# A Petri Net Approach for Performance Measurement of Supply Chain in Agile Virtual Enterprise

Reggie Davidrajuh
Narvik Institute of Technology, PO Box 385, N-8505 Narvik, Norway
Email: Reggie.Davidrajuh@hin.no

## Abstract

**This paper is about performance measurement of supply chain in agile virtual enterprise environment with an approach that uses a new high level Petri net. Measurement of the performance of supply chain is a good indicator of the efficiency of collaboration of the enterprises involved; thus the main assembler can determine whether making changes among the collaborating partners may improve the overall performance. The approach discussed in this paper uses cost, quantity, and delivery as the key performance indicators.**

**Due to the large number of participating enterprises and due to the dynamic nature of collaboration, classical approaches that use Petri nets are not suitable modeling supply chain in agile virtual enterprise. The new approach discussed in this paper models supply chain in an agile virtual enterprise by taking the topology of connection between the enterprises as the starting point, forming a connected system to represent the virtual enterprise. The tool AgileSIM used in this approach then generates an interim Petri net model from the connected system, replacing each enterprise by a transition-place pair, where the transition represents the production at the enterprise and the place represents the output buffer for products. The interim Petri net model is then converted into C++ code for compilation into an executable program. Thus this approach is very suitable for automating performance measurement, for example as an on-line monitor for agile virtual enterprise activities.**

**Keywords**: **Petri net, Supply chain, performance measurement, agile virtual enterprise**

# 1 • Introduction

Performance measurement of supply chain is a very important part of the strategic supply chain management system of an agile virtual enterprise. This is because, the management need a transparent and real-time view of the supply chain so that they can quickly react to the changing market conditions ('agility'), by associating with new collaborating enterprises or eliminating some existing ones who are performing less satisfactorily ('virtual enterprise'). It is important for management of the agile virtual enterprise to rely on simple, quick, and yet effective tools for performance measurement of the supply chain; in today's competitive environment, management simply cannot afford to wait for a long time for the information they need.

In the next section, the classical approaches for modeling supply chain with Petri nets are presented. It is shown that while this approach is very suitable to find bottle necks in the production systems, this is not suitable for performance measurement of supply chain in agile virtual enterprise due to the large number of participating enterprises and due to the dynamic nature of collaboration. In the third section, a new approach is presented along with a presentation of a new high level Petri net. A sample problem is worked out in the fourth section.

In the following subsection of this introductory section, the key terms such as agile virtual enterprise and supply chain are introduced.

## 1.1 Introducing agile virtual enterprises

Agile virtual enterprise is based on two concepts; *agility* is the ability to react quickly to changing conditions; *virtual enterprise* means the ability to have fuzzy business boundaries for a project, extending the enterprise to include newer collaborating enterprises [1]. In figure-1, the main producer or assembler integrates a number of collaborating enterprises (suppliers and distributors), to manufacture a certain class of product. When market conditions change, a new class of product or an improved version of the product should be quickly turned out to meet the new market requirements; this ability is termed as agility. In this case, the main assembler may seek for a new combination of suppliers and distributors that are more suitable to manufacture the new class of products [2].

As shown in figure-1, our view of the collaboration is 'main assembler-centered'. This means, the main assembler- the enterprise that owns the trademark of the product being produced by collaboration, decides whether to change the collaborating enterprises (accept any new enterprise into collaboration or to reject any existing collaborating enterprises), change the volume and properties of the product, etc.

The life cycle of agile virtual enterprise

includes phases such as *business opportunity identification*, *partner selection*, *formation*, *operation*, and *reconfiguration* [3]. Before forming an agile virtual enterprise, profitability of a product (that is going to be produced) has to be assessed. This is done in the opportunity identification phase. Profitability of a product is assessed by extensive market analysis and research. After the opportunity identification phase, the right collaborating enterprises must be found to manufacture the product (partner selection phase); thus the supply chain is established at this phase.

After forming an agile virtual enterprise, the collaboration is put to use for producing a class of products; this is the operation phase. During the operation phase and after, the main assembler constantly monitor the performance of the supply chain so as to determine whether to establish a new supply chain for producing the same class of product or a new class of product, by changing the collaborating enterprises; this phase is called the reconfiguration phase. Clearly, formation and reconfiguration phases are similar. In reconfiguration phase, the performance of an existing collaborating enterprise is compared with a new potential supplier. Mathematical models are available for assessment of existing and new suppliers [4, 5].

Purely mathematical models for performance measurement are better for academic purposes than for practical use as mathematical models are difficult to use, need experts to program it, and they are difficult to combine with other existing enterprise resource planning (ERP) modules and software. In this paper, a simple yet effective approach is introduced for performance

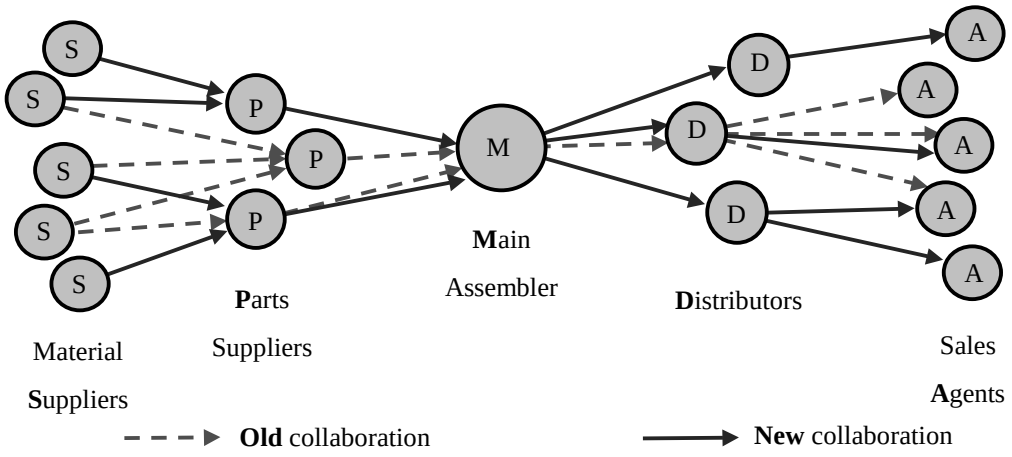



Figure 1: Dynamic collaboration in an agile virtual enterprise

measurement of supply chain, which uses a new high level Petri net. The tool that is developed for this approach is called AgileSIM. AgileSIM is developed for MATLAB simulation environment. The approach presented in this paper is very relevant for automating supply chain management system and to integrate performance measurement as a module within a wider system that consists of a data collection system and an inference engine [2, 6].

### 1.2 supply chain, key performance indicators

Figure-1 shows a simplified supply chain spanning from procuring raw materials from the raw materials suppliers to the delivering the finished good to the consumers via sales agents. The enterprises on the supply side and on the distribution side are grouped into two tiers (raw material suppliers and parts or semi-finished product suppliers; distributors and sales agents). But in reality (e.g. in automobile production), the participating enterprises fall into many levels thus establishing a multi-tier supply chain. The supply chain management system is an important part of an enterprise's strategy to optimize planning and execution processes to respond to the changes in the market. It is not an exaggeration to say that supply chain management system have now become more important than the manufacturing processes themselves [7].

Traditional methods of measuring supply chain performance through purely functional metrics (which are often financially oriented) are no longer sufficient to make decisions for an agile virtual enterprise. An agile virtual enterprise need to capture critical quantitative data and qualitative insight into the supply chain. In order to do that, the selection of key performance indicators (KPI) must be limited a few most important ones. We have selected *cost* (right price), *quantity* (right amount), and *delivery* (right time) as the key performance indices. The approach discussed in section-3 for performance measurement make use of theses three key performance indicators.

### 1.3 push/pull material flow control systems

When modeling supply chain, frequently, distinction is made between two kinds of material flow control systems, the push systems (e.g. MRP) and pull system (e.g. Kanban operated JIT). However, most practical supply chains consists of both push and pull systems [8]. Fresh fruits and vegetable production systems is a good example for push system, whereas production and sales of some custom made specialized instruments are typical example for pull system. The Toyota system, though originally invented the pull system, uses a push system for distributing produced vehicles; but uses Kanban for input raw material and other parts. Thus, Toyota is a good example for hybrid push/pull type [8].

A fundamental difference observed in push and pull systems is the direction of information flow; in push systems, the information flow is in the same direction as the material flow. Whereas, in pull systems, the information flow is in opposite direction to the material flow. Figure-2 shows the differences between push and pull systems, for a part supplier.

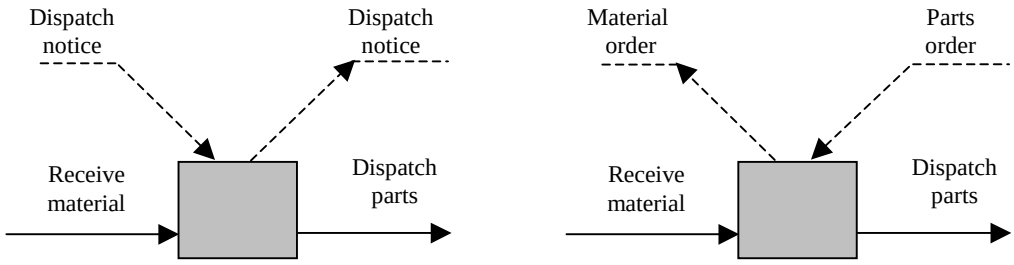
The advantages and disadvantages of push and pull systems and the similarities and dissimilarities between them are out-of scope of this work, thus interested reader is referred to [8]. For simplicity, the examples shown in this paper are based on pull system principle only. With some minute changes, the Petri net model versions for push system can be Obtained.

## 2 • Classical approaches with Petri nets

In this section, classical approaches for modeling and performance measurement of supply chain are presented. By going through this section, it will become clear that the classical approaches using Petri nets are more suitable for modeling material flow within an enterprise (that is 'intra-enterprise') and not useful for modeling a virtual enterprise ('inter-enterprise') where the business boundaries are fuzzy, and collaborating partners are numerous and often change.

For intra-enterprise use, classical approaches can be used to check whether the Petri net model has desired properties such as liveness, boundedness, etc. These desired properties characterize the dynamic behavior of a well-designed production system. For instance, in the Petri net model of an enterprise, the liveness ensures that blocking will never occur due to lack of raw materials for production line, and boundedness guarantees that the number of in-process parts is upper bounded which in turn guarantees the stability of the material flow through the enterprise.

To check the properties of a Petri net model,




2a: The push system for a part supplier 2b: The pull system for a part supplier

Figure-2: The fundamental difference between push and pull systems

the methods such as incidence matrix, coverability tree, and invariant analysis could be used [9]. Unfortunately, these methods do not apply to large-size Petri net models, as in the case of supply chain for agile virtual enterprises.

**2.1 Modeling supply chain: the integrated approach**

The integrated model of a supply chain can be obtained by two ways: The first way is the reduction of Petri nets model for a complete supply chain; A Petri net model of a supply chain can be built from the first principles of conditions and events of

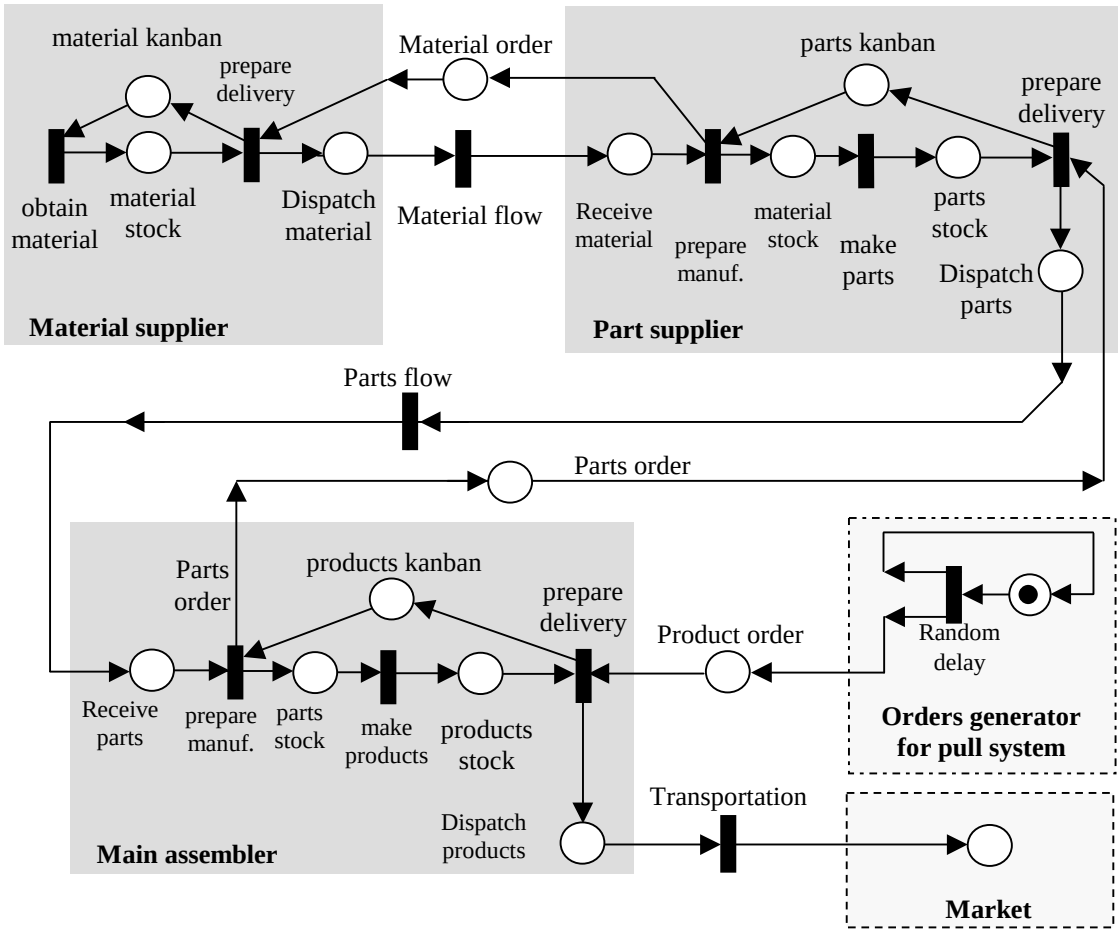



Figure-3: A simple Petri net model of a pull system, consisting of a material supplier, part supplier and an assembler

production systems in agile virtual enterprise (see figure-3). Since the Petri net model will be huge, it can be reduced according to some proposed reduction rules while preserving properties. The main disadvantage of this approach lies in the difficulty of finding the reducible sub-Petri nets [10].

The second way is to build an acceptable Petri net model by starting with a simplistic model and then adding more and more details to it so that the desired properties are guaranteed without analysis. The basic idea is to build the desired properties in the model instead of checking properties after modeling the system; thus the modeling approach is neither easy nor suited for automation [10].

Figure-3 (adapted from [8]) shows model of a supply chain build after integrated approach. Firgure-3 contains only a material supplier, a part supplier and an assembler, which is far too simple than any real-life supply chain, which contains probably hundreds of participating enterprises. Therefore, it is clear that it will be very difficult to build analyze the complete model of any real-life supply chain by the integrated approach, as the model will contain too many nodes. Also, this modeling approach does not offer flexibility to

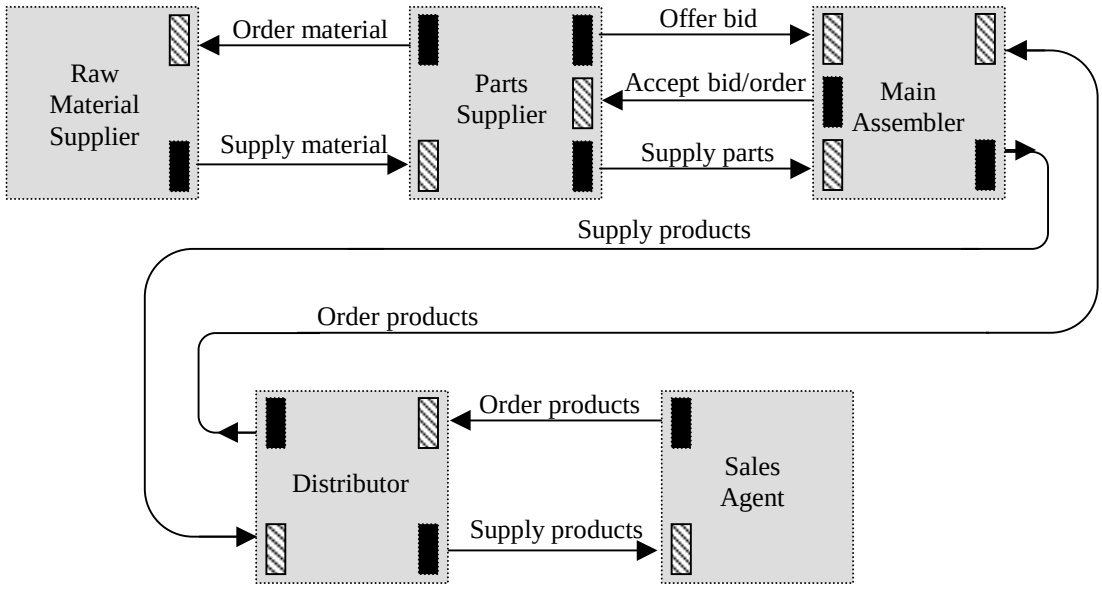



Figure-4: Connecting modules through i/o ports in modular modeling approach

support the dynamic collaboration (removal and addition of participating enterprises) that is characteristic of agile virtual enterprises. A better approach is to model each participating enterprise as a module first, and then connecting these modules together to form a virtual enterprise without loosing any properties due to connection. This approach is called the modular modeling approach, which is explained in the following subsection.

### 2.2 Modeling supply chain: the modular approach

This approach is well suited to model systems composed of independent and easily identifiable subsystems, as in the case of agile virtual enterprise. The modular approach consists of mainly, modeling individual enterprises as modules, with *event graphs*, and connecting the modules together through their input/output ports. The modular modeling approach is a five step process, the first step being identification of the independent subsystems in the overall system.

After identifying the independent subsystems, then the second step is to identify the input/output ports (i/o ports) of the subsystems. It is through the i/o ports, the subsystems are going to be connected to obtain the complete model. Figure-4 shows input and output ports of enterprises participating in an virtual enterprise environment.

The third step is to model the individual subsystems with Petri nets; these models are called *modules*. Figure-5 shows modules for different types of enterprises, modeled using event graph. It must be made sure that the modules representing enterprises must be event graphs (event graphs are Petri nets where all the places have exactly one input transition and one output transition); this restriction imposed so that the modules can be reduced to a minimal size (reduction in internal transitions and places) using reduction theorems discussed in [10].

The fourth step is to reduce the size of the modules; [10, 11] shows a reduction method that is suitable for reducing modules that contain event graphs surrounded by i/o ports (transitions). According to the reduction theorem, each module can be replaced by an equivalent event model called minimal representation; the minimal representation has the same set of input and output transitions as the initial module, but has fewer internal places and do not has any internal transitions [10]. Therefore, the removal of internal transitions and reduction in internal places greatly reduces the size of the connected system, thus simplifies the model analysis.

The fifth and final step is to connect modules for different (types of ) enterprises together to form a complete model of the supply chain. When connecting the modules through their i/o ports, the ports are replaced by transitions. Since the i/o ports are transitions, the connecting line (arc) should

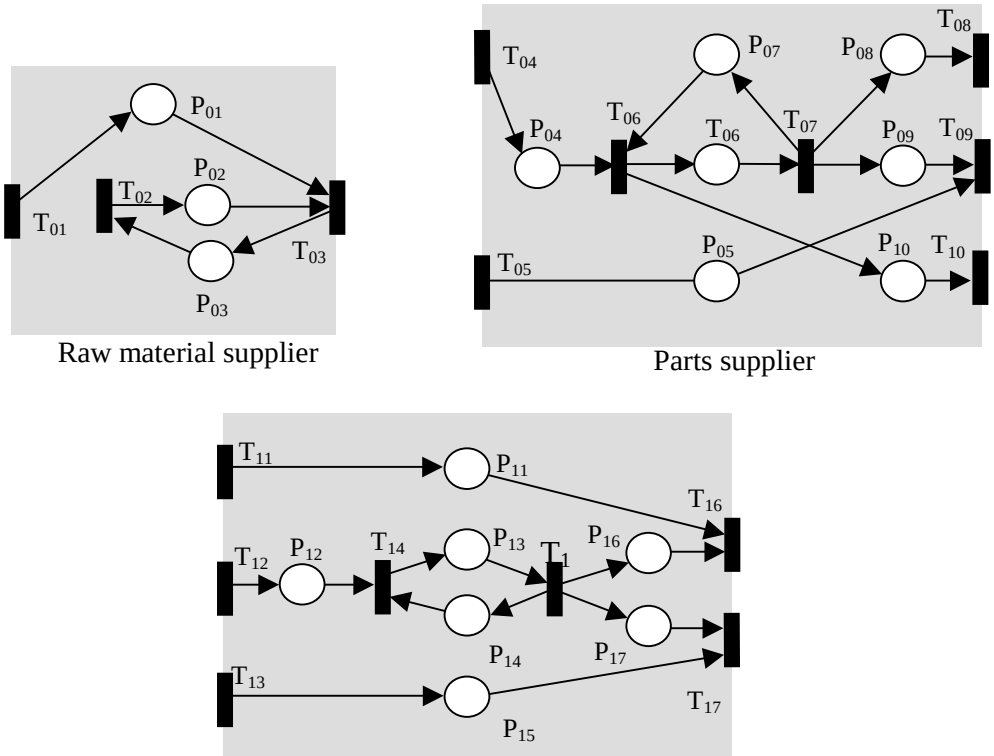



Figure-5: Petri net models of different kinds of enterprises

contain a place.

In figure-5, different types of enterprises are modeled as modules. It must be remembered that modules representing enterprises must be event graphs, where all the places have exactly one input transition and one output transition. The description

of places and transitions of the modules shown in figure-5 is given in table-1. It is obvious that the Petri net models for the different kinds of enterprises are much simplified diagrams, but since we are more interested in creating a complete model by connecting modules together, to get the overall performance, let us keep the modules simple.

The module for a raw material supplier in figure-5 has three transitions, one input transition (marked $T_{01}$), one output transition ($T_{03}$), and an internal transition ($T_{02}$); it is clear that $T_{01}$ refers to the input transition and $T_{03}$ refers to its output transition. Also, the module has three internal places $P_{01}$ - $P_{03}$. In the Petri net module for a sales agent is shown in figure-5, sale of goods is represented by the transition $T_{24}$ (see table-1 too). Though sales means goods leaving agent, therefore an output

Table-1: Description of the modules for different kinds of enterprises

| | | | |
|---|---|---|---|
| **A raw material supplier** | | | |
| $T_{01}$ | Receive order from part supplier/assembler | $P_{01}$ | Orders for material received. |
| $T_{02}$ | Production of raw material. | $P_{02}$ | Material ready for shipment. |
| $T_{03}$ | Ship material to part manufacturer | $P_{03}$ | Volume of material to produce. |
| **A part supplier** | | | |
| $T_{04}$ | Material arrive (from mat. supplier) | $P_{04}$ | Unloaded material in queue |
| $T_{05}$ | Orders received from assembler | $P_{05}$ | Received orders for parts |
| $T_{06}$ | Production calculations | $P_{06}$ | Ready for production of parts |
| $T_{07}$ | Manufacture of parts | $P_{07}$ | Monitor of quantity produced |
| $T_{08}$ | Order raw materials | $P_{08}$ | Monitor of materials used |
| $T_{09}$ | Ship parts to main assembler | $P_{09}$ | Parts ready for shipment |
| $T_{10}$ | Send bid to main assembler | $P_{10}$ | Bid for parts (to assembler) |
| **Main assembler** | | | |
| $T_{11}$ | Bids for parts and raw materials | $P_{11}$ | Received bids from suppliers |
| $T_{12}$ | Parts and raw materials arrive | $P_{12}$ | Unloaded parts in queue |
| $T_{13}$ | Orders received | $P_{13}$ | Ready for manufacture |
| $T_{14}$ | Production calculations | $P_{14}$ | Monitor of quantity produced |
| $T_{15}$ | Manufacture of products | $P_{15}$ | Received orders for products |
| $T_{16}$ | Order parts and raw materials | $P_{16}$ | Monitor of parts/materials used |
| $T_{17}$ | Ship products to distributors | $P_{17}$ | Products ready for shipment |
| **A distributor** | | | |
| $T_{18}$ | Goods arrive from main assembler | $P_{18}$ | Stock of goods |
| $T_{19}$ | Receive orders for goods from sales agent | $P_{19}$ | Received orders for goods |
| $T_{20}$ | Supply and distribution calculation | $P_{20}$ | Monitor of free capacity |
| $T_{21}$ | Order goods from main assembler | $P_{21}$ | Goods ready for shipment |
| $T_{22}$ | Ship goods to sales agents | | |
| **A sales agent** | | | |
| $T_{23}$ | Goods arrive from distributor | $P_{21}$ | Stock of goods |
| $T_{24}$ | Sales | $P_{22}$ | Monitor of free capacity |
| $T_{25}$ | Order goods from distributor | $P_{23}$ | Monitor of sales |

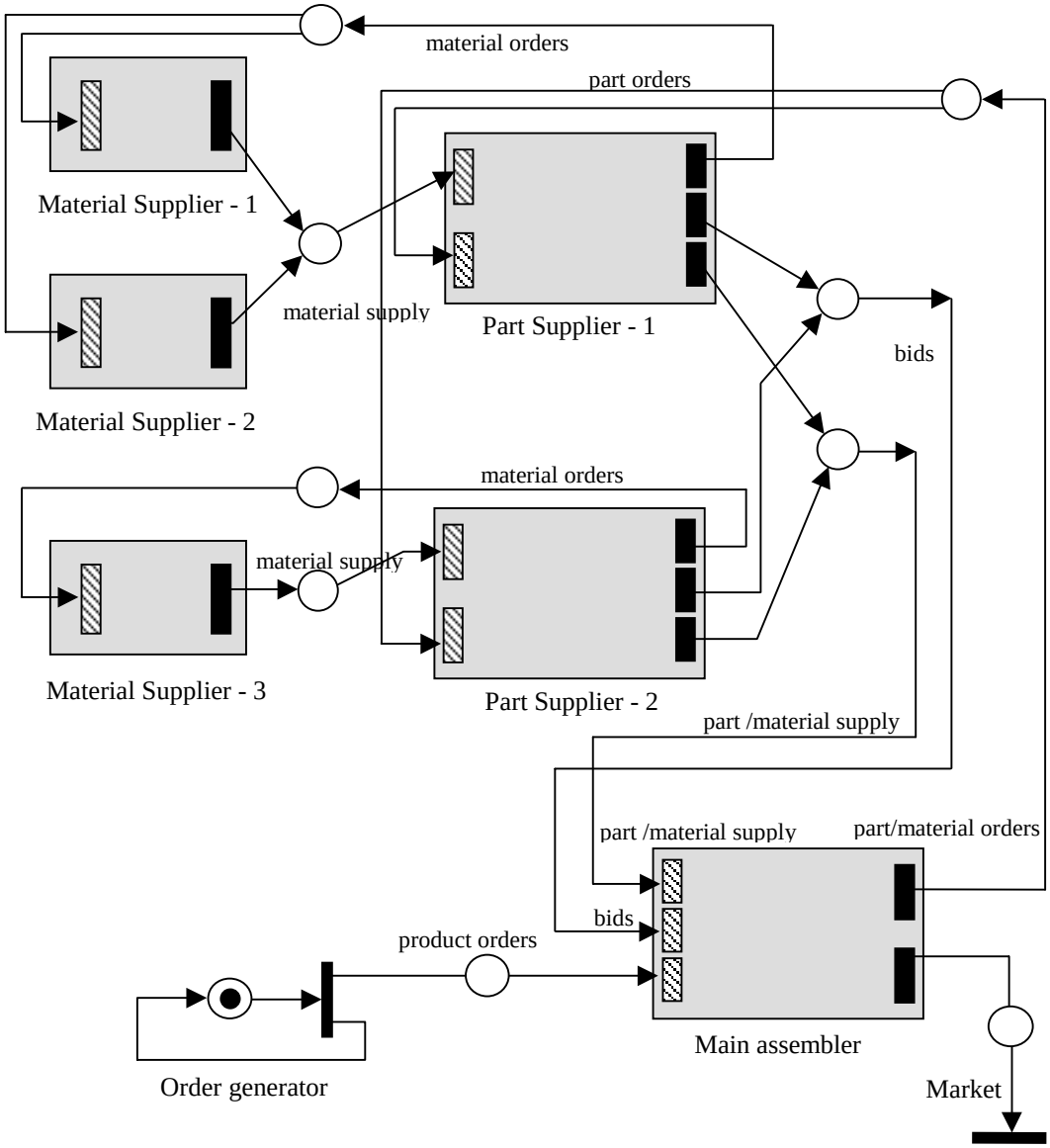



Figure-6: Model of a trivial supply chain obtained by connecting the modules.

activity, it is shown as an internal transition because sales is not coupled directly with a distributor.

The supply chain model of a virtual enterprise is obtained by connecting the modules for different kinds of enterprises through their i/o ports. Since the i/o ports are transitions, the connecting line (arc) should contain a place, see figure-6.

Figure-6 shows a small system consisting of

just six enterprise, the main assembler (there will just one main assembler in an agile virtual enterprise, as our view of agile virtual enterprise is a 'main assembler-centered' view) two parts suppliers (first-tier suppliers) and three raw material suppliers (second-tier suppliers). Even for this small system model, there are too many places (39 places totally) and transitions (30 transitions); thus, analyzing a real-world agile virtual enterprise consisting hundreds of supply and distribution enterprises will be impossible. There must be a way of reducing individual modules to a minimum size while keeping the topology of connection between enterprises intact and preserving its properties. The next subsection is about reduction of modules.

**2.3 Reduction of modules**

By reduction theorem, simple event graph module called minimal representations replace more complex modules for different kinds of enterprises. By reduction, the liveness and boundedness properties of the original module are preserved. The procedure of obtaining a minimal representation is given in [10].

The minimal representations for the different types of enterprises are shown in figure-7. If we replace the original modules in the model shown in figure-6 by the reduced modules, then the connected model will have 21 transitions and 24 places totally. Thus, the reduction is the total number of nodes (transitions 30% and places 38%) is significant; If

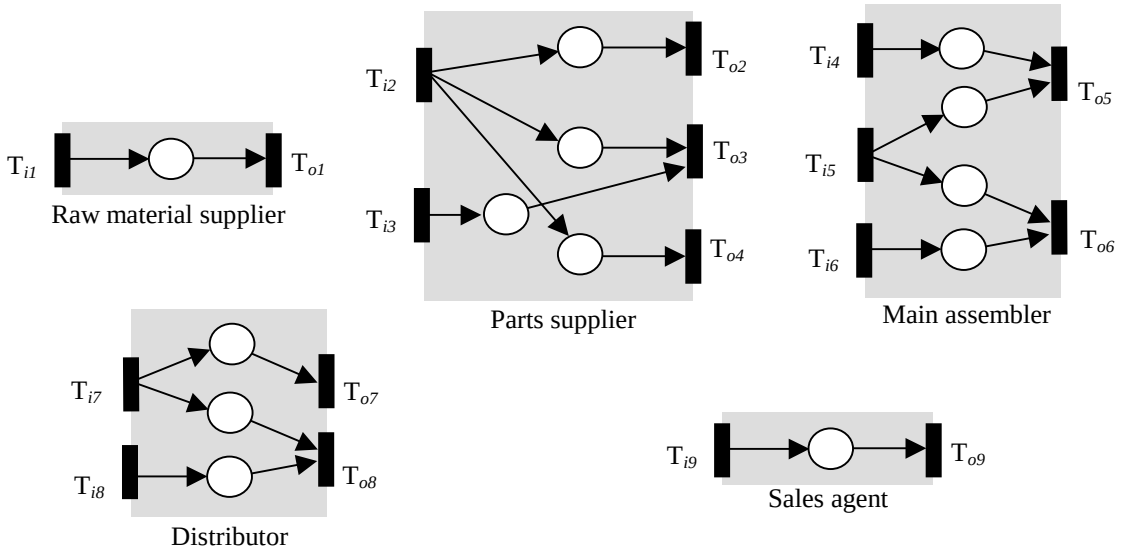



Figure-7: Minimal representations for different types of enterprises

we modeled the enterprises more realistically (more complex), then the difference due to reduction will be very significant.

It is obvious that the complete model of a supply chain obtained by the modular approach (connecting minimal representation (reduced modules) of participating enterprises) will be of much smaller size than the model obtained by integrated approach. Therefore, it will be easy to analyze the model obtained by the modular approach , using any standard mathematical analysis techniques for Petri nets. Material flow amounts, arrival times, starvation due to lack of material (liveness) or over production and stocking and WIP can be calculated using standard techniques for Petri nets. By assigning realistic data, these simulations could be carried-out.

It must be noted that the modular approach too is not quite suitable for supply chain in agile virtual environment where the participating enterprises dynamically change. In the next section, a new approach is introduced, together with a tool that operates on a new high level Petri net. The new approach is especially developed for performance measurement of supply chain in agile virtual environment.

## 3 • A new approach for performance measurement

A new tool called AgileSIM is under development at the department of Computer Assisted Production Engineering - Narvik Institute of Technology, especially for performance measurement of supply chain in agile virtual enterprise. AgileSIM has the following features:

- Performance measurement with key performance indicators cost, quantity, and time,
- Easy evaluation of potential suppliers (adding new partners to collaboration), and
- Simple user interface

AgileSIM offers,

1. A model structure of a realistic agile virtual enterprise,
2. An interim Petri net model for verification, and simulation
3. Modeling and simulation functions that are flexible and expandable as it is programmed in industry standard MATLAB development system,
4. Implementation, as the MATLAB system automatically generate portable C++ code that can be compiled into executable code,
5. Faster simulation results, so that can be used as a part of automated supply chain monitoring system for real-time use, and
6. Easy to use

In this section, a new approach for modeling,

simulation and implementation for performance measurement of agile virtual enterprise is introduced. AgileSIM- the tool used in this approach, is to establish an agile virtual enterprise consisting of a number of collaborating enterprise, to run performance calculation, and to evaluate improvements in performance due addition/removal of any new/existing enterprise.

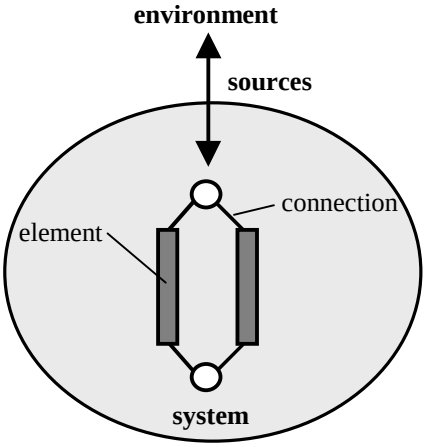



Figure-8: Concept of a system in manufacturing systems theory

### 3.1 Manufacturing systems theory

The new approach described in this paper is based on Manufacturing system theory. Manufacturing system theory in the form it is presented here is due to the work of the Scandinavian School of Systems Theory for the past 30 years [12]. For detailed study of manufacturing system theory, the standard textbook used for graduate study by Professor Øyvind Bjørke should be referred [12]. By using manufacturing system theory, not only physical systems but logical systems too can be modeled and analyzed. Also, modeling approach by manufacturing system theory seems to be more general, systematic, unified and effective than the other conventional methods [13].

As shown in figure-8, a system consists of three fundamental components, namely *elements*, *connections*, and *sources*. The elements carry all the physical properties of the system. Elements are the building blocks of the physical system. For example, in an electrical LRC circuit, inductors, resistors, and capacitors are the elements; the property of a resistor is its admittance, whereas in manufacturing- a machine element's property could be its processing time, ratio between the number of input items and output items, scrap percentage etc.

When there is no connection between the elements, the set of isolated elements is called *the primitive system*. The connections reflect how the elements in the primitive system influence each other and it represents the structure of the system. The set of connected elements is called *the connected system*.

Finally, the sources reflect the influence between the total system and the environment. Sources are the environment's influence on the system; in an electrical circuit, sources are current sources or voltage sources; in production planning, demand of products is a typical source.

### The formulation methodology

The objective of the new approach based on manufacturing systems theory is to offer a strategy by which the behavior of complex systems could be determined from the known behavior of its individual elements.

As stated above, a system in manufacturing systems theory is made of 1) Elements: systems parts, 2) Connections: system structure, and 3) Sources: external influence on the system. The mathematical formulation approach by manufacturing systems theory can be summarized as follow [14]:

- **Phase 1**: **identifying the primitive system**

1. Break up the system into its basic parts, i.e., elements, we call this group of isolated elements "the primitive system".
2. Set up the governing equation of each element independent of other elements, by that, we isolate the variables in the individual elements.
3. Concurrently, by the process of measurement, we will create an abstract model of the whole system defining the topological structure of the whole system.

- **Phase 2**: **making the connected system**

By means of the topological structure, we will connect together the variables in the individual elements. That is to set up the governing equations of the whole system, or "the connected system".

- **Phase 3**: **applying the sources, and solving the connected system**

To apply sources and solve the governing equations of the connected system; solving the system refers to determining the behavior of the system under the applied sources.

These three phases are our guidelines in the modeling process for a huge variety of discrete systems.

### 3.2 Using the approach for performance measurement

Performance measurement of virtual enterprise is done in three stages. Figure-9 shows the different stages. The first stage is the modeling stage where we use manufacturing systems theory approach to establish a connected system representing an agile virtual enterprise; when establishing a connected system, the properties of the elements are assigned with realistic values (system specification) using the tool AgileSIM.

The second stage is the simulation stage. After establishing a connected system, AgileSIM automatically generates the interim Petri net model of the connected system. The tool PenSIM (Petri net

simulator) can be used to carry-out simulations on the Petri net model. Thus the dynamic collaboration of the virtual enterprise is expressed by the connected system, while the dynamic activities of the virtual enterprise is represented by the places and transitions of the Petri net model.

The third stage is the implementation stage. At this stage, the Petri net model is converted to C++ programming language code and compiled into a executable system. This executable system will co-operate with at least two other executable systems, the inference engine and the data collection system. Together, these executable systems will automate supplier selection procedure, interested reader is referred to [2, 6].

**The connected system**

Considering supply chain in agile virtual enterprise as a system, there exist two types of elements - the participating enterprises and the

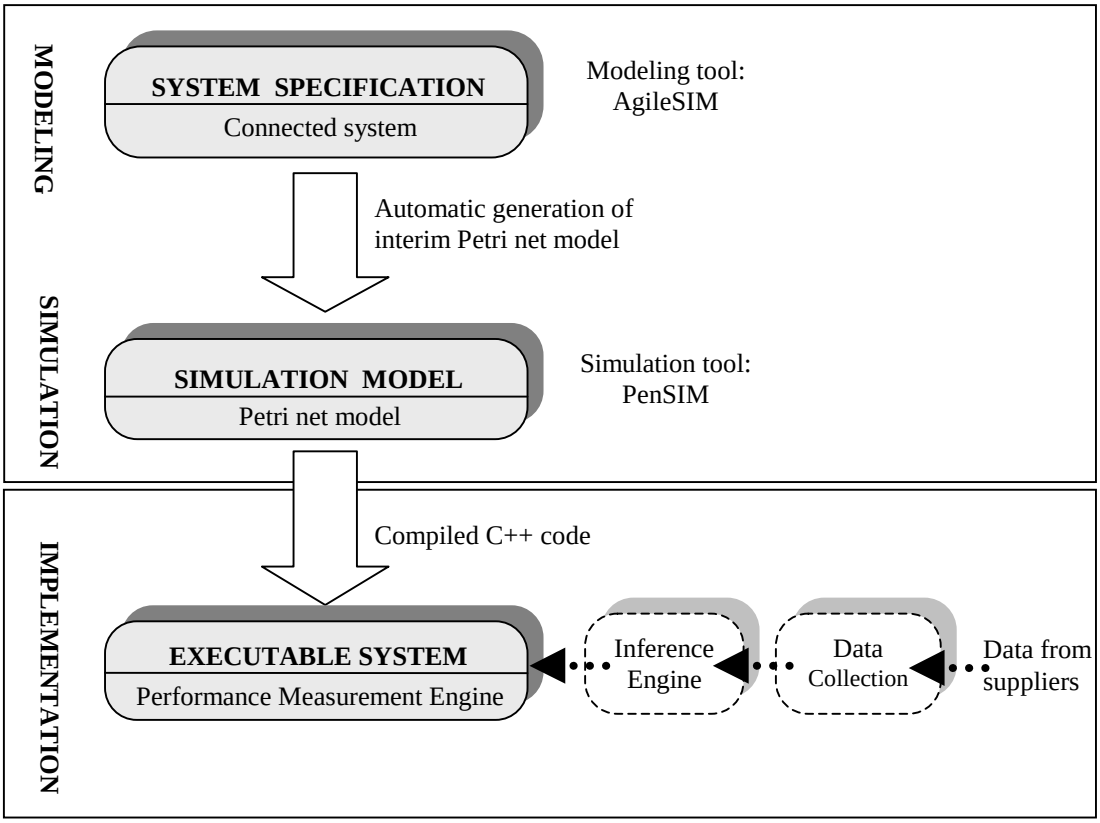



Figure-9: Different stages of modeling, simulation and implementation of performance measurement engine for agile virtual enterprise

transportation between them. Thus, a group of isolated enterprises and transportation make-up the primitive system. An enterprise element has *properties* like overall production costs per unit, production time, etc. A transportation element has some properties like selling enterprise (from) and purchasing enterprise (to) etc., see figure-10. A *collaboration* is a set of transportation, connecting together enterprise elements in the primitive system to establish the connected system. Figure-10 shows two enterprise elements and a transportation element of the primitive system linked together forming a connected system.

## 3.3 The Petri net model

Once by system specification a connected system is made, the interim Petri net model of the connected system is generated by AgileSIM, which is based on a new high-level Petri net. We say a new high-level Petri net because, in the Petri net model, the places does not simply hold numerals (or tokens) as in the case of ordinary Petri nets, but data like the inventory levels, accumulated costs etc. Also, the transition does not fire just because the input place has enough tokens, but fires if the logical conditions attached to it (e.g. firing will not overflow the buffer in the output place). Before formally defining the new high-level Petri net used in AgileSIM, let us first define the ordinary Petri net:

■ Definition D1 (ordinary Petri net):

A Petri net is a four-tuple

$$(P, T, A, x_0)$$

Where,

$P$ is the set of places (representing conditions or number of parts), $P = [p_1, p_2, \ldots, p_n]$

$T$ is the set of transitions (corresponds to events), $T = [t_1, t_2, \ldots, t_m]$

$A \subseteq (P \times T) \cup (T \times P)$ is a set of arcs from places to transitions and from transitions to places, and

$x = [x(p_1), x(p_2), \ldots, x(p_{n1})] \in \mathrm{N}^n$ is

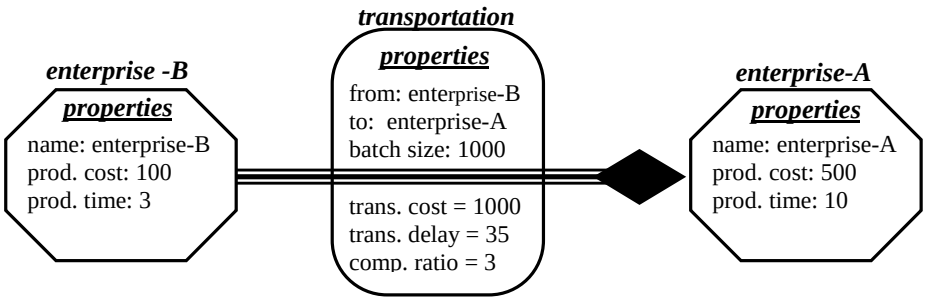



Figure-10: A connected system, consisting of three primitive elements

the row vector of markings (tokens) on the set of places, $x_0$ is the initial marking. ■

Let $I(t_j)$ represent the *set of input places* to transition $t_j$ and $O(t_j)$ represent the *set of output places* to transition $t_j$. Then,

$$I(t_j) = \{p_i \in P : (p_i, t_j) \in A\},$$
$$O(t_j) = \{p_i \in P : (p_i, t_j) \in A\}$$

Similarly, let $I(p_i)$ and $O(p_i)$ be the set of input transitions and the set of output transitions to the place $p_i$ respectively.

*The weight of an arc* from $p_i$ to $t_j$ : if there are $k$ arcs directed from transition $t_j$ to place $p_i$ $p_i \in O(t_j)$ then the weight of the arc $w(t_j, p_i) = k$.

Having defined the ordinary Petri net, now the enabled transitions (conditions that make a transition in ordinary Petri net to fire) is defined:

■ Definition D2 (Enabled transition in ordinary Petri net) [9]:
A transition $t_j \in T$ in an ordinary Petri net is said to be *enabled* if

$$x(p_i) \geq w(p_i, t_j) \quad \text{for all} \quad p_i \in I(t_j).$$ ■

In other words, transition $t_j$ in the ordinary Petri net is enabled when the number of tokens in $p_i$ is at least as large as the weight of the arc connecting $p_i$ to $t_j$. When a transition fires, the vector of markings change:

■ Definition D3 (Firing in ordinary Petri net):
A transition $t_j \in T$ fires in an ordinary Petri net, if and only if $x(p_i) \geq w(p_i, t_j)$ for all $p_i \in I(t_j)$, changes occur in markings

$$x'(p_i) = x(p_i) - w(p_i, t_j) + w(t_j, p_i)$$
$$, \quad i = 1, \ldots, n.$$

■

With these definitions, now it is high time to go through the definitions for the new high-level Petri net.

### 3.4 New High-Level Petri net

In this subsection, definitions for the new high-level Petri net used by the AgileSIM is given.

■ Definition D4 (the new high-level Petri net):
The high-level Petri net used by AgileSIM is a four-tuple $(P, T, A, x_0)$
where $(P, T, A, x_0)$ is an ordinary Petri net

$P$ is the set of places partitioned into subsets $P_E$ and $P_C$ , such that $P = P_E \cup P_C$, and

$T$ is the set of transitions partitioned into subsets $T_E$ and $T_C$ , such that $T = T_E \cup T_C$.

Also,

For a transition $t_{jE} \in T_E$, there is only one input place, i.e. $\|I(t_{jE})\| = 1$, and

For a transition $t_{jC} \in T_C$, there is only one input place and only one output place. i.e. $\|I(t_{jC})\| = 1$, and $\|O(t_{jC})\| = 1$.■

Subscripts *E* and *C* refers to enterprise and transportation respectively.

■ Definition D5 (Firing in the high-level Petri net):
When transition $t_j \in T$ fires in the high-level Petri net, changes occur in markings in the following way:
When transition $t_{jE} \in T_E$ fires:

For the input place $p_{iE} \in I(t_{jE})$, $x'(p_{iE}) = 0$

For all the output places $p_{oC} \in O(t_{jE})$,
$x'(p_{oC}) = x(p_{iE}) \times w(t_{jE}, p_{oC})$,
$o = 1, \ldots, n.$
When transition $t_{jC} \in T_C$ fires:

For the input place $p_{iC} \in I(t_{jC})$, $x'(p_{iC}) = 0$

For the output place $p_{oE} \in O(t_{jC})$,

$$x'(p_{oE}) = m \times w(p_{iC}, t_{jC}) \triangleright x(p_{iC}),$$
$$m \in \aleph^+$$

which means, markings on the output place is a multiple of the weight of the arc $w(p_{iC}, t_{jC})$ just greater than the markings on the input place of the transition. ■

■ Definition D6 (Enabled transition in the high-level Petri net):
A transition $t_j \in T$, $T = T_e \cup T_c$ in the high-level Petri net is said to be *enabled* if
$x(p_i) \geq 0$ for all $p_i \in I(t_j)$, and
$x'(p_o) \leq \lceil x(p_o) \rceil$ for all
$p_o \in O(t_j)$, ■

In other words, transition $t_j$ is enabled when there is *any* number of tokens in the input place $p_i$ and if the tokens added to the outplace by firing will not exceed the limit (maximum) attached to that output place.

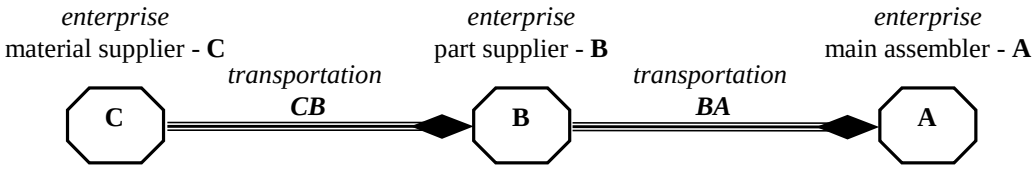
### 3.5 Generating Petri net model from connected system

When AgileSIM automatically generates the Petri net model of the connected system, the following rules are observed:

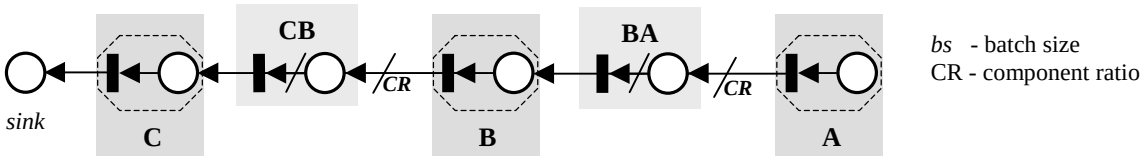
1. Each enterprise is replaced by transition-place pair ($t_E$ - $p_E$). The transition represents the (overall) production at the specific enterprise

and the place represents the out-buffer for products of that enterprise. See figure-11.

2. Each transportation is replaced by transitions-place pair ($t_C$ - $p_C$). The transition is connected to the supplying enterprise ('from' enterprise), and the place is connected to the purchasing enterprise ('to' enterprise). The weight of the arc between the transition and the place is the batch size.
3. The direction of flow in Petri net model is opposite to that of the connected system model. Since the pull system for material flow control is assumed, Petri net model depicts the order flow (or information flow) which should be in opposite direction to connected system model which shows the material flow direction.
4. The weight of the arcs between the places of the transportation ($p_C$) and the purchasing enterprise transition ($t_E$) is the *component ratio*. Component ratio means how many units of parts flowing through a transportation is needed to make a unit of product at the purchasing enterprise.
5. The raw material suppliers or any other enterprises that are not connected to any supplying enterprises in the connected system become *sinks* in the Petri net model. Therefore, extra places are attached to them to accumulate the tokens.




11a: A simple connected system




11b: Petri net model for the connected system in 11a.

Figure-11: Generating Petri net model from a connected system

Figure-11 shows the rules for generating Petri net model from the connected system pictorially. Though the Petri net model is shown pictorially in figure-11b, AgileSIM creates interim Petri net model in a matrix form called *the incidence matrix*. The incidence matrix representing a Petri net is actually a matrix of input places and output places of the transitions in the Petri net model. Figure-12 shows the incidence matrix for the Petri net model shown in figure-11b

**Simulations with Petri net model**

We choose Petri net as the interim system model for simulations because we wanted to make sure that the simulations (time, amount and cost calculations) will be done in linear time. The methodology that is used to manipulate the Petri net model in calculations make sure the it will done in linear time $O(n)$ where $n$ is the number of transportation. This requirement on linear time is a necessity, otherwise because of the size of the system model (large number of enterprises and transportation), non-linear simulation time will make the implementation inappropriate as an on-line real-time monitor.

Calculations are done in two traversals. In the first traversal (*'walk-through'*), the simulation follows the order-flow (for example, in figure-11a, simulation starts at the right-end of the Petri net model- enterprise 'A') with demands on products (sources) as the initial markings. During this traversal, the material flow amount and time profile are calculated. In the second traversal (*'back-tracking'*), the simulation start at the left-end of the Petri net model, following the material-flow. To do this, the flow direction of the Petri net model must be reversed. This reversing is done simply by

$$incidence\_matrix = \begin{array}{r} \\ t_A \\ t_B \\ t_C \\ t_{BA} \\ t_{CB} \end{array} \begin{array}{c} \begin{array}{cccccc} p_A & p_B & p_C & p_{BA} & p_{BC} & p_{\sin k} \end{array} \\ \left[\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & bs_{BA} & 0 & 0 \\ 0 & 0 & 0 & 0 & bs_{CB} & 0 \end{array}\right. \\ \text{input places} \end{array} \begin{array}{c} \begin{array}{cccccc} p_A & p_B & p_C & p_{BA} & p_{BC} & p_{\sin k} \end{array} \\ \left.\begin{array}{cccccc} 0 & 0 & 0 & CR_{BA} & 0 & 0 \\ 0 & 0 & 0 & 0 & CR_{CB} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right] \\ \text{output places} \end{array}$$

Figure-12: Incidence matrix for the Petri net model shown in figure-11b.

swapping the input places and output places in the incidence matrix (see figure-12). During this second traversal, the cost calculations are done.

## 4 • Application example

A small connected system consisting of 10 enterprises (A-B-C-D-E-F-G-H-I-J) and 10 transportation (BA-CA-DB-EB-EC-AF-AG-FH-FI-GJ) is shown in figure-13; the properties of the enterprises and of the transportation are given in table-2 and in table-3 respectively. Suppose some customers order a number of products at the sales agent 'H'. The first part of this example is to compute the time taken (time profile) to satisfy the demand, the inventory left in the participating enterprises and how the costs are added to the product as the product is developed through the supply chain. The second part of this example is about evaluating a potential supplier/distributor.

Let us assume that the demand at sales agent 'H' is 30. This will be the source of the connected model. Now that all the necessary data are given for the system specification (table-2/3), given below is the mathematical formulation approach for problem solving:

### 4.1 Mathematical formulation of the problem

We shall use the mathematical formulation approach by the manufacturing system theory:

***Phase 1: identifying the primitive system***

The primitive systems consists of 10 enterprise elements, and their properties are given in

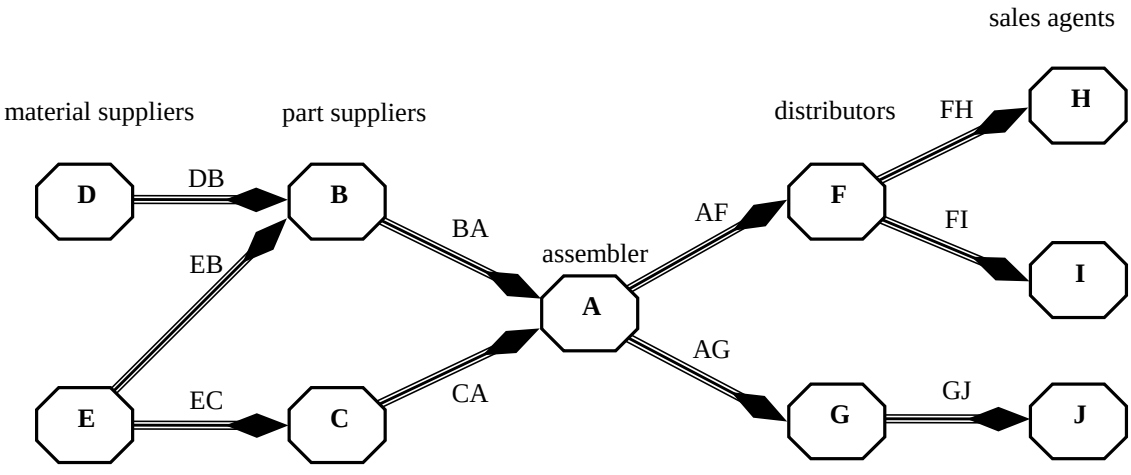



Figure-13: The connected system for the application example

| Table-2: Properties of the enterprises | | | |
|---|---|---|---|
| | *Properties* | | |
| *Enterprise* | *Production Cost (per unit)* | *Production Time (hours)* | *Max. inventory (in units)* |
| A | 250 | 0.3 | 1000 |
| B | 50 | 0.1 | 2000 |
| C | 25 | 0.1 | 3000 |
| D | 20 | 0.05 | 4000 |
| E | 25 | 0.04 | 5000 |
| F | 170 | 0.1 | 1000 |
| G | 175 | 0.1 | 1000 |
| H | 285 | 0.1 | 100 |
| I | 290 | 0.1 | 100 |
| J | 290 | 0.1 | 100 |

table-2. Using AgileSIM these enterprises are to be defined first. For example, to define enterprise 'A' (the main assembler) using the properties given in table-2,

```
> A = enterprise(prod_time, prod_cost, 'A');
```

Similarly, the other enterprises are defined.

***Phase 2: making the connected system***

There are also 10 transportation (properties given in table-3). For example, to define the transportation 'BA',

```
> BA = transportation (c_ratio, batch_size, t_delay, t_cost, B, A, 'BA');
```

After defining the other 9 transportation too, the enterprises are to be connected together to form the connected system. This is done in AgileSIM,

| Table-3: Properties of the transportation | | | | | | |
|---|---|---|---|---|---|---|
| | *Attributes* | | | | | |
| transport-ation | *From* | *To* | *Batch Size* | *Trans. Cost per batch* | *Transport Time hours* | *Component Ratio* |
| BA | B | A | 1000 | 100 | 48 | 2 |
| CA | C | A | 1500 | 100 | 24 | 3 |
| DB | D | B | 1500 | 200 | 48 | 3 |
| EB | E | B | 1000 | 140 | 72 | 2 |
| EC | E | C | 2000 | 140 | 72 | 1 |
| AF | A | F | 100 | 250 | 168 | 1 |
| AG | A | G | 100 | 250 | 168 | 1 |
| FH | F | H | 10 | 50 | 48 | 1 |
| FI | F | I | 100 | 50 | 48 | 1 |
| GJ | G | J | 100 | 45 | 24 | 1 |

> VE1=collaboration(BA, CA, DB, EB, ED, AF,AG, FI, FH, GJ, 'simple Virtual Enterprise');

***Phase 3: applying the sources, and solving the connected system***

There is only one source for this system, which is 30 units in enterprise 'H'. Thus,

> source1=sources(H, 30);

> Petri_model1 = convert (VE1, source1);

By executing the above program code, AgileSIM generates the Petri net model. Petri net model can be simulated using the Petri net simulator- PenSIM which is an integral part of AgileSIM.

> [time_profile, material_flow, cost_calculation] = pensim(Petri_model1);

The results of the simulations, time series ('time_profile'), flow amounts ('material_flow') and incurred costs ('cost_calculation') are in matrix form, therefore can be used by any mathematical analysis packages. Plotting these matrices with MATLAB system gives the figures-14 to -16. Figure-14 shows the time profile. It shows the time taken for production at different enterprises ('A' to 'H') and the time taken due to transportation ('BA' to 'FH') between enterprises. From this figure, it is easy to identify the most time consuming task.

Figure-15 shows how the cost of a product increases as it flows through different enterprises and their connections (transportation), starting from

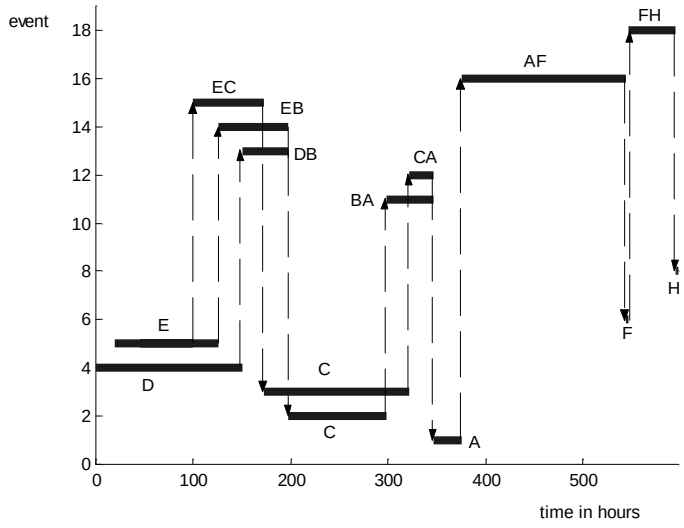



Figure-14: Time profile of the product flowing through the supply chain

the material suppliers 'D' and 'E', through the main assembler 'A', to the final destination- the sales agent 'H'. From this figure, it is easy to identify the most expensive/least expensive production node and transportation. Figure-16 shows the inventory left at different enterprises after production. From this figure, it is easy to identify the unnecessary inventory accumulation along the supply chain.

## 4.1 Evaluation of a potential supplier

An important criteria for AgileSIM is that, there is support for dynamic selection of collaborating enterprises. Suppose the a new enterprise (lets call it enterprise 'X'), a part supplier offers competitive prices, and the decision makers wants to see whether replacing enterprise 'C' by 'X' will improve the overall performance. Using AgileSIM, this kind of studies can be done efficiently, using a few program codes:

***Phase 1: identifying the primitive system***

First we want to remove enterprise C from the primitive system.

```
> VE2 = remove_element(C, VE1)
```

Now the new enterprise 'X' is defined so that it can be added to the primitive system:

```
> X = enterprise(0.1, 35, 'X');
```

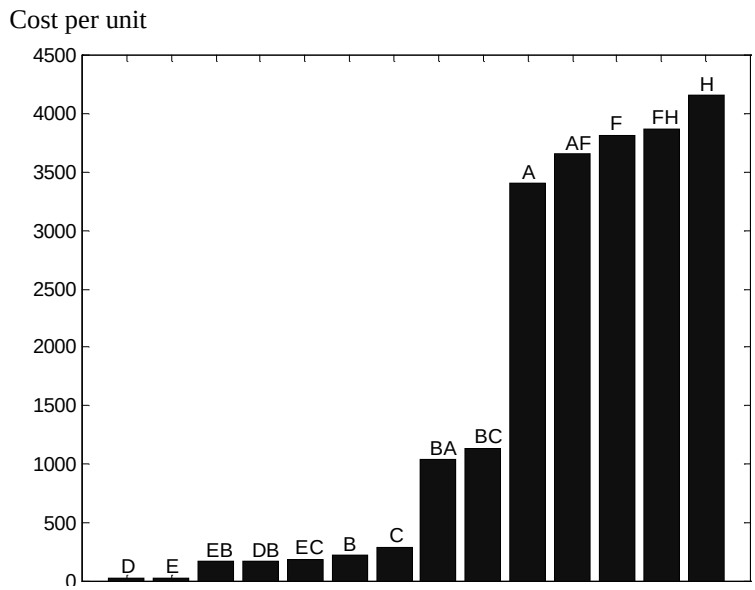



Figure-15: Cost of a unit product as it flow through the supply chain

***Phase 2: making the connected system***

When an enterprise is removed from primitive system, all the connections it has with the rest of the connected system are also removed automatically. We need only to define the new transportation the new enterprise has with the rest of the system. E.g.:

```
> XA=transportation(3,1000,24, 90, X, A, 'XA');
```

And finally, to form the connected system,

```
> AVE2=add_collaboration(XA, VE2);
```

***Phase 3: applying the sources, and solving the connected system***

AgileSIM code 'convert' is executed again to generate the new system model. By this, the Petri net model is obtained, and then simulations can be carried out.

```
> Petri_model2=convert (AVE2, source1);
```

The simulation results due to the new collaboration with enterprise 'X' can now be compared with the results obtained from older collaboration, to determine whether replacing 'C' with 'X' will improve the overall performance of the supply chain. Decision can be made based on the policies like *fastest response time* (by comparing figure-14 with the time profile of the new collaboration), *cheapest price* (by comparing figure-15 with the costs calculations from the new

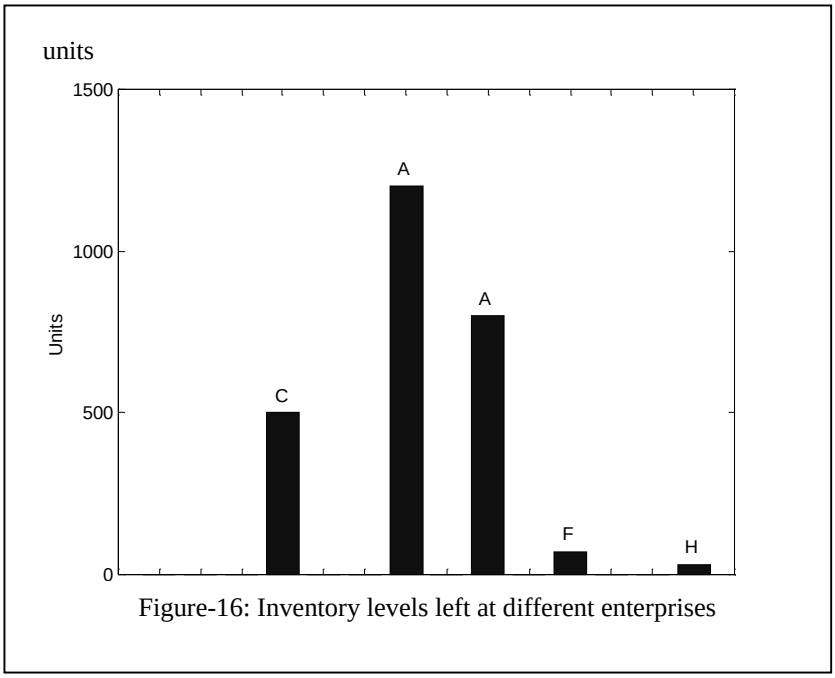



Figure-16: Inventory levels left at different enterprises

collaboration), or *lowest inventory levels* (comparing figure-16 with the inventory levels from the new collaboration). Since the simulation results from the old and new collaborations are in matrix form, any mathematical packages can be used for comparison.

Figure-17 compares the time profiles of the two collaborations. After receiving the customer order at the sales agent 'H', total time taken to furnish finished products at 'H' is about the same in both collaborations. This means, the enterprises 'C' and 'X' have negligible influence on the whole production time.

Figure-18 shows how the costs (due to production/transportation) are added to the product development. By the new collaboration, the total cost of a unit product at the sales agent 'H' is lower (3660 monetary units) compared with older collaboration (4155 monetary units). Finally, figure-18 compares the inventory levels left at different enterprise due to production. In comparison, the new collaboration fares well, as the inventory levels at 'C' is completely eliminated (because 'C' is no longer a collaborating enterprise), and the inventory levels at the main assembler 'A' is reduced; 'A' has two types of inventory, therefore 'A' is given two

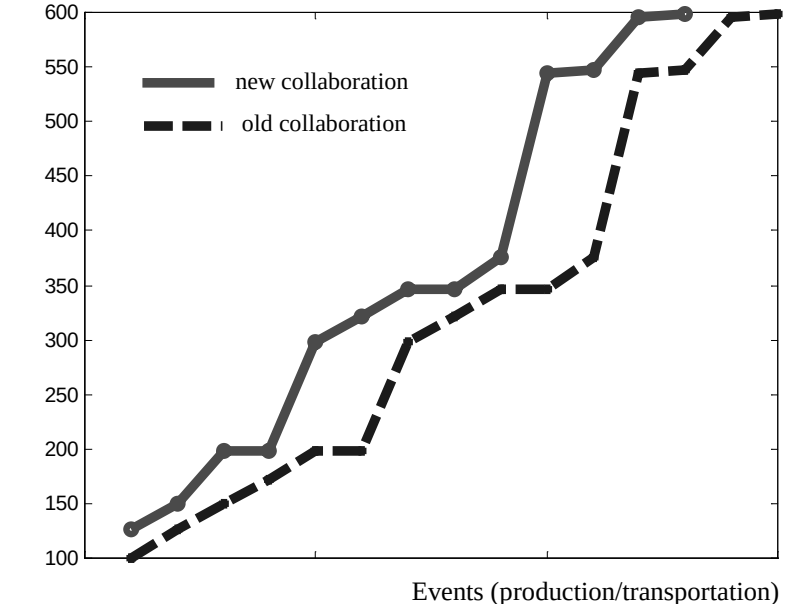



Figure-17: Comparison of the time profiles of the collaborations

columns in figure-18. The inventory levels at 'F' remains same for both collaborations. In figure-18, the inventory level at sales agent 'H' indicates the customer orders

## 5 • Conclusion

The new approach based on manufacturing system theory provides a simple yet effective solution for modeling, simulation and implementation of performance measurement of supply chain in agile virtual enterprise. For the modeling stage, a connected system- a realistic model of an actual virtual enterprise is established, using the topology of connection of the enterprises participating in the virtual enterprise as the starting point. For simulation, an interim Petri net model is generated by the tool AgileSIM; the Petri net can then be analyzed by the Petri net simulator PenSIM. For implementation, the MATLAB system on which the tools AgileSIM and PenSIM run, generates C++ code for compilation and execution on any computer platform. Thus, this approach is very useful for automating a part of supply chain management system, for example as an on-line

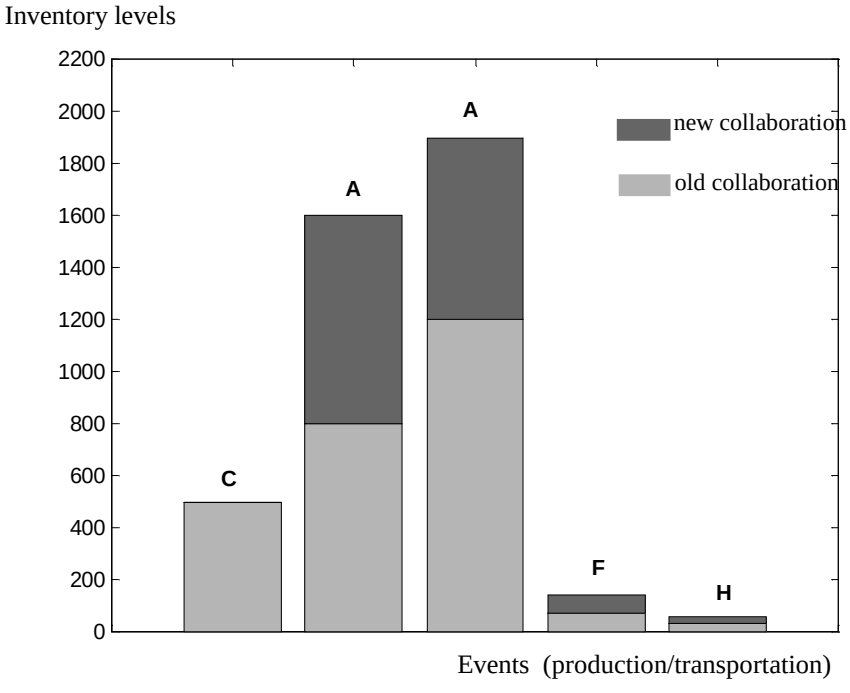



Figure-18: Comparison of the inventory levels left in enterprises

monitor for agile virtual enterprise activities. By selecting a few but most important performance key indicators, the approach discussed in this paper, gives an overview of the performance, which is not that visible in the classical approaches.

## References


[1] Goranson, H. T. *The Agile Virtual Enterprise: cases, metrics, tools*, Quorum Books, ISBN 1-56720-264-0, 1999.

[2] Davidrajuh, R., and Deng, Z. "An Autonomous Data Collection System for Virtual Manufacturing System" , *Int. J. Agile Management Systems, Vol. 2, No.1, 2000*

[3] Enator. "Virtual Enterprising", http://195.100.12.162

[4] Min, H. and Galle, W. P. "International purchasing strategies of multinational US firms", *International Journal of Purchasing and Material Management*, Vol. 27, No.3, 1991

[5] Thompson, K. N. "Vendor profile analysis", *Journal of Purchasing and Material Management*, Vol. 26, No.1, 1990

[6] Davidrajuh, R. and Bjørke, Ø. "A Logic Toolbox for Manufacturing Systems Applications" , Submitted for publication in the *Int. J. Agile Management Systems*, August 2000.

[7] Yam, R., and Lo, W., and Tang, P. Y. "Enhancement of global competitiveness for Hong Kong/China manufacturing industries through i-agile virtual enterprising"**,** Managing Innovative Manufacturing Conference (MIM2000), Aston Business School, U.K

[8] Bonney, M. C. "Are push and pull systems really so different?", *Int. J. Production Economics, vol. 59, pp. 53-64, 1999*

[9] Cassandaras C. G. and Lafortune, S. L. *Introduction to discrete event systems*, Kluwer Academic Publishers, ISBN 0-7923-8609-4, 1999.

[10] Savi, V. M. and Xie, X. "Liveness and Boundedness Analysis for Petri nets with Event Graph Modules", *Petri nets, Lecture Notes in Computer Science series*, Springer-Verlag, 1992

[11] Harhalakis, G., Proth, J. M., Savi, V. M. and Xie, X. "A stepwise specification of a manufacturing system using Petri nets", Proceedings of the 1991 IEEE International Conference on Systems, Man and Cybernetics, Charlottesville, Virginia, October 1991

[12] Bjørke, Ø. *Manufacturing Systems Theory - A Geometrical Approach to Connection*. Tapir, Trondheim, Norway, 1995.

[13] Wang, K. "A New Modeling and Analyzing Approach to Material Flow and Productivity", International IFIP Conference on Computer Applications in Production and Engineering, Beijing, China, 1995

[14] Hussein, B. "On modeling mechatronic systems: a geometrical approach. Department of Production and Quality Engineering", Norwegian University of Science and Technology, Report NTNU 97007, 1997.

Reggie Davidrajuh received Masters degree in Control systems engineering in 1994, and Ph.D. in Industrial engineering in February 2001, both from the Norwegian University of Science and Technology (NTNU). He is currently an assistant professor of computer science at the department of technology at Narvik Institute of Technology, Norway. His current research interests include e-commerce, agile virtual enterprises, discrete event systems and modeling of distributed information systems. Email: rd@hin.no, Fax: +47 96766810, Phone: +47 76966000; URL: http://www.hin.no/~rd