

## **A TWO-TIER METHOD FOR EVALUATING ALTERNATIVE POLICIES TO SUPPORT INTERACTIVE ANALYSIS OF 3D MATERIAL FLOW SIMULATIONS**

Wilhelm Dangelmaier  
Kiran Mahajan  
Daniel Huber  
Bengt Mueck

Heinz Nixdorf Institute  
University of Paderborn  
Fürstenallee 11, 33102 Paderborn, GERMANY

### **ABSTRACT**

Discrete-event material flow simulation tools have long been offering real-time 3D visualization. This feature allows less experienced users to analyze the underlying system. Beyond this, visualization is not used to interact with the simulated (underlying manufacturing) system to improve or control the material flow, especially under disturbances. This paper presents a simulation based 2-tier framework, which seeks to control or improve material flow by means of real-time user immersive visualization. The first tier uses static optimization to compute the material flow by selecting from a large number of alternative policies based on deterministic disturbances. The second tier is a reactive algorithm which computes solutions for probabilistic disturbances. The results of the two tiers are used for interacting with the underlying system using visualization. We show that the proposed system is able to handle complex alternative policies, which supports interactive analysis of 3D material flow simulations.

### **1 INTRODUCTION**

These days, manufacturing systems are getting more and more complex. Simulation and visualization can help in understanding the underlying manufacturing system very well. However, it is not possible today, to control the material flow and results of the simulated system using bi-directional interactive analysis between the simulated system and visualization. An integration of real-time decision-making algorithms and virtual reality would provide advantages like the ability to visualize the entire plant, whilst still being able to control and analyze it real-time by interacting with the system in the event of disruptions or unforeseen events. This calls for an integration of existing and new work in areas like simulation and visualization, and real-time production scheduling.

There have been numerous works, which focus on real-time scheduling of the material flow based on simulation, some of which are referenced here. (Harmonosky et al. 1997) present their work in the areas of real-time selective re-routing and scheduling algorithms based on simulation. They iteratively use simulation as a tool to find out the best policy from a set of alternative policies in real-time. (Chong, Sivakumar 2003) mention about reactive and predictive simulation based scheduling for dynamic manufacturing. They also use on-line simulation to evaluate the selected approaches and the corresponding schedules to determine the best solution. The selected solution was used until the deviation of actual performance from the estimated one exceeded a given limit. (Leon, Wu and Storer 1994) use a graph based approach to control a flexible manufacturing system for predictive system disruptions. They assumed the disruptions to be deterministic, and act a priori to adjust the original schedule. While the simulation-based approach for real-time scheduling seems very intuitive, efforts are required to integrate these algorithms to provide decision making abilities in 3D visualizations.

Using simulation as a tool for real-time scheduling presents several benefits and drawbacks. The most important benefit is that simulation can provide accurate information about a certain policy. Secondly, simulation proves to be effective when the system behavior is probabilistic, and cannot be easily captured by analytic methods. The ability to provide accurate information about a certain policy becomes a drawback when it comes to modeling a complex manufacturing system. As the underlying manufacturing system gets bigger and as the number of “control points” (location where a decision to handle a part has to be taken) increase with each point providing several alternatives, the more difficult it is to employ simulation, *especially in the real-time virtual world* to obtain information about the best alternative policy. As a small example, three control points (one after the other) with three decisions

each can result in a minimum of 9 alternatives! In such cases, simulation can prove to be very time consuming to deliver a decision. Secondly, the length of the look-ahead planning horizon, i.e. the length of the simulation run used to compare a set of alternatives, has little scientific backing. The more time each evaluative simulation runs, the longer it will take to make scheduling and control decisions, thus failing to provide decisions in “real-time”.

In the areas of simulation and visualization, (Mueck et al. 2003) report the development of a real-time walk-through system coupled with a simulation system along with bi-directional information flow. The motive was to provide a completely immersive environment for the user, allowing him to interact with a certain process. However, the indications to lead the user to a *significant process/object* (a process which can recover the system from disturbances) were lacking. The user was not guided to the *significant process* automatically. In other words, the user was unaware of events in the simulation, and the effect of those events on the performance of the system. Apart from the identification of the *significant process*, the user had no means to employ corrective policies in the virtual world.

In this paper, we describe a framework to integrate work in these areas and discuss related issues. Besides a general framework, we specifically take an example problem and solve it going through some of the phases of the framework. In brief, the system will pre-calculate the material flow using best policies (from a range of alternative policies), based on deterministic input before hand by performing a static calculation. The material flow is then visualized in real-time and pre-simulated offline to handle disturbances. Section 2 describes the assumptions in our work. Section 3 shows the overall framework of our methodology.

Section 4 describes an example manufacturing system and some of the modeling work. Section 5 explains the static calculation and results of applying the algorithm on the example. Section 6 then describes pre-simulation and visualization followed by showing some results obtained by implementation (using Technomatix *eM-Plant*). Preliminary results demonstrate the feasibility of the concept for a typical manufacturing problem. The paper is concluded by mentioning new challenges and further research directions.

## 2 ASSUMPTIONS

In this work, we do not yet consider the connection of the physical manufacturing system to the visualized simulation. This implies, that we do not consider process and yield variations, etc originating from the physical manufacturing system and methods to bring back the deviations using control. All disturbances we consider result from either orders (jobs) and machine breakdowns in the virtual world. We specifically consider mass manufacturing environments with limited product variety.

## 3 AN OVERALL FRAMEWORK

Figure 1 shows the overall framework of the envisaged system. The user has primarily two tasks. The first is to model the manufacturing system with all the (deterministic) parameters. Here, we let the user define the alternative control policies at so called “control points” in the simulation model. We also let the user decide which KPI’s (Key Performance Indicators: make-span, costs, work in progress and equipment utilization) he wishes to achieve throughout the lifetime of a particular order. The user is

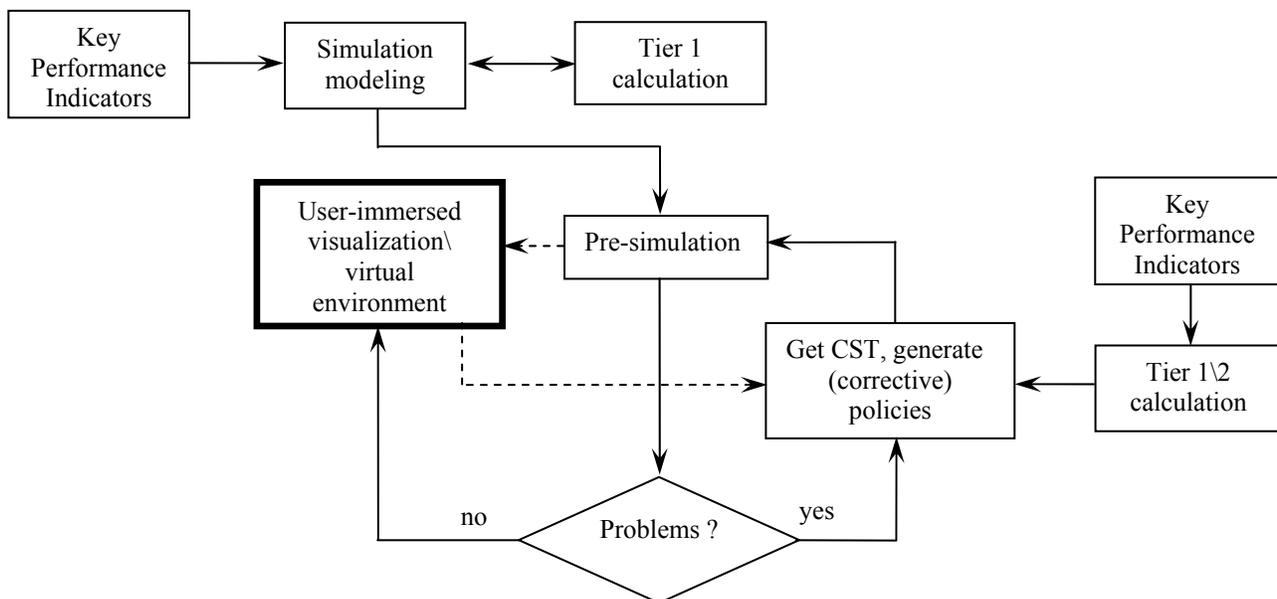


Figure 1: Overall Framework for Interactive Analysis in Virtual Environments

typically given an option of setting priorities over the KPI's. All this information is modeled during simulation modeling and passed on to the tier 1 calculation. Based on this information, the tier 1 calculation will compute a sub-optimal material flow to be used in the system, which includes part routing decisions. The calculation considers production rates, buffer sizes and current production loads (if any) to compute the flow. This flow is again given back to the simulation model (Figure 1). This material flow is calculated in such a way that it seeks to avoid problems like bottlenecks. Section 5 explains a method to do this. The tier 1 calculation primarily seeks to reduce complexity due to the alternative policies and to compute control policies for deterministic system disturbances.

Then, this material flow is pre-simulated in an offline fashion and visualized in a 3D environment. The visualization is the place where we seek to control and analyze the underlying system (bold box in Figure 1). The pre-simulation is done for three reasons. First, to “drive” the visualization (as is done traditionally), secondly, to validate tier 1 and tier 2 calculations, and thirdly, to simulate the effect of disturbances which might arise in due course of time. As seen in Figure 2, the pre-simulation, at time  $t_{Sim}$ , is a fast simulation, and leads the user-immersed real-time visualization, at time  $t_{VR}$ , (from now on real-time visualization is termed as *virtual simulation*) by time  $\Delta T$ . Note however, that the pre-simulation is started from the same starting state as the virtual simulation (more details in next paragraph). The pre-simulation and the virtual simulation together handle disturbances in the system. The disturbances we consider are deterministic (arrival\cancellation of orders and machine maintenance (breakdown) schedules) and probabilistic (arrival\cancellation of new orders and machine breakdowns). Table 1 references the work that has been done to handle these disturbance types by other researchers. Assume, that at time  $(t_{Sim} + x)$  or  $(t_{Sim} - x)$ , both  $> t_{VR}$ , information about a new order is available (deterministic disturbance).

Table 1: Disturbances Considered and References

Disturbance type	References
Machine Breakdowns	Akturk and Gorgulu 1999; Hormonosky et. al 1997, Yamamoto and Nof 1985; Leon, Wu and Storer 1994
Rush (hot) orders	Abumaizar and Svestka 1997; Jain and Elmaraghy 1997; Kim and Kim 1994
Order cancellation	Abumaizar and Svestka 1997; Jain and Elmaraghy 1997; Li, Shyu, and Adiga 1993

The pre-simulation stops after it goes into a complete irreversible damage state (due to the new order, assuming that the new order has completely different characteristics) at time  $(t_{Sim} + z)$ . In other words, the pre-simulation determines the effect of the new order\disturbance. Meanwhile the virtual simulation continues, and tier 1 part of the calculation is then invoked which determines the material flow (resulting in corrective policies) based on the current (virtual) simulation state (CST), the KPI, the effect of the new order arrival (estimated by the pre-simulation), and the available alternative policies. On the other hand, when the disturbance is sudden machine failure (probabilistic disturbance, detected at, for instance, time  $t_{VR}$ ), the virtual simulation continues, while the pre-simulation now comes back to the state at  $t_{VR}$  ( $=t_{VR}^1$ ) (Figure 2) and rapidly simulates until the time where the pre-simulation goes into an irreversible damage state, due to the machine breakdown. The tier 2 calculation is then invoked which again considers the CST, KPI, the effect of the machine breakdown (obtained by the pre-simulation), and alternative policies to compute possible alternative part routes or corrective policies (refer Figure 1). An example of the irreversible damage state is the state when a buffer is full and the preceding machine stops, because it can no longer send parts to the buffer. More on the issues related to considering the CST

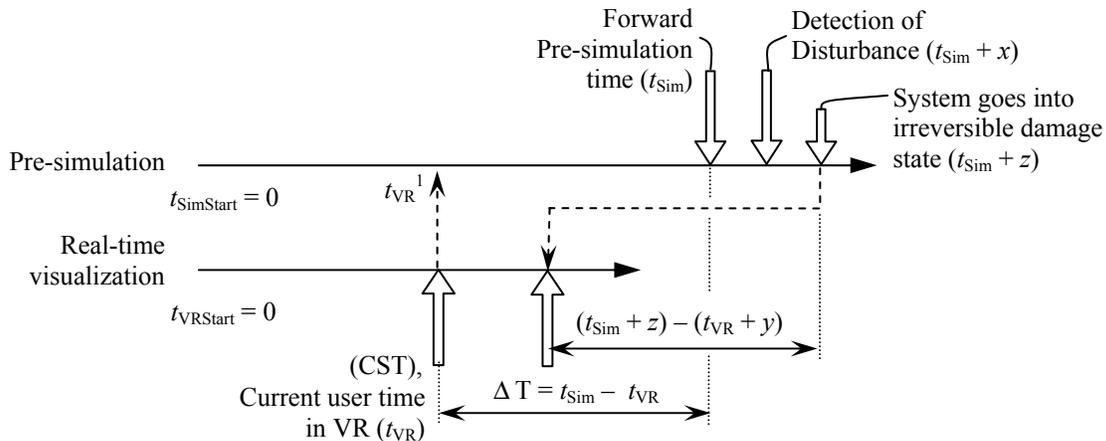


Figure 2: Time-Lag Approach to Detect Disturbances and Apply Corrective Policies

are discussed later. Now, we offer the following conjectures to the reader to help things fall in place: *Conjecture 1*: All deterministic disturbances will be detected by the pre-simulation mechanism, and will be handled by the tier 1 calculation phase. *Conjecture 2*: All probabilistic disturbances will be detected by the virtual simulation, and will be handled by the tier 2 calculation phase.

The computed corrective policies are then presented to the user. The sub-problems are to determine, by how much time does the pre-simulation lead the virtual simulation ( $\Delta T$ ), when does the user have to apply the corrective policy  $((t_{sim} + z) - (t_{VR} + y))$  (dotted lines with arrows in Figure 2) in the virtual world for all cases of disturbances, and for how long. It is also interesting to investigate if we could apply corrective policies as late as possible, with a view to keep the system as stable as possible, whilst still achieving the desired performance. The answer to when the user has to apply the corrective policies will have to be obtained not only by considering the CST, but also the information from the pre-simulation about how the virtual simulation will evolve over time.

When the user has these answers, his second task starts. A particular corrective policy will lead him to a certain location – a machine, a buffer, etc (this is explained in more details in the next section), where he navigates to in the virtual factory. This location can now be termed as a *significant process* that (Mueck et al. 2003) referred. We anticipate that when the number of corrective policies is too large, it may be wise to implement them at once, instead of navigating at numerous locations. The navigation leads to an immersive experience for the user. Situations where the user has to navigate through large physical distances between locations can also be handled in this setting, by stopping (and restarting) the virtual simulation between navigating to the locations. After implementing the corrective policy (assume, at time  $((t_{sim} + z) - (t_{VR} + y))$ ) (shown by dotted lines with

arrows in Figure 2), the virtual simulation state and time is sent to the pre-simulation mechanism with the corrections (Figure 1). Again, both the virtual simulation and the pre-simulation mechanisms simulate the flow (starting both with state at time  $((t_{sim} + z) - (t_{VR} + y))$ ), but with the addition that they now include the influence of the user’s corrective policy. The time-lag mechanism repeats as explained earlier. On the other hand, when there is no problem, the user continues to navigate in the system in a supervisory role using the walk-through system. In the following sections, the tier 1 calculation is explained in detail for a specific problem. Tier 2 calculations are briefly addressed at the end of the paper. The next section shows the example to demonstrate the concept of alternative policies.

#### 4 AN EXAMPLE MANUFACTURING SYSTEM

Figure 3 shows a manufacturing system with CP1, and CP2 representing the control points where decisions are to be made. The control points can be implemented as objects in the simulation system. Thus, a control point can be anything from a forklift to a machine. The user is free to define the location and the definition of control policies. Alternative policies can be anything from an alternative material flow route within the factory or specific instructions for a specific part, to different steps of processing speeds for a machine. Alternative policies can be standard or special. Standard policies include, for instance, material flow routes which cannot be changed, due to technological constraints. Policy 1 in CP1 is of type standard. Policy 2 in CP1 is of type special, because a decision needs to be made here. Similarly, CP2 contains only special policies. In this work special policies are modeled separately, while standard policies are “hard coded” during implementation. This means the system is able to handle both these policy types simultaneously.

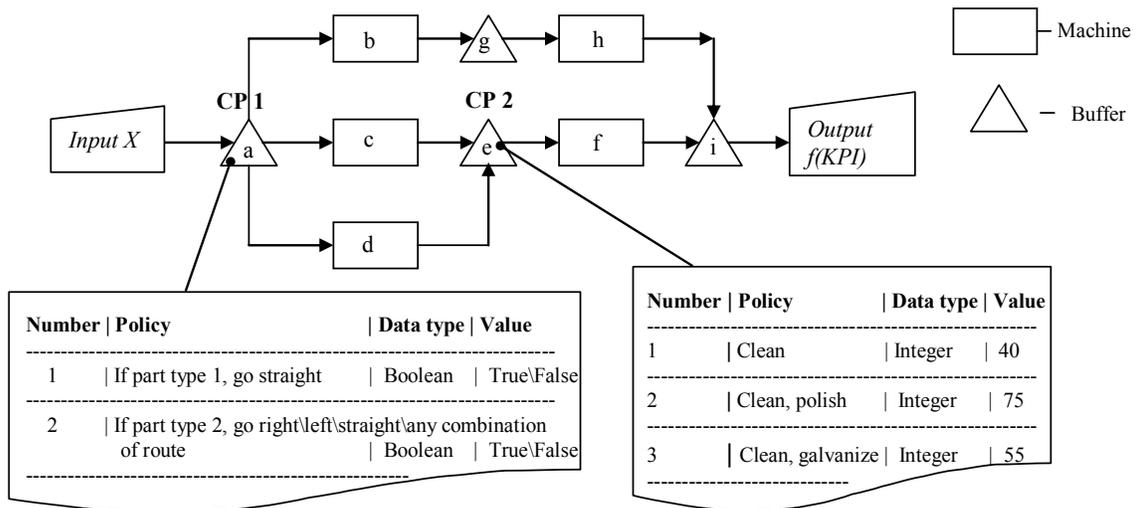


Figure 3: A Typical Manufacturing System with Standard and Special Policies

Table 2 and 3 show the data like machine processing speeds, buffer sizes used for the system along with standard and special policy definitions. These policy definitions when expanded, will result in a tree of alternative policies, which is approximated here as a network of alternative policies. Refer to the network of alternative policies in Figure 4 to read the policy definitions in Table 3.

It can be seen that the network closely resembles the physical system configuration with the addition that if a machine has three policies to handle a part, then the machine is replicated thrice in the form of a node to reflect the three policies. Modeling the policies this way, makes it possible for the virtual user to relate a policy to one generic object (a machine, for instance). Hence, in Figure 3, machine d, and stock point e are shown as  $d_1, d_2, d_3$  and  $e_1, e_2,$

and  $e_3$ , respectively in Figure 4. As seen in Figure 4, time, cost and production load derivatives required for handling one unit of flow in node  $j$  (which can, for instance be a machine) are placed on the preceding arc  $i \rightarrow j$ . It is assumed that each policy will result in cost, time derivatives, while the production load derivatives will depend on number of parts assigned less the number of parts completed at any instant of time. Wherever it is not possible to place time and/or cost factors, either of these will be reduced to zero. If the underlying manufacturing system has more than one source and more than one sinks, then this can be modeled by placing dummy arcs from the *super-source* and *super-sink* to the several sources and sinks respectively and further adding zero costs, time and production load derivatives. The next section explains the tier 1 calculation for the example discussed so far.

Table 2: Data for Model

Object	Property	Value	Unit
a (Buffer)	Capacity	50	No.
b (machine)	Processing speed	120	Sec
c (machine)	Processing speed	80	Sec
e (Buffer)	Capacity	50	No.
f (machine)	Processing speed	60	Sec
g (Buffer)	Capacity	80	No.
h (machine)	Processing speed	70	Sec
i (Buffer)	Capacity	40	No.
Product	Quantity	1000	No.
Part inter-arrival	Constant	20	Sec
Part handling	Sequence	FIFO	

Table 3: Data on Policies

Policy definition		Type
300 products are routed $a \rightarrow b \rightarrow g \rightarrow h \rightarrow i$		Standard
280 products are routed $a \rightarrow c \rightarrow e_3 \rightarrow f \rightarrow i$		Standard
At node e, products can either be cleaned (policy $e_1$ ), cleaned and polished ( $e_2$ ) or cleaned and galvanized ( $e_3$ ) with processing times of 40, 75, and 55 seconds respectively		Special
At node d, products can be processed with any one of the three processing speeds, of 90 (policy $d_1$ ), 30 (policy $d_2$ ), 70 (policy $d_3$ ) seconds.		Special
At node a, 420 products can take either or any policy b, c, $d_1, d_2, d_3$ .		Special

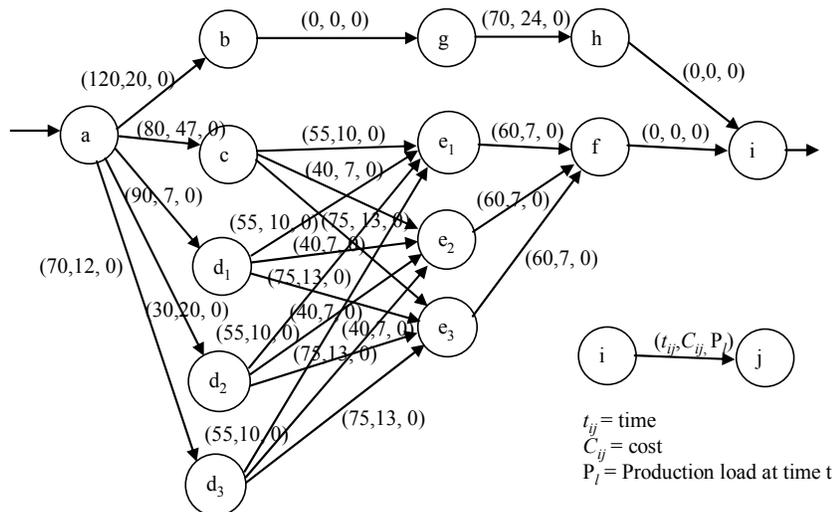


Figure 4: Network of Policies

**5 TIER 1 CALCULATION:  
MATHEMATICAL MODEL**

The tier 1 will compute a sub-optimal material flow based on the user KPI's like time (make-span) and costs. Other KPI's are discussed later in this section.

For the moment, no production loads on the factory are considered. The following are the known input to the mathematical model:

- time  $t$  required for each part  $a$  on machine  $j$ .
- number of parts  $n$ .
- cost  $k$  of operating each machine  $j \rightarrow$  hourly rate.
- number of buffers each with capacity  $B_{fi}$ .
- production rates  $\xi_{in(i)}$  and  $\xi_{out(j)}$  (measured in parts\min) before and after nodes  $i$  and  $j$  respectively.
- selective routing of parts  $n_p$ ,  $0 \leq n_p \leq n$ , through the factory due to technological constraints of manufacturing  $n_p$  parts in the factory, if any.

The cost of operating each machine is obtained from the time product  $n$  took on machine  $j$ . Production rates were obtained from the machine processing speeds. Then, the problem can be formulated as a network based LP as follows:

Minimize  $Z_1 =$  Make span of  $n$  products, so,

$$Z_1 = \sum_{i=0}^{i=n} \sum_{j=0}^{j=n} t_{ij} \times x_{ij} \tag{1}$$

Similarly, Minimize  $Z_2 =$  Total manufacturing costs, so,

$$Z_2 = \sum_{i=0}^{i=n} \sum_{j=0}^{j=n} C_{ij} \times x_{ij} \tag{2}$$

Where,  $t_{ij}$  is the time to send a part through  $i \rightarrow j$ ,  $x_{ij}$  is the flow through  $i \rightarrow j$ , and  $C_{ij}$  is the cost of sending each unit of the material flow through nodes  $i \rightarrow j$ . The make span of  $n$  products will include, in general, part waiting times and machining times. Equation 1, does consider the part machining times, but not the part waiting times. We want to estimate the material flow, which will possess least waiting and total times, and at the same time avoid problems (in this case, bottlenecks). This material flow with the lowest overall time in general, will also result in a lower WIP (work in progress), and consequently, lower equipment utilization. Hence, estimating the make-span is sufficient to derive additional, related KPI's.

We first re-formulate equation 1. In the present case, let  $t_w$  be the total time  $x_{ij}$  products will wait before proceeding to flow through  $i \rightarrow j$ . Because we especially want to model part-waiting times, we let  $i$  be the stock point with capacity  $B_{fi}$ , and  $j$  be a machine. Clearly, if  $\xi_{in(i)} < \xi_{out(j)}$ , then it can be assumed that the speed of the outcoming part will equal  $\xi_{in(i)}$ . This is because, in such a case, parts will never be delayed. On the other hand if  $\xi_{in(i)} > \xi_{out(j)}$ , each part will have to wait for some-time before it can be serviced or before it can enter the next station. The waiting time for  $x_{ij}$  parts, through  $i \rightarrow j$  can then be calculated using Little's law by the following equation,

$$t_w = \left( \frac{B_{fi}}{\xi_{in(i)}} \right) \times x_{ij} \tag{3}$$

The above equation is based on steady state estimates, meaning that no more than  $B_{fi}$  parts can enter the buffer if the buffer is full. Hence, the above equation will not result in correct waiting times if  $\xi_{in(i)} > \xi_{out(j)}$  and  $x_{ij} > B_{fi}$  (the case of a bottleneck). To avoid the bottleneck, we first need to find the initial flow, and then adjust the production rate and buffer size variables. It can also be argued that we might as well select an alternative policy, leading to an alternative route to avoid the bottleneck. However, because the aim of the re-formulated equation, is to find the least part waiting times for a certain number of products, the resulting material flow is already the best. In other words, if there was a better material flow route resulting in lower waiting times, it would have already selected that. This leaves us to alter the production rate and/or buffer size variables. The following equation calculates, for instance, the correct buffer size based on the flow amount, and the production rates:

$$B_{fi} = \left( \frac{\xi_{in(i)} - \xi_{out(j)}}{\xi_{in(i)}} \right) \times x_{ij} \tag{4}$$

We use the following approach. We first estimate the flow amount resulting using equation 5 (which is equation 1 + waiting times  $t_w$ ) with the value of  $t_w$ , obtained by using equation 3. Then wherever bottlenecks occur, we use the flow value  $x_{ij}$  obtained by solving equation 5, and substitute it in equation 4. This will result in a buffer size, which is a steady-state size, meaning that all the flow can pass through the buffer without there being a bottleneck. Once we get the values of the buffer sizes, the new part

waiting times can be re-calculated by using equation 5, which is written as, minimize  $Z_3$ , where,

$$Z_3 = \sum_{i=0}^{i=n} \sum_{j=0}^{j=n} ((t_{ij} \times x_{ij}) + t_{wij}). \quad (5)$$

Equation 2 and 5, will further be subject to material flow constraints through nodes. The constraints are:

$$\sum_{j=1}^{j=n} x_{ij} - \sum_{j=1}^{j=n} x_{ji} = b_i, \text{ for all } i's,$$

where,  $b_i$  is the net flow generated. For the example problem described (refer Figure 3), at node a, the constraint can be written as,

$$x_{ab} + x_{ac} + x_{ad_1} + x_{ad_2} + x_{ad_3} = n_{final}. \quad (6)$$

Similarly, at node i as,

$$-x_{hi} - x_{fi} = -n_{final}, \quad (7)$$

where,

$$n_{final} = n - n_p,$$

and, at node  $e_1$ , the constraint can be written as,

$$-x_{ce_1} - x_{d_1e_1} - x_{d_2e_1} - x_{d_2e_1} + x_{e_1f} = 0, \quad (8)$$

and all  $x_{ij} \geq 0$ .

Equations 6 and 7 suggest that the net flow or the number of parts entering the system and the number of parts leaving the system is equal. All other nodes except a and i, are formulated in a similar fashion as equation 8, to indicate they do not retain the material flow. Deviations to the above constraints can also be formulated when modeling assembly of parts and part rejection, by assigning gain or loss factors to the nodes and to the net material flow (Minieka 1978).

In this work, both time and cost functions (KPI's) are minimized simultaneously. This concept is borrowed from the theory of goal programming. We make use of the non-preemptive category where both functions are of roughly comparable importance (for detail treatment see Hillier, Lieberman 2001, page 332). It is a well known fact that with preemptive goal programming one can control the importance of one function over the other. Our approach is to first solve for each KPI individually and then use these results (for example, of the best flow at least cost and at least time) as goal values to perform goal programming.

Table 4 shows the values resulting from solving each KPI individually (make span includes machining times + waiting times respectively) for the example in the previous section and the penalties placed to further perform goal programming.

Table 4: Setting of KPI (Goals) and Penalty Weights

Key Performance Indicator (KPI)	KPI Values (Goals)	Penalty weights
(Z <sub>3</sub> ) Make-Span (Minutes)	(909 + 7243) = 8152	1
(Z <sub>2</sub> ) Cost (Euros)	8820	1

Given the penalty weights, let Z be the number of penalty points incurred by missing the above goals. The overall objective function can then be written as minimize,

$$Z = \sum_{i=1}^{i=n} \text{Weight} \times (\text{Amount over/under KPI } n).$$

Penalties are placed by the priority weights for exceeding the individually calculated KPI value, while no penalties are placed for going under the KPI value. (This is because there cannot be values, lower than the ones already obtained). The solution procedure tries to reduce the (weighted) sum of deviations of these KPI from their respective goals (values obtained in Table 4) (Hillier, Lieberman 2001).

Note that it is feasible to have a large number of control points and the policies contained within the network shown in Figure 4. This is because, the network simplex method can solve problems with several thousand nodes, having several thousand constraints in a matter of a few seconds. Intermediate values of the goals and the weights used were set in Table 4.

### 5.1 Results of applying tier 1 calculations

In this work, the model was formulated and solved in the commercial optimization package MPLCPLEX. Table 5 provides the estimated material flow. The current example took 0.06 seconds to give a solution. The final value of Z was 5550, while the time KPI was exceeded by 3109 and the cost KPI by 2441.

The total flow in Table 5 amounts to 1000 parts which includes the estimated quantity and the pre-decided quantity. Given this, according to the statement made earlier that a bottleneck occurs when  $\xi_{in(i)} > \xi_{out(j)}$  and  $x_{ij} >$

$B_{fi}$ . When the flow and the network is analyzed, it is clear that node a, is a bottleneck. As also mentioned earlier, the bottleneck can be avoided by changing the system dimensions.

Table 5: Estimated Material Flow

From	To	Estimated Quantity	Pre-decided Quantity	Total flow
a	b		300	300
a	c		280	280
a	d <sub>1</sub>	16		16
a	d <sub>3</sub>	404		404
b	g		300	300
c	e <sub>3</sub>		280	280
d <sub>1</sub>	e <sub>2</sub>	16		16
d <sub>3</sub>	e <sub>2</sub>	404		404
g	h		300	300
e <sub>2</sub>	f	420		420
e <sub>3</sub>	f		280	280
h	i		300	300
f	i	420 + 280	420 + 280	700

Here we choose to change the input speed of incoming parts using equation 4. Equation 4 has to be slightly modified to suit the situation, in this case, for node a, we have one input channels and four output channels, namely, b, c, d<sub>1</sub> and d<sub>2</sub> for the flow. After calculating the correct input speed (which comes out to be 0.8 part/min), we pre-simulate the material flow, and Figure 5 shows the plots obtained which shows the number of parts in each buffer over the life time of the entire pre- simulation run. The results have validated the calculations of tier 1 by showing that the number of parts in each buffer are well below their maximum capacities.

Further, when the flow is analyzed, the path a → d<sub>2</sub>→e<sub>1</sub>\e<sub>2</sub>\e<sub>3</sub>... (note that e<sub>1</sub>\e<sub>2</sub>\e<sub>3</sub> is one object) was avoided because of the goal programming. However, we could still choose to send a few parts using the above path by making decisions in real-time, to possibly result in reduced times at the expense of higher costs. Over-riding the flow calculated by tier 1 is not yet considered in this paper.

## 6 PRE-SIMULATION, VISUALIZATION AND INTERACTIVE ANALYSIS

The flow obtained in the previous section, is then pre-simulated, and visualized using Technomatix *eM-Plant*. We use the inbuilt time-lag offered by *eM-Plant* to model the time difference between pre-simulation and virtual simulation. Figure 6 shows visualization screen shots.

Continuing with the previous example, assume that machine c randomly breaks down at some point in time during the virtual simulation. Because of the time-lag approach the virtual simulation is actually behind the pre-simulator by time ΔT. Hence, we parameterize the breakdown at the pre-simulation level, but ask it to occur at time (ΔT + time when breakdown occurs in the pre-simulation) during the virtual simulation. As seen in Figure 6, the pre-simulation automatically detected a disturbance and has indicated where the disturbance occurred and when. Similarly,

other disturbances can also be detected automatically by feedback from the pre-simulation or the virtual simulation.

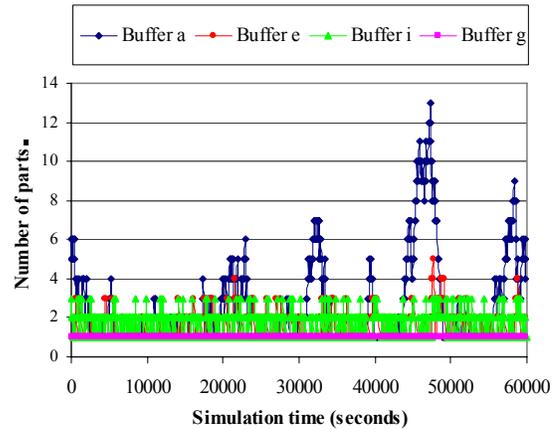


Figure 5: Buffer Sizes During Pre-simulation Run

As indicated earlier, the tier 2 calculations will be invoked to recover from the random machine breakdown. Tier 2 considers alternative policies, KPI, etc to compute corrective policies. Linking these corrective policies back to the objects will result in automatic detection of significant objects (processes). Tier 2 will mostly include algorithms which deal with reactive disturbances (several heuristic based algorithms have been developed, see Table 1). Because, the system immediately detects disturbances after they occur, it is possible that some time-lapse occurs before the system goes into a complete irreversible damage state. Hence, both tier 1 and tier 2 will have to provide control instructions which include decisions like when should the user be interacting with a significant process or processes.

If we assume that the result of the tier 2 calculation phase is to re-route the material flow (which was originally to be sent to machine c) to machine b, then our significant process would be object\buffer a. This is because (refer Figure 4), buffer a is the point where we defined the alternative control policies for the alternative material flow route. As discussed earlier the user may decide to navigate (for automatic motion planning algorithms see Mahajan et. al 2005) to the location and interact with the (significant process) object. The user will typically be given an option to implement the corrective policies from his current position, or to navigate to the significant process. Upon reaching the significant process, he will be able to select the process, and implement the corrective policy. This interaction could be done by sending messages to the pre-simulation mechanism, from the visualization. In the present case, the message would be instructions to change the control policy at buffer a. At this point, the loop is completed. Re-visiting Figure 1, if a new disturbance occurs, this time a deterministic order arrival, the tier 1 calculation would be activated, which would consider the algorithm described earlier.

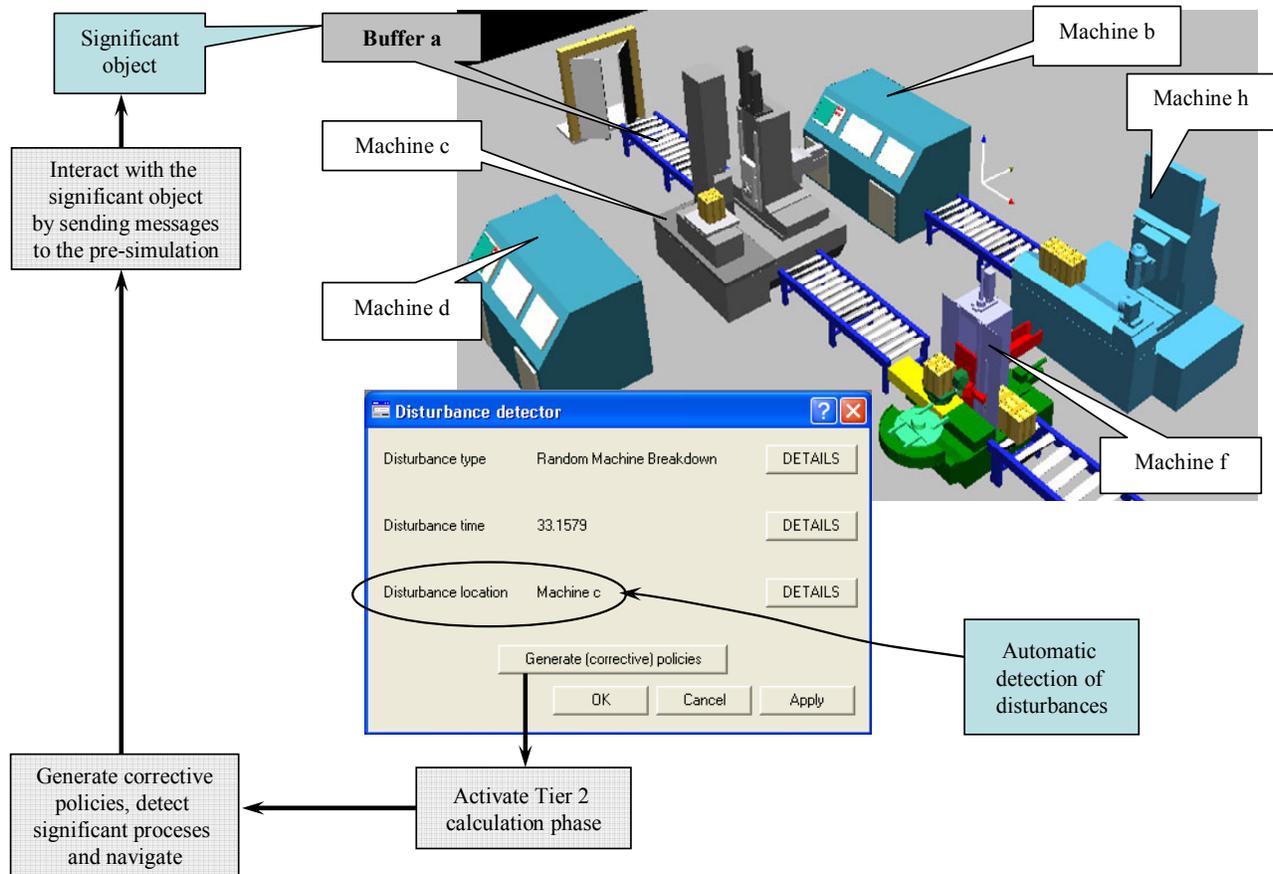


Figure 6: Visualization, Automatic Disturbance Detection and Mechanism For Interactive Analysis

## 7 CONCLUSION AND FUTURE WORK

Integration of visualization and real-time decision making algorithms discussed here serve as an inevitable starting point for the development of more sophisticated control algorithms in virtual environments. The real benefit from such an integration would be expected when the virtual simulation is made to emulate a real production facility.

We have presented a theoretical framework to integrate work in the areas of visualization and real-time decision making. We partially demonstrated the tier 1 calculation phase with the help of an example. We especially considered several alternative policies, and the selection of the best policies. We then pre-simulated and virtually simulated the flow. We also showed how problems are automatically detected by the pre-simulation mechanism. Further, we discussed how significant processes could be identified, and how interaction between simulation and visualization could take place to correct an example problem.

The tier 1 calculation phase seems to prove beneficial because it optimizes the material flow for the cases where the system parameters are deterministic. It was also seen that the results of the tier 1 calculation phase matched with

those of the pre-simulation mechanism, which also acted as a validating tool.

Much work remains to be done in the areas of tier 1 and tier 2 calculation phase, which will include algorithms for real-time decision-making based on the discussed disturbances. We did not yet consider refining the flow obtained from tier 1 calculations during real-time. Work in these area has been done in the past, but not in the context of virtual simulations and interactive analysis. In addition to this, the selection of the time difference between pre-simulation, and virtual simulation, and a basis for deciding when the virtual user should interact with the simulation, and for how long has to be established. Our future work will concentrate in these directions.

## ACKNOWLEDGEMENTS

The authors wish to thank Prof. Leena Suhl for her support, the DFG (German Research Foundation) grant DA155/29-1 and the International Graduate School of Dynamic Intelligent Systems to fund this research.

## REFERENCES

- Abumaizar, R. J., and J. A. Svestka. 1997. Rescheduling job shops under random disruptions. *International Journal of Production Research* 35 (7): 2065-2082.
- Akturk, M. S., and E. Gorgulu. 1999. Match-up scheduling under a machine breakdown. *European Journal of Operational Research* 112 (1): 81-97.
- Chong, C. S., A. I. Sivakumar, and R. Gay. 2003. Simulation-based scheduling for dynamic discrete manufacturing. In *Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sanchez, D. Ferrin, and D. J. Morrice, 1465 - 1473. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.
- Harmonosky, C. M., R. H. Farr, and M. C. Ni. 1997. Selective rerouting using simulated steady state system data. In *Proceedings of the 1997 Winter Simulation Conference*, ed. S. Andradottir, K. J. Healy, D. H. Withers, and B. L. Nelson, 1293 - 1298. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.
- Hillier, F. S., and G. J. Lieberman. 2001. Introduction to Operations Research. McGraw-Hill, ISBN 007232169.
- Jain, A. K., and H. A. Elmaraghy. 1997. Production scheduling/rescheduling in flexible manufacturing. *International Journal of Production Research* 35 (1): 281-309.
- Kim, M. H., and Y. Kim. 1994. Simulation based real time scheduling in a flexible manufacturing system. *Journal of Manufacturing Systems* 13 (2): 85-93.
- Leon, V. J., S. D. Wu, and R. H. Storer. 1994. A game-theoretic control approach for job shops in the presence of disruptions. *International journal of production research* 32: 1451-1476.
- Li, R-K., Y.-T. Shyu, and S. Adiga. 1993. A heuristic rescheduling algorithm for computer-based production systems. *International Journal of Production Research* 31 (1): 1815-1826.
- Mahajan, K. R., C. Laroque, W. Dangelmaier, S. Soltenborn, M. Kortenjan, and D. Kuntze. 2005. d<sup>3</sup> FACT insight: A motion-planning algorithm for material flow simulations in virtual environments. In *Proceedings of the 2005 Simulation and Visualization Conference*, ed. T. Schulze, G. Horton, B. Preim, S. Schlechtweg, 115 - 126. Erlangen Germany: SCS publishing house e.V.
- Minioka, E. 1978. Optimization algorithms for networks and graphs. Marcel Dekker, Inc. ISBN 0824766423.
- Mueck, B., Dangelmaier, W. and Fischer, M. 2003. Components for the active support of the analysis of material flow simulations in a virtual environment, In *Proceedings of the 15th European Simulation Symposium*, ed. A. Verbraeck and V. Hlupic, 367-371. Erlangen Germany: SCS publishing house e.V.
- Yamamoto, M., and S. Y. Nof. 1985. Scheduling/rescheduling in the manufacturing operation system environment. *International Journal of Production Research* 23 (4): 705-722.

## AUTHOR BIOGRAPHIES

**WILHELM DANGELMAIER** studied Mechanical Engineering at the University of Stuttgart, Germany. Since 1981, he was director and head of the Department for Corporate Planning and Control at the Fraunhofer Institute for Manufacturing (Fraunhofer IPA). In 1991, Dr. Dangelmaier became Professor for Business Computing at the Heinz Nixdorf Institute; University of Paderborn, Germany. In 1996, he founded the Fraunhofer Center for Applied Logistics. His e-mail address is [dangelmaier@hni.upb.de](mailto:dangelmaier@hni.upb.de) and his Web address is <http://www.hni.upb.de/cim/>.

**KIRAN MAHAJAN** studied mechanical engineering at the Delft University of Technology in The Netherlands with a specialization in production engineering. Since 2004, he is a research assistant at the Heinz Nixdorf Institute; University of Paderborn, Germany. He has a bachelor's degree from the University of Pune (COEP), India. His research interests are interactive analysis of manufacturing simulations and 3D visualization. His e-mail address is [kiran@hni.upb.de](mailto:kiran@hni.upb.de) and his Web address is <http://www.hni.upb.de/cim/>.

**DANIEL HUBER** studied industrial engineering at the University of Paderborn, Germany. Since 2005, he is a research assistant at the group of Prof. Dangelmaier, at the Heinz Nixdorf Institute. His research interest is simulation modeling of industrial processes. His e-mail address is [huber@hni.upb.de](mailto:huber@hni.upb.de) and his Web address is <http://www.hni.upb.de/cim/>.

**BENGT MUECK** studied computer science at the University of Paderborn, Germany. In 2004, he received a Ph.D. for his work on dynamic multiresolution modeling of huge simulation systems. His research interest is development of simulation tools for logistic systems. His e-mail address is [mueck@hni.upb.de](mailto:mueck@hni.upb.de) and his Web address is <http://www.hni.upb.de/cim/>.