 2. First step in our case study is to design the disassembly line and balance it. In order to do that we would need to define the knowledge base for the example. Knowledge base will define the part precedence relation ships, time to disassemble each part (individual task times), whether a part is hazardous or not, and respective demand rates for each part. In our case study we use the same product McGovern & Gupta used [15]. Table 1 shows the knowledge base.

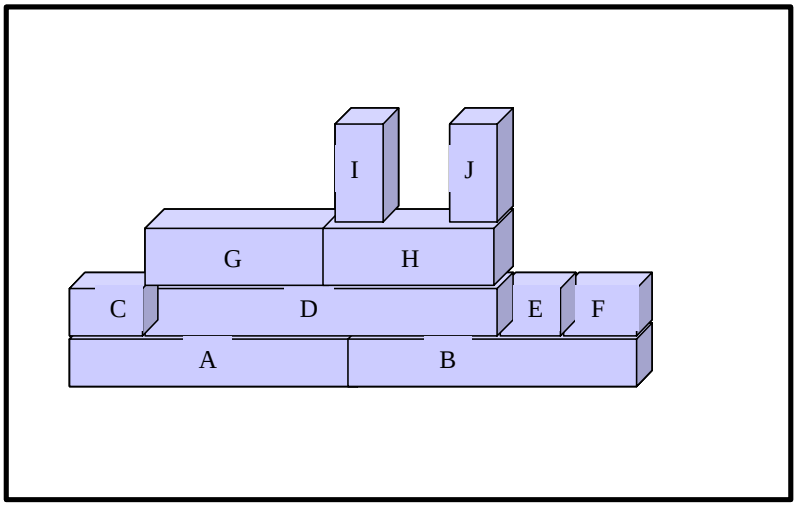

Figure 2. Case Study Product with its respective parts

Table 1. Case Study Product Knowledge Base

Part	**Precedence**	**Task**	**Disassembly Time**	**Hazardous**	**Scrap Rate -Part**	**Demand**
I		1	23	No	0.25	No
J		2	16	No	0.25	750
G		3	18	No	0.25	No
H	I, J	4	20	Yes	0.25	295
C		5	14	No	0.25	No
E		6	14	No	0.25	360
F		7	10	No	0.25	No
D	G, H	8	36	No	0.25	No
A	C, D, E, F	9	10	No	0.25	500
B	C, D, E, F	10	12	No	0.25	No

Returned product arrives to the returns warehouse with respect to Poisson distribution with a mean rate of 8. Capacity of the returns warehouse is assumed to be 100 returned products and anything above and beyond that is disposed at a disposal rate cost to the environment. In reality it is assumed that we never run out of returned products our system. After applying the 2-Opt Heuristic to our Disassembly Line we end up with 5 Disassembly Stations. Table 2 shows which parts are being disassembled at which station what are the disassembly times at each station and the estimated idle times. Idle times are based on a defined cycle time of 40 seconds. This is based on the speed at which the disassembly line is designed to operate at.

Table 2. Disassembly Line Design & Task Assignments

	DS-1	**DS-2**	**DS-3**	**DS-4**	**DS-5**
Parts	I, J	H, E	C, G	D	A, B, F
Disassembly Time (secs)	39	34	32	36	32
Idle Time	1	6	8	4	8

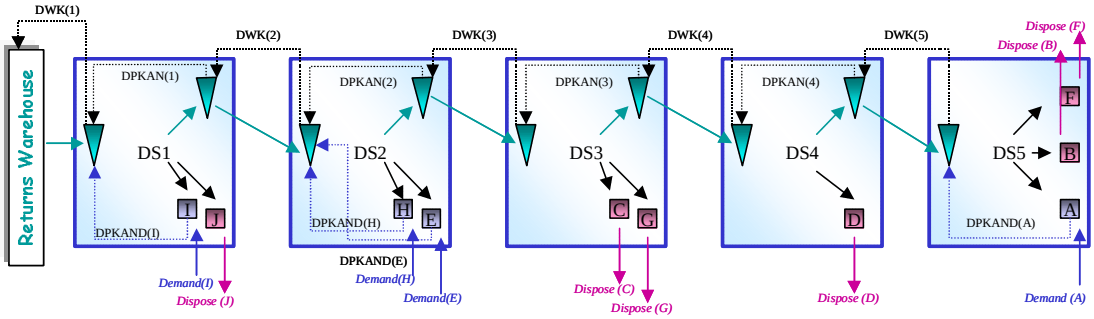

Figure 3. Case Study Balanced Disassembly Line

Each station has one operator that disassembles the product and retrieves that station's demanded part. Our assumption is that the disassembly line is balanced and each workstation has on the average similar processing times. Demand for each part is shown on Table 1.

5.1 Summary of considerations modeled in this case study:

5.1.1 *Parts Considerations*: Quality of the returned product and parts and Quantity of the parts in the returned product.

5.1.2 *Product Considerations:* There is only one product that is returned to this facility, which consists of 10 different parts. (Figure 2.)

5.1.3 *Line Considerations:* The layout of our case study disassembly line is serial, and we are designing an unpaced disassembly line with a cycle time of 40 seconds.

5.1.4 *Demand Considerations:* 4 out of 10 parts being disassembled are demanded. (Table 1.) As you can see from Figure 3 the demanded parts are being disassembled at different DSs and there are DSs, which have to perform the disassembly regardless of absence of demand, due to demand in preceding DSs's parts. For example, to get 1 part A, which is demanded we will have 1 of J, C, G, D, F and B disassembled and disposed.

5.1.5 *Assignment Considerations:* Table 1 shows all the aspects of the case study product for the disassembly task assignments. As you can see the precedence relationships, hazardous parts are taken into account in the 2-Opt Heuristics algorithm to make task assignments to each disassembly station.

5.2 DKSDL Parameters for the Case Study:

We calculated and derived to those parameters presented in Table 3, for our DKSDL model using the previously explained formulation in section 2 and 4.1 of our paper. First column of the table shows the parameters of the model such as base number of kanbans for DPKAND and DPKAN, ceiling values, etc. Columns 2 thru 5 show the individual values for each part for A through J in the case study product. Constant for the buffer capacities is assumed to be 5 across the board for each demanded part. It is assumed to be 0 for those parts, which are not demanded at all. Ceiling constant is assumed to be 4. Since all calculations are based on "hourly rates" and there is an 8-hour shift we will go to 4-hour worth of kanban size as being the ceiling. Detailed calculations for base number of DPKAND and values are presented in Table 4.0 below. We have also only considered

5.3 Results & Analysis:

We have modeled the DKSDL and MKSDL (Modified Kanban System for Disassembly Line) using Arena 8.0 [12]. The models were run 24000 minutes for warm up period and data was collected in the next 480 minutes of the simulation run as one day's production. For the MKSDL we have used the base number of Kanbans calculated for the

DKSDL. Of course in the MKSDL model the number of Kanbans are kept constant and they are not changed. In our experiments we have run we have found out that DKSDL adapts to changing conditions such as operational and parts considerations described in section 3.0 much better then its counterpart MKSDL. In order to compare the two systems without any added variability we have first run the simulation model with constant demand arrival rates. This allowed us to compare the most important criteria demand satisfaction between the two models. Table 5.0 shows the performance of both systems MKSDL and DKSDL under constant demand arrival rates. As you can see DKSDL satisfies more demand in the same period of time compared to MKSDL, which in turn results in more revenues for the facility and satisfied customers.

Table 3. DKSDL Case Study Parameter Setting

Parameter	DS-1, I & J	DS-2, H&E	DS-3, C&G	DS-4, D	DS-5, A&B&F
$N_{DPKAND_{ij}}$	0, 12	31, 17	0, 0	0	11, 0, 0
N_{DPKAN_j}	47	0	11	11	N/A
$Ceiling_{DPKAND_{ij}}$	0, 48	124, 68	0, 0	0	44, 0, 0
$Ceiling_{DPKAN_j}$	188	0	44	44	N/A
PB_{ij}	0, 60	155, 85	0, 0	0	55, 0, 0
$C_{DPKAND_{ij}}$	1, 13	32, 18	1, 1	1	12, 1, 1
C_{DPKAN_j}	48	1	12	12	N/A
$R_{DPKAND_{ij}}$	0, 12	31, 17	0, 0	0	54, 0, 0
R_{DPKAN_j}	47	0	11	11	N/A
$X_{i\ for\ +1}$	0.1, 0.05	0.1, 0.1	0.15, 0.1	0.05	0.05, 0.05, 0.05
$Y_{i\ for\ -1}$	0.1, 0.1	0.1, 0.1	0.1, 0.1	0.1, 0.1	0.1, 0.1

Table 4. DKSDL Base DPKAND Calculations

Item *i*	DS *j*	Processing Time (min)	Production Rate/hr	Demand/day	Demand/hr	Holding Cost/hr	Returns Cost/hr	N-DPKAND
I	1	0.65	39	0	0	0.125	0.125	**0**
J	1	0.65	39	750	94	0.125	0.125	**12**
H	2	0.57	34.2	295	37	0.125	0.125	**31**
E	2	0.57	34.2	360	45	0.125	0.125	**17**
C	3	0.53	31.8	0	0	0.125	0.125	**0**
G	3	0.53	31.8	0	0	0.125	0.125	**0**
D	4	0.6	36	0	0	0.125	0.125	**0**
A	5	0.53	31.8	500	63	0.125	0.125	**11**
B	5	0.53	31.8	0	0	0.125	0.125	**0**
F	5	0.53	31.8	0	0	0.125	0.125	**0**

Once we have achieved superior results with the DKSDL model we have run it under variable demand rates. We have used Poisson distribution with a mean of Demand/hr shown in Table 4.0 for each of the items being disassembled and demanded. We have also compared other criteria under the variable demand experiments since variable demand environment is what we are interested in and it is more realistic to. Table 5.A and Chart 1.0 shows the performance of both systems MKSDL and DKSDL under variable demand arrival rates for each component disassembled. DKSDL yet again outperforms the MKSDL under the variable demand environment as well.

Chart 1.0
Shortage Comparison MKSDL vs. DKSDL

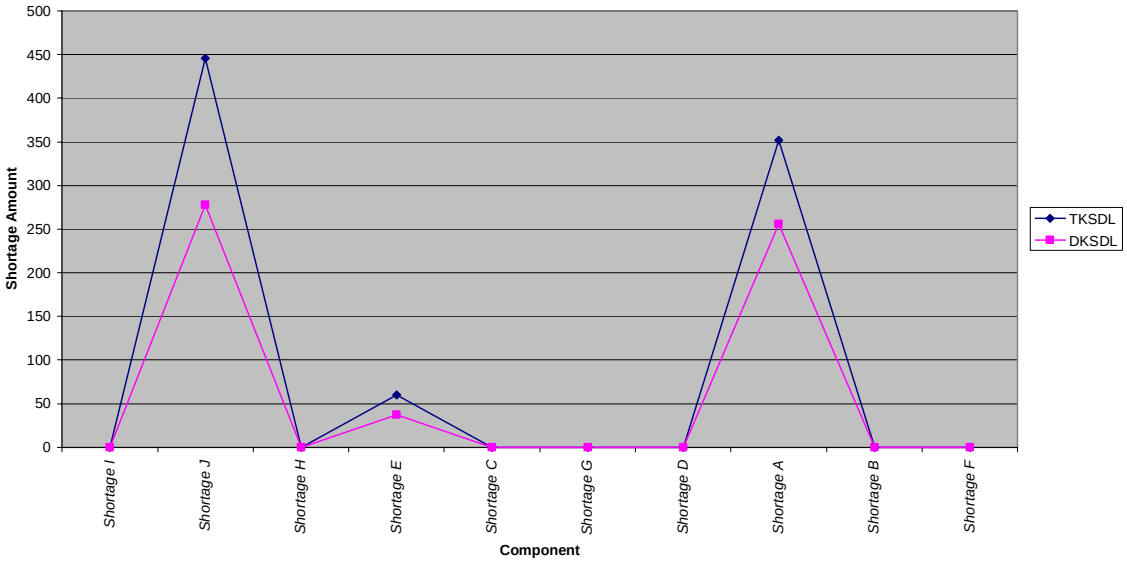

Table 5. Demand Satisfaction		
	MKSDL	DKSDL
Demand I	0	0
Demand J	847	847
Demand H	334	334
Demand E	406	406
Demand C	0	0
Demand G	0	0
Demand D	0	0
Demand A	568	568
Demand B	0	0
Demand F	0	0
Demand I Satisfied	0	0
Demand J Satisfied	264	538
Demand H Satisfied	194	334
Demand E Satisfied	196	384
Demand C Satisfied	0	0
Demand G Satisfied	0	0
Demand D Satisfied	0	0
Demand A Satisfied	160	264
Demand B Satisfied	0	0
Demand F Satisfied	0	0
Shortage I	0	0
Shortage J	583	309
Shortage H	140	0
Shortage E	210	22
Shortage C	0	0
Shortage G	0	0
Shortage D	0	0
Shortage A	408	304
Shortage B	0	0
Shortage F	0	0

Table 5.A Demand Satisfaction-Variable		
	MKSDL	DKSDL
Demand I	0	0
Demand J	890	823
Demand H	336	327
Demand E	399	416
Demand C	0	0
Demand G	0	0
Demand D	0	0
Demand A	598	546
Demand B	0	0
Demand F	0	0
Demand I Satisfied	0	0
Demand J Satisfied	444	545
Demand H Satisfied	336	327
Demand E Satisfied	339	379
Demand C Satisfied	0	0
Demand G Satisfied	0	0
Demand D Satisfied	0	0
Demand A Satisfied	246	290
Demand B Satisfied	0	0
Demand F Satisfied	0	0
Shortage I	0	0
Shortage J	446	278
Shortage H	0	0
Shortage E	60	37
Shortage C	0	0
Shortage G	0	0
Shortage D	0	0
Shortage A	352	256
Shortage B	0	0
Shortage F	0	0

Table 6.0 shows the parts considerations we had in variable demand scenario for the MKSDL and the DKSDL models. In general we had a higher number of scrapped components for the DKSDL model as well as higher number of times when the disassembly was completed the operators found that the demanded component was missing (Count Yi). Both MKSDL and DKSDL had comparable amounts of extra parts when disassembled as also shown on Table 6. (Count Xi) Regardless of stochastic behavior of the environment and the returned product, DKSDL model adopted faster and still outperformed the MKSDL by satisfying a higher amount of the demand.

Table 7. WIP Level-Variable		
	MKSDL	**DKSDL**
WIP Type I	189	216
WIP Type II	75	123

Table 7.0 shows the WIP Type I and II levels which shows that DKSDL has slightly more WIP levels than MKSDL. However as you can see this is not a major increase. This is an expected behavior for the DKSDL model since in order to satisfy more of the demand the dynamic kanban module increases the number of production kanbans and hence we end up with higher average levels of WIP inventories.

Table 6. Part Considerations-Variable		
	MKSDL	**DKSDL**
Scarp I	99	131
Scrap J	90	114
Scarp H	44	53
Scrap E	51	61
Scrap C	10	11
Scrap G	11	15
Scrap D	11	15
Scrap A	1	9
Scrap B	4	16
Scrap F	4	10
countXI (+1)	38	41
countXJ	17	33
countXH	18	24
countXE	24	19
countXC	4	6
countXG	8	5
countXD	5	1
countXA	2	1
countXB	1	0
countXF	1	1
countYI (-1)	30	49
countYJ	38	49
CountYH	20	24
countYE	14	19
countYC	4	7
countYG	4	3
countYD	2	2
countYA	2	3
countYB	2	4
countYF	2	10

Table 8. Disposal Quantities-Variable		
Component		
	MKSDL	**DKSDL**
Disposal I	420	518
Dsposal J	0	0
Disposal H	0	0
Disposal E	0	7
Disposal C	260	278
Disposal G	255	279
Disposal D	260	283
Disposal A	0	0
Disposal B	242	281
Disposal F	242	280
Total	**1679**	**1926**
Partial Disassembly		
	MKSDL	**DKSDL**
Product Disp Station 1	0	0
Product Disp Station 2	161	254
Product Disp Station 3	147	180
Product Disp Station 4	1	1
Total	**309**	**435**

Table 8.0 shows the disposal quantities for both models under variable demand scenario. Overall both the component and partially disassembled product disposal rates are slightly higher for the DKSDL compared to MKSDL. This is due to the fact that we are disassembling more products in order to satisfy the demand and hence we end up with more of the not demanded components, which are disposed off once the buffer sizes are reached. However if you see the disposal rates for the components they are comparable 1679 for MKSDL versus 1926 for DKSDL. On the other hand the partially disassembled product rate is much more higher for DKSDL for DS 2 and 3. This is due to the fact that DS 3 and 4 have components, which are not demanded. As you can see in both models the higher amount of partially disassembled

product disposals are higher at the no demand stations, however it is higher on the DKSDL due to the fact that DKSDL increases number of production kanbans to satisfy more of the component A at DS 5 hence causing timing issues and disposals for the partially disassembled product.

6. CONCLUSIONS

In this paper we have introduced a new model for shop floor control for a disassembly environment called the DKSDL. We have detailed the intricacies of the model. We have also compared the DKSDL model to our previously proposed MKSDL model. In our previous work we had shown that the MKSDL model out performs the Push System. In this paper however, we have shown that our improved model, which modifies the number of production kanbans with respect to demand levels and WIP inventory levels, performs better than the MKSDL model. We have run two sets of experiments with multiple days of data collection to support this finding.

REFERENCES

[1] Brennan, L., Gupta, S. M., and Taleb, K. N., "Operational Planning Issues in an Assembly/ Disassembly Environment", *International Journal of Operations and Production Management*, Vol.14, No.9, 57-67, 1994.

[2] Guide V.D.R, Srivastava, R. M., Spencer, S., "Are Production Systems Ready for The Green Revolution", *Production and Inventory Management Journal*, Vol.37, No.4, 70-76, 1996.

[3] Gungor, A. and Gupta, S. M., "Issues in environmentally conscious manufacturing and product recovery: a survey", *Computers and Industrial Engineering*, Vol. 36, No.4, 811-853, 1999.

[4] Gungor, A. and Gupta, S. M., "Disassembly line in product recovery", *International Journal of Production Research*, Vol. 40, No.11, 2569-2589, 2002.

[5] Gupta, S. M. and Brennan, L., "Implementation of Just-In-Time Methodology in a Small Company", *Production Planning and Control*, Vol.6, No.4, 358-364, 1995.

[6] Gupta, S. M. and McLean, C. R., "Disassembly of Product", *Computers and Industrial Engineering*, Vol.31, No.1-2, 225-228, 1996.

[7] Gupta, S. M. and Taleb, K. N., "Scheduling Disassembly", *International Journal of production Research*, Vol.32, No.8, 1857-1866, 1994.

[8] Gupta, S. M., Al-Turki, Y. and Perry, R. F., "Flexible kanban system", *International Journal of Operations and Production Management*, Vol. 19, No. 10, 1065-1093, 1999.

[9] Gupta, S. M. and Al-Turki, Y., "An algorithm to dynamically adjust the number of Kanbans in stochastic processing times and variable demand environment", *Production Planning and Control*, Vol. 8, No.2, 133-141, 1997.

[10] Gupta, S. M. and Al-Turki, Y., "Adapting just-in-time manufacturing systems to preventive maintenance interruptions", *Production Planning and Control*, Vol. 9, N0.4, 349-359, 1998

[11] Gupta, S. M. and Al-Turki, Y., "The effect of sudden material handling system breakdown on the performance of a JIT system", *International Journal of Production Research*, Vol. 36, No.7, 1935-1960, 1998

[12] Kelton, D., Sadowski, R. P. and Sadowski, D. A., "Simulation with ARENA", McGraw-Hill, New York, NY, 1998.

[13] Kizilkaya, E. and Gupta, S. M., "Material flow control and scheduling in a disassembly environment", *Computers and Industrial Engineering*, 35(1-2), 93-96, 1998

[14] Law, A. M. and Kelton, D., "Simulation Modeling and Analysis", McGraw-Hill, New York, NY, 1991.

[15] McGovern, S. M. and Gupta, S. M., "2-Opt Heuristic for the Disassembly Line Balancing Problem", *Proceedings of the SPIE International Conference on Environmentally Conscious Manufacturing III*, Providence, 71-84, October 29-30, 2003.

[16] Moore, K. E. and Gupta, S. M., "Stochastic colored Petri net (SCPN) models of traditional and flexible kanban systems", *International Journal of Production Research*, Vol. 37, No. 9, 2135-2158

[17] Moyer, L. and Gupta, S. M., "Environmental Concerns and Recycling/Disassembly Efforts in the Electronics Industry", *Journal of Electronics Manufacturing*, Vol.7, No.1, 1-22, 1997.

[18] Taleb, K. and Gupta, S. M., "Disassembly of Multiple Product Structures", *Computers and Industrial Engineering*, Vol.32, No.4, 949-961, 1997.

[19] Taleb, K., Gupta, S. M. and Brennan, L., "Disassembly of Complex Products with Parts and Materials Commonality", *Production Planning and Control*, Vol.8, No.3, 255-269, 1997.

[20] Takashi, K. and Nakamura, N., "Agile control in JIT ordering systems", *International Journal of Agile Management Systems*, Vol. 2/3, 242-252, 2000.

[21] Tardif, V. and Maaseidvaag, L., "An Adaptive approach to controlling kanban systems", *European Journal of Operations Research*, Vol.132, 411-424, 2001.