OBJECT-ORIENTED MULTI-INDENTURE MULTI-ECHELON SPARE PARTS SUPPLY CHAIN SIMULATION MODEL.

Manuel D. Rossetti and Soncy Thomas

Abstract

In supply chain networks, the flow of material occurs between various business entities such as suppliers, manufactures, distributors and retailers, which can be considered as the various echelons (levels) in a multi-echelon supply chain network. Spare parts supply chain management deals with optimizing stocking, purchasing, repairing and disposing activities of purchased parts and their components for maintaining a successful user service level for a system involving repair and operating functions. A spare parts inventory system has products which will have relationships with other products. A multi-indenture structure defines the structure of the products that flow through the supply chain system. In this article, we present a standardized, object-oriented, data-driven, simulation framework for the evaluation of multi-echelon multi-indenture spare part supply chain networks.

Key Words

Spare parts, inventory, object-oriented simulation, multi-indenture, multi-echelon, supply chain.

1. Introduction

A spare parts supply chain system can be considered as a network of facilities and distribution options that operate to obtain raw materials, transform these materials to finished products, distribute these finished products to the customers depending upon

their demand, and repair the failed products. The importance of spare parts supply chain management has increased in the past decades to reduce the down time of critical equipment such as computer equipment, medical equipment, and military equipment. Fast and effective supply of spare parts is required for effective corrective maintenance which leads to the need of spare parts supply chain management [1]. In this research our goal is to analyze and identify the fundamental elements necessary for modeling generic spare parts supply chain scenarios via simulation. We provide an object-oriented, generic simulation framework for multi-indenture, multi-echelon (MIME) spare parts supply chain networks. The Unified Modeling Language (UML), which has emerged as a standard for object-oriented modeling, has been used for depicting the design details of the framework. This framework presents a reusable design of a supply chain network by representing the various behaviors of the supply chain entities as a set of abstract classes which allow the user to plug in different behaviors without altering the structure of the framework. The framework can be used for the evaluation of new or existing supply chains, testing of analytical optimization algorithms, and embedding in optimization algorithms. In the following, we briefly review literature in the area of spare parts inventory management to motivate our work and as a basis for the testing of the framework discussed later in this paper.

The Multi Echelon Technique for Recoverable Item Control (METRIC) theory has formed the basis for theoretical results in supply chains that involve repairable items. In the METRIC theory, the objective function is to minimize the expected number of back orders. The METRIC theory calculates the optimal stock level at different bases (echelons) in a multi-echelon system for every first indenture item in the system [2], [3].

METRIC theory assumes that the units under repair depend on the total number of units in service, the repair rate, and the delay for obtaining the spares [4].

The multi-indenture problem can be modeled accurately by taking into account the dependencies between the indentures using VARI-METRIC theory [2]. This resulted in a ten-fold improvement over the standard METRIC results [2], [3]. In the VARI-METRIC technique, the calculation of the expected number of back orders is done by taking into account not only the mean pipeline values but the variance as well. The VARI-METRIC technique assumes a Poisson demand with a constant mean. The VARI-METRIC technique also assumes that all units can be repaired and that there exists no lateral supply [3]. In VARI-METRIC theory, a product failure is caused by at most one subcomponent failure and all products are considered equally critical.

IBM has developed an optimizer for flexible and optimal control of service levels and spare parts inventory. The optimizer helped in reducing the inventory investment and operating cost and improving the service levels for IBM [5]. The objective of the optimizer was to determine the stock control policy for each location and for each part that would minimize the expected costs for the whole system. The cost function included replenishment cost (which includes transportation, handling and order setup costs), emergency cost, and inventory holding cost [5].

The previously mentioned methods, utilize analytical modeling techniques based on stochastic inventory theory. These techniques are limited in their ability to fully describe real spare part networks. Among the various alternative methodologies that are available, simulation is highly recommended for analyzing complex systems [6]. Dong [7] considered simulation as a better technology for designing supply chain systems due

to the system variation and interdependencies. In a supply chain network, since the demand forecast is the most important variable that affects the movement of material in the network and demand forecast is susceptible to a large amount of variation, simulation is one of the best methods to analyze the network when the key driver is variance [8]. Ingalls [8] also suggested that simulation is the best tool for evaluating rule-based systems in which the customers or certain groups of customers have priority in selecting the products.

Among the major problems for using simulation for complex supply chain analysis is that most of the simulation models are specific to a particular problem and have a limited use [9]. The two major problems that are associated with simulation models are 1) they take a long time to develop and 2) they are very specific and have limited reuse [10]. In order to address the reusability of the simulation models, Rossetti and Chan [9] developed the Supply Chain Simulation Framework (SCSF), which facilitates the dynamic analysis of multi-echelon supply chain systems. We enhance and build upon their work for modeling the multi-echelon aspects of the spare parts supply chain simulation model.

2. Spare Pats Supply Chain Simulation Framework Development

The product structure of failed products and the subcomponents has to be accounted for when analyzing the repairable products network. In a MIME system, a product may consist of critical and non-critical subcomponents. The working condition of all the critical subcomponents in a multi-indentured product together with the functioning of the

end product simultaneously determines the availability of the end product. For example, airplanes in a military weapon system or copiers in a commercial supply chain system depend on the operational capabilities of the critical and non-critical subcomponents. The failure of non-critical subcomponents in a product will make the product partially mission capable (PMC) and the failure of critical subcomponents will make the product failed due to subcomponents. In addition, a component in the MIME system has a chance of failing independently of any of the product's subcomponents status.

A series of events occur when an end item arrives at a facility (the product's primary operating base) after its operational cycle, starting with the diagnosis of the operational capability of the end item and its subcomponents. The products that are failed, either independently of subcomponents or because of the failed subcomponents, create demand for spare parts. A list of demands constitutes an order. The order is placed at the primary facility for repair or replacement. The facility tries to repair the failed products. The failed products that are not repairable at the local facility are sent to another facility for repair. A warehouse, which supports the facility, will try to satisfy the demand that was created due to the failed product with its stocked inventory. If the warehouse is out of stock, the demand is backordered until a new product becomes available. Replenishment for the warehouse can come from either the local repair station or from another echelon. A diagrammatic representation of a single echelon multi-indentured product is shown in Fig. 1. Fig. 2 shows a diagrammatic representation of a multi-echelon system. The bases in the Fig. 2 can be considered as the first echelon, the depots as the second echelon and the contractor as the third echelon. The demand for the failed products will occur at a base from the end items that are associated with that base.

The failed products that cannot be repaired at the base facility will generate demands for the depot and the failed products that cannot be repaired at the depot generate demands for the contractor. The flow of demands and replenishments that occur in the system are also illustrated in Fig. 2.

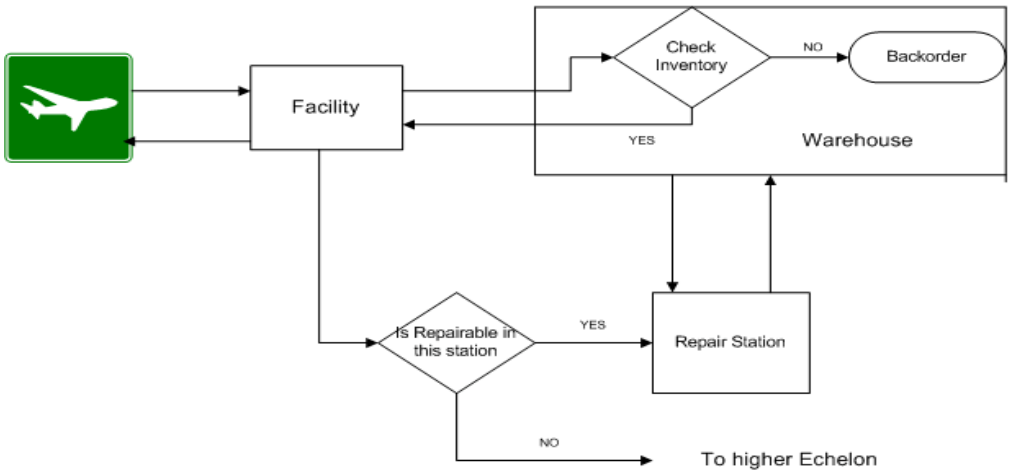

Figure 1 A single echelon system

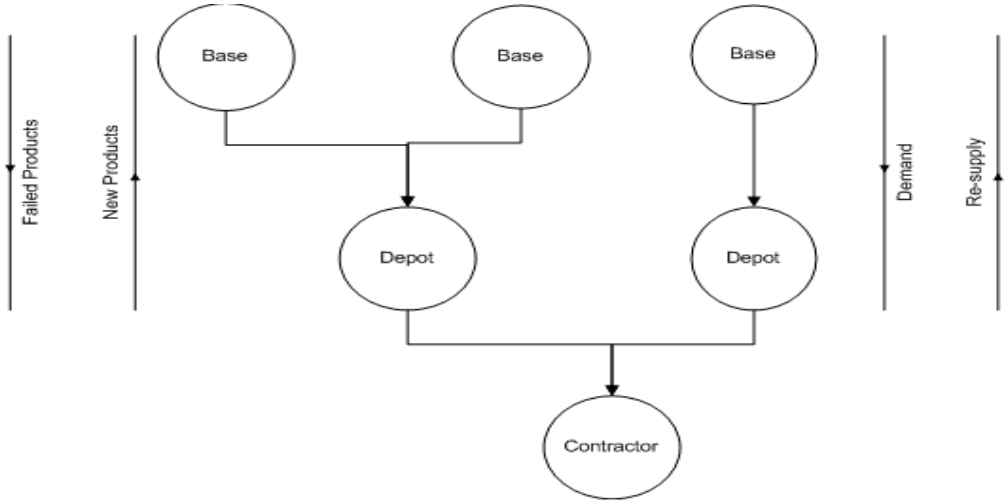

Figure 2 Multi-echelon system

A detailed analysis of the MIME system is possible by analyzing the multi-indenture architecture and the multi-echelon architecture. The multi-indenture architecture depends on the relationships that exist between a product and its subcomponents. The bill of

materials (BOM) can be considered as a data structure that stores the relationship between the end items and the lower-level items [11], [12]. In supply chain networks where the customers are given a high degree of freedom for their product specification, the design and maintenance of BOM structures for these products are difficult [13]. In this research, we use a database to hold the BOM structures for modeling the MIME spare parts supply chain system. This will help users change the simulation model according to their requirements.

Each product in the MIME system has an operation time (run time), which is the random amount of time the product will work when the product is assigned to do an operation. The failure time of a product can be defined as the random amount of time the product works before it fails. In other words, if the operation time of a product exceeds the failure time associated with the product, failure will occur. In our framework, the end item or the product that does not have a parent is assumed to operate for a random or scheduled amount of time. All the children of this main product will assume the same operating time as that of the operation time of the parent product. The failure of a product or the failure of the subcomponents associated with the product defines the different state changes associated with the product.

The *working state* is defined as the state of the product while the product is operating. As illustrated in Fig. 3, a product can reach the *working state* only when the product is in the *available state* or in *PMC state*, in other words, a product can work only when is completely functional or partially functional (some non-critical components are not operationally capable). The *failed state*, as explained earlier happens if the operation time of a product exceeds the failure time associated the product, the product goes into

the *failed state*. A product may reach the *failed state* only from the *working state* because failure can occur only if the product is working.

A product is in its *Partially Mission Capable (PMC) state* when the product is available for operation with one or more of the subcomponents in the *failed state*. The *PMC state* of a product can be reached only from the *working state* of the product. If a product is ready for operation with none of the subcomponents of the product in the *failed state*, the product is in the *available state*. The *available state* of a product can be reached from the *PMC state*, *working state*, and the *ready to issue state*. The *failed due to subcomponent state* of a product occurs when the product is not available for operation because it contains at least one failed subcomponent, which was critical for the functioning of the product. This state can be reached only from the *working state* of the product. Once a failed product is repaired by a repair station, the state of the product becomes the *repaired state*. A product in the *repaired state* may not confirm that all the subcomponents of the product are in a state that is ready for functioning. The product may still have some of the subcomponents in the *failed state*. A product reaches the *repaired state* only from the *failed state*, because repair is needed only when a product is failed.

The *ready to issue state* of the product conveys the idea that the product is in inventory. A product in the *ready to issue state* confirms that the product and all its subcomponents are completely ready for operations and that it can be used as inventory. A product in the *failed due to subcomponent state* or in the *failed state* can be transformed into the *ready to issue state*. A product is in the *initial state* before a simulation replication begins and after the simulation replication ends. This *initial state* is

useful because this state allows all products to have a common state before the beginning of a simulation replication. From the *initial state* a product can change only into the *ready to issue state.* The state transformation procedure of the product is illustrated in Fig. 3.

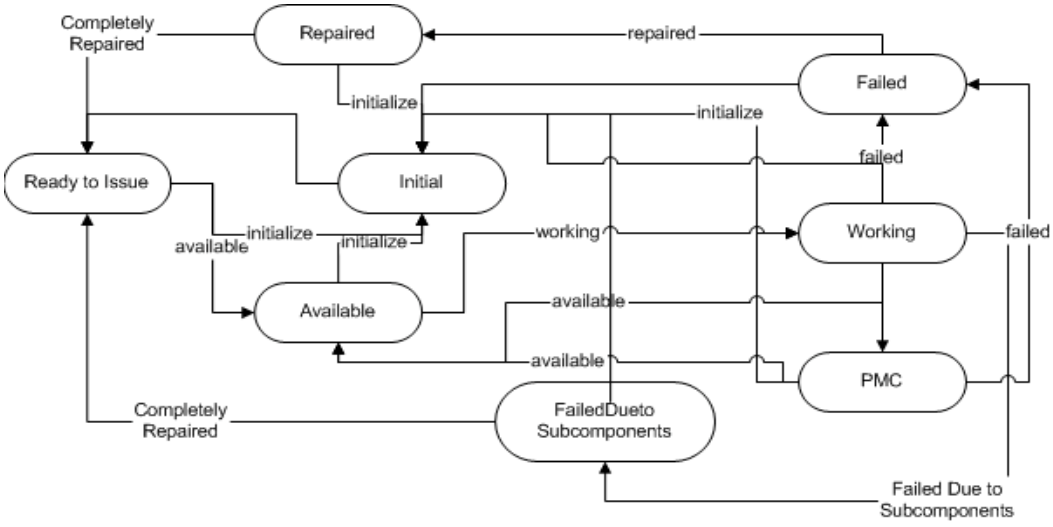

Figure 3 State change pattern

This framework allows the higher indentured products to work without having the failed subcomponents contained in the product, when the subcomponents are taken to the facility for repair or replenishment. The higher indentured products in the system should be aware of the missing subcomponents in the product that were sent to a facility for repair or replenishment. The list of these missing components will be used for re-attaching the repaired or new subcomponents back into their respective positions.

The orders that were created because of the failure of the products in the system are sent to a facility for repair or replenishment. Various behaviors that are associated with a facility are assigned to the agents of the facility. The *Order Receiving Agent* of a facility processes the orders that arrive at the facility from another facility and sends the failed products to the repair station attached to the facility. The *Order Sending Agent*

creates demands for failed products that a facility cannot repair at the local repair station, and sends the created demands as an order to a different facility. The *Shipment Receiving Agent* of a facility breaks down the shipments of good products received from another facility, and sends those products to the warehouse of the facility.

The *Shipment Sending Agent* makes the shipment with the orders that were satisfied at the facility and then sends the shipments to the customer facility that created the order. The *End Item Scheduling Agent* schedules another operational cycle for the end items that arrive at the facility provided the end items are operationally capable. Fig. 4 illustrates the facility and the functionality of its agents.

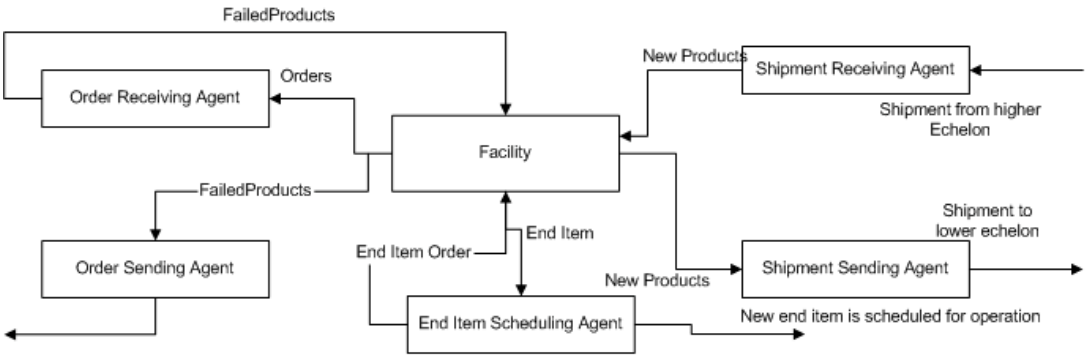

Figure 4 Agents of a fFacility

The agents facilitate easy behavioral changes for a facility. For example, a contractor is a special kind of facility that receives demands for products that are not repairable at any of the repair stations of the facilities. A contractor is distinguished from a facility by the agents it uses.

Another major component of a facility is the *Warehouse*. The function of a warehouse in a facility is to store the products that are ready for issue. The warehouse of a facility is capable of storing any number of products and maintaining inventory

associated with all of the products. The warehouse deals with two different kinds of orders. External orders represent the orders that are flowing in and out of a facility and internal orders represent orders generated at a facility's repair station by a product containing demands for the missing components in the product (failed removable subcomponents that were removed for repair) that were created while disassembling the product. Internal orders also include orders from the end item scheduling agent of the facility for a good end item. Fig. 5 illustrates the function of a warehouse in a facility.

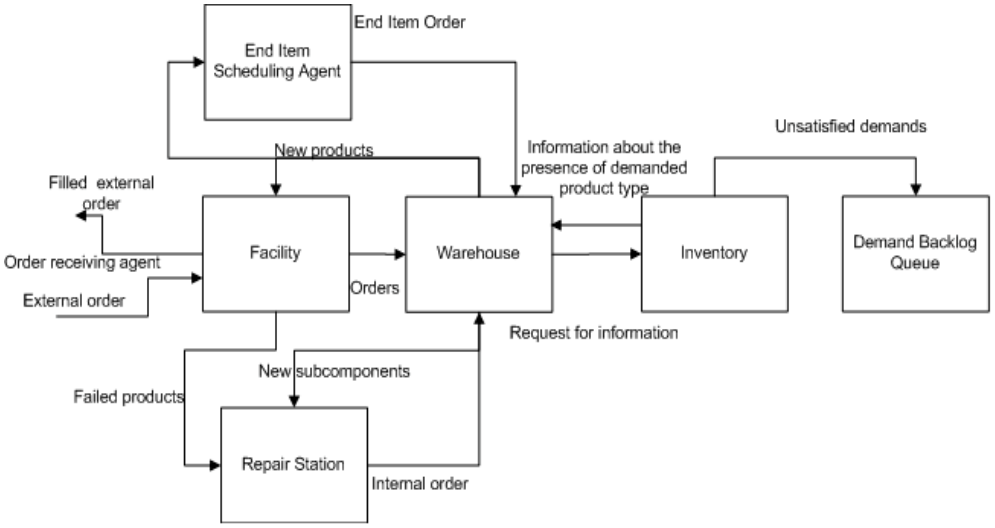

Figure 5 Warehouse of a facility

The *Repair Station* of a facility is the component which performs the repair work on failed products that arrive at a facility. The major behaviors of a repair station are assigned to its components so as to enhance the flexibility of the framework. The *Disassembling Unit* of the repair station disassembles the failed products and the *Assembling Unit* attaches the components into their respective positions within products that were disassembled for repair. The *Work Station* of the repair station acts as a resource in the repair station and repairs the failed products. The *Inspection Unit* of the

repair station checks for any failed subcomponents in the product that are not repaired locally. If the product contains any failed parts, because they cannot be repaired locally, the product will be sent to the order sending agent of the facility where it will be sent to a different facility for repair. The *Dispatcher* of the repair station makes repair jobs for the different functional units of the repair station and then assigns the repair jobs according to the availability of the functional units. Fig. 6 illustrates the relationship between the repair station and its components.

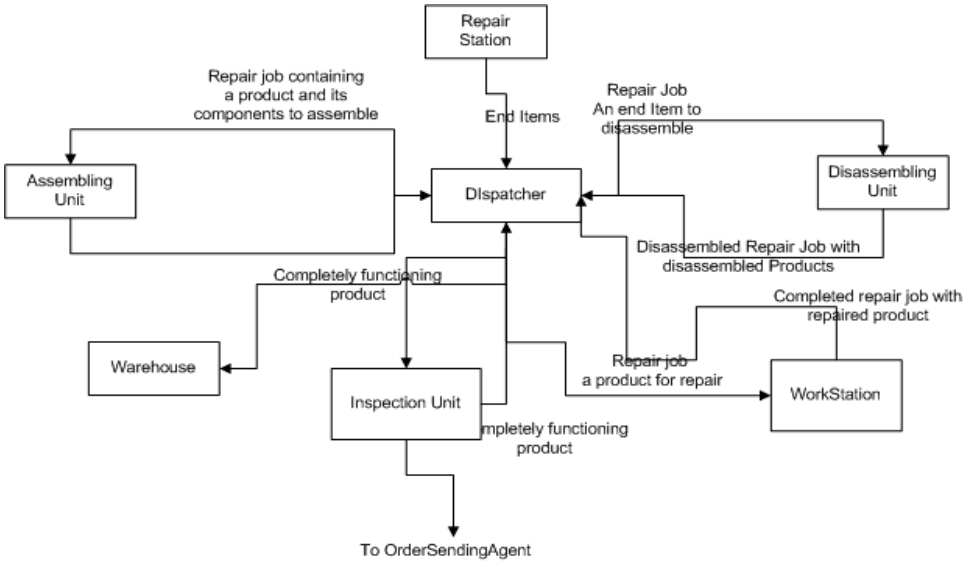

Figure 6 Repair station and its components

We designed the spare parts supply chain framework by dividing the framework into 4 subcomponents which includes database, preprocessor, MIME Application, and a postprocessor. In order to make a generic MIME supply chain simulation model, the model receives all the data that is required to create the multi-echelon and multi-indenture architecture from a database. In our implementation, we have decided to use Microsoft Access as the database. The design is not dependent on the use of Microsoft Access. Other database engines such as MySQL, Oracle, DB2 can be easily substituted as

the database store. The database consists of 6 tables. Table 1 shows the list of tables used in the database and their functions.

Table 1 List of Tables Used in the Database

ProductType Table	Stores the information necessary for making product types necessary for a system.
BOM Table	Stores the parent-child relationship that exists between the products which is the fundamental relation behind the indenture structure of a product
Facility Table	Stores the data identifier of all the facilities that are required for the network and the class names of the agents which each facility needs to make.
FacilityRelationship Table	Stores information required for building multi-echelon structure (customer-supplier relationship).
ProductFacilityRelationship Table	Stores information that is dependent on both product type and facility such as inventory, mean repair time, repair probability etc.
RepairLocation Table	Stores information of the facilities that can repair a particular product type
EndItemFacilityAssociation Table	Stores the ProductTypes of the end items and the number of end items associated with each facility.

To avoid excessive database traffic within the model, we have used a preprocessor (MIMEStructure class) to query all information required for building the system from the database before the simulation begins. The preprocessor also has the responsibility of obtaining the Java Database Connectivity (JDBC) which allows applications to access SQL databases. The connection between the Java class (MIMEStructure) and MS Access is obtained by using the JDBC - ODBC (Open Database Connectivity) Bridge. Fig. 7 illustrates the components of the MIME simulation environment. The database facilitates restructuring of the MIME systems by altering the data input of the database. The simulation model is constructed entirely from the database. Hence the time taken for modeling different systems using the MIME simulation framework will entirely depend upon the time required to collect the essential data for the system such as the product structure, the failure time of the products, the repair time, etc.

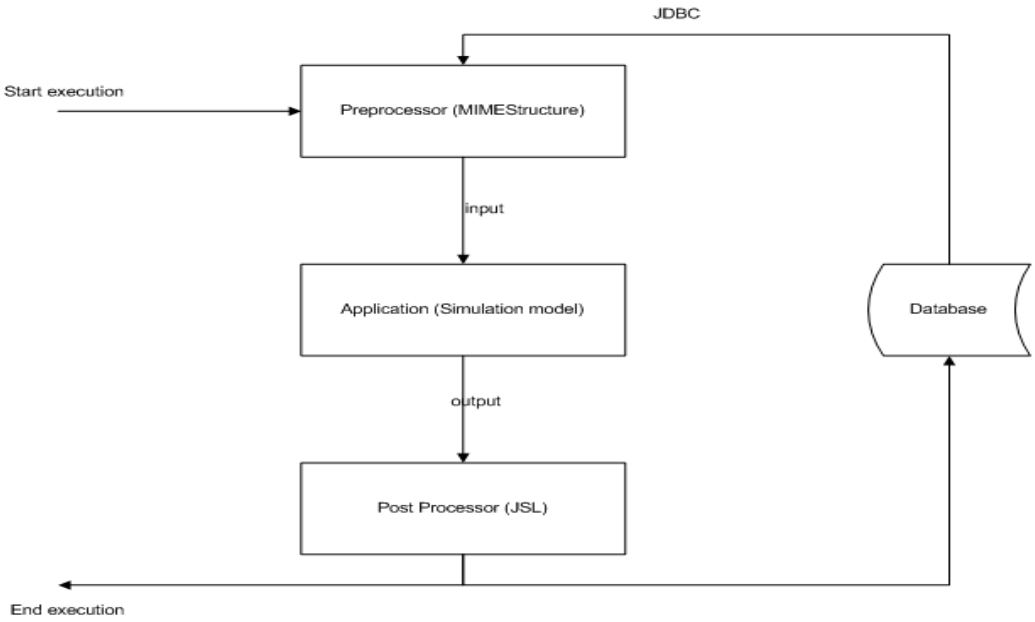

Figure 7 MIME system design

The application includes the design of the multi-echelon and multi-indenture architectures. The facilities and the relationships (supplier-customer) between facilities are created by the "MIMEStructure" class (Preprocessor). The MIMEStructure uses a network pattern to create all the facilities in a system and its relationship with other facilities. The general network pattern has a network class and two additional classes, one for nodes and the other one for arcs. Fig. 8 illustrates the UML representation of the design which we used to create a facility and its relationships with other facilities.

In Fig. 8 the *MIMEStructure* represents the general class in the network pattern. The facility class represents the node and the *FacilityRelationship* represents the arc. A customer can have many suppliers and a supplier can have many customers. The *FacilityRelationship* contains information such as the time taken for the shipment of a particular product type from the supplier to the customer. In the network, every relationship between facilities can exist only when a *ProductType* is associated with it.

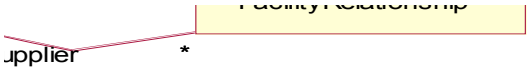

Figure 8 Design of facility and its relationship with other facilities

The facility segregates the different functions that should be performed by the facility into the component classes which it has created for doing the specific functions. The association between the facility and its components is an aggregation which implies that the facility contains all its components. A diagrammatic representation for the aggregation association of a facility and its components is illustrated in Fig. 9.

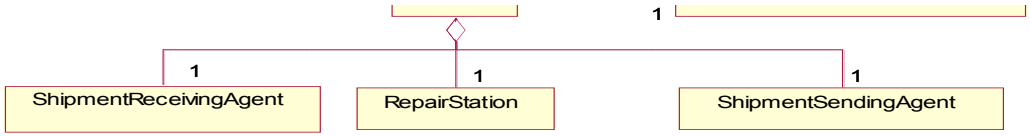

Figure 9 Facility and its components

When a warehouse is made by a facility for storing the inventory associated with the facility, the warehouse will make an inventory class for each product type in the system. The relationship that exists between *Warehouse* and *Inventory* is aggregation, which implies that all the inventory objects created are contained in Warehouse.

When a facility creates a repair station, the subcomponents of the repair station are also created. The association between the repair station and its components is aggregation with a link attribute which defines the number of subcomponents that should

be created by the repair station. The aggregation association implies that the repair station contains the subcomponents that were created. The link attributes which the repair station uses for creating the subcomponents are derived from the Facility depending upon the type of subcomponents to be created. The UML representation of the relationship that exists between the repair station and its subcomponents is illustrated in Fig. 10

Figure 10 Repair station and its components

In order to build the multi-indenture architecture, a network pattern was again used for creating the products. The class, *MIMEStructure* represents the network class; *ProductType* and *ProductRelationship* are the classes that represent the nodes and the arcs respectively. *ProductType* represents the different types of products that are present in the system and *ProductRelationship* represents the relationship that exists between the products.

A product type can have many product relationships in which it can act as parent to other product types or a child to several product types. But a product relationship has only one parent and only one child contained in it. This represents two, one-to-many, relationships. Aggregation is used to represent the relationship between the MIMEStructure and its components because it contains the *ProductType* and the *ProductRelationship* classes. The above representation is shown in Fig. 11.

Figure 11 Design of multi-indenture architecture

The post processor of the simulation model simulates and gives the output of the simulation. The Java Simulation Library (JSL) package is a simulation resource library, which we utilized to simulate the MIME, supply chain system [9]. In this system, all the supply chain elements are connected to the JSL, which is described in reference [14]. We have designed the network in such a way that all the supply chain elements such as the facility, warehouse, repair station, order, etc. are derived from a JSL class called ModelElement which allows our model to access the functionalities of the JSL such as scheduling events, access of random number generators etc. The events scheduled by the subclasses of the ModelElement advance the simulation clock associated with the JSl. The JSL writes the output of the simulation to a database.

As discussed above the MIMEStructure class (Preprocessor) deals with the data interaction that is only done prior to the system simulation. Therefore a singleton pattern was used to implement MIMEStructure. A singleton pattern ensures the creation of only one instance and provision of a global point to access it [15]. While implementing the multi-echelon structure, we used a mediator pattern for the Facility and the RepairStation classes. A mediator (Facility and Repair Station in our framework) facilitates loose coupling by keeping objects from referring to each other directly, which allows an

independent interaction between the agents of the Facility and the components of the RepairStation.

We have used the delegation pattern for implementing the agents of the facility and the components of the Repair station. The delegation pattern is a method where an object outwardly expresses certain behaviors for example (facility's behavior of order reception) but in reality delegates responsibility for providing that behavior to an associated object (Order Receiving Agent of Facility). All the agents of a Facility and the components of the Repair Station are defined as abstract classes. This allows the user to instantiate a new agent of different behavior and attach to the facility with out altering the design of the model. Fig. 12 shows the abstract class implementation of the agents of a Facility. The delegation pattern used for partitioning the system into many agents and the mediator pattern used for reducing the interconnections between the objects enhances the reusability of the framework.

Figure 12 Abstract implementation of the agents of a facility

To simulate the MIME supply chain model, a class named MIMEApplication was created. MIMEApplication creates a model by instantiating the class Model in the JSL.

An experiment is then setup for the model by instantiating an Experiment class in the JSL. The experiment will allow the user to define the simulation parameters such the replication length, the warm up period and the number of replications. In order to collect the statistical report for an experiment, the user should turn on the report collection methods of the Experiment class. Fig. 13 shows the connection between the JSL and the MIME simulation framework.

Figure 13 JSL and MIME simulation framework

3. Framework Testing and Validation

The MIME simulation model was validated by comparing the results with a VARI-METRIC model [2]. We also simulated two other spare parts supply chain systems, where we used some of the additional features provided by our framework. The VARI-METRC example involves a three echelon, three indentured spare part supply chain system. The first echelon consists of 50 submarines, where each group of 10 is supported by a ship (the second echelon), that can be accessed in one day. The supply ships have a 39 day average re-supply from the depot (third echelon), when the depot has stock on its shelf. In this example, it is assumed that item 9 is the same as item 3, item 10 the same as item 5 and item 11 the same as item 6. Table 2 and Table 3 represent the initial

parameters for the VARI-METRIC system. The initial parameters such as the product structure, the echelon structure, the failure time, repair probability and repair time of a particular product type at each facility etc. are specified in the database. The inventory policy at every echelon is assumed to be an (S-1, S) policy. Because of the space limitations, we refer the reader reference [2] for further details of the VARI-METRIC problem.

Table 2 Repair Probabilities of Product Types at each Echelon.

Item	First echelon Repair Probability	Second echelon Repair Probability	Third echelon Repair Probability
1	0.8	0.2	0.5
2	0.6	0.4	0.5
3	0.4	0.5	0.5
4	0.2	0.5	0.5
5	0.2	0.5	0.5
6	0.2	0.6	0.5
7	0.7	0.3	0.5
8	0.5	0.5	0.5

Table 3 Initial Stock Level and the Demand.

Item	Indenture	Base Demand	Depot Stock	Ship 1 Stock	Ship 2 Stock	Ship 3 Stock	Ship 4 Stock	Ship 5 Stock
1	**1**	**0.7**	**2**	**0**	**0**	**0**	**0**	**0**
2	**2**	**0.2**	**1**	**0**	**0**	**0**	**0**	**0**
3	**3**	**0.1**	**2**	**0**	**0**	**0**	**0**	**0**
4	**3**	**0.1**	**1**	**0**	**0**	**0**	**0**	**0**
5	**2**	**0.9**	**10**	**1**	**1**	**1**	**1**	**1**
6	**3**	**0.1**	**4**	**1**	**1**	**1**	**1**	**1**
7	**1**	**1.1**	**3**	**0**	**0**	**0**	**0**	**0**
8	**2**	**0.3**	**2**	**0**	**0**	**0**	**0**	**0**

The fill rates of the products obtained from the simulation framework and the analytical model are compared for the validation of the framework. Table 4 represents the analytical (top cell) and simulation results (bottom cell) of the fill rates at the depot and Table 5 represents the fill rates at the ship. Since fill rate is defined as the percentage of demands that can be met at the time they are placed, the fill rate at the ships for the product types with zero inventory are zero. This is the reason why only the fill rates of product type 5 and 6 are provided. For the VARI-METRIC model and the rest of the models discussed in this section, the simulations were run for 30 replications with replication length of 300 time units and a warm up period of 50 time units. In the simulation results, the values in the parentheses represent the half width of a 95% confidence interval. Clearly, from these results we can reproduce the results of the VARI-METRIC problem.

Table 4 Fill Rates for First Echelon (Depot)

Item	1	2	3	4	5	6	7	8
Depot	26.82	29.22	46.41	47.64	62.57	62.56	11.74	27.37
	27.6 (0.6)	29.8 (0.6)	46.4 (0.6)	48.2 (0.8)	61.9 (0.7)	61.0(0.7)	11.7 (0.4)	27.1 (0.4)

Table 5 Fill Rates for Second Echelon (Ship)

Item	Ship 1	Ship 2	Ship 3	Ship 4	Ship 5
5	53.8	53.8	53.8	53.8	53.8
	53.4 (0.5)	53.1 (0.5)	53.5 (0.5)	53.3 (0.5)	53.6 (0.5)
6	83.44	83.44	83.44	83.44	83.44
	83.3(0.6)	82.5 (0.6)	83.5 (0.6)	83.3 (0.4)	82.9 (0.6)

In order to test some of the features that we have implemented in our framework, we made two additional models. Among the two models, model 1 is used as the reference model and model 2 is compared with the results of the model 1 to see the effects of the changes made.

In model 1, we are using the same multi-echelon structure as that of the VARI-METRIC model and a different multi-indenture structure. In this model product 8 has two product 3 (only one in VARI-METRIC model) and product 5 has two product 6 as subcomponents (only one in VARI-METRIC model). The structure of the products is illustrated in Fig. 14. The change of structure was easily done by changing the attribute "quantity" in the BOM table of the database which allows a product to have more than one subcomponent of the same product type.

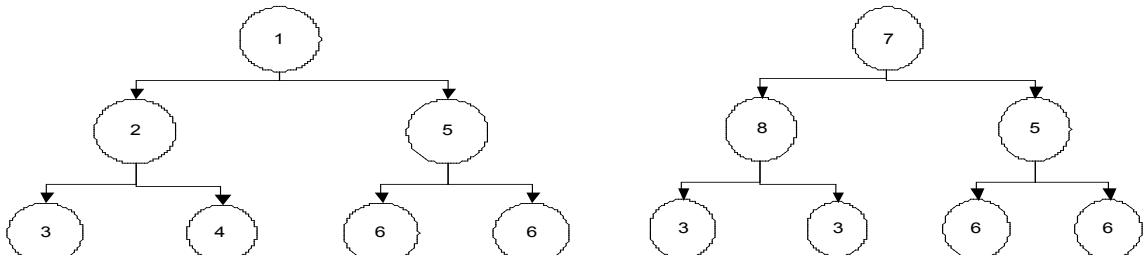

Figure 14 Product structure for model 1

In each submarine, an end item 7 and two end item 1 are set into operation at the beginning of the simulation. All the three end items will run for randomly generated operational time. Failure of more than one subcomponents and the end item are possible in this model. Instead of generating demands for submarines according to the demand rates as in the VARI-METRIC model, in model 1, demands are created for the failed subcomponents or the end item after the completion of each operational cycle of the end item depending upon their respective product states. Hence the demands of failed products depend on the number of end items allocated to each submarine. In other words, the more the number of end items operating in a facility the more the demands for the failed products. The fill rates for the depot are given in Table 6 and Table 7 gives the fill rates for the ships.

Table 6 Fill Rates for First Echelon (Depot)

Item	1	2	3	4	5	6	7	8
Depot	17.2 (0.4)	14.8 (0.5)	33.4 (0.5)	32.3 (0.6)	35.7 (0.5)	49.1 (0.6)	38.8 (0.5)	18.8 (0.4)

Table 7 Fill Rates for Second Echelon (Ship)

Item	Ship 1	Ship 2	Ship 3	Ship 4	Ship 5
5	41.5 (0.4)	41.3 (0.4)	41.5 (0.4)	41.4 (0.4)	41.5 (0.4)
6	70.3 (0.5)	70.5 (0.6)	69.8 (0.7)	70.4 (0.7)	70.3 (0.6)

The explanation of the results of model 1 is given in comparison with the results of model 2 after the following discussion. In model 2, we depict the effects of some of the attributes which we have been assigned for the product and the resulting state changes the product will go through during the simulation. As explained in the earlier sections, a product can fail due to the failure of its subcomponents or failed subcomponents can make a product partially mission capable and a product may not be removable from its parent product. These attributes can be set in the BOM table of the database. In model 2, we discuss the effects of these two attributes by comparing the results with model 1.

In the indenture structure provided in Fig. 14, model 2 assumes that failure of product 6 will make the product 5 partially mission capable. This model also assumes that product 3 is a subcomponent not detachable from its parent product 8. Table 8 and 9 shows the fill rates of depot and ships obtained from simulating model 2.

Table 8 Fill Rates for First Echelon (Depot)

Item	1	2	3	4	5	6	7	8
Depot	32.9 (0.5)	14.8 (0.3)	84.6 (0.6)	31.8 (0.3)	73.3 (0.5)	63.2 (0.5)	49.3 (0.7)	6.0 (0.2)

Table 9 Fill Rates for Second Echelon (Ship)

Item	Ship 1	Ship 2	Ship 3	Ship 4	Ship 5
5	59.3 (0.4)	59.6 (0.4)	59.4 (0.4)	59.1 (0.5)	59.3 (0.5)
6	71.0 (0.6)	70.3 (0.5)	70.1 (0.5)	70.6 (0.4)	70.8(0.5)

Comparing the two results given in Tables 6, 7 (model 1) and Tables 8, 9 (model 2), the fill rate at the depot for product 3 is increased from that of model 1 because in model 2, product 3 was considered as not removable from the parent product 8 in the end item 7 hierarchy. Therefore individual demands for failed product 3 will not occur when product 3 is a subcomponent of product 8, because product 3 is not separable while working as a subcomponent in product 8. An individual demand for product 3 will occur only when it is failed, working as a subcomponent of product 2. On the other hand the fill rate at the depot for product 8 has decreased in model 2 compared to model 1. This is because of the fact that whenever product 3 is failed we have to replace product 8 instead of replacing product 3 which is a not removable from the parent product.

In model 1, the demand for product 5 occurs when product 5 is failed due to the failure of subcomponent 6 and when product 5 fails by itself. In the case of model 2, the demand for product 5 occurs only when product 5 is failed, because failure of subcomponent 6 will not cause product 5 to fail but makes product 5 a partially mission capable product. This result in a reduced demand for product 5 and hence increased the fill rate of product 5 in model 2 compared to model 1. The fill rate of product 1 and product 7 (end items) are also increased in model 2 compared to model 1 because in model 2, the failure of product 6 will only result in making the end items partially

mission capable. Hence the demands for the end items in model 2 are reduced, resulting in the increase of fill rate of the end items at the depot. Fill rates of product 6 are increased in model 2 at both the ships and the depot because when product 6 is failed, it is taken out from the end item for repair with out affecting the operational capability of the end items, which hold the product 6. This reduces the failure chance of product 6 thereby increasing the fill rates in model 2 compared to model 1.

4. Summary.

The goal of this research was to develop an object-oriented simulation framework that captures the fundamental elements for modeling generic spare parts supply chain networks. The modeling framework was built on an agent plug-in control mechanism which allows great flexibility for extending and incorporating additional behaviors into models. This article has described in detail the object-oriented architecture of the framework and the underlying modeling assumptions. The framework also provides a well defined database representation for modeling instances.

We illustrated the use of the framework by developing and testing models for multi-echelon, multi-indenture supply chain systems based on the classic problem presented in reference [2]. The modeling of such systems is easily accomplished by changing the database and the results presented clearly demonstrate the potential use for the framework. In the future, we plan to incorporate part cannibalization and provide object-oriented primitives for modeling the transportation components of these types of systems. In addition, agents for more complex scheduling of end-item operation, repair part dispatching, and transportation planning are planned. This framework will be used

in the development and testing of MIME spare part inventory models and algorithms. In addition, the framework can be used to estimate performance of supply chain configurations. Finally, the models developed from this framework can be embedded in optimization models for planning the strategic, operational, and tactical execution of large-scale spare part supply chain networks.

References

1. A. V. Sleptchenko, *Integral inventory control in spare parts networks with capacity restrictions* doctoral diss., Universiteit Twente (The Netherlands), 2002.

2. C. Sherbrooke, *Optimal inventory modeling of systems: multi-echelon techniques* (New York: Wiley, 1992).

3. I. R. Hester, *Analysis of the effect of centralizing management of mobility readiness spares package assets*, Thesis, Air Force Institute of Technology, Ohio, 2001.

4. G. C. Schaefer, J.D. Haas, A simulation model to investigate the impact of health and usage monitoring systems (HUMS) in helicopter operations and maintenance, *Proceedings of America Helicopter Society 58th Annual Forum*, Montreal, Canada, 2002.

5. M. Cohen, P. V. Kamesam, P. Kleindorfer, H. Lee, & A. Tekrian, Optimizer: IBM's Multi-Echelon Inventory System for Managing Service Logistics, *Interfaces 20 (1)*, 1990, 65-82.

6. A. M. Law, & W.D. Kelton, *Simulation modeling and analysis*, (Boston: McGraw-Hill, 2000).

7. M. Dong, *Process modeling, performance analysis and configuration simulation in integrated supply chain network design*, doctoral diss., Virginia Polytechnic Institute and State University, 2001.

8. R. G. Ingalls, The value of simulation in modeling supply chains, *Proc. of Winter Simulation Conference*, eds. D. J. Medetros, E. F. Watson, J. S. Carson, and M. S. Manivannan, Washington, DC, 1998, 1371-1375. Institute of Electrical and Electronic Engineers, Piscataway. New Jersey.

9. M. D. Rossetti & H. T. Chan, A Prototype Object-Oriented Supply Chain Simulation Framework, *Proc. of Winter Simulation Conference*, eds. S. Chick, P. J. Sanchez, D.

Ferrin, and D. J. Morrice, New Orleans, Louisiana, 2003, 1612-1620. Institute of Electrical and Electronic Engineers, Piscataway. New Jersey.

10. J. M. Swaminathan, S.F. Smith, & N.M. Sadeh, Modeling supply chain dynamics: A multiagent approach, *Decision Sciences*, *29(3)*, 1998, 607.

11. W. J. Hopp, & M. L. Spearman, *Factory physics: foundations of manufacturing management* (Boston: Irwin/McGraw-Hill, 2000).

12. L. Silverston, *The data model resource book* (New York: Wiley, 2001).

13. J. Jiao, M.M. Tseng, Q. Ma, & Y. Zou, Generic bill-of-materials-and-operations for high-variety production management, *Concurrent Engineering Research and Applications 8(4)*, 2000, 297-321.

14. M. D. Rossetti, B. Aylor, R. Jacoby, A. Prorock, and A.White, Simfone′: An object-oriented simulation framework, *Proc. of the Winter Simulation Conference,* eds. J. Joines, R. Barton, P. Fishwick, and K. Kang, Orlando, Florida, 2000, 1855-1864. Institute of Electrical and Electronic Engineers, Piscataway. New Jersey.

15. E. Gamma, R. Helm, R. Johnson, & J. Vlissides., *Design patterns: elements of reusable object-oriented software* (Reading, Mass: Addison-Wesley, 1995).

Biographies

Manuel D. Rossetti, Ph. D., P. E. is an Associate Professor in the Industrial Engineering Department at the University of Arkansas. He received his Ph.D. in Industrial and Systems Engineering from The Ohio State University. Dr. Rossetti has published over thirty journal and conference articles in the areas of transportation, manufacturing, health care and simulation and he has obtained over $1.5 million dollars in extra-mural research funding. His research interests include the design, analysis, and optimization of manufacturing, health care, and transportation systems using stochastic modeling, computer simulation, and artificial intelligence techniques. He serves as an Associate Editor for the International Journal of Modeling and Simulation and is active in IIE, INFORMS, and ASEE.

Soncy Thomas, M.S.I.E. is an Associate Consultant in Infosys Technologies Ltd. He received his M.S. in Industrial Engineering from the University of Arkansas and B.Tech. in Mechanical Engineering from the Kerala University, India. His areas of interest include computer simulation, and the design, analysis and implementation of supply chain systems.