DEVELOPMENT OF GENETIC ALGORITHM-BASED STOCHASTIC MODEL TO STUDY AND OPTIMIZE SINGLE-ECHELON VS MULTI-ECHELON INVENTORY SYSTEMS

A Thesis

Presented to

the Faculty of the College of Science and Technology

Morehead State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

---

by

Nadeera Ekanayake

December 2013

Accepted by the faculty of the College of Science and Technology, Morehead State University, in partial fulfillment of the requirements for the Master of Science degree.

Dr. Nilesh Joshi
Director of Thesis

Master's Committee: Dr. Ahmad Zargari, Chair

Dr. Nilesh Joshi

Dr. Yuqiu You

11/18/2013
Date

DEVELOPMENT OF GENETIC ALGORITHM-BASED STOCHASTIC MODEL TO STUDY AND OPTIMIZE SINGLE-ECHELON VS MULTI-ECHELON INVENTORY SYSTEMS

Nadeera Ekanayake, M.S.
Morehead State University, 2013

Director of Thesis: Dr. Nilesh Joshi

In today's global market, managing the entire supply chain efficiently becomes a crucial factor for a successful business. Performance efficiency of a supply chain network depends on how the inventories are managed across the entire network. Inventory management in a supply chain network is a complex problem due to the nature of interdependencies among different nodes of the network, and can rarely be solved using closed-form mathematical solutions. These problems can be broadly classified in to two categories: single-echelon and multi-echelon. In single-echelon inventory control problems, the focus is on determining the appropriate level of inventory for an individual unit within the supply chain network. On the contrary, multi-echelon inventory optimization takes a holistic approach by focusing on the correct levels of inventory across the entire supply chain network. The goal of this research is to use stochastic modeling approach to develop a scalable multi-tier supply chain model that can accommodate multiple inventory items, and to experiment with the model to study and compare its behavior under single-echelon vs. multi-echelon inventory systems. A genetic algorithm based multi-objective optimization method is used to optimize model's behavior with two conflicting objectives: minimizing average inventory across the end to end supply chain and maximizing overall fill rate or service level. The results show that the solutions generated using multi-echelon optimization can be quite different than the solutions generated using single-echelon optimization. Under single echelon settings, network behaves as a decentralized system and as a result, entire supply chain network suffer with higher inventory levels and lower fills rates. In contrast, multi echelon network behaves as a centralized system and provides lower inventory levels while maintaining higher fill rates for the entire supply chain network. This makes sense since the former takes a far-sighted systems level view of the problem as against the short-sighted individual unit level approach taken by the latter. However, distribution centers failed to provide optimal values when performing under multi echelon configuration. In the best interest of the system as a whole, distribution centers have to compromise on their individual performance.

ACKNOWLEDGEMENTS

First and foremost I offer my sincere gratitude to my supervisor, Dr. Nilesh Joshi, for giving me the opportunity to work on such a challenging research and supporting me throughout my thesis with his patience and knowledge. He has taught me the methodology to carry out the research and to present the research work as clearly as possible. It was a great privilege and honor to work and study under his guidance.

Secondly, I would like to express my gratitude for Dr. Ahmad Zargari, chair of department of Applied Engineering &Technology and my graduate program advisor. I will forever be thankful to him for believing in me and encouraging me throughout my graduate career at Morehead State University.

I 'am grateful to Dr. Yuqiu You for her comments and suggestions which helped me to improve my research.

I would also like to thank my parents, my brother and sister in law. They were always supporting me and encouraging me with their kind wishes.

Finally, I would like to thank my husband, Lakmal Molligoda. If not for him, I would not have contemplated this path. He was always there to cheer me up and stood by me through the good and bad times.

Table of Contents

Chapter 1: INTRODUCTION

1.1 Overview

In today’s business environment, most of the companies are seeking ways of maximizing profits by reducing costs and improving efficiency. Supply chain management (SCM) is considered as the best solution for this because of the impact that SCM has on business is significant and exponential. Supply chain management is the systematic and strategic coordination management for supplying goods and products required by the end customer. Key objective of an efficient supply chain is to get “the right quantity at the right time and in the right place”. A supply chain is a network of nodes (Stores, Distribution Centers, Suppliers, etc.) and each of these nodes carries out different practices to provide goods and services to end customer. SCM directly helps to boost customer service by making sure the right product and quantity are delivered in a timely fashion. SCM also has a huge impact on decreasing inventory holding costs which helps to maximize profits. Thus, it is evident that inventory serves a useful purpose in supply chain and inventory management plays a key role in supply chain management. An inventory control policy determines how the company moves its inventory throughout the supply chain. Efficient inventory management helps to minimize the need for excess inventory in the system by carefully managing the factors that drive inventory levels up. Supply chain inventory systems can be classified in to two main categories: single-echelon and multi-echelon inventory systems. In single-echelon inventory systems, each node is independent from one another and responsible for their own inventory policies. On the other hand, multi-echelon systems focus on the inventory levels of all the nodes across the network. All of the inventory parameters are determined simultaneously, taking into account the interrelationships among all the connected nodes in the network. Therefore, multi-echelon systems are far more complex as compared to single-echelon inventory systems.

This research is aimed to look at different available inventory control policies and develop a genetic algorithm-based stochastic modeling method to compare and analyze single echelon vs. multi-echelon inventory systems for supply chains with multiple inventory items.

1.2 Problem Statement

Inventory management plays a key role in supply chain management. The basic purpose of inventory management is to specify "**when items should be ordered"** and "**how large the order should be"**. Selecting the most economical method or policy for inventory management is a challenge to any organization, because of the variety of policies available in the industry. Inventory policies vary in two aspects,

- Procedure used to trigger generate orders : "Time to order"
- Decision rule that determine the order size: " Quantity to order"

Each and every node in a supply chain network has their own individual inventory policy defined and these policies can be varying from one node to another based on each nodes operations and organization policies. Supply chain networks can be categorized as Single-echelon or as Multi-echelon supply chain systems based on how they optimize their inventory across several stages of the network. Single-echelon supply chain system follows a decentralized approach, where each node is responsible for its own inventory policy and ignores the fact that excess inventory or lower fill rates in the system are produced by interrelationships between nodes. In contrast, Multi-echelon supply chain system follows a centralized approach where inventory optimization is treated holistically across entire supply chain network. Implementing Multi-echelon system is more complex compared to Single-echelon system because it takes into account multiple factors impacting demand and supply variability across the entire supply chain network. The objective of this research is to model, simulate and optimize the inventory levels of a supply chain consisting of three levels (Store, Distribution Center and Supplier) with multiple inventory items using a single echelon control system and then a multi-echelon control system, utilizing a system dynamic approach with the Siemens Plant Simulation software.

1.3 Research Objectives

Objectives of this research are:

1. Research currently available single-echelon and multi-echelon inventory control policies.

2. Design a prototype supply chain network as a foundation for this study.
3. Formulate a multi-objective inventory optimization problem for the prototype network with focus on reducing average inventory across the network and improving the fill rate.
4. Model the network and associated multi objective inventory optimization problem using Plant Simulation software.
5. Design and simulate various experimental scenarios to compare single echelon vs. multi-echelon model.
6. Analyze results of experimentation.

1.4 Definition of Terms

(*R, S*) Policy - a periodic review policy in which the inventory is observed in intervals (R). S represents the order level.

(*s, Q*) Policy - also called the reorder point (s), order quantity (Q) system. This model is under continuous review so a new order can be made when the inventory goes below the specified reorder point.

(*s, S*) Policy - the reorder point (s), order level (S) policy.

Continuous System - a system that has state variables that change constantly over time.

Dynamic Simulation Model - models changing over time rather than just a specific point in time.

Economic Ordering Quantity (EOQ) - Ordering quantity method that reduces the balance of cost between reorder costs and inventory holding costs.

Fill Rate – the rate at which customer orders are able to be filled from the current inventory. A high fill rate correlates to the high customer satisfaction/expectation.

Inventory - records and supply levels of stock.

Logistics - the movement and flow of material, money, and information between suppliers and customers.

Model - a depiction of a system so it can be studied and evaluated.

Multi-Echelon Supply Chain - A supply chain with entities located on multiple tiers.

Poisson distribution - a discrete probability distribution that communicates the likelihood of a specified number of events occurring in a fixed amount of time (Banks, Carson II, Nelson, & Nicol, 2001).

Simulation system - a group of objects that are connected in interaction to accomplish a purpose.

Stochastic Simulation Model - a model that has one or more random variable inputs which will provide random outputs.

Stock - physical goods and materials that contain economic value to an organization. Stock is held in various forms while awaiting processing, packaging, transport, or use at a later time.

Supply Chain - a system where materials flow from the source to the end customer while information is being transported in both directions through suppliers, manufacturers, distributors, retailers, and customers.

System – a collection of entities (such as people or machines) that interact with one another to try to achieve a logical goal.

SKU - Stock keeping units

Triangular Distribution - a continuous probability distribution defined by a minimum value, a maximum value, and a most likely value, or the mode.

1.5 Organization of Thesis

The rest of the thesis is organized as follows. Chapter 2 provides a literature review on supply chain management, inventory systems, different inventory policies, single echelon and multi-echelon systems, genetic algorithms, and various simulation software packages currently available in the market. Recent research efforts in single echelon and multi-echelon systems are also discussed. Chapter 3 provides detail description on methodology developed in the research. Chapter 4 provides the simulation results and analysis. Finally, in Chapter 5, conclusions drawn from the research are discussed.

Chapter 2: REVIEW OF LITERATURE

2.1 Overview

This chapter explores literature in five different areas: Supply chain management, Inventory systems, Different Inventory policies, Single echelon and Multi echelon comparison, Genetic Algorithm, Simulation software's and recent research efforts in single echelon and multi echelon supply chain systems.

2.2 Supply Chain Management

A supply chain is an integrated process which includes all activities associated with the flow of transformation of goods from a supplier through to the end user. There are many different ways of describing Supply chain. However the basic idea behind each of these different definitions is the same. The Supply Chain Council (1997) uses the definition: "The supply chain - a term increasingly used by logistics professionals - encompasses every effort involved in producing and delivering a final product, from the supplier's supplier to the customer's customer. Four basic processes - plan, source, make, deliver - broadly define these efforts, which include managing supply and demand, sourcing raw materials and parts, manufacturing and assembly, warehousing and inventory tracking, order entry and order management, distribution across all channels, and delivery to the customer". Stevens (1989) defines a supply chain as "A connected series of activities which is concerned with planning, coordinating and controlling materials, parts, and finished goods from supplier to customer. It is concerned with two distinct flows (material and information) through the organization". Quinn (1997) defines the supply chain as "all of those activities associated with moving goods from the raw-materials stage through to the end user. This includes sourcing and procurement, production scheduling, order processing, inventory management, transportation, warehousing, and customer service. Importantly, it also embodies the information systems so necessary to monitor all of those activities". Interest in supply chain management has gradually increased since the 1980s when companies saw the benefits of collaborative relationships when each node works together rather than performing individually (Oliver & Webber, 1992). Primary purpose of supply chain management is to improve performance and maximize profit by integrating all the entries to a single unit.

Figure 2.1 provides an example of the flow of a typical retailer supplier chain network. System starts with a supplier who provides the products to Distribution Centers and the products flow through City Hubs, Retail Stores to end Customer.

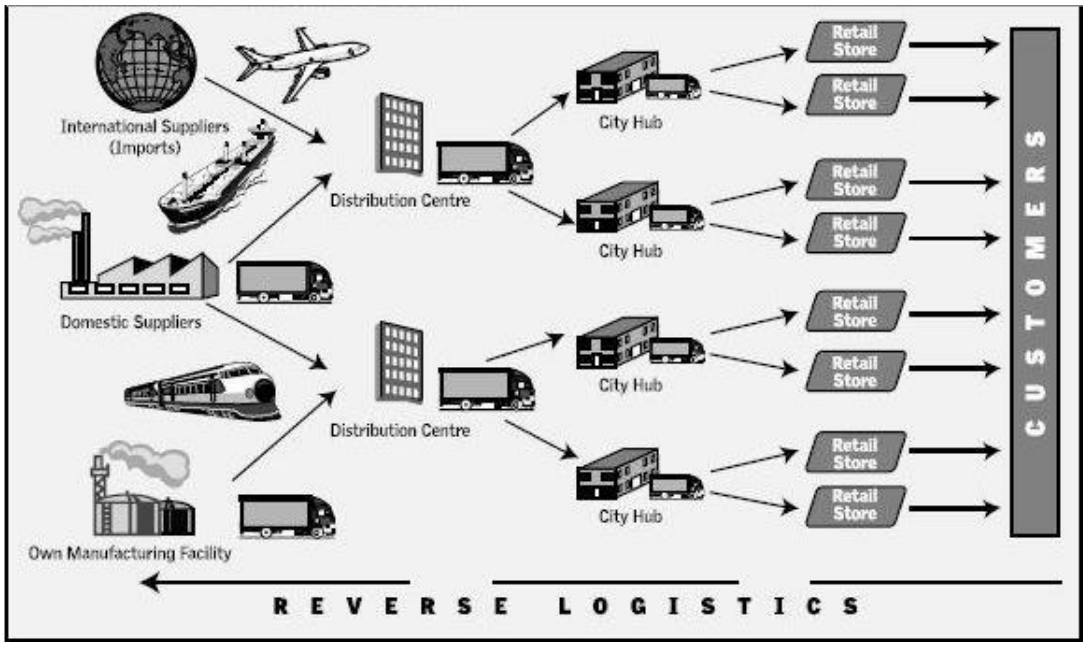


Figure 2.1: Retailer supply chain flow (Agarwal, 2007)

2.3 Inventory Management Systems

Inventory management is a key element in Supply Chain Management. In a supply chain, in addition to managing relationships among different entities such as suppliers, distribution centers, retailers, and customers, it is also important to maintain the inventory flow between the nodes. Demand and supply relationship between each node in the network, controls the overall product flow of the network. Cost of inventory for each node is a key factor in determining whether that specific node generates profit or not. Without proper inventory management, there would not be an efficient supply chain process.

Implementing efficient inventory management policies within the organization has always been a challenging task for the supply chain managers. In general, inventory management is considered as organizing and balancing the inventory levels in a company (IBM, 2012). Having large amounts of inventory or having less inventory would affect the responsiveness of a supply chain. For example, when a distribution center has low inventory on hand, its fill rate gets lower due to the inventory unavailability and it affects the stores available inventory situation which eventually creates dissatisfaction among customers. Thus one can argue that a company should keep more than enough stock on hand which will result in 100% fill rate every time. Even though it seems like a viable solution, keeping more on hand inventory results in increasing costs and decreasing profits. The carrying cost of inventory has four main components: capital cost, storage cost, inventory maintenance cost, and inventory risk cost. Capital cost is the cost spent on purchasing the inventory items. Storage cost involves with storage expenses; rent or mortgage, moving cost, utility cost, etc. Inventory maintenance cost involves with insurance and taxes paid on the inventory. And finally, inventory risk cost involves with risk associated with holding inventory such as product expiration and damaged products. The longer the inventory remains untouched, the more it will cost in upkeep (Adeyemi & Salami, 2010).

Therefore, it's obvious that whenever the inventory level of an entity is poorly managed the whole supplier chain would suffer. The primary purpose of Inventory control is to find out "When to order" and "How much to order". Optimizing inventories involve integrating inventory management with supply chain network design (Chopra & Meindl, 2003). Currently there are many different inventory control methods available in the industry and determining the best method is always a challenge to any organization. Some of these methods include ABC analysis, economic ordering quantity (EOQ), materials requirement planning (MRP), enterprise resource planning (ERP), etc. ABC analysis is based on dividing items in to three categories, A, B and C, where category “A” consists of the most valuable items, and category “C” consists of the least valuable ones. This method aims to draw decision maker’s attention on the critical few (A-items) and not on the trivial many (C-items) (Inventory Management, 2012). On the other hand, the basic Economic Order Quantity (EOQ) model seeks to find the balance between ordering cost and carrying cost by determining the most economic quantity to obtain by the distributor (Onawumi, Oluleye, & Adebiyi, 2011).

2.4 Inventory Models/Policies and their comparison

Inventory management establishes the optimal inventory levels that must be maintained to meet expected service levels for demand fulfillment. Inventory systems are divided into two main models: single-period inventory models and multi-period inventory models. A single-period inventory model is based on a one-time purchasing decisions and assuming that the product will not be reordered. In contrast multi-period inventory models are based around an item that is planned to be purchased periodically and the item inventory levels are closely observed. Reordering decision can be driven by an event trigger or by a time trigger (Jacobs & Chase, 2011). This research is focused on multi-period inventory systems.

Many industrial companies uses multi-period inventory model. There are two types of multi-period inventory models: Fixed-Order Quantity Model and Fixed-Time Period Model. Fixed-Order Quantity Model is also defined as Q models. Fixed-Order Quantity model is "Event triggered". For an example an event is triggered when running out of stock. Fixed-Time Period Model is also defined as periodic review system, fixed-order interval system, and also as the P-model (Jacobs & Chase, 2013). Fixed-Time Period Model is "Time triggered". Orders are replaced on predetermined time intervals.

Table 2.1 provides a basic comparison between these two models. Q represents constant order quantity, q represents variable order quantity, R represents the reorder level/point and T and represents the periodic review interval.

Table 2.1: Fixed–Order Quantity and Fixed–Time Period Differences (adopted from (Jacobs & Chase, 2013)**)**

Feature	Fixed Order Quantity Model	Fixed- Time Period Model
Order quantity	Order quantity is same for each time(Q-constant)	Order quantity varies each time order is placed(q-variable)
When to place order	When inventory position drops to the reorder level R.	When the review period T arrives.
On hand inventory levels	Each time a withdrawal or addition is made	Reviewed only at review period

Fixed–Order Quantity system focuses on order quantities and reorder points. Each time a unit is taken out of inventory, the withdrawal is updated and the on hand inventory is immediately compared to the reorder point. If it is less than or equal to the reorder point, an order for Q items is placed. If it has not, then the system will not trigger an event and remains in idle state until the next demand occurs. In the Fixed–Time Period system, for each review period, on hand inventory is compared with reorder point level and the decision to place an order is made.

2.4.1. Economic Order Quantity (EOQ) model:

The EOQ model is defined as a continuous replacement method, which means that inventory is compared with the reordering level to decide whether to order or not. EOQ model is one of the oldest and most recognized inventory control technique (Slack, Chambers, & Johnston, 2006).

According to Slack, Chambers & Johnston (2006) EOQ model is based on following main assumptions:

1. Demand is known and constant
2. Lead time is known and constant
3. Receipt of inventory is instantaneous
4. Quantity discounts are not available
5. Variable costs are limited to: ordering cost and holding cost
6. If orders are placed at the right time, stock outs can be avoided

The Total Inventory Cost for a basic EOQ model is defined as:

Total annual cost = annual purchase cost + annual ordering cost + annual holding cost

$$TC = DC + \left({}^{D}/_{Q}\right) * S + \left({}^{Q}/_{2}\right) * H$$

where:

TC = Total annual cost,

D = Annual demand,

C = Cost per unit,

Q = Quantity to be ordered,

S = Setup cost or cost of placing an order,

H = Annual holding and storage cost per unit of average inventory.

Knowing when to reorder materials is a key to minimizing unnecessary inventory holding costs. Three similar inventory policies are the (s, Q) policy, the (R, S) policy, and the (s, S) policy. R represents the review interval or the order cycle, Q represents the order quantity, S represents the order level and s represents the re-order level.

(s, Q) Inventory Policy

(s, Q) inventory policy is also defined as reorder point (s), order quantity (Q) system. This model is continuously reviewed and when the inventory level drops to reorder point s, an order is placed for a lot size of Q. The order arrives to refill the inventory after a lead time, L (Jensen & Bard, 2002).Figure 2.3 represents the inventory pattern of a (s,Q) system.

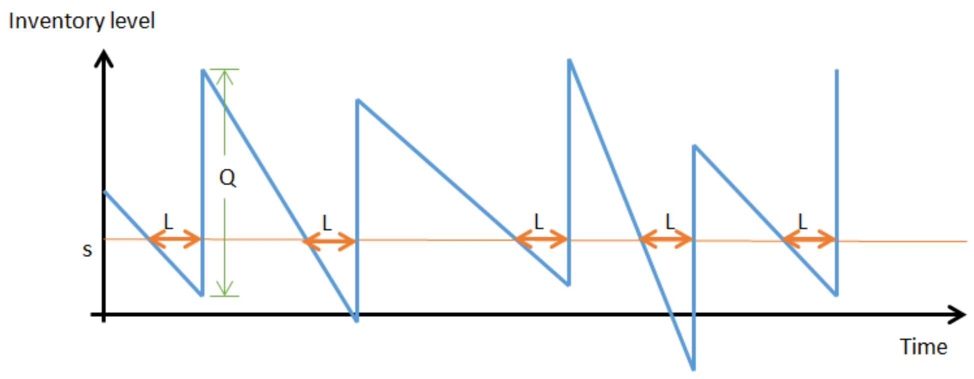


Figure 2.3: Inventory pattern for reorder point (s), order quantity (Q) system.

Adopted from: (Jensen & Bard, 2002)

(R, S) Inventory Policy

The (R, S) policy is a periodic review policy where the inventory level is observed in pre-defined time intervals of length R. S represents the reorder level. If the inventory level is y upon the reviewing time, then an order of (S-y) lot size will be placed. The order arrives to refill the inventory after a lead time of L. Figure 2.4 represents this inventory policy model, in which the dotted lines are representing the inventory position while the solid lines are representing the inventory level (Jensen & Bard, 2002).

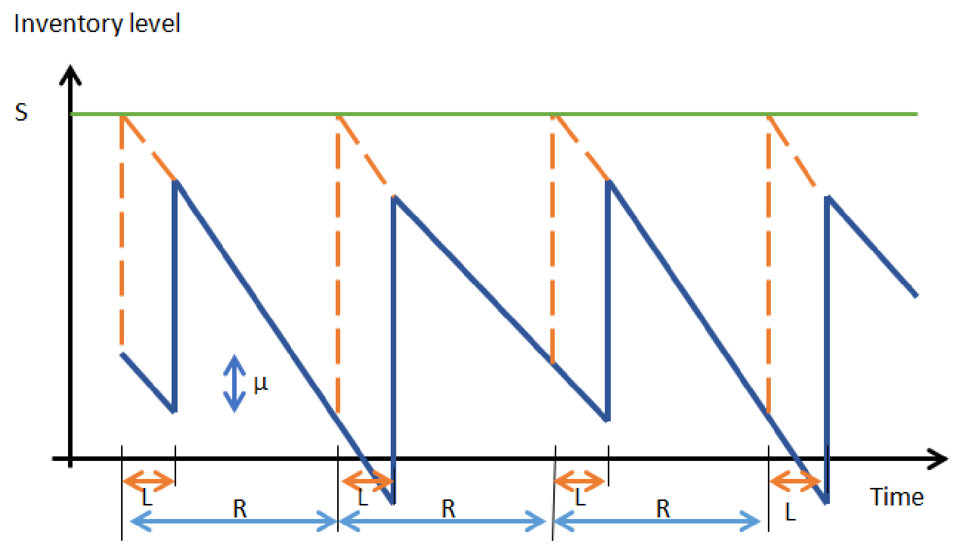


Figure 2.4: (R, S) Inventory Policy. Adopted from: (Jensen & Bard, 2002)

(s, S) Inventory Policy

In (s, S) inventory policy, demand is considered to have variability. Order up to level or upper stock id defined as "S" and reorder level or safety stock level is defined as "s". With (s, S) inventory policy, inventory level is reviewed with pre-defined time intervals of R and new orders aren't taken until the current inventories fall to or below (s). The orders that are placed will not go beyond the defined order up to level S. Figure 2.5 is an example of what an inventory could look like following the (s, S) policy.

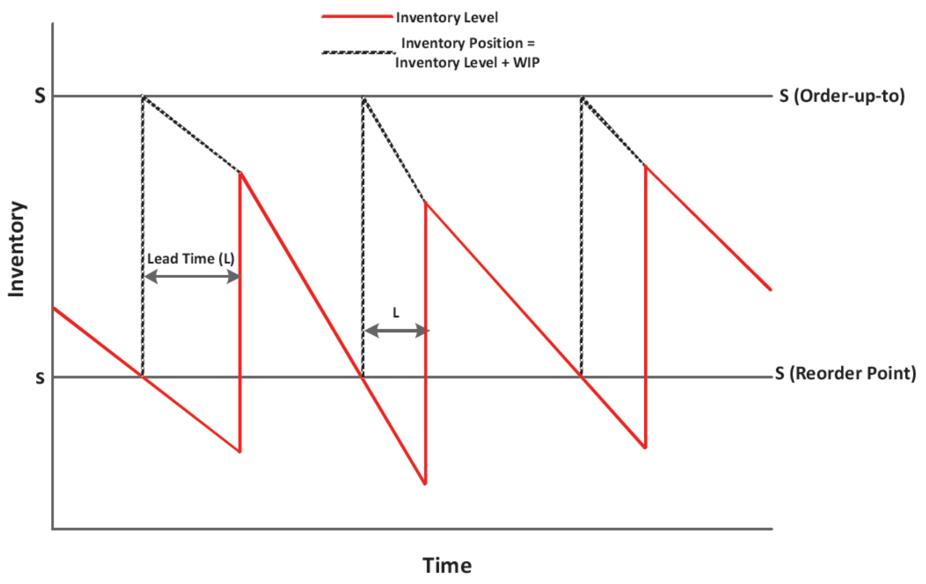


Figure 2.5: (s, S) Inventory Policy. Source: (Joines & Roberts, 2012)

2.5 Single echelon supply chain vs. Multi echelon supply chain

Supply chain systems can be divided into two main categories based on the inventory optimization method that is followed in the network: Single echelon and Multi echelon. In a single echelon system each node is independent from one another and responsible for their own stocking policies. Once all the nodes determine their policies, their combined operations will create a demand process for orders placed at the connected node. Single echelon supply chain systems can be defined as a decentralized system where each node acts individually from one another. When each node tries to optimize its own inventory levels without considering the impact of those decisions on other levels of the supply chain, the interrelationships among nodes are ignored. Such a micro view of the problem results in the overall supply chain network having high inventory levels and low fill rates (Hausman & Erkip, 1994). In contrast in a multi echelon inventory system, all the inventory parameters are determined simultaneously, taking into account the interrelationship between each connected nodes. The macro-level system performance objective is optimized by the application of the multi echelon inventory policy. Therefore multi echelon system can be defined as a centralized system where each node is dependent on one another and each of the inventory parameters in a node are related to connected nodes and vice versa (Lee, 2003). With this approach supply chain is considered as a whole and tries to identify what is appropriate for the entire network, not for each specific node individually. Followings are some of the main questions that need to be considered when developing a multi echelon system (Multi Echelon Inventory Optimizer, 2013).

- How much inventory should be kept and in which node?
- How to distribute the inventory so the overall investment is minimized with higher service level?

This research is focused on developing a supply chain system model to compare single echelon and multi echelon inventory systems.

2.6 Supply chain Simulation

2.6.1 Methods of evaluating supply chain performance

A fundamental problem when implementing a supply chain is evaluating the performance. Followings are some of the methods that can be used to evaluate performance of a supply chain (Law & Kelton, 2000):

- Analytical method – Complex mathematical models.
- Physical experiments (Example: Industrial pilot implementations)
- Simulation methods

Simulation methods have many advantages over analytical methods and physical experiments. Analytical methods use complex mathematical models and are more suitable for simple problems. It can be difficult, if not impossible, to develop analytical models for complex problems. On the other hand, physical experiments such as a pilot run on the actual system can be cost and time-intensive and not possible in every case (Beamon, 1998).

In contrast to that, simulation approach provides the opportunity to explore complex and large scale supply chain problems without much technical difficulties and in a very low budget. Simulations can also be used in forecasting (Imagine That Inc., 2012). The ability to use “What-If” analysis in simulation approach helps to build the “best” supply chain model configuration which also strengthens the reasons of choosing simulation models over other evaluation methods (Bhaskaran, 1998).

2.6.2 Modeling tools and their comparison

Currently there are several discrete event simulation software packages available that can be used in different domains (Cimino, Longo, & Mirabelli, 2010). Most of these tools use common techniques, rules and logics. However, each of these tools has limitations when there is a need to implement a more complex and real world scenario like supply chain modeling. This section will look into different available discrete event simulation software tools.

Anylogic

Anylogic is a Java based simulation application and it is mainly used for processes analysis and optimization, forecasting and strategic planning and processes visualization. Anylogic supports Agent Based, Discrete Event and System Dynamics modeling and simulation (Cimino, Longo, & Mirabelli, 2010). It supports both flow charts and graphical modeling. Anylgoic is widely used for supply chain modeling, logistics, manufacturing, health care and modeling business process.

Arena

Arena is provided by Rockwell Automation and it uses SIMAN language as the simulation language. Arena provides the users animation at run time and it allows to import CAD drawings to enhance animation capabilities (Cimino, Longo, & Mirabelli, 2010).Arena is considered as the most widely used simulation software in the industry (ACTOperationsResearch, 2013).

Automod

Automod is developed by Applied Materials Inc. It is based on the domain-specific simulation language Automod. It has additional features like Auto View devoted to support simulation animation with AVI formats (Cimino, Longo, & Mirabelli, 2010).

Simio

Simio is designed by Dennis Pegden, the co-creator of Arena Simulation software and of SIMAN simulation language. Simio is mostly known for its modeling speed and 3D functionality (Pegden, 2009). With Simio, advanced analytics, airports, manufacturing, supply chain, healthcare, military, mining, ports, lean six sigma can be easily modeled (ACTOperationsResearch, 2013).

Technomatix Plant simulation

Technomatix Plant Simulation is owned by Siemens Inc. It is a discrete, event-controlled simulation program which only inspects those points in time, at which events take place within the simulation model. Technomatix Plant Simulation allows users to create well-defined, hierarchical models of production facilities, lines and processes (Cimino, Longo, & Mirabelli, 2010). It is built on an object-oriented architecture with hierarchy and inheritance with multiple interface support. Technomatix Plant Simulation software provides extensive analysis tools, such as bottleneck analysis, statistics and charts which can help a user to evaluate different manufacturing scenarios (Siemens, 2012). Experiments Design functionalities are also provided to conduct design of experiments. Simulation optimization is carried out by using Genetic Algorithms and Artificial Neural Networks (Cimino, Longo, & Mirabelli, 2010).

For this research, Technomatix Plant simulation software is used to simulate the multi echelon supply chain model.

2.7 Genetic algorithm for optimization

Simulation is often used to improve a system and the ultimate goal of a simulation is to find an optimal solution for the entire system. Developing an optimal or near-optimal design configuration which achieves the goal of simulation at a minimum cost is an important factor for system designers. Genetic Algorithms (GAs) have proven to be a powerful tool for solving optimization problems. In 1960, John Holland invented Genetic Algorithms and it was further modified by Holland and his students and colleagues at the University of Michigan during 1960s and the 1970s (Melanie, 1999). GAs are probabilistic algorithms based on biological evaluation process (Konagaya, 1992). Although the algorithm does not guarantee to get the optimal solution for a problem, it can find nearly optimal or useful solutions for the problem. Because of their broad applicability, ease of use, and global perspective, GAs has been increasingly applied to various search and optimization problems in the recent past. Some of the recent research in GA area include: GA based airlines booking terminal open/close decision systems (George, Rajakumar, & Binu, 2012) and GA optimization technique in beam steering of circular array antennas (Adebola, Abd-Alhameed, & Abousitta, 2012).

GAs uses the process of natural evolution to generate the optimal solution. GAs are based on the following foundations (Goldberg & Holland, 1988):

- Population initialization process
- Selection process
- Crossover operation
- Mutation operation and
- Replace population with offspring

This research is based on finding the best parameters for (s, S) which provides the minimum value for average inventory while maintaining a higher service level for the entire system. Genetic algorithm is used as a tool to generate best set of experiment scenarios for this study.

2.8 Recent research in Single-echelon and Multi-echelon areas

There are several research efforts in the recent past, focused on single-echelon and multi-echelon inventory optimization problems. Caballini and Revetria (2008) developed a System Dynamics simulation for a non multi echelon supply chain including the retailing, wholesaling, distribution, and production processes, with the main goal of searching for inventory policies that yield reduced costs and/or increased revenues. In their research they analyzed the effect of increased delays on the behavior of the system. However, this research was only focused on a non multi echelon supply chain and on one single inventory item.

Takahashi (2011) implemented a two-echelon dual-channel supply chain model with setup of production and delivery and developed a new inventory control policy for the supply chain. However, this research was limited to two echelon dual channel model.

Nyberg, Grossmann and Westerlund (2012) explored a problem of determining an efficient reformulation of the multi echelon stochastic inventory system with uncertain demands. According to their research "It is shown that by reformulating the three-stage multi echelon inventory system with specific exact linearization, larger problems can be solved directly with mixed-integer linear programming (MILP) without decomposition. The new formulation is significantly smaller in the number of continuous variables and constraints. An MILP underestimation of the problem can be solved as part of a sequential piecewise approximation scheme to solve the problem within a desired optimality gap" (Nyberg, Grossmann, & Westerlund, 2012). However, their research is based on complex mathematical formulation for a very specific problem, and the model flexibility is limited.

Smith (2012) focused on design and implementation for semiconductor supply chain systems with multi-echelon simulation with multi-echelon forecast biasing and optimization. The model was specifically built to analyze the supply chain in a semiconductor industry. Graphical User interfaces, which allows a designer to better visualize the model was not presented in Smith's research and it was considered as a limitation in the research (Smith, 2012).

Shu, Li and Huang (2012) conducted a research on how demand selection decisions are made for a multi-echelon inventory distribution system. In this research they have studied an integrated demand selection and multi-echelon inventory control problem that generalizes the classical deterministic single distribution center (DC) multi-retailer model by incorporating demand selection decisions. They have also explained some interesting managerial insights obtained from the numerical experiments from their research (Shu, Li, & Huang, 2012).This research was lacking from model flexibility perspective.

The literature review shows that a wide variety of models have been developed to address single-echelon inventory control problems, and multi-echelon inventory control problems for single inventory items, but the same cannot be said about multiple inventory items for single-echelon inventory control problems, and multi-echelon inventory control problems. Additionally, many of the models developed previously have limited flexibility. Thus, there are opportunities for further research in single-echelon control systems and multi-echelon control systems with multiple inventory items.

Chapter 3: METHODOLOGY

3.1 Methodology Flow Chart

This research focuses on studying and optimizing the behavior of single echelon vs. multi echelon inventory systems. Figure 3.1 outlines the major steps of our research methodology. First, a conceptualization and specification of the problem has been identified and a prototype of the supply chain network is designed. Next, the supply chain model is implemented using Siemens Plant simulation software. After a phase of testing of the model to prove its validity, various experimental scenarios were developed using Genetic Algorithms and have been simulated for Single echelon and Multi-echelon problems and results have been evaluated and presented in graphical format.

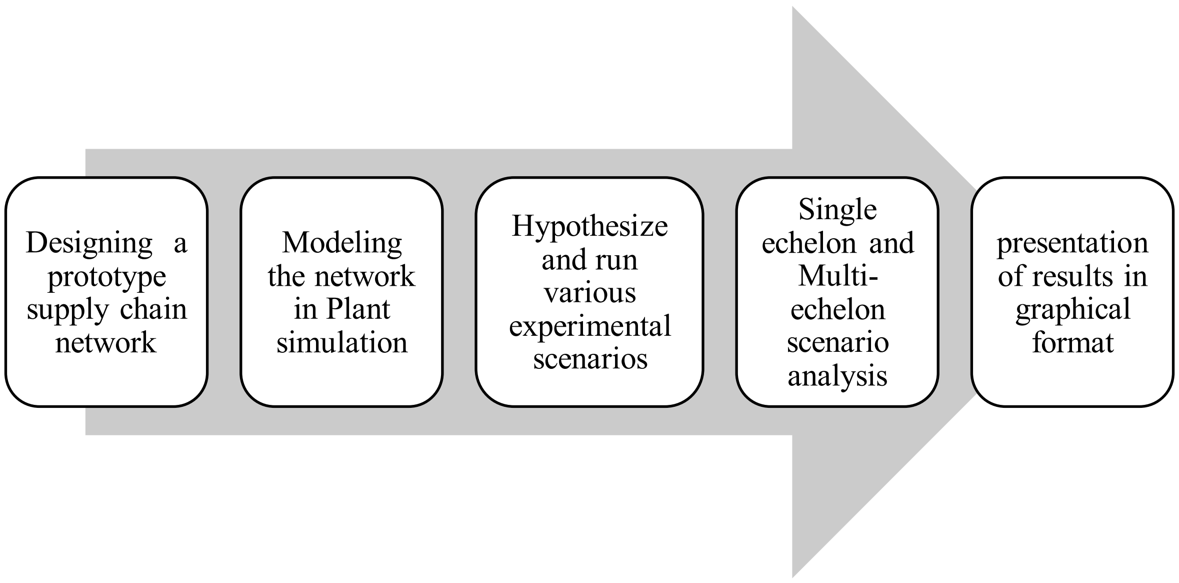

Designing a prototype supply chain network
Modeling the network in Plant simulation
Hypothesize and run various experimental scenarios
Single echelon and Multi-echelon scenario analysis
presentation of results in graphical format

Figure 3.1:Flow chart of Methodology

3.2 Model description

The purpose of this research is to develop an integrated methodology that allows supply chain decision makers to analyze the performance of their supply chain in a fast, shareable and easy to use format. The simulation model used in this research is developed using Siemens Plant simulation software.

3.2.1 Model Assumptions

Like any other simulation model, our multi-echelon supply chain model is based on certain assumptions. These assumptions are listed below:

- Inventory policies used in this research are based on (s,S) inventory policy.
- Every store and distribution center has common SKUs in their inventory databases.
- Individual stores can place orders only with a single predefined distribution center in the supply chain.
- Customer demand is generated using the Poisson distribution, and processing times at various nodes of the supply chain network are modeled using the triangular distribution.
    - Triangular distribution is a continuous probability distribution defined by a minimum value, a maximum value, and a most likely value, or the mode.

  - Poisson distribution is a discrete probability distribution that communicates the likelihood of a specified number of events occurring in a fixed amount of time.

- The run length of the simulation will be limited to 365 days. All experiments will be run for this length of time for consistency.
- The backordering in the model is not allowed. In case the inventory level reaches zero, the customers demand is not fulfilled.
- Distribution center demand will be 100% fulfilled by the suppliers and the suppliers will have all the resources to fulfill the demand.

3.2.2 Supply chain elements

The supply chain model includes suppliers, distribution centers and stores. In the developed model a single network node can be a store, a distribution center or a supplier. A supply chain network begins with suppliers and ends with one or more stores. Stores satisfy customer demands, distribution centers satisfies store demands and finally suppliers satisfies distribution centers demand. Figure 3.2 shows a very simple three tier supply chain where customers place the orders for items with the stores. The stores review their inventory levels on a periodic basis and place orders for items with the distribution centers as and when necessary. Distribution centers also review their inventory levels on a periodic basis and place orders for items with the supplier.

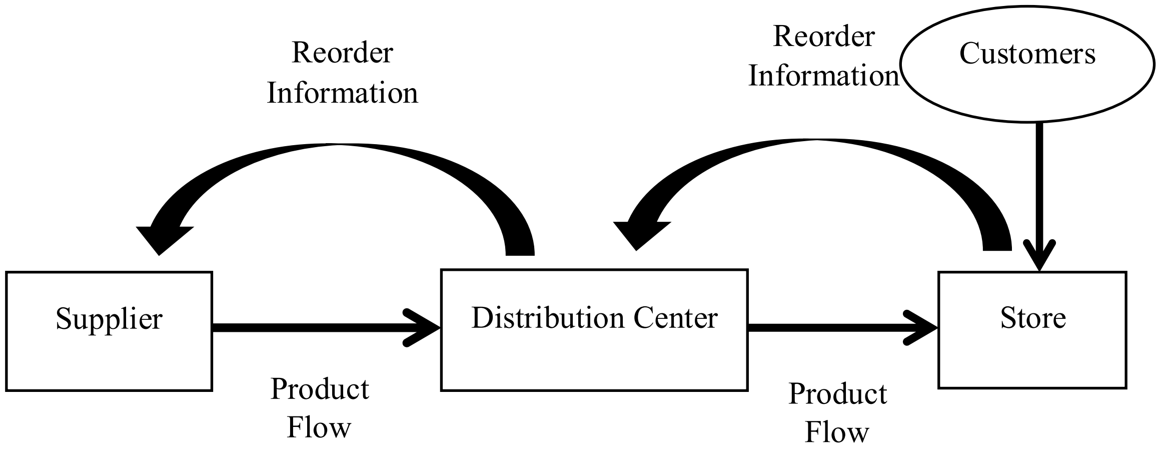


Figure 3.2: Simple Three Tier Supply Chain

Stores:

Stores are places that sell goods and merchandise to other businesses or individuals and are placed at the end of the supply chain. Basically, stores satisfy customer demands. Figure 3.4 displays the operations flow of store behavior in the designed model in this research. Store operations flow starts with the customer demand generation.

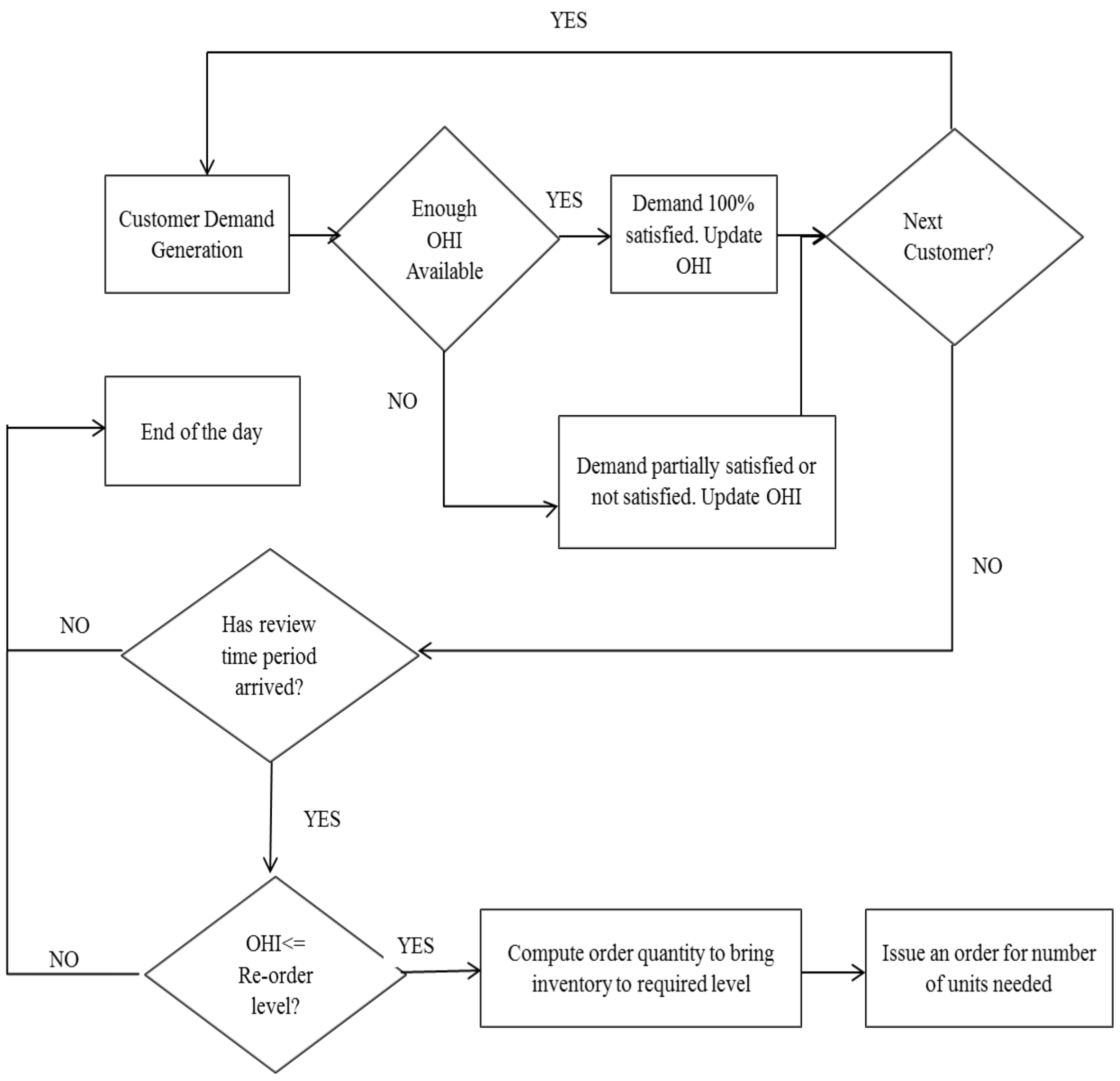

YES
Customer Demand Generation
Enough OHI Available
YES
Demand 100% satisfied. Update OHI
Next Customer?
End of the day
NO
Demand partially satisfied or not satisfied. Update OHI
NO
NO
Has review time period arrived?
YES
OHI<= Re-order level?
NO
YES
Compute order quantity to bring inventory to required level
Issue an order for number of units needed

Figure 3.4: Store operation flow chart

Demand per item arrives according to a Poisson process with parameter λ with mean equal to average daily demand.

$$D_i = Poisson(\lambda_i)$$

$$Where;\ \lambda_i = Average\ daily\ demand\ for\ item\ i$$

$$D_i = \textit{Demand for item } i$$

For each item, demand process is assumed to be an independent arrival process. Figure 3.3 is an example for a distribution plot for item i with average daily demand (λ_i) = 40.

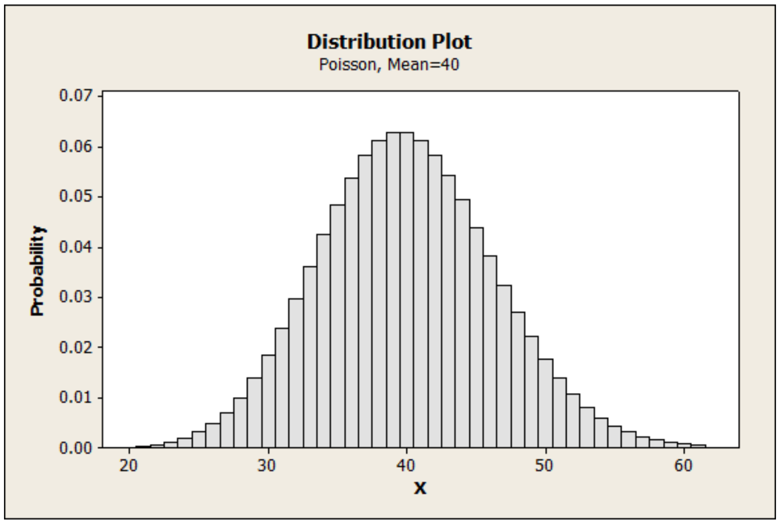


Figure 3.3: Distribution plot, Poisson with λ_i = 40

Once the demand quantity is generated for each item, the quantity required is compared with the stores individual On Hand Inventory (OHI) and the order will be eventually satisfied. We used a binary decision variable.

Decision:

$$D_i \leq OHI_{s,i}$$

$$\textit{Where; } D_i \quad = \textit{Demand for item } i$$

$$OHI_{s,i} \quad = \textit{On Hand Inventory for inventory item } i \textit{ in store } s$$

If true, then:

Demand is 100% satisfied.

If false, then:

Demand is partially satisfied or not satisfied.

Satisfied quantities will be recorded to calculate the fill rate. At the end of each day, based on the (s,S) inventory policy used, the stores decide whether to place an order to the distribution centers or not.

Decision:

$$OHI_{s,i} \leq s_{s,i}$$

Where; $s_{s,i}$ *= Reorder level for item i in store s*

If true, then:

$$Q_{s,i} = S_{s,i} - OHI_{s,i}$$

Where; Q *= Order quantity for item i in store s*

$S_{s,i}$ *= Order up to quantity for item i in store s*

Issue an order to distribution center for number of units needed ($Q_{s,i}$*).*

If false, then:

No orders will be placed.

Distribution Centers:

Distribution centers serve as a warehouse for a variety of products. They are strategically located according to company's needs. Figure 3.6 displays the operations flow of a distribution center in the designed model. Distribution Center operations flow starts with the store order arrival.

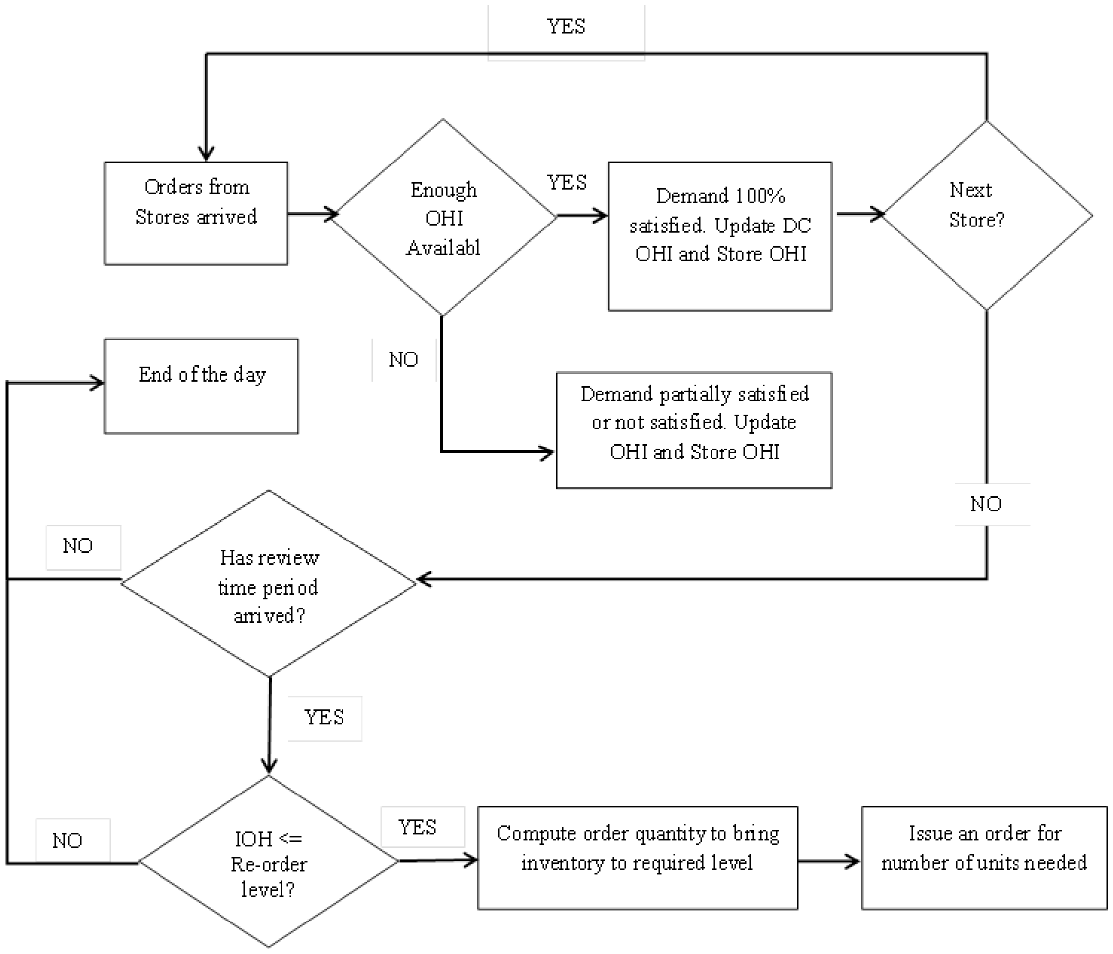

YES
Orders from Stores arrived
Enough OHI Availabl
YES
Demand 100% satisfied. Update DC OHI and Store OHI
Next Store?
End of the day
NO
Demand partially satisfied or not satisfied. Update OHI and Store OHI
NO
NO
Has review time period arrived?
YES
NO
IOH <= Re-order level?
YES
Compute order quantity to bring inventory to required level
Issue an order for number of units needed

Figure 3.6: Distribution Center Operations flow chart

Each time a store places an order, its order quantity is compared with the distribution center on hand inventory and the order will be satisfied accordingly.

Decision:

$$D_{s,i} \leq OHI_{DC,i}$$

Where; $D_{s,i}$ = Demand from store s for item i

$OHI_{DC,i}$ = DC On Hand Inventory for item i

If true, then:

Demand is 100% satisfied.

If false, then:

Demand is partially satisfied or not satisfied.

Satisfied quantities will be recorded to calculate the fill rate. Distribution centers will take a processing time to process each order and it is calculated using triangular distribution. The probability density function for triangular distribution is given by:

$$f(x|a,b,c) = \begin{cases} 0 & for\ x < a, \\ \dfrac{2(x-a)}{(b-a)(c-a)} & for\ a \leq x \leq c, \\ \dfrac{2(b-x)}{(b-a)(b-c)} & for\ c < x \leq b, \\ 0 & for\ b < x, \end{cases}$$

Where; a = lower limit /minimum

b= upper limit /maximum

$$c = mode$$

$$with,\ a < b \text{ and } a \leq c \leq b$$

Distribution centers will take between *a (minimum)* and *b (maximum)* hours with most likely *c (mode)* hours to process each order. Figure 3.5 is an example of the distribution plot for triangular distribution, when a= 10 hours, b= 30 hours and c= 20 hours.

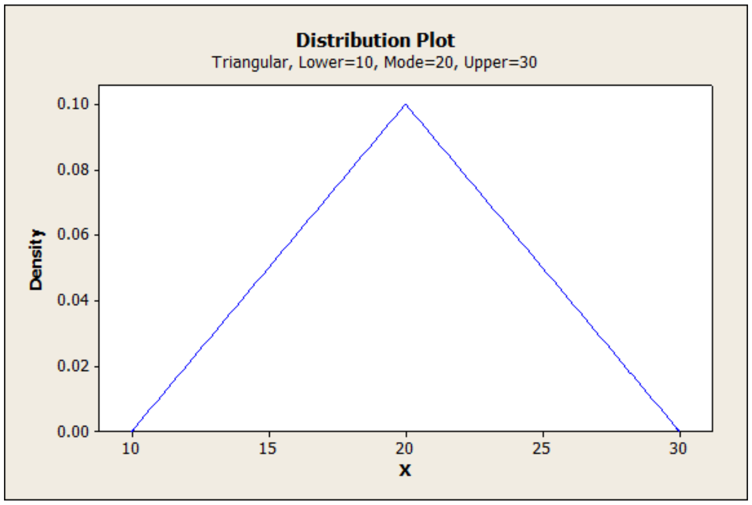


Figure 3.5: Distribution plot for triangular distribution when a= 10 hours, b= 30 hours and c= 20 hours

Total processing time for each order is calculated as:

$$Processing\ time = Triangular(a, c, b) * D_{s,i}$$

Similar to stores, distribution centers will also review their on hand inventory level periodically based on a reorder inventory policy, (s,S) policy and will decide whether to place an order to the supplier or not.

Decision:

$$OHI_{DC,i} \leq s_{DC,i}$$

Where; $s_{DC,i}$ = *Reorder point for item i in distribution center DC*

If true, then:

$$Q_{DC,i} = S_{DC,i} - OHI_{DC,i}$$

Where; $Q_{DC,i}$ = Order Quantity for distribution center DC for item i

$S_{DC,i}$ = *Order up to Quantity for item i in distribution center DC*

Issue an order to the supplier for number of units needed ($Q_{DC,i}$).

If false, then:

No orders will be placed.

Suppliers:

A supplier, in a supply chain is an enterprise that contributes goods or services in a supply chain. A supplier manufactures inventory items and sells them to the next link in the chain, which is distribution center in our model. Figure 3.8 shows the supplier operation flow. Once distribution centers places the order to supplier, supplier will manufacture those items and will complete the order.

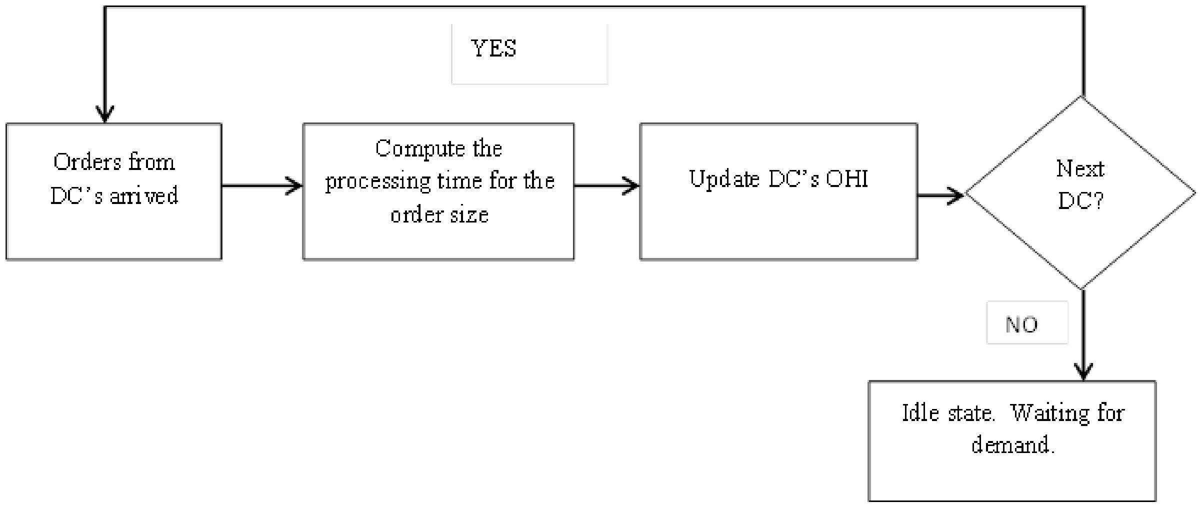

YES
Orders from DC's arrived
Compute the processing time for the order size
Update DC's OHI
Next DC?
NO
Idle state. Waiting for demand.

Figure 3.8: Supplier Operations flow chart

Supplier will take some time to process the order and the processing time is calculated using triangular distribution. The supplier will take between *a* and *b* minutes with the most likely *c* minutes to manufacture item i. Figure 3.7 displays the distribution plot for triangular distribution when a= 10 minutes , b=30 minutes and c=15 minutes.

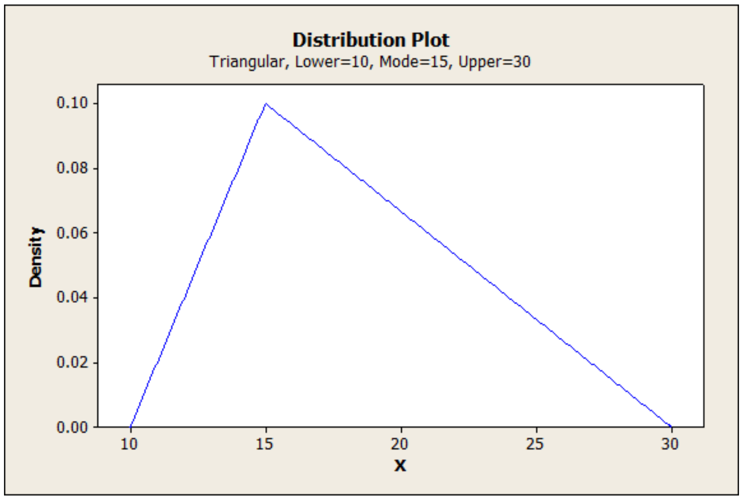


Figure 3.7: Distribution plot for triangular distribution when a= 10 minutes, b=30 minutes and c=15 minutes

Total processing time for item i with $Q_{DC,i}$ order quantity is calculated as:

$$Processing\ time = Traingular(a, c, b) * Q_{DC,i}$$

For this research, it is assumed that the distribution center demand will be 100% fulfilled every time by the suppliers and the suppliers will have all the resources that they need to manufacture products to fulfill the demands.

Objective Function Formulation:

For each store and distribution center node, fill rate and total inventory level is recorded daily for performance measurement calculations.

Average Inventory for one single day for the k^{th} node:

$$AvgInv_{ik} = \sum_{j=1}^{n} OHI_{ij} \Big/ n$$

Where; n = no of inventory items

OHI_{ij} = On Hand Inventory for the jth inventory item on the ith day.

Average Inventory for d days for the kth node:

$$AvgInv_k = \sum_{i=1}^{d} AvgInv_{ik} \Big/ d$$

Where; d = no of days

$AvgInv_k$ = Average Inventory for kth node for d days

Fill rate for the jth inventory item in one single day is estimated as follows:

$$FR_{ij} = MIN(1, \frac{OHI_{ij}}{OrdQty_{ij}})$$

Where; OHI_{ij} = On Hand Inventory for jth item on the ith day.

$OrdQty_{ij}$ = Order Quantity for jth item on the ith day.

We used weighted average method to come up with overall fill rate, where the weights are representative of the importance of the specific inventory items.

Thus, the overall Fill rate for all the items for one single day for kth node can be calculated as:

$$FR_{jk} = {\sum_{j=1}^{n} W_j FR_j} \Big/ {\sum_{j=1}^{n} W_j}$$

Where; Sum (W_j) = 1

n = no of inventory items that has a demand

Average Fill rate for d days:

$$FR_k = {\sum_{i=1}^{d} FR_i} \Big/ d$$

Where; d = no of days

Total average inventory for the entire supply chain network for d days:

$$TotalAvgInv = \sum_{k=1}^{n} AvgInv_k$$

Where; n = no of nodes

Average fill rate for the entire supply chain network for d days:

$$FR = {\sum_{k=1}^{n} FR_k} \Big/ n$$

Where; n = no of nodes

Goal of this research is to select the best (s,S) policy which gives the minimal total average inventory level with higher fill rate. In this model these two objectives are typically conflicting. As the inventory level is minimized the fill rate goes down. Therefore, we need to convert this multi objective problem in to a single objective.

Objective 1: Minimize average total inventory

Objective 2: Maximize average fill rate

Objective 1 and Objective 2 are combined to create a single objective.

Inv/Fill rate = Average total inventory/ Average fill rate

As the fill rate goes up and the inventory goes down, the modified objective function's value goes down.

Modified objective: Minimize (Inv/Fill rate)

Inv/Fillrate for kth node for d days:

$$(INV/FR)_k = \frac{AvgInv_k}{FR_k}$$

Inv/Fillrate for the entire supply chain network for d days:

$$(INV/FR) = \frac{TotalAvgInv}{FR}$$

3.3 Simulation model, Supply Chain configuration and Design of Simulation Experiments

3.3.1 Simulation model

Making a simulation model more flexible and easy to use for any simulation scenario is an important factor for any decision based tool. Supply chain conceptual model is translated into a computerized simulation model using Siemens Technomatix plant simulation software. In this study, user interfaces were introduced to allow more flexibility for the supply chain simulation model. Detail description of the model and the user interfaces are included in the Appendix I and SIMTALK coding related to the model is available in Appendix III.

3.3.2 Supply Chain configuration

The simulation model used for this research includes a supply chain network with four stores, two distribution centers, one supplier and seven different product items. Figure 3.9 displays the supplier chain network used for this study.

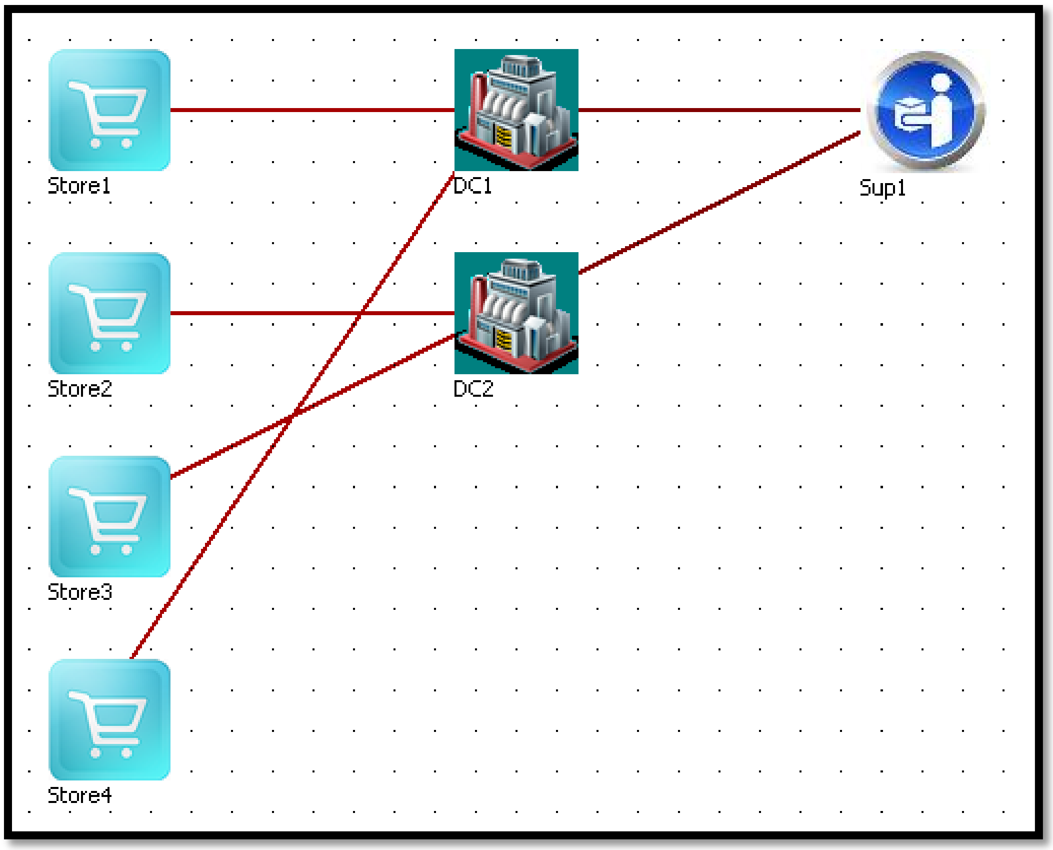


Figure 3.9: Supply chain network model used for the study

Stock keeping units (SKU) table contains all the product information including product identification number, product details, minimum, maximum and most likely time for manufacturing in minutes. Table 3.1 displays the SKU table used for this study. Seven different product items were used for this study.

Table 3.1: SKU table

SKU_ID	Item Name	Most likely time to manufacture(minutes)	Minimum time to manufacture(minutes)	Maximum time to manufacture(minutes)
Item1	A	01:00.0	00:50.0	01:50.0
Item2	B	02:00.0	00.50.0	02:50.0
Item3	C	01:00.0	00.50.0	01:50.0
Item4	D	00:40.0	00.30.0	01:00.0
Item5	E	01:10.0	01:00.0	01:50.0
Item6	F	02:00.0	01:00.0	02:50.0
Item7	G	01:00.0	00:50.0	01:50.0

At the beginning of the simulation, each store is assumed to have the same inventory items, initial inventory, re order level, order up to quantities and average demand value for each item. Table 3.2 is an example of a store inventory table.

Table 3.2: Inventory table for Store 1

ID	Initial Inventory	Reorder level	Order Up to Qty	Mixture
Item1	400	200	400	50
Item2	500	250	500	80
Item3	600	300	600	40
Item4	400	200	400	35
Item5	500	250	500	40
Item6	600	300	600	40
Item7	800	400	800	75

Similar to stores, each distribution center is assumed to have the same inventory items, initial inventory, re order level and order up to quantities for each item at the beginning of the simulation. Table 3.3 is an example of a distribution center inventory table.

Table 3.3: Inventory table for DC 1

ID	**Initial Inventory**	**Reorder level**	**Order Up to Qty**
Item1	2000	1000	2000
Item2	1800	900	1800
Item3	2400	1200	2400
Item4	1600	800	1600
Item5	2000	1000	2000
Item6	3000	1500	3000
Item7	2400	1200	2400

The run length of the simulation will be limited to 365 days. All experiments will be run for this length of time for consistency.

3.3.3 Design of Simulation Experiment scenarios

The goal of this research is to determine parameters for (s,S) inventory policy that minimizes the average total inventory and maximizes fill rate value. In order to create valid experimental scenarios, each pair must satisfy following condition,

$$0 \leq s_g \leq S_g \leq C$$

Where; $g = 1,2 \ldots\ldots 50$

$s_g = Reorder\ ratio$

$S_g = Order\ up\ to\ quanitity\ ratio$

C = (Maximum inventory level possible/Initial inventory)

Since the research problem involves multiple inventory items, one approach to represent the (s,S) parameters is by using as a ratio value compared to initial inventory. Let (s_1,S_1) be the experiment scenario 1, where s_1 is the re order ratio and S_1 is the order up to quantity ratio for experiment scenario 1 . Following formulas will be used to update the reorder level and order up to quantity amount for item *i* in node *k*.

Re order quantity for item i = Ceiling [s_1 * Initial inventory for item i]

Order Up to quantity for item i = Ceiling [S_1* Initial inventory for item i]

Example:

Let's assume (s_1,S_1) = (0.25,0.75)

Based on the data provided in table 3.3 for Item 1;

Initial Inventory = 2000

Reorder level = 1000

Order up to quantity = 2000

Once the (s_1,S_1) ratios are applied for Item 1;

Reorder level = Ceiling [s_1 * Initial Inventory] = Ceil [0.25*2000] = 500

Order up to quantity = Ceiling [S_1 * Initial Inventory]

= Ceiling [0.75* 2000] =1500

Once the initial configuration is completed, genetic algorithm procedure is used to generate best possible experimental scenarios for this research.

3.3.4 Genetic algorithm

Genetic algorithm is used as a tool in this research to generate the best possible experiment scenarios and to get a nearly optimal and useful solution for the research problem. Figure 3.10 displays the flow of the genetic algorithm used for this research. Algorithm starts with an initial population and followed by fitness evaluation, crossover, mutation and replacement.

As stated above, the goal of this research is to find out the best (s,S) policy which gives minimal *average total inventory* and maximum *average fill rate*. Therefore, the goal for the genetic algorithm is to generate best parameters for s and S. Each (s,S) pair is considered as a gene and consist of fractions. To generate a valid gene, each pair must satisfy following condition and this must be satisfied throughout the Genetic Algorithm procedure:

$$0 \leq s_g \leq S_g \leq C$$

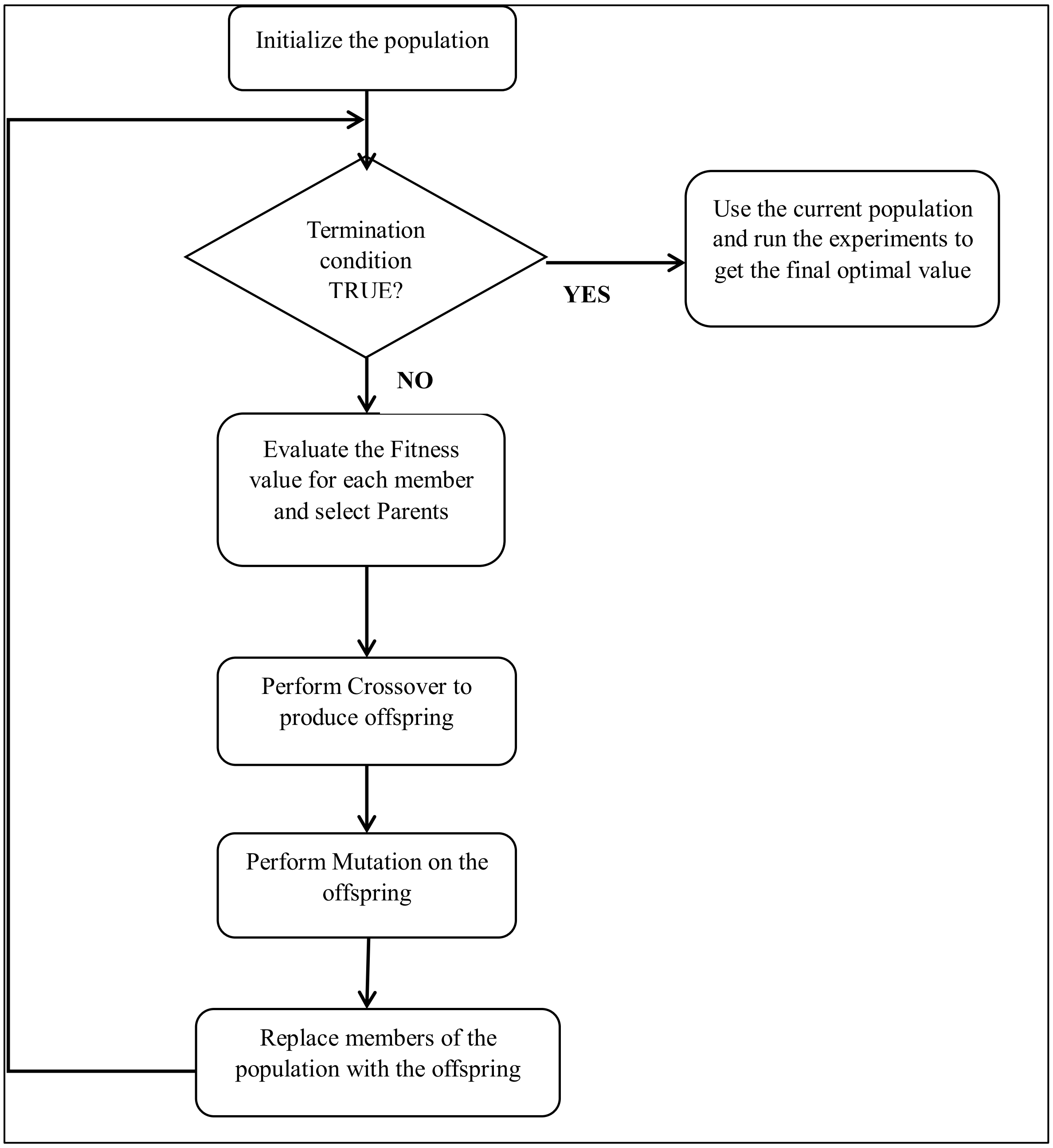

Initialize the population
Termination condition TRUE?
YES
Use the current population and run the experiments to get the final optimal value
NO
Evaluate the Fitness value for each member and select Parents
Perform Crossover to produce offspring
Perform Mutation on the offspring
Replace members of the population with the offspring

Figure 3.10: Genetic algorithm (GA) flow chart

Initialize population

Each variable for (s_g,S_g) pair should be generated randomly and each generated (s_g,S_g) pair should satisfy $0 \leq s_g \leq S_g \leq C$ condition. Following steps will be followed to generate a valid (s_g,S_g) gene.

Step 1: s_g is assigned with a fraction value between 0 and C-0.01 and s_g is generated randomly from a uniform distribution on a set {0,0.01,0.02…….C-0.01} .

Equivalent simtalk code used to generate s_g.

s_g = z_uniform(0.01,0,C-0.01)

Step 2: S_g is assigned with a fraction value between s_g+0.01 and C and S_g is generated randomly from a uniform distribution on a set { s_g+0.01 , s_g +0.02……….C }.

Equivalent simtalk code used to generate S_g.

S_g = z_uniform(0.01, s_g+0.01,C)

Above two step initialize procedure, ensures that each variable in (s_g,S_g) pair meets the required condition.

Evaluate the population and select Parents

Once the population is initialized each gene should be validated based on their fitness. For each and every (s_g,S_g) pair/gene in the population, fitness value is calculated based on the fill rate value. Fill rate>90% is considered as a higher fitness as a parent and genes that generated fill rate>90% is selected as *Parents* for crossover.

Perform Crossover to generate offspring

Crossover operation is performed on two selected Parents and genes will be interchanged between two parents to produce two offspring that derive genes from their parents. Each new offspring should satisfy $0 \leq s_g \leq S_g \leq C$ condition. Following steps will be followed to create valid offspring.

Let ($s_{g,1}$,$S_{g,1}$) and ($s_{g,2}$,$S_{g,2}$) be the two parents that were selected for crossover operation. One of the following methods will be used to perform the interchange with equal probability (1/2).

- If ($s_{g,1}< S_{g,2}$) then,

  Replace $s_{g,2}$ with $s_{g,1}$.
- If ($s_{g,2}< S_{g,1}$) then,

  Replace $s_{g,1}$ with $s_{g,2}$.

If none of the above conditions are satisfied, no interchange will be performed as it will create invalid genes.

Perform Mutation on the offspring

With the mutation procedure, the value of a gene in an offspring can be mutated based on a predefined mutation rate. Mutation helps to generate alternative solutions to get the optimal results. Each offspring is subjected to a mutation based on the predefined mutation rate. One of the following methods can be used to perform mutation with equal probability (0.5) in order to satisfy the $0 \leq s_g \leq S_g \leq C$ condition.

- Value of s_g is changed to a random fraction value on {0.01,0.02……., S_g-0.01}
- Value of S_g is changed to a random fraction value on { S_g+0.01, S_g +0.02,..C-0.01}

Replace members of the population with the offspring

Once the mutation operation is completed, current population should be replaced with the newly generated offspring members and those become the new population for the next round. The number of created offspring for each round is equal to the number of members in the current population.

Termination

Iteration process of the genetic algorithm will be terminated when it reaches predefined maximum number of rounds. Once the iteration is terminated, current population will be used as the experiment scenarios and the final simulation will be executed to find out which scenario provides the minimal *Average total inventory* and maximum *fill rate.*

The parameter setting of the GA is as follows: The number of population (g) is 50, inventory capacity ratio (C) is 1.5, parent selection probability is 0.7, cross over rate is 0.8, mutation rate is 0.1, number of rounds is 20, and number of trials is 3.

3.3.5 Single echelon experiment setup

Experimental scenarios that were generated through genetic algorithm are now used to experiment each node in the supplier chain network. Experiments were conducted using "Experiment manager" utility in Siemens plant simulation. For each store and distribution center, experiments were conducted using following parameter settings:

Input variables: Reorder ratio(s), Order up to ratio(S)

Output variables: Average fill rate, Average total inventory and (Average total inventory/Average fill rate) for the selected node

Number of experiments: 50

Number of trials : 3.

For each experiment, selected node (s, S) inventory policy will vary according to the experimental scenarios and all the other nodes in the supplier chain network will use a default inventory (s, S) policy of (0.50, 1).

Experimental data is collected for each store and each distribution center to find out the best (s, S) parameter for each individual node.

3.3.6 Multi echelon experiment setup

In multi echelon optimization, the whole system is considered as a centralized system and the goal is to determine the best inventory policy for stores and distribution centers that will give minimal value for (average total inventory/average fill rate) for the whole system. Let's assume that we have five experiment scenarios from (s_1, S_1)…… (s_5, S_5). To analyze the entire system performance, we need to create a 5x5 matrix for this.

Stores will run five (s,S) inventory policies((s_1,S_1)…….(s_5,S_5)) and distribution centers will run five (s,S) inventory policies((s_1,S_1)…….(s_5,S_5).).For this configuration, all possible experiment scenarios are developed using table 3.4.

Table 3.4: Example of a Design matrix used to generate Multi echelon experiment scenarios

		Stores				
Distribution Centers		(s_1,S_1)	(s_2,S_2)	(s_3,S_3)	(s_4,S_4)	(s_5,S_5)
	(s_1,S_1)	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate
	(s_2,S_2)	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate
	(s_3,S_3)	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate
	(s_4,S_4)	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate
	(s_5,S_5)	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate	AvgInv/Fillrate

Total number of experiments for the above example: 5*5 = 25 experiments

Multi echelon experiments used for this study are conducted using following parameter settings:

Input variables: Distribution center reorder ratio (s_{DC}), Distribution center order up to ratio (S_{DC}), Store reorder ratio (s_s), Store order up to ratio (S_S)

Output variables: Average fill rate, Average total inventory and Average total inventory/Average fill rate for the entire supply chain network.

Number of experiments: 50 *50 = 2500

Number of trials : 3.

Chapter 4: SIMULATION RESULTS AND ANALYSIS

4.1 Supply chain model base simulation

Prior to commencement of experiments, supply chain model was used for base simulation experiment to test and verify the model. Figure 4.1 represents how the total on hand inventory level and average inventory level changes during the time period of 360 days for the entire supply chain network.

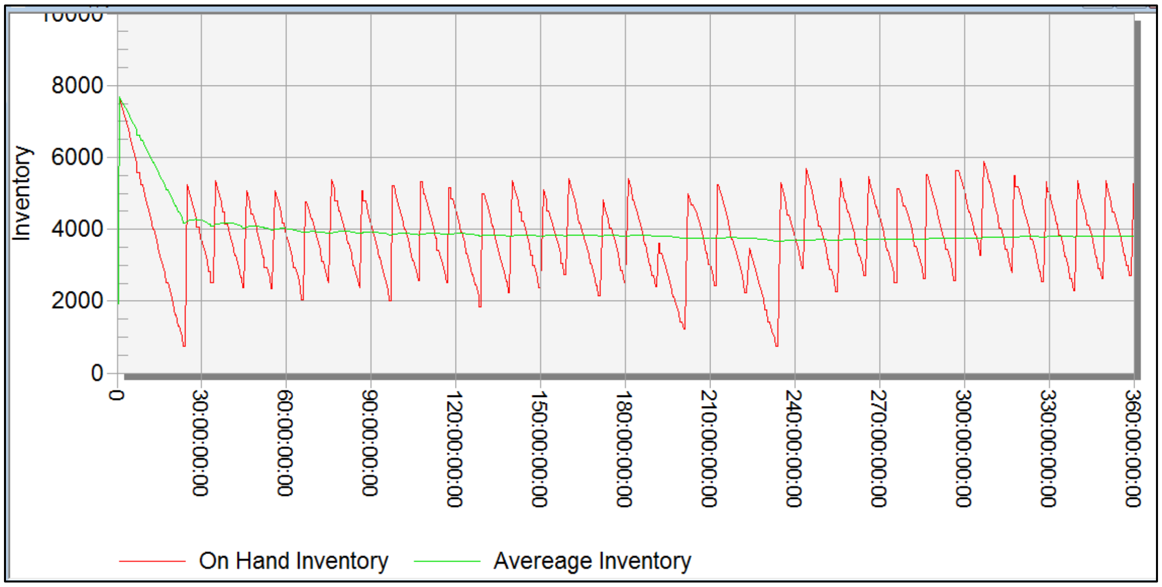


Figure 4.1 Inventory pattern for entire supply chain network with (s, S) inventory policy (0.5, 1) applied.

4.2 Experiment scenarios generation

Using the Genetic algorithm, 50 experiment scenarios were selected for the research study. Generated experiment scenarios are included in table II.1 in Appendix II.

4.3 Single Echelon Experiments

For single echelon optimization, experiments are conducted on one single node at a time, while keeping all the other nodes constant. All the nodes except the node selected for the experiment are configured to s=0.5 and S=1 ratios before the commencement of the experiments. Fifty experiment scenarios were used to analyze each node performance in single echelon inventory system and results with tables are included in table II.2 and table II.3 in Appendix II.

Once the results are collected, experiment scenarios were sorted based on the INV/FR value in ascending order to find out those with minimum INV/FR for the specific node. The objective of this experiment is to minimize the INV/FR value for the selected node while keeping the service level (FR) above 90%. Overall supply chain network performance measurements are ignored for these experiments since the focus is on minimizing objective function value for the selected individual node.

If the selected node is k:

Minimize: $(INV/FR)_k$ for the kth node.

Subject to: $FR_k >= 0.90$

Selected experimental scenarios are then validated to see if the (s, S) values provide the best value not only for $(INV/FR)_k$ but also for FR_k values for node k. Following results were found as the best optimal solution for stores and distribution centers individually.

Experiment results for Distribution centers:

Since configuration settings were similar for all the distribution centers: DC 1and DC 2, outputs were very similar with small variations.

DC results for best optimal solution:

$s_{DC} = 0.65$

$S_{DC} = 0.94$

$(INV/FR)_{DC} = 983.688$

$(FR)_{DC} = 0.95 = 95\%$

Where;

s_{DC} = Reorder level ratio for DCs

S_{DC} = Order up to quantity ratio for DCs

Using Box plots to determine best values for (s, S):

Box plots offer the ability to see the variability and confidence interval for each experiment. Twelve experiments were selected from the results tables with 6 best values and 6 worst values for$(INV/FR)_{DC}$. The primary response plots are displayed in Figure 4.2 and Figure 4.3.

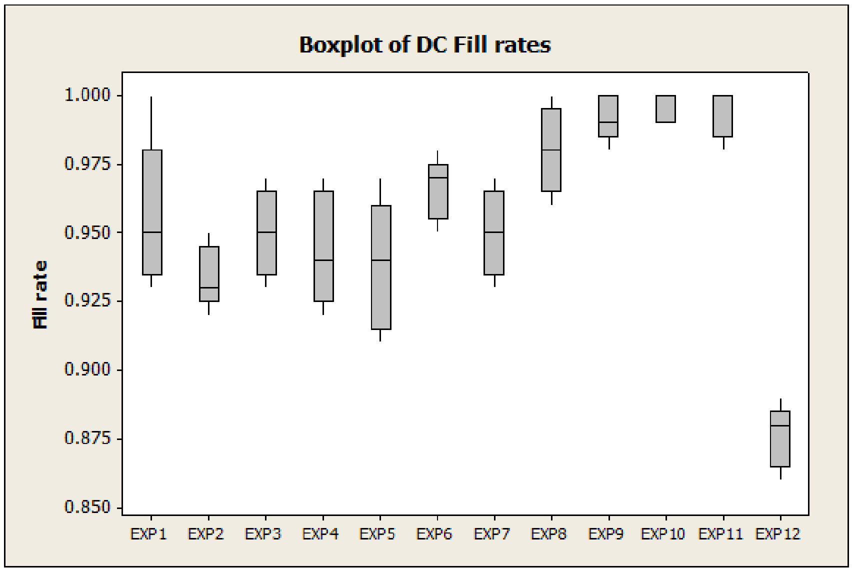


Figure 4.2 Boxplot of average fill rate (FR_{DC}) for the distribution centers

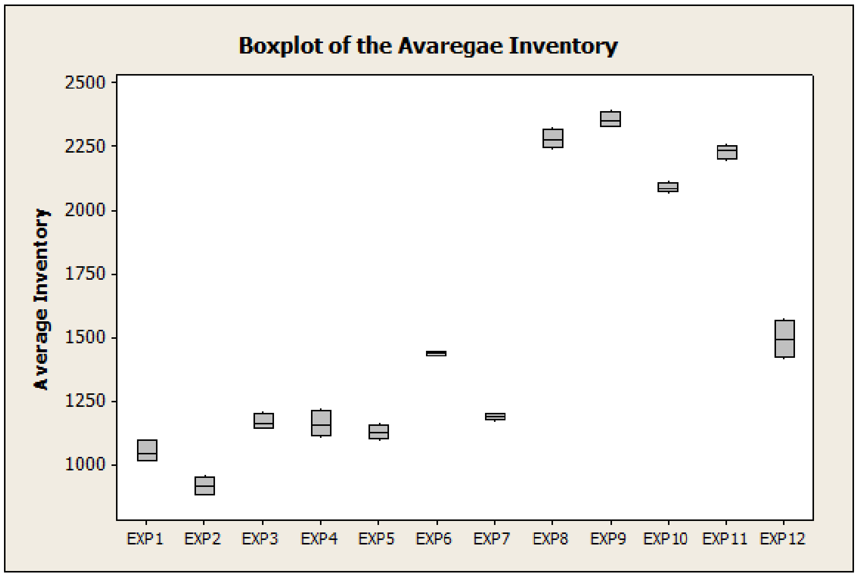


Figure 4.3: Boxplot of average inventory ($AvgInv_{DC}$) for the distribution centers

Based on the Fill rate plot, Exp 9, Exp 10 and Exp 11 provide the best values for FR_{DC}. However, based on the Average Inventory plot Exp1 and Exp2 provides the best values for $AvgInv_{DC}$.

Figure 4.4 provides the box plot for combined $(INV/FR)_{DC}$ response. Based on the combined response plot, best values for $(INV/FR)_{DC}$ is provided by *Exp 2*.

Exp 2 (s_{DC}, S_{DC}) = (0.65, 0.94)

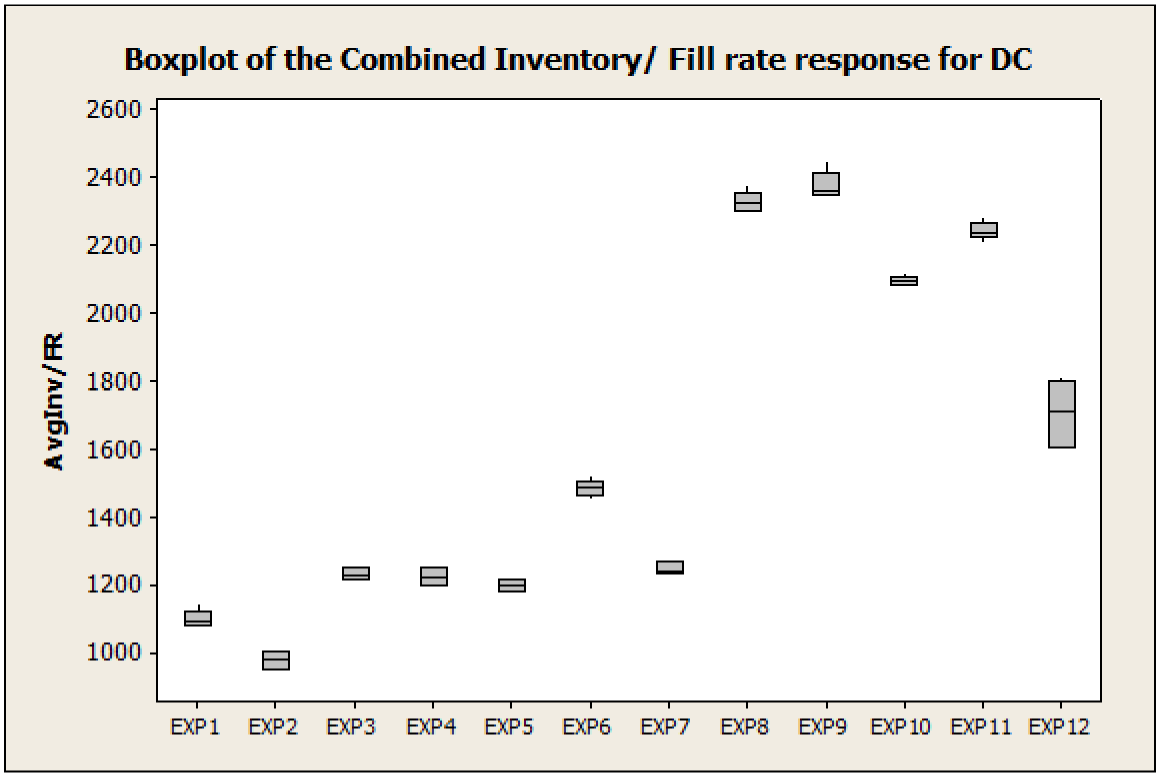


Figure 4.4 Box plot for combined response $(INV/FR)_{DC}$ for the distribution centers

Table 4.1 represents the new reorder levels and order up to quantity levels for DC1 when (s_{DC}, S_{DC}) = (0.65, 0.94) is applied.

Table 4.1: DC1 with optimal reorder and order up to quantity levels for single echelon optimization

ID	Initial Inventory	Reorder level	Order Up to Qty
Item1	2000	1300	1880
Item2	1800	1170	1692
Item3	2400	1560	2256
Item4	1600	1040	1504
Item5	2000	1300	1880
Item6	3000	1950	2820
Item7	2400	1560	2256

Experiment results for stores:

Since configuration settings were similar for all the stores: store 1, store 2, store 3 and store 4 in the model, outputs were very similar with small variations.

Store results for the best optimal solution:

$s_s = 0.47$

$S_s = 1.08$

$(INV/FR)_s$ = 474.68

$(FR)_s$ = 0.98 = 98%

Where;

s_s = Reorder level ratio for stores

S_s = Order up to quantity ratio for stores

Using Box plots to determine best values for (s_s,S_s):

Box plots offer the ability to see the variability and confidence interval for each experiment. Twelve experiments were selected from the results table with 6 best values and 6 worst values for $(INV/FR)_S$. The primary response plots are displayed in Figure 4.5 and Figure 4.6.

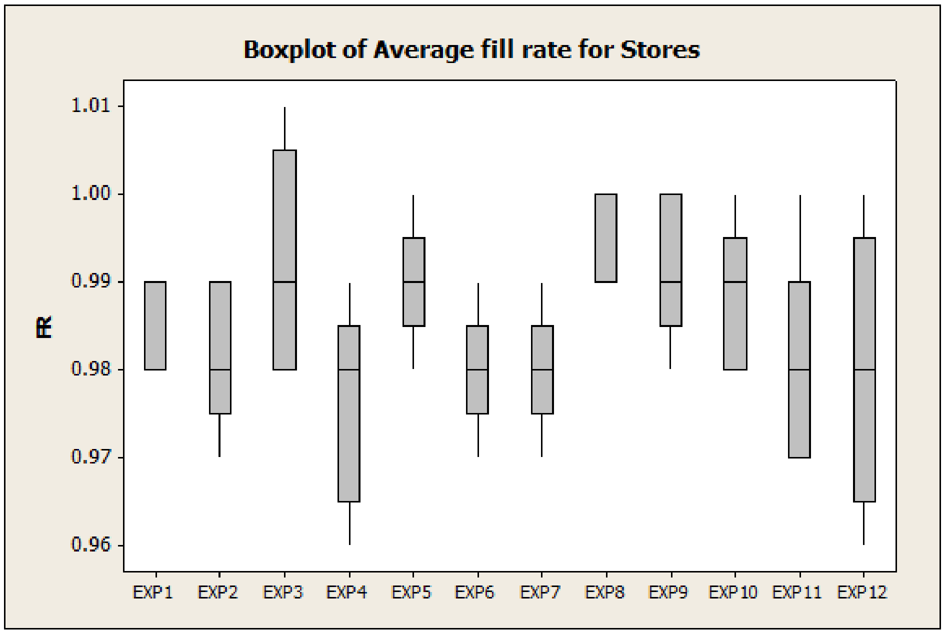


Figure 4.5 Boxplot of average fill rates (FR_s) for stores

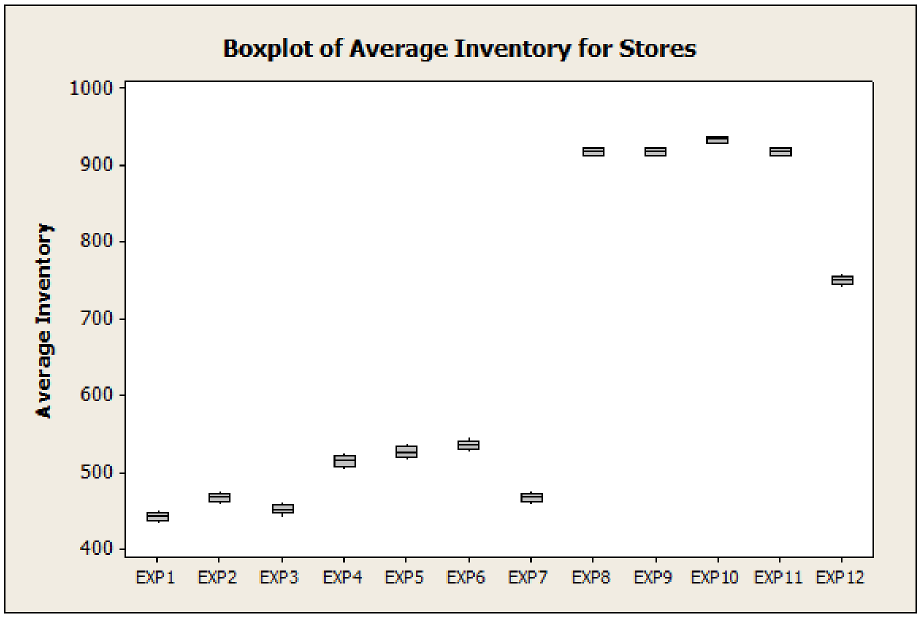


Figure 4.6: Average inventory ($AvgInv_s$) boxplots for stores

Based on the Fill rate plot, Exp 3 provides the best values for FR_s. However, based on the Average Inventory plot Exp1 and Exp3 provides the best values for $AvgInv_s$.

Figure 4.7 provides the box plot for combined $(INV/FR)_s$ response.

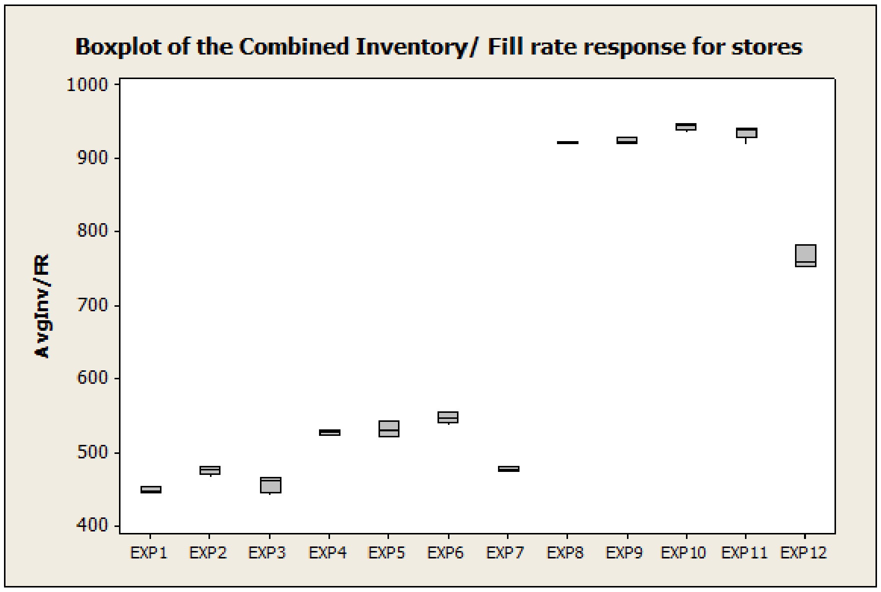


Figure 4.7: Box plot of combined response $(INV/FR)_s$ for stores

Based on the combined response plot, a best value for (INV/FR)s is provided by Exp 1.

Exp 1 $(s_s,S_s) = (0.47,1.08)$

Table 4.2 represents the new reorder levels and order up to quantity levels for Store1 when $(s_s,S_s) = (0.47,1.08)$ is applied.

Table 4.2: Store1 with optimal reorder and order up to quantity levels for single echelon network

ID	Initial Inventory	Reorder level	Order Up to Qty	Mixture
Item1	400	188	432	50
Item2	500	235	540	80
Item3	600	282	648	40
Item4	400	188	432	35
Item5	500	235	540	40
Item6	600	282	648	40
Item7	800	376	864	75

Results for entire supply chain network with optimal parameters for distribution centers and stores:

Once the optimal parameters for (s, S) were collected both for stores and distribution centers, it was used to examine the overall network performance when the optimal parameters are applied for each node.

s_{DC} = 0.65

S_{DC} = 0.94

s_s = 0.47

S_s = 1.08

(INV / FR) for the entire supply chain network = 4137.069

FR for entire supply chain network = 0.96 = 96%

Figure 4.8 displays the inventory pattern for entire supply chain network with the optimal inventory parameters in single echelon settings. According to figure 4.8, total average inventory level resides between 5000 and 4000 when optimal (s,S) parameters are applied for single echelon network.

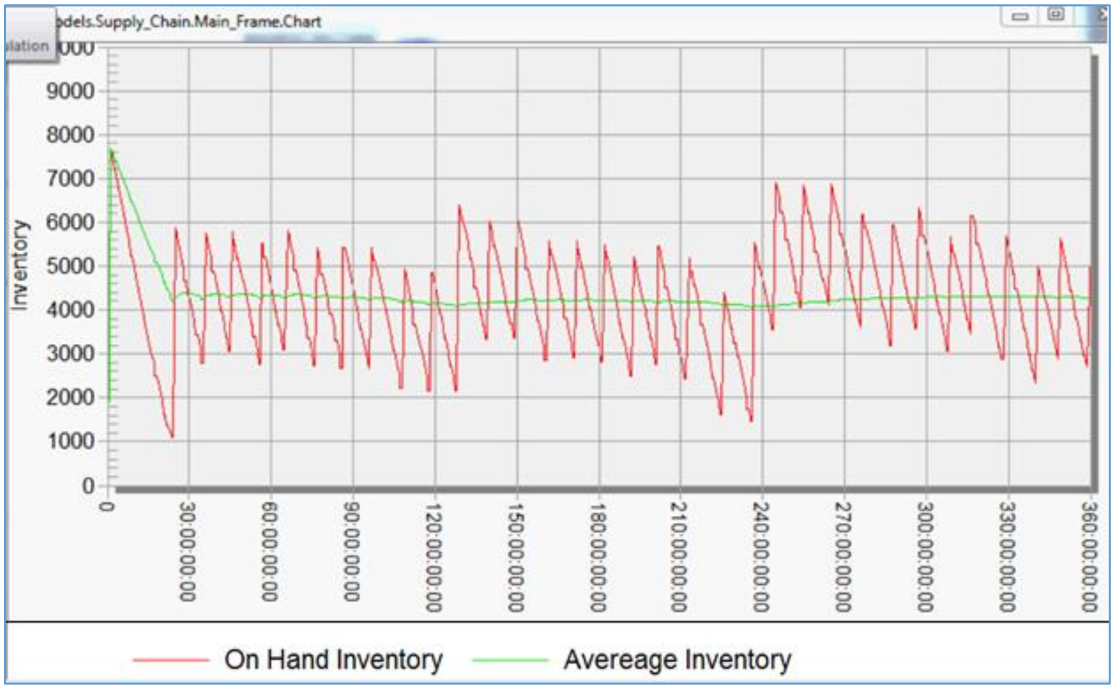


Figure 4.8: Inventory pattern for entire supply chain network with the optimal inventory parameters in single echelon settings

Based on the overall performance measurements, it can be observed that the single echelon optimization does not produce optimal values for the entire system since each individual node acts independently in the system.

4.4 Multi Echelon Experiments

For multi echelon experiments, all nodes are used for the experiment. Each experiment scenario is built with DC reorder ratio, DC order up to ratio, Store reorder ratio and Store order up to ratio information.

A total of 2500 experiment scenarios were used to analyze the multi echelon inventory system behavior and result tables are included in table II.4 in Appendix II.

Once the results are collected, experiment scenarios were sorted based on the INV/FR column in ascending order to find out those with minimum (INV/FR) values. The goal is to minimize the (INV/FR) while keeping the FR above 90%.

Minimize: (INV/FR)

Subject to: $FR >= 0.90$

Selected experimental scenarios are then validated to see if the (s, S) values for both DC and store provides the best value not only for (INV/FR) but also for FR. Following results were found as the most optimal solution for the entire network through the data analysis.

DC :$(s_{DC},S_{DC}) = (0.76,0.86)$

Store :$(s_s,S_s) = (0.47,1.08)$

$INV/FR = 3859.82$

FR = 0.95 = 95%

Using Box plots to determine best values for $((s_{DC},S_{DC}),(s_s,S_s))$:

Box plots offer the ability to see the variability and confidence interval for each experiment. Fourteen experiments were selected from the results tables with 7 best values and 7 worst values for (INV/FR). The primary response plots are displayed in Figure 4.9 and Figure 4.10.

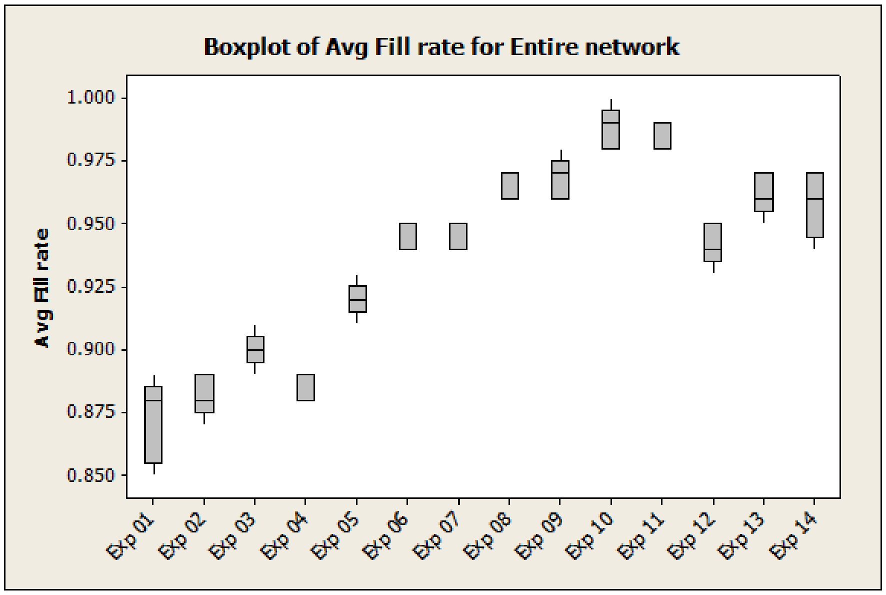


Figure 4.9: Boxplot of Average Fill rate (FR) for the entire system

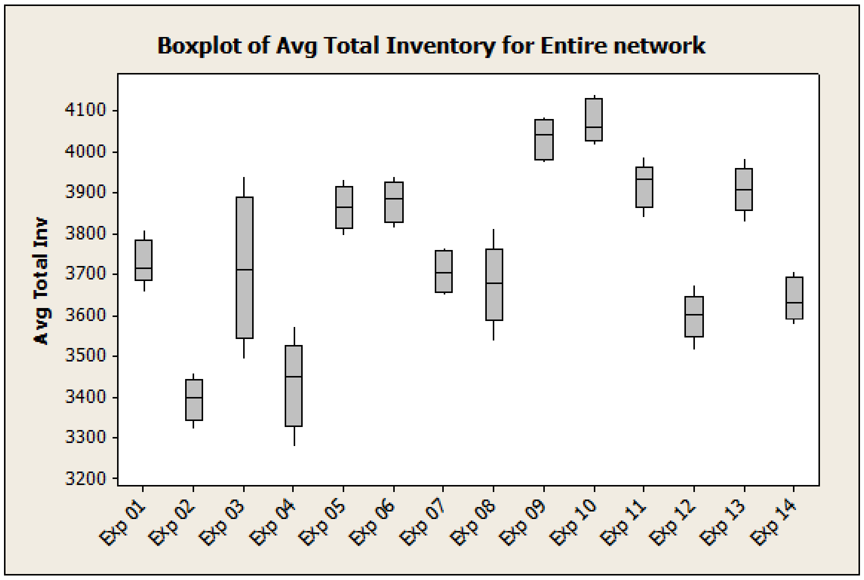


Figure 4.10: Boxplot for Average Inventory (AvgInv) for the entire system

Based on the Fill rate plot, Exp 10 and Exp 11 provide the best values for *FR*. However, based on the Average Inventory plot, Exp2 and Exp4 provide the best values for *AvgInv*.

Figure 4.11 provides the box plot for combined (INV/FR) response.

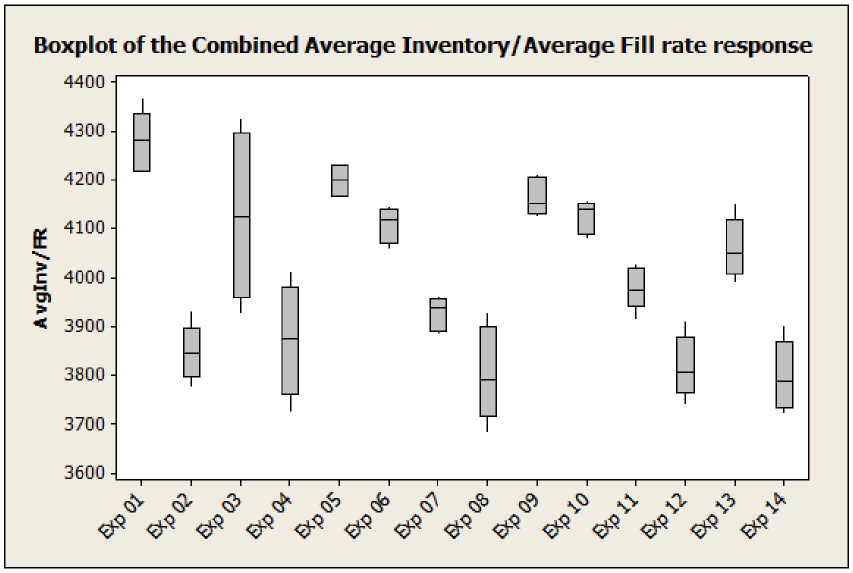


Figure 4.11: Box plot for combined response (INV/FR) for multi echelon optimization

Based on the combined response plot, the best value for (INV/FR) is provided by Exp 14.

Exp 14 $((s_{DC},S_{DC}),(s_s,S_s)) =$ (0.76,0.86),(0.47,1.08)

Table 4.4 represents the new reorder levels and order up to quantity levels for Store1 when (s_s,S_s) = (0.47,1.08) is applied.

Table 4.4: Optimal inventory levels for store1 in multi echelon network

ID	Initial Inventory	Reorder level	Order Up to Qty	Mixture
Item1	400	188	432	50
Item2	500	235	540	80
Item3	600	282	648	40
Item4	400	188	432	35
Item5	500	235	540	40
Item6	600	282	648	40
Item7	800	376	864	75

Table 4.5 represents the new reorder levels and order up to quantity levels for DC1 when (s_{DC},S_{DC}) = (0.76,0.86) is applied.

Table 4.5: Optimal inventory levels for DC1 in multi echelon network

ID	Initial Inventory	Reorder level	Order Up to Qty
Item1	2000	1520	1720
Item2	1800	1368	1548
Item3	2400	1824	2064
Item4	1600	1216	1376
Item5	2000	1520	1720
Item6	3000	2280	2580
Item7	2400	1824	2064

Figure 4.12 displays the inventory pattern for entire supply chain network with the optimal inventory parameters in multi echelon settings. According to figure 4.12, total

average inventory level resides between 4000 and 3000 when optimal (s,S) parameters are applied for multi echelon network.

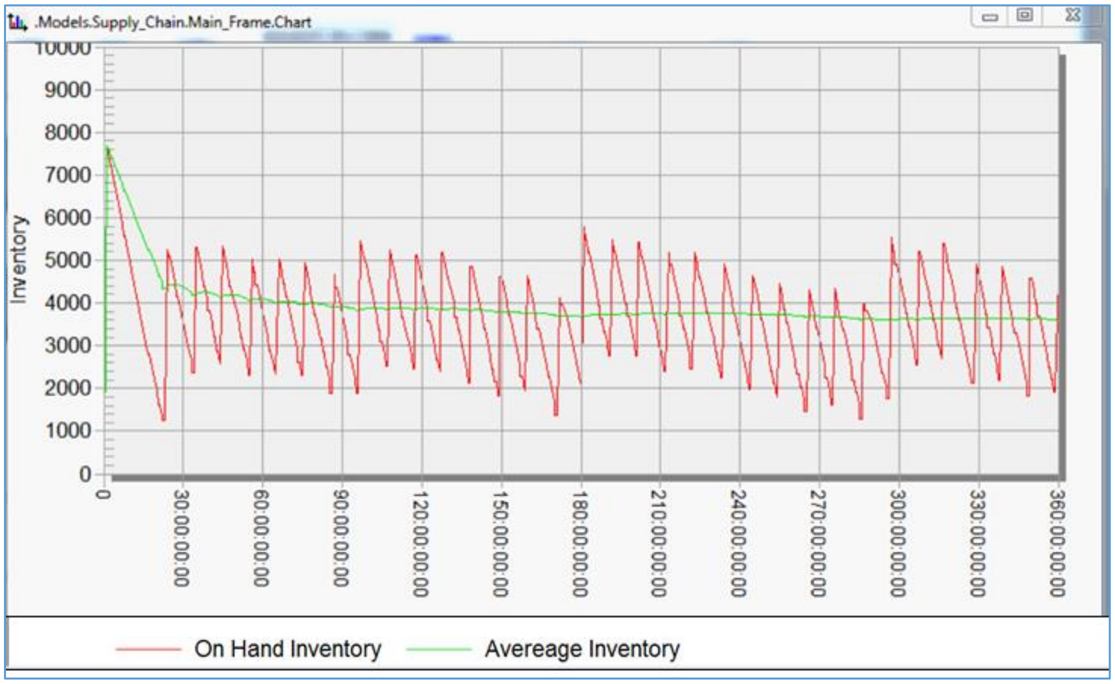


Figure 4.12: Inventory pattern for entire supply chain network with the optimal inventory parameters in multi echelon settings

Results for distribution centers and stores with entire supply chain network optimal parameters:

Once the optimal parameters for $((s_{DC},S_{DC}),(s_s,S_s))$ were collected for the entire supply chain network ,those parameters were used to examine the individual performance of each node.

s_{DC} = 0.76

S_{DC}	$= 0.86$
s_s	$= 0.47$
S_s	$= 1.08$
$(INV/FR)_s$	$= 460.29$
FR_s	$= 0.98 = 98\%$
$(INV/FR)_{DC}$	$= 1005.71$
FR_{DC}	$= 0.93 = 93\%$

Based on the individual component performance measurements, optimal parameters for the entire system produces best parameters for stores, however it does not provide the best parameters for distribution centers.

Chapter 5: DISCUSSION

5.1 Summary

Managing inventories in a supply chain network is an important topic that has received attention from organizations because of its major impact on the economic performance of the organization. Supply chain network can be broadly classified into two categories based on the method used for inventory optimization: single-echelon and multi-echelon. In this research, a simulation based methodology is developed to compare and analyze single echelon inventory systems and multi-echelon inventory systems for supply chains with multiple inventory items. Simulation module is developed using Siemens plant simulation software and experiment scenarios were generated through genetic algorithm procedure. Experiments were carried out to collect performance measurement data for single echelon systems and multi echelon systems. Collected data was then used to derive the optimal inventory policy for single-echelon and multi-echelon systems using ranking and selection methods.

5.2 Conclusion

This research identified that when single echelon optimization is used, each individual node will get the optimal values for its average inventory and fill rate, however they fail to provide the optimal values for the entire supply chain network as a whole. This is due to the fact that single echelon supply chain systems operate as a decentralized system where each node acts individually from one another. When interdependencies among nodes are ignored and each node tries to optimize its own inventory levels and fill rates without considering the impact on connected nodes, it ends up with the overall supply chain having high inventory levels and low fill rates.

In contrast, multi echelon supply chain network performs as a centralized system where each node is dependent on one another and each of the inventory parameters in a node are related to connected nodes and vice versa. The optimal (s,S) parameters for the entire supply chain network provides the optimal inventory levels for stores , however it failed to produce the best optimal inventory levels for distribution centers. Distribution centers produced low inventory levels during the single echelon network settings compared to multi echelon network settings. However, the differences between the values are very low and can be ignored. The satisfaction of the customer is the most important objective of any successful business. Based on the customer perspective, it can be concluded that the stores provided the optimal inventory levels with higher service levels even though distribution centers doesn’t provide the optimal inventory levels under multi-echelon configuration. Therefore, entire network is performing on its optimal performance settings in multi echelon supply chain network.

Finally, based on the results it can be concluded that multi echelon systems provides the optimal inventory levels for the entire supply chain network and strong results for individual nodes as well if not the most optimal. In the best interest of the system as a whole, individual nodes have to give up on further marginal individual performance improvements that may be possible.

5.2 Future Research

Results gathered in this research, showed that there is a conflict between the individual node performance and system performance. When the multi echelon optimal parameters were applied for the entire system, distribution centers showed lower results compared to single echelon performance measurements. For future research, the model can be further extended to incorporate a strategic decision making game-theoretic model. This will help to analyze how John von Neumann's and Oskar Morgenstern's Game theory can be incorporated to resolve above mentioned performance conflicts.

Data used for this research was generated using various probability distributions that best represent the related real life processes. There is an opportunity to further test and verify the developed simulation model with actual data scenarios. With actual data, business logic used to develop the model and the distributions used for the model can be further verified to make sure that the model is working as expected for real time industrial supply chain scenarios.

In this study, Genetic algorithm procedure was used as a tool to generate experimental scenarios. As a further development, genetic algorithm procedure should be modified to generate the final optimal solution for single echelon and multi echelon scenarios.

For future analysis, data collected from this research can be further analyzed using statistical methods in order to find out the significant differences in single echelon vs. multi echelon inventory systems.

References

Supply Chain Council. (1997). Retrieved August 15, 2013, from Supply Chain Council: http://supply-chain.org/

Inventory Management. (2012). Retrieved September 7, 2012, from Management Study Guide: http://www.managementstudyguide.com/inventory-management.htm

ACTOperationsResearch. (2013). *Dynamic and discrete-event simulation*. Retrieved 2013, from Operations Research ACT: http://www.act-operationsresearch.com/Core_Competence/Simulation.html

Adebola, A., Abd-Alhameed, R., & Abousitta, M. (2012). Application of Genetic Algorithm Optimisation Technique in Beam Steering of Circular Array Antenna.

Adeyemi, S. L., & Salami, A. O. (2010). *Inventory Management: A Tool of Optimizing Resources in a Manufacturing Industry A Case Study of Coca-Cola Bottling Company, Ilorin Plant.* Retrieved September 18, 2012, from Kamla-Raj Enterprises.

Agarwal, V. (2007). Supply chain and retail: The means to the end . *Business Daily from THE HINDU group of publications*.

Arkieva. (2013). *Multi Echelon Inventory Optimizer*. Retrieved 8 27, 2013, from arkieva: http://www.arkieva.com/multiechelon.asp

Banks, J., Carson II, J. S., Nelson, B. L., & Nicol, D. M. (2001). *Discrete-Event System Simulation.* Upper Saddle River, New Jersey: Prentice Hall.

Beamon, B. M. (1998, August 15). Supply chain design and analysis:: Models and methods. *International Journal of Production Economics*, pp. 281-294.

Bhaskaran, S. (1998, June 7). Simulation Analysis of a Manufacturing Supply Chain. *Decision Sciences*, pp. 633–657.

Caballini, C., & Revetria, R. (2008). Simulating A Non Multi Echelon Supply Chain Model Using Berkeley MadonnaSoftware: A System Dynamics Approach. *International Journal of Mathematical Models and Methods in Applide Sciences*.

Chopra, S., & Meindl, P. (2003). *Supply Chain Management: Strategy, Planning and Operation.* NJ: Prentice Hall: Saddle River.

Cimino, A., Longo, F., & Mirabelli, G. (2010). A General Simulation Framework for Supply Chain Modeling:State of the Art and Case Study. *IJCSI International Journal of Computer Science*.

George, A., Rajakumar, B. R., & Binu, D. (2012). Genetic algorithm based airlines booking terminal open/close decision system. *International Conference on Advances in Computing, Communications and Informatics* , (pp. 174-179).

Goldberg, D. E., & Holland, J. H. (1988). Genetic Algorithms and Machine Learning. *Machine Learning*.

Hausman, W. H., & Erkip, N. K. (1994, May 1). Multi-Echelon vs. Single-Echelon Inventory Control Policies for Low-Demand Items. *Management Science*.

IBM. (2012). *Inventory Management with Optimization Technology*. Retrieved September 7, 2012, from IBM: http://www-01.ibm.com/software/websphere/optimization/inventory-management/

Imagine That Inc. (2012). *Solutions: Simulation Software Overview*. Retrieved September 20, 2012, from ExtendSim: http://www.extendsim.com/sols_simoverview.html

Jacobs, F. R., & Chase, R. B. (2011). *Operations and Supply Chain Management* . McGraw-Hill.

Jacobs, R. F., & Chase, R. B. (2013). *Inventory Management.* Retrieved September 20, 2013, from Operations and Supply Chain Management: The Core.

Jensen, P. A., & Bard, J. F. (2002). *Operations Research Models and Methods.*

Joines, J. A., & Roberts, S. D. (2012). *Simulation Modeling with Simio: A Workbook* (2nd ed.). Sewickley, Pennsylvania: Simio LLC.

Konagaya, A. (1992). New topics in Genetic Algorithm research. *New Generation Computing*, 423-427.

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

Lee, C. B. (2003). Multi-Echelon Inventory Optimization. *Evant White Paper Series*.

Melanie, M. (1999). *An introduction to genetic algorithms.* Bradford Book.

Nyberg, A., Grossmann, I. E., & Westerlund, T. (2012). An efficient reformulation of the multiechelon stochastic inventory system with uncertain demands. *AIChE Journal*.

Oliver, R., & Webber, M. (1992). Supply-chain management: logistics catches up with.

Onawumi, Oluleye, & Adebiyi. (2011). An Economic Order Quantity Model with. *Int. J. Emerg. Sci*, 465-477,.

Pegden, C. D. (2009). *An Introduction to Simio for Arena Users*. Retrieved September 25, 2012, from Simio: Forward Thinking.

Quinn, F. J. (1997). What's the buzz? *Logistics Management*, 43-47.

Shu, J., Li, Z., & Huang, L. (2012). Demand selection decisions for a multi-echelon inventory distribution system. *Journal of the Operational Research Society* .

Siemens. (2012). *Plant Simulation*. Retrieved 11 14, 2013, from Siemens: http://www.plm.automation.siemens.com

Slack, N., Chambers, S., & Johnston, R. (2006). *Operations Management.* Pearson.

Smith, J. (2012). Scalable Knowledge Interchange Broker: Design and Implementation for Semiconductor Supply Chain Systems.

Stevens, G. C. (1989). Integrating the Supply Chain. *International Journal of Physical Distribution & Logistics Management*, 3-8.

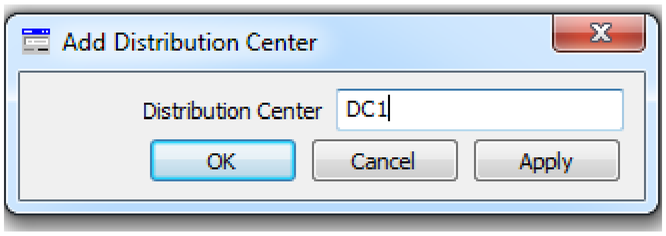
Takahashi, K. (2011). Inventory control in a two-echelon dual-channel supply chain with setup of production and delivery. *International Journal of Production Economics*, 403–415.

Appendix I: Simulation model

Following sections will provide detail descriptions of the user interfaces introduced in the developed application in Siemens Technomatix plant simulation software.

Distribution Center:

"Add Distribution Center "dialog box is used to add a Distribution center to the model. Figure I.1 displays how a user adds a Distribution Center named: DC1.

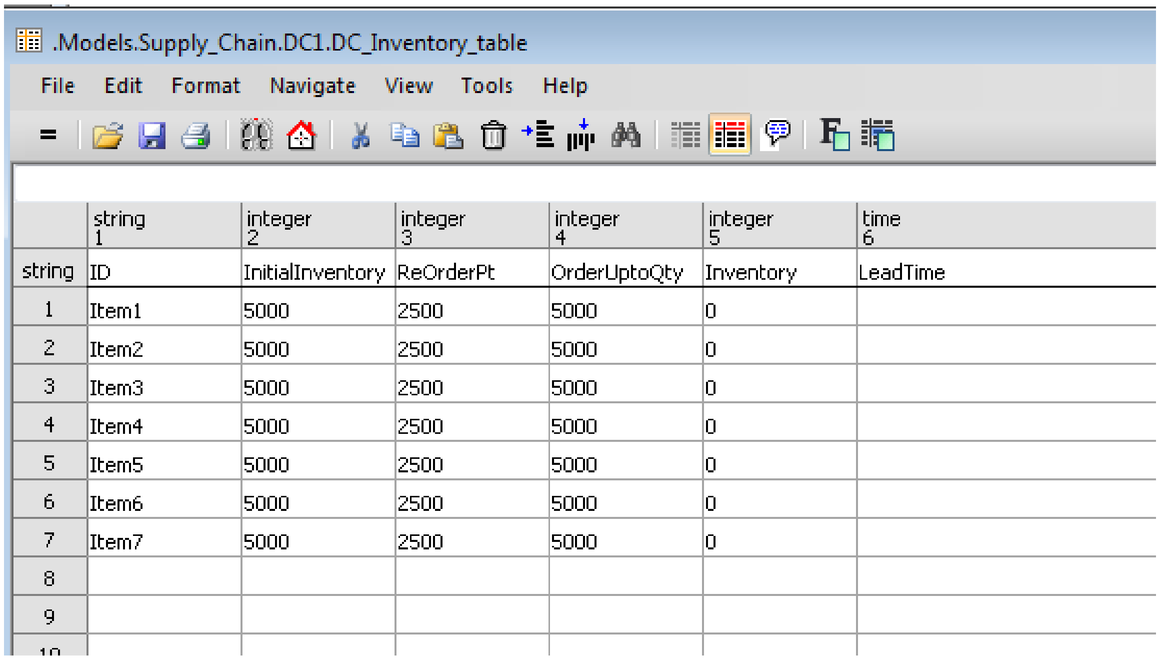


Distribution Center DC1

Figure I.1: Add Distribution Center Dialog box

Once the user has entered the Distribution center there will be a validation to check if the Distribution center already exists and if it does then the system will prompt an error message.

Once the validation process is done the user will be prompted to enter the inventory data for the newly created Distribution center. For each and every item defined in the SKU table a new row with the item Id will be created automatically in the newly created Distribution center Inventory table. User needs to enter the Initial Inventory, Re Order Point and Order Up to level manually. Figure I.2 shows the DC1 inventory table after data entry.


.Models.Supply_Chain.DC1.DC_Inventory_table
File Edit Format Navigate View Tools Help

	string 1	integer 2	integer 3	integer 4	integer 5	time 6
string	ID	InitialInventory	ReOrderPt	OrderUptoQty	Inventory	LeadTime
1	Item1	5000	2500	5000	0	
2	Item2	5000	2500	5000	0	
3	Item3	5000	2500	5000	0	
4	Item4	5000	2500	5000	0	
5	Item5	5000	2500	5000	0	
6	Item6	5000	2500	5000	0	
7	Item7	5000	2500	5000	0	
8						
9						

Figure I.2: DC1 Inventory table

Initial Inventory: Initial inventory is the starting inventory.

Re Order point: Threshold at which Distribution centers should order more products to prevent shortages while also avoiding overstock.

Order Up to level: The maximum inventory position or the target inventory level allowed by the Distribution center. This also can be defined as “base stock level”.

Figure I.3 displays how the newly created distribution appears in the model.

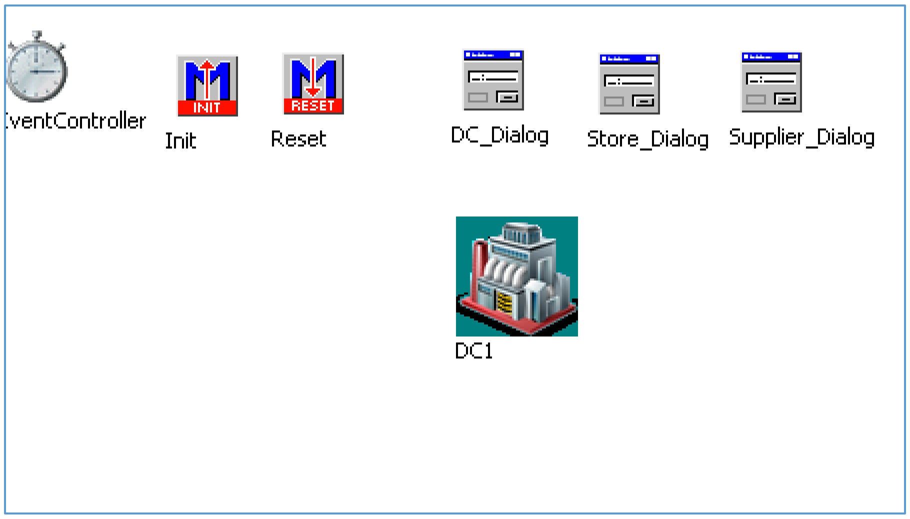


Figure I.3: Snap shot of the model after the first Distribution center is added

Same steps will be followed to add more distribution centers as shown in Figure I.4.

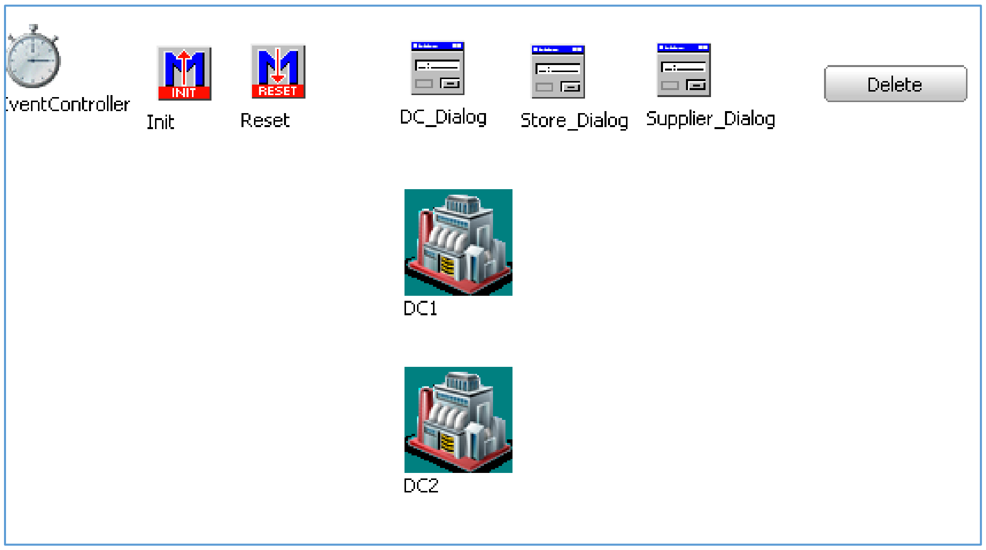
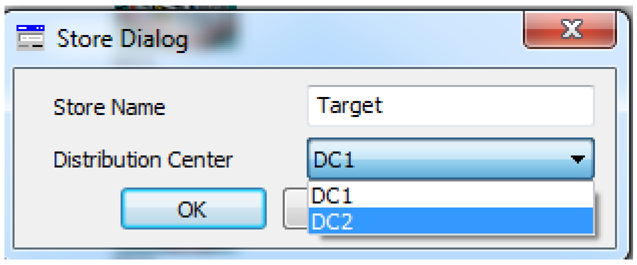


Figure I.4: Snapshot of the model after the addition of the second distribution center

Store:

Store Dialog box allows users to add any number of stores. A store should always connect to one Distribution Center and no more than that. As shown in figure I.5, Store dialog box has the list of available Distribution centers defined in a Dropdown list. To create a Store, user needs to select the desired Distribution center from the drop down list and enter the name of the Store.

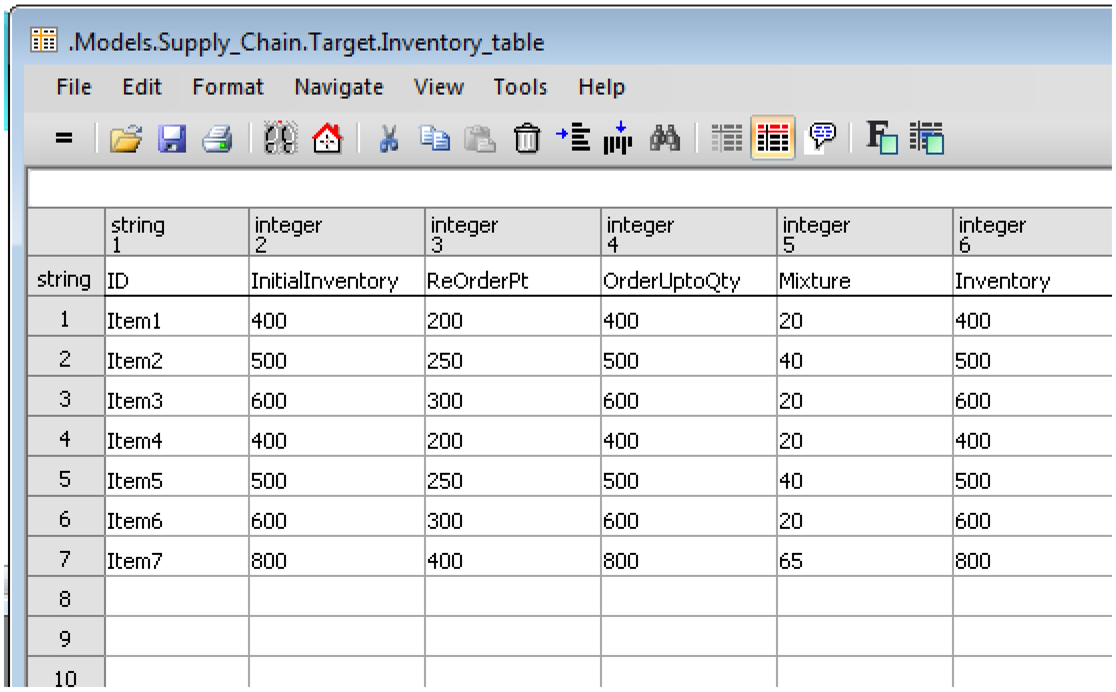


Store Name: Target

Distribution Center: DC1

Figure I.5: Store Dialog box

Similar to Distribution center dialog box, a validation will be done to identify if the user entered store name is unique and once it’s done the user will be prompted to enter the inventory data for the store. Figure I.6 shows a screenshot of the store inventory table.


.Models.Supply_Chain.Target.Inventory_table
File Edit Format Navigate View Tools Help

	string 1	integer 2	integer 3	integer 4	integer 5	integer 6
string	ID	InitialInventory	ReOrderPt	OrderUptoQty	Mixture	Inventory
1	Item1	400	200	400	20	400
2	Item2	500	250	500	40	500
3	Item3	600	300	600	20	600
4	Item4	400	200	400	20	400
5	Item5	500	250	500	40	500
6	Item6	600	300	600	20	600
7	Item7	800	400	800	65	800
8						
9						
10						

Figure I.6: Screenshot of the store inventory table

Initial Inventory, Re-order point and Order Up to level are attributes that are similar to the distribution center inventory table attributes that were discussed earlier. The only new attribute that is introduced in Store Inventory table is the "Mixture" attribute. Store is the place we generate customer demand and to create these demand, probability distribution will be used. And the mixture will represent the average demand value for each and every item.

Same steps can be followed to add more Stores to the model (see Figure I.7).

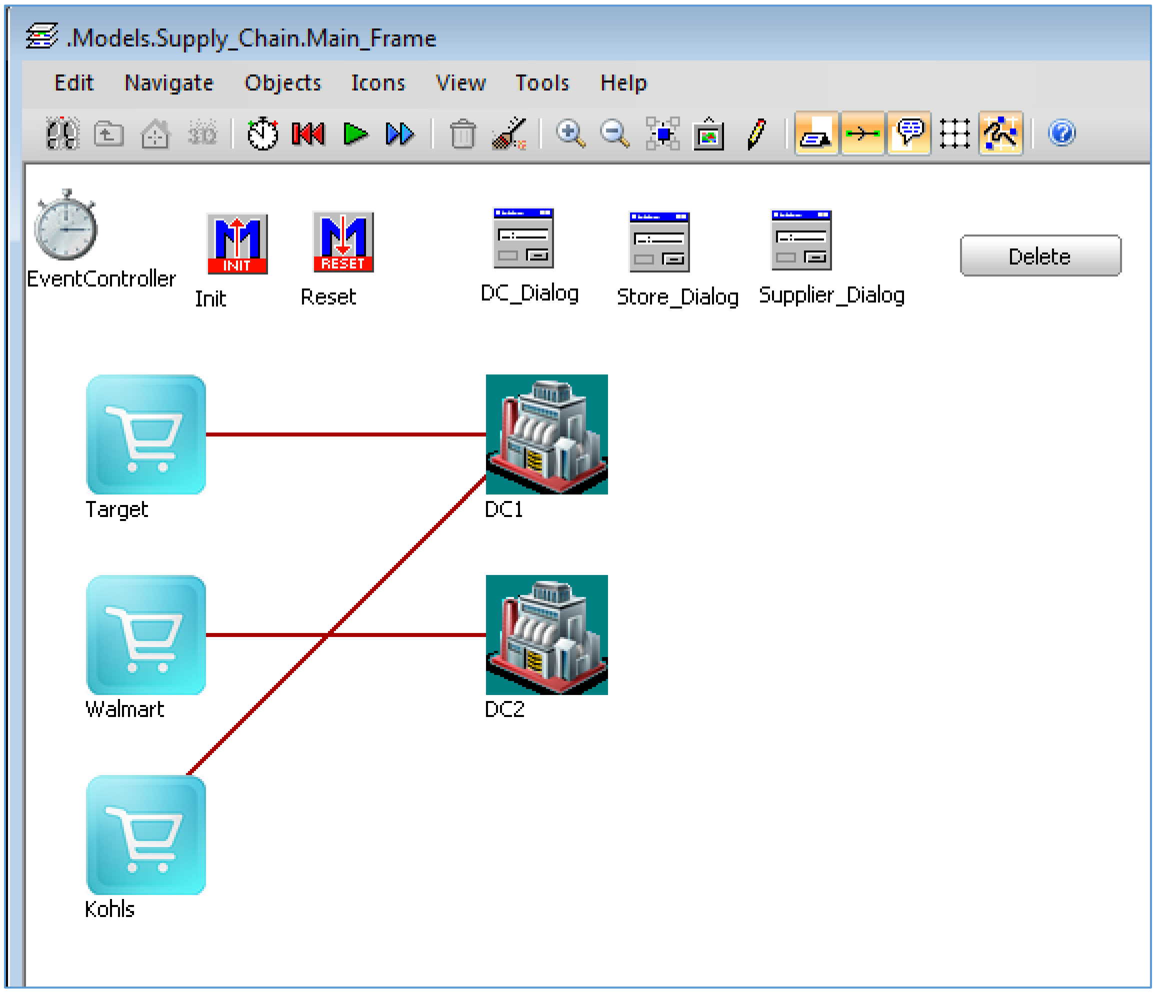


Figure I.7: Snapshot of the model after the stores are added

Supplier:

Supplier dialog allows a user to add suppliers for the supply chain model. All the items that are defined in SKU table will be listed in this dialog and user has the control on selecting which supplier supplies the defined items. Figure I.8 shows how a user can add a new Supplier.

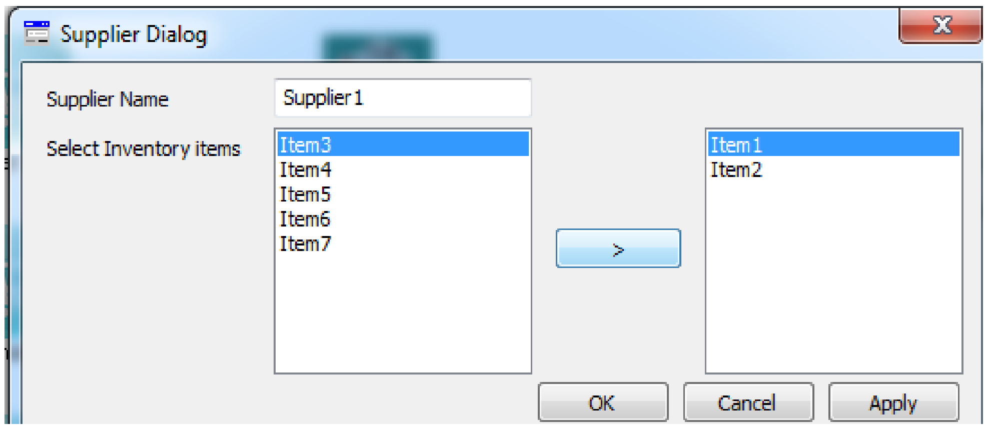


Figure I.8: Supplier Dialog box

Once the configurations are completed, developed supply chain model is displayed as in figure I.9.

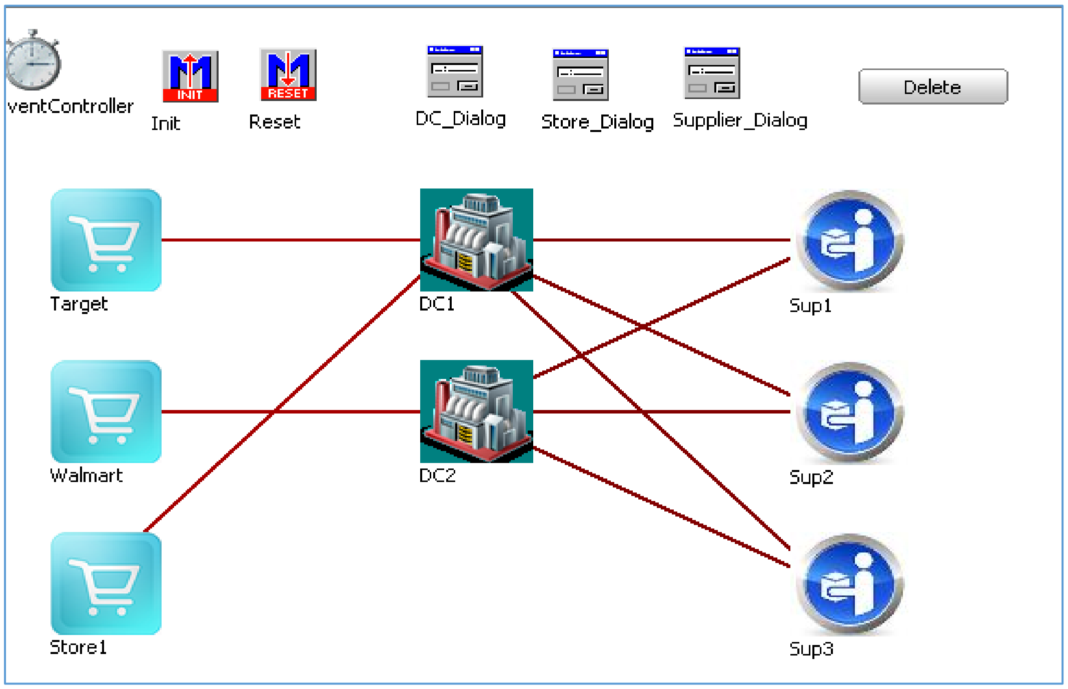


Figure I.9: Completed simulation model

Appendix II: Experiment Results

Experiment scenarios generated through Genetic Algorithm:

Table II.1 lists the 50 experiment scenarios generated using the Genetic Algorithm.

Table II.1: Experiment scenarios generated through genetic algorithm

Experiment	Re order ratio	Order up to quantity ratio
Exp 01	0.52	1.50
Exp 02	0.58	1.45
Exp 03	0.52	1.11
Exp 04	0.64	1.50
Exp 05	0.58	1.23
Exp 06	1.05	1.24
Exp 07	1.20	1.47
Exp 08	0.39	1.25
Exp 09	0.76	1.45
Exp 10	0.54	1.46
Exp 11	0.91	1.11
Exp 12	0.58	1.24
Exp 13	1.10	1.45
Exp 14	0.95	1.45
Exp 15	0.60	1.11
Exp 16	0.52	1.11
Exp 17	0.52	1.34
Exp 18	0.62	1.34
Exp 19	0.73	1.11
Exp 20	0.67	1.30
Exp 21	0.62	1.30
Exp 22	1.06	1.45
Exp 23	1.10	1.45
Exp 24	0.60	1.34
Exp 25	0.76	1.49
Exp 26	0.60	1.11
Exp 27	0.41	1.42
Exp 28	1.05	1.50
Exp 29	0.39	1.25
Exp 30	0.52	1.30
Exp 31	1.11	1.45
Exp 32	0.52	1.17
Exp 33	0.47	1.08

Exp 34	0.65	0.94
Exp 35	0.76	1.34
Exp 36	0.52	1.30
Exp 37	0.64	1.50
Exp 38	0.56	1.30
Exp 39	0.67	1.35
Exp 40	0.73	1.50
Exp 41	1.10	1.24
Exp 42	0.62	1.24
Exp 43	0.76	0.86
Exp 44	1.10	1.11
Exp 45	0.62	1.50
Exp 46	0.52	1.45
Exp 47	0.62	1.24
Exp 48	1.28	1.30
Exp 49	0.25	1.24
Exp 50	0.64	1.11

Experiment results table for a store in single echelon optimization:

Table II.2 documents the experiment results for store 1 in single echelon optimization. As seen from Table II.2, Exp. 38 provides the minimal $(INV/FR)_S$values with (s_s,S_s) =(0.47,1.08) parameters.

Table II.2: Experiment results for Store 1 in single echelon optimization

Experiment	Re order ratio	Order up to quantity ratio	Fill rate	Average Total Inventory	Average Total Inventory /Fill rate
Exp 01	0.52	1.50	0.96	616.76	780.42
Exp 02	0.58	1.45	0.98	726.12	709.93
Exp 03	0.52	1.11	0.99	469.99	557.67
Exp 04	0.64	1.50	0.98	774.13	690.79
Exp 05	0.58	1.23	0.99	569.43	643.49
Exp 06	1.05	1.24	1.00	755.47	693.25
Exp 07	1.20	1.47	1.00	935.21	875.58
Exp 08	0.39	1.25	0.97	573.24	713.49
Exp 09	0.76	1.45	0.98	737.64	692.74
Exp 10	0.54	1.46	0.99	665.82	706.12
Exp 11	0.91	1.11	0.99	654.95	660.57
Exp 12	0.58	1.24	0.99	576.29	610.40

Exp 13	1.10	1.45	1.00	919.58	805.57
Exp 14	0.95	1.45	1.00	751.65	809.20
Exp 15	0.60	1.11	0.99	485.76	578.36
Exp 16	0.52	1.11	0.99	469.99	476.98
Exp 17	0.52	1.34	0.98	649.83	608.93
Exp 18	0.62	1.34	0.98	652.04	658.86
Exp 19	0.73	1.11	0.99	655.03	660.65
Exp 20	0.67	1.30	1.00	619.36	633.08
Exp 21	0.62	1.30	0.99	619.67	623.83
Exp 22	1.06	1.45	1.00	919.58	820.14
Exp 23	1.10	1.45	1.00	919.58	919.58
Exp 24	0.60	1.34	0.99	652.04	745.35
Exp 25	0.76	1.49	1.00	769.51	732.28
Exp 26	0.60	1.11	0.99	485.76	586.42
Exp 27	0.41	1.42	0.98	536.48	531.05
Exp 28	1.05	1.50	0.99	894.54	783.41
Exp 29	0.39	1.25	0.97	573.24	696.22
Exp 30	0.52	1.30	0.97	615.19	619.95
Exp 31	1.11	1.45	0.99	919.50	822.39
Exp 32	0.52	1.17	0.99	518.87	658.20
Exp 33	0.47	1.08	0.98	465.18	474.68
Exp 34	0.65	0.94	0.98	517.45	501.49
Exp 35	0.76	1.34	1.00	657.19	614.34
Exp 36	0.52	1.30	0.99	615.19	636.50
Exp 37	0.64	1.50	0.98	774.13	741.17
Exp 38	0.56	1.30	0.99	626.22	681.53
Exp 39	0.67	1.35	1.00	659.92	648.88
Exp 40	0.73	1.50	0.99	777.40	743.17
Exp 41	1.10	1.24	1.00	755.47	765.70
Exp 42	0.62	1.24	1.00	571.39	634.79
Exp 43	0.76	0.86	0.99	455.11	499.32
Exp 44	1.10	1.11	0.99	654.95	596.41
Exp 45	0.62	1.50	0.98	761.88	744.25
Exp 46	0.52	1.45	0.97	639.18	698.83
Exp 47	0.62	1.24	0.99	571.39	596.81
Exp 48	1.28	1.30	1.00	802.36	725.30
Exp 49	0.25	1.24	0.94	417.01	572.10
Exp 50	0.64	1.11	0.96	525.37	503.92

Experiment results table for a DC in Single echelon network:

Table II.3 documents the experiment results for DC 1 in single echelon optimization. As seen from Table II.3, Exp. 34 provides the minimal $(INV/FR)_{DC}$values with (s_{DC},S_{DC}) =(0.65,0.94) parameters.

Table II.3: Experiment results DC 1 Single echelon network

Experiment	Re order ratio	Order up to quantity ratio	Fill rate	Average Total Inventory	Average Total Inventory /Fill rate
Exp 01	0.52	1.50	0.94	1492.37	2300.38
Exp 02	0.58	1.45	0.96	1463.61	1562.11
Exp 03	0.52	1.11	0.95	1376.16	1470.24
Exp 04	0.64	1.50	0.97	1569.42	1574.53
Exp 05	0.58	1.23	0.97	1128.38	1319.60
Exp 06	1.05	1.24	0.98	1723.02	1533.67
Exp 07	1.20	1.47	1.00	2276.88	2086.72
Exp 08	0.39	1.25	0.93	1027.13	1539.20
Exp 09	0.76	1.45	0.96	1609.54	1456.42
Exp 10	0.54	1.46	0.95	1431.78	1554.87
Exp 11	0.91	1.11	0.96	1418.59	1491.01
Exp 12	0.58	1.24	0.96	1193.88	1340.30
Exp 13	1.10	1.45	0.99	2234.27	1886.02
Exp 14	0.95	1.45	1.00	2083.66	2139.07
Exp 15	0.60	1.11	0.96	1398.68	1683.59
Exp 16	0.52	1.11	0.94	1376.16	1462.88
Exp 17	0.52	1.34	0.96	1245.96	1372.28
Exp 18	0.62	1.34	0.98	1374.08	1342.19
Exp 19	0.73	1.11	0.98	1430.22	1436.25
Exp 20	0.67	1.30	0.98	1339.09	1398.72
Exp 21	0.62	1.30	0.98	1291.57	1349.97
Exp 22	1.06	1.45	1.00	2215.61	1888.75
Exp 23	1.10	1.45	1.00	2234.27	2235.09
Exp 24	0.60	1.34	0.99	1374.08	1683.49
Exp 25	0.76	1.49	0.99	1702.06	1590.90
Exp 26	0.60	1.11	0.96	1398.68	1559.01

Exp 27	0.41	1.42	0.91	1351.76	1488.56
Exp 28	1.05	1.50	0.96	2348.54	2090.52
Exp 29	0.39	1.25	0.93	1027.13	1562.65
Exp 30	0.52	1.30	0.95	1163.75	1189.30
Exp 31	1.11	1.45	0.99	2230.26	1868.85
Exp 32	0.52	1.17	0.96	1048.96	1517.12
Exp 33	0.47	1.08	0.93	1154.40	1211.04
Exp 34	0.65	0.94	0.95	934.505	983.688
Exp 35	0.76	1.34	0.98	1863.16	1616.65
Exp 36	0.52	1.30	0.98	1163.75	1445.45
Exp 37	0.64	1.50	0.98	1569.42	1466.37
Exp 38	0.56	1.30	0.98	1255.59	1378.94
Exp 39	0.67	1.35	0.99	1399.60	1380.76
Exp 40	0.73	1.50	0.99	1673.46	1582.95
Exp 41	1.10	1.24	0.99	1723.02	1724.70
Exp 42	0.62	1.24	0.98	1396.69	1530.37
Exp 43	0.76	0.86	0.94	916.54	1137.55
Exp 44	1.10	1.11	0.96	1437.11	1330.33
Exp 45	0.62	1.50	0.97	1589.56	1583.07
Exp 46	0.52	1.45	0.94	1411.36	1556.86
Exp 47	0.62	1.24	0.96	1396.69	1468.15
Exp 48	1.28	1.30	0.99	1892.78	1755.10
Exp 49	0.25	1.24	0.87	996.20	1450.11
Exp 50	0.64	1.11	0.91	1390.01	1365.57

Experiment results for Multi echelon network optimization:

2500 experiment were carried out and first 100 results are presented in Table II.4. As seen from Table II.4, Exp. 59 provides the minimal (INV/FR) values with ((s_{DC},S_{DC}), (s_s,S_s)) =((0.76,0.86), (0.47,1.08)) parameters.

Table II.4: Experiment results for Multi echelon network

Experiment	DC Reorder ratio	DC Order up to ratio	Store Reorder ratio	Store Order up to ratio	Average fill rate	Average total inventory	On hand inventory	Avg total Inventory/Avg fill rate
Exp 0001	0.52	1.50	0.52	1.50	0.92	4698.71	3683.67	5193.62
Exp 0002	0.52	1.50	0.58	1.45	0.94	5741.89	6641.00	5808.91
Exp 0003	0.52	1.50	0.52	1.11	0.95	4970.20	2806.33	5540.97
Exp 0004	0.52	1.50	0.64	1.50	0.95	5999.76	6248.00	5967.48
Exp 0005	0.52	1.50	0.58	1.23	0.95	5078.58	4166.00	5642.56
Exp 0006	0.52	1.50	1.05	1.24	0.97	5866.58	5408.67	5840.91
Exp 0007	0.52	1.50	1.20	1.47	0.97	6517.72	7177.33	6517.23
Exp 0008	0.52	1.50	0.39	1.25	0.95	5161.43	3046.33	5901.22
Exp 0009	0.52	1.50	0.76	1.45	0.94	5774.57	5354.00	5983.29
Exp 0010	0.52	1.50	0.54	1.46	0.94	5375.02	3613.67	5759.46
Exp 0011	0.52	1.50	0.91	1.11	0.96	5219.57	4056.33	5539.46
Exp 0012	0.52	1.50	0.58	1.24	0.95	5218.02	5503.67	5478.21
Exp 0013	0.52	1.50	1.10	1.45	0.96	6469.67	7113.33	6303.93
Exp 0014	0.52	1.50	0.95	1.45	0.96	5856.75	5971.00	6270.27
Exp 0015	0.52	1.50	0.60	1.11	0.95	4921.16	2581.67	5545.82
Exp 0016	0.52	1.50	0.52	1.34	0.95	5501.83	4342.67	5508.11
Exp 0017	0.52	1.50	0.62	1.34	0.95	5413.93	4899.00	5734.54
Exp 0018	0.52	1.50	0.73	1.11	0.96	5238.26	4056.33	5598.02
Exp 0019	0.52	1.50	0.67	1.30	0.95	5443.54	5411.33	5623.95
Exp 0020	0.52	1.50	0.62	1.30	0.95	5320.19	5465.33	5633.35
Exp 0021	0.52	1.50	1.06	1.45	0.96	6352.89	7113.33	6279.49
Exp 0022	0.52	1.50	0.60	1.34	0.95	5368.62	5574.67	5979.69
Exp 0023	0.52	1.50	0.76	1.49	0.94	5833.90	5722.67	6012.99
Exp 0024	0.52	1.50	0.41	1.42	0.91	4154.53	2473.00	5102.17
Exp 0025	0.52	1.50	1.05	1.50	0.95	6560.70	4986.33	6165.81
Exp 0026	0.52	1.50	0.52	1.30	0.95	5373.42	5069.67	5994.69
Exp 0027	0.52	1.50	1.11	1.45	0.96	6514.68	7113.33	6417.41
Exp 0028	0.52	1.50	0.52	1.17	0.94	5088.22	3125.67	5888.33
Exp 0029	0.52	1.50	0.47	1.08	0.94	4949.82	3062.00	5351.77
Exp 0030	0.52	1.50	0.65	0.94	0.96	4933.82	3533.00	5201.87
Exp 0031	0.52	1.50	0.76	1.34	0.95	5584.49	5574.67	5565.15
Exp 0032	0.52	1.50	0.56	1.30	0.95	5300.64	5465.33	5606.42
Exp 0033	0.52	1.50	0.67	1.35	0.95	5488.94	5133.33	5831.70

Exp 0034	0.52	1.50	0.73	1.50	0.95	5889.70	6256.67	6055.88
Exp 0035	0.52	1.50	1.10	1.24	0.97	5866.58	5408.67	6092.25
Exp 0036	0.52	1.50	0.62	1.24	0.95	5183.43	3316.00	5756.32
Exp 0037	0.52	1.50	0.76	0.86	0.96	4647.01	3591.67	4985.12
Exp 0038	0.52	1.50	1.10	1.11	0.96	5219.57	4056.33	5257.31
Exp 0039	0.52	1.50	0.62	1.50	0.94	5957.13	7225.67	6070.96
Exp 0040	0.52	1.50	0.52	1.45	0.93	5118.70	5391.33	5659.27
Exp 0041	0.52	1.50	1.28	1.30	0.97	6228.18	6111.00	6221.18
Exp 0042	0.52	1.50	0.25	1.24	0.91	4176.62	4926.33	5191.88
Exp 0043	0.52	1.50	0.64	1.11	0.95	4872.60	3141.00	4887.78
Exp 0044	0.52	1.50	0.69	1.08	0.96	5193.25	3939.33	5335.25
Exp 0045	0.52	1.50	0.41	1.42	0.91	4154.53	2473.00	4819.59
Exp 0046	0.52	1.50	0.45	1.50	0.92	4777.21	3870.67	5032.84
Exp 0047	0.52	1.50	0.92	1.25	0.96	5859.36	6473.33	5750.14
Exp 0048	0.52	1.50	1.17	1.39	0.97	6318.26	6921.33	6342.36
Exp 0049	0.52	1.50	1.06	1.45	0.96	6364.83	7113.33	6640.44
Exp 0050	0.52	1.50	1.20	1.47	0.97	6501.70	7177.33	6638.22
Exp 0051	0.58	1.45	0.52	1.50	0.94	5700.95	5650.00	6266.72
Exp 0052	0.58	1.45	0.58	1.45	0.94	5780.86	3519.67	6073.05
Exp 0053	0.58	1.45	0.52	1.11	0.96	4843.60	2704.67	5385.35
Exp 0054	0.58	1.45	0.64	1.50	0.94	5902.87	4290.33	5923.34
Exp 0055	0.58	1.45	0.58	1.23	0.95	5243.51	5657.33	5782.25
Exp 0056	0.58	1.45	1.05	1.24	0.97	5716.36	3064.00	5632.49
Exp 0057	0.58	1.45	1.20	1.47	0.97	6364.70	3725.33	6421.40
Exp 0058	0.58	1.45	0.39	1.25	0.95	5244.40	2504.33	5844.66
Exp 0059	0.76	0.86	0.47	1.08	0.96	3704.64	3986.60	3859.82
Exp 0060	0.58	1.45	0.54	1.46	0.94	5708.32	4726.67	6068.68
Exp 0061	0.58	1.45	0.91	1.11	0.98	5226.83	2697.67	5529.51
Exp 0062	0.58	1.45	0.58	1.24	0.95	5314.53	5484.33	5518.65
Exp 0063	0.58	1.45	1.10	1.45	0.97	6288.91	3661.33	6102.12
Exp 0064	0.58	1.45	0.95	1.45	0.96	6060.45	7166.67	6477.37
Exp 0065	0.58	1.45	0.60	1.11	0.96	4824.41	2584.33	5385.10
Exp 0066	0.58	1.45	0.52	1.34	0.95	5340.48	5663.33	5467.33
Exp 0067	0.58	1.45	0.62	1.34	0.96	5795.43	3792.00	5878.54
Exp 0068	0.58	1.45	0.73	1.11	0.97	5359.22	2723.33	5692.59
Exp 0069	0.58	1.45	0.67	1.30	0.96	5395.45	5663.67	5590.73
Exp 0070	0.58	1.45	0.62	1.30	0.97	5558.33	4438.67	5792.60
Exp 0071	0.58	1.45	1.06	1.45	0.97	6373.27	3661.33	6248.58

Exp 0072	0.58	1.45	0.60	1.34	0.96	5767.09	5263.67	6122.29
Exp 0073	0.58	1.45	0.76	1.49	0.97	5972.79	4013.33	6070.58
Exp 0074	0.58	1.45	0.41	1.42	0.94	5823.99	7264.33	6310.83
Exp 0075	0.58	1.45	1.05	1.50	0.97	6339.97	4430.33	6393.00
Exp 0076	0.58	1.45	0.52	1.30	0.95	5271.59	4660.33	5917.88
Exp 0077	0.58	1.45	1.11	1.45	0.98	6230.62	3661.33	6086.72
Exp 0078	0.58	1.45	0.52	1.17	0.96	4883.23	3232.33	5398.85
Exp 0079	0.58	1.45	0.47	1.08	0.95	4653.35	2822.67	5090.35
Exp 0080	0.58	1.45	0.65	0.94	0.98	4695.90	2519.33	4733.38
Exp 0081	0.58	1.45	0.76	1.34	0.96	5529.56	4815.00	5468.66
Exp 0082	0.58	1.45	0.56	1.30	0.96	5719.03	4618.00	5899.01
Exp 0083	0.58	1.45	0.67	1.35	0.97	5844.51	4320.00	5934.71
Exp 0084	0.58	1.45	0.73	1.50	0.95	6069.61	4653.00	6418.71
Exp 0085	0.58	1.45	1.10	1.24	0.98	5862.85	3012.33	6029.87
Exp 0086	0.58	1.45	0.62	1.24	0.96	5245.47	4063.33	5740.48
Exp 0087	0.58	1.45	0.76	0.86	0.97	4399.56	2491.67	4746.68
Exp 0088	0.58	1.45	1.10	1.11	0.98	5226.83	2697.67	5067.89
Exp 0089	0.58	1.45	0.62	1.50	0.94	5969.51	4645.00	5917.68
Exp 0090	0.58	1.45	0.52	1.45	0.94	5426.28	3994.67	6027.01
Exp 0091	0.58	1.45	1.28	1.30	0.97	5900.88	3215.33	5909.60
Exp 0092	0.58	1.45	0.25	1.24	0.91	4237.47	4373.67	5224.35
Exp 0093	0.58	1.45	0.64	1.11	0.97	4920.02	2571.33	4879.78
Exp 0094	0.58	1.45	0.69	1.08	0.97	5182.06	4708.33	5291.32
Exp 0095	0.58	1.45	0.41	1.42	0.94	5823.99	7264.33	5901.41
Exp 0096	0.58	1.45	0.45	1.50	0.94	6094.73	5028.33	6364.37
Exp 0097	0.58	1.45	0.92	1.25	0.98	5891.29	3067.67	6128.40
Exp 0098	0.58	1.45	1.17	1.39	0.97	6298.95	3472.33	6327.37
Exp 0099	0.58	1.45	1.06	1.45	0.97	6364.26	3661.33	6614.08
Exp 0100	0.58	1.45	1.20	1.47	0.97	6364.70	3725.33	6534.06

Appendix III: SimTalk Program code

Store customer demand generation method:

```
is
   j,NextRow:integer;
   demand:integer;
   obj:object;
do
   @.OrderAmt :=0;
   demand:=0;
   NextRow:=0;
    for j := 1 to Class.Inventory_table.yDim loop
         NextRow:= Class.Demand_table.yDim + 1;
         Class.Demand_table["Item_ID",NextRow]:= Class.Inventory_table["ID",j];
         demand:=z_poisson(2,Class.Inventory_table["Mixture",j]);
         Class.Demand_table["Order_qty",NextRow] := demand;
         @.OrderAmt := @.OrderAmt + demand;
        demand:=0;
   next;
```

Customer order processing method:

```
is
   now:time;InvCosts,HoldingCosts,BacklogCosts:real;
   processedtime : time;
   j,i:integer;
   OrderQty:integer;
   Inventory_total: integer;
do
   for j := 1 to Class.Demand_table.yDim loop
      Class.Inventory_table.CursorY := 1;
      OrderQty :=0;
          if
         Class.Inventory_table.finden(`["ID",1]..`["ID",Class.Inventory_table.yDim],C
         lass.Demand_table["Item_ID",j]) then
          if Class.Inventory_table["Inventory",Class.Inventory_table.CursorY]>=
         Class.Demand_table["Order_qty",j] then
```

```
            @.fillrate := @.fillrate +
          min(1,Class.Inventory_table["Inventory",Class.Inventory_table.CursorY]/Cla
          ss.Demand_table["Order_qty",j]);
           end;
           OrderQty:=
          Class.Inventory_table["Inventory",Class.Inventory_table.CursorY]-
          Class.Demand_table["Order_qty",j];
          if OrderQty <0 then
            OrderQty :=0;
           end;
          Class.Inventory_table["Inventory",Class.Inventory_table.CursorY] :=
OrderQty;
      end;
           Inventory_total := Inventory_total +
          Class.Inventory_table["Inventory",Class.Inventory_table.CursorY];
   next;
  @.fillrate := @.fillrate / Class.Demand_table.yDim;
  Class.Demand_table.delete;

end;
```

Store Order generation method:

```
is
  processedtime : time;
  test : string;
  j,i:integer;
  OrderQty:integer;
  Inventory_total: integer;

 do
      for j := 1 to Class.Demand_table.yDim loop
      Class.Inventory_table.CursorY := 1;
      OrderQty :=0;
      if
Class.Inventory_table.finden(`["ID",1]..`["ID",Class.Inventory_table.yDim],Class.De
mand_table["Item_ID",j]) then
           if Class.Inventory_table["Inventory",Class.Inventory_table.CursorY]>=
Class.Demand_table["Order_qty",j] then
```

```
            @.fillrate := @.fillrate +
min(1,Class.Inventory_table["Inventory",Class.Inventory_table.CursorY]/Class.Dem
and_table["Order_qty",j]);
          end;
         OrderQty:=
Class.Inventory_table["Inventory",Class.Inventory_table.CursorY]-
Class.Demand_table["Order_qty",j];
         if OrderQty <0 then
            OrderQty :=0;
          end;
         Class.Inventory_table["Inventory",Class.Inventory_table.CursorY] :=
OrderQty;
      end;
      Inventory_total := Inventory_total +
Class.Inventory_table["Inventory",Class.Inventory_table.CursorY];
    next;
   @.fillrate := @.fillrate / Class.Demand_table.yDim;
   Class.Demand_table.delete;

end;
```

DC Order Processing method:

```
is
   processedtime : time;
   test : string;
   j,i:integer;
   OrderQty,FulFilledQty :integer;
   Inventory_total,DCInvOnHand: integer;
   store_path : object;
do

   Inventory_total:=0;
   @.fillrate:=0;
   DCInvOnHand:=0;
   Fill_rate := 0;

if  @.OrderLotAmount >0 then
   for j := 1 to Class.DC_Demand_table.yDim loop
       Class.DC_Inventory_table.CursorY := 1;
      OrderQty :=0;
```

```
        FulFilledQty :=0;
        if
Class.DC_Inventory_table.finden(`["ID",1]..`["ID",Class.DC_Inventory_table.yDim],
Class.DC_Demand_table["Item_ID",j]) then
             --if DC_Inventory_table["Inventory",DC_Inventory_table.CursorY]>=
DC_Demand_table["Order_qty",j] then
                @.fillrate :=  @.fillrate +
min(1,Class.DC_Inventory_table["Inventory",Class.DC_Inventory_table.CursorY]/Cl
ass.DC_Demand_table["ReOrdQty",j]);
          --   end;
            OrderQty:=
Class.DC_Inventory_table["Inventory",Class.DC_Inventory_table.CursorY] -
Class.DC_Demand_table["ReOrdQty",j];
            if OrderQty <0 then
               OrderQty :=0;
                FulFilledQty :=
Class.DC_Inventory_table["Inventory",Class.DC_Inventory_table.CursorY];
            else
               FulFilledQty := Class.DC_Demand_table["ReOrdQty",j];
            end;

           .Models.Supply_Chain.Store_Information.CursorY := 1;

            if
.Models.Supply_Chain.Store_Information.finden(`["Store_ID",1]..`["Store_ID",.Mod
els.Supply_Chain.Store_Information.yDim],Class.DC_Demand_table["Store",j]) then

store_path:=.Models.Supply_Chain.Store_Information["Path",.Models.Supply_Chain
.Store_Information.CursorY ];
               store_path.Inventory_table.CursorY := 1;
                  if  store_path.Inventory_table.finden(`["ID",1]..`["ID",
store_path.Inventory_table.yDim],Class.DC_Demand_table["Item_ID",j]) then
                       store_path.Inventory_table["Inventory",
store_path.Inventory_table.CursorY] :=  store_path.Inventory_table["Inventory",
store_path.Inventory_table.CursorY]+ FulFilledQty;
                  end;
             end;

Class.DC_Inventory_table["Inventory",Class.DC_Inventory_table.CursorY] :=
OrderQty;
        end;
        Inventory_total := Inventory_total +
Class.DC_Inventory_table["Inventory",Class.DC_Inventory_table.CursorY];
```

```
     next;
    @.fillrate := @.fillrate / Class.DC_Demand_table.yDim;
    --Fill_rate := @.fillrate;
    Class.DC_Demand_table.delete;

end;
if @.fillrate = 0 then
    Fill_rate := 1;
else
    Fill_rate := @.fillrate;
 end;
for j := 1 to Class.DC_Inventory_table.yDim loop
    DCInvOnHand := DCInvOnHand + Class.DC_Inventory_table["Inventory",j];
next;
.Models.Supply_Chain.Inventory_data.CursorY := 1;
if.Models.Supply_Chain.Inventory_data.finden(`["DC_ID",1]..`["DC_ID",.Models.S
upply_Chain.Inventory_data.yDim],Class.Name) then

.Models.Supply_Chain.Inventory_data["Fill_rate",.Models.Supply_Chain.Inventory_
data.CursorY] := Fill_rate;

.Models.Supply_Chain.Inventory_data["Total_Inventory",.Models.Supply_Chain.Inv
entory_data.CursorY] := DCInvOnHand;
 end;
Total_DC_Inv :=Total_DC_Inv + DCInvOnHand;
count_DC := count_DC+1;
end;
```

Supplier Demand generation method:

```
is
    j,i,NextRow,index:integer;
    DCobj:object;
    SKUlst:list;
    Processing_time,Maxtime,lastevent,now,Maxnow:time;
    Name:string;
do
    lastevent := .Models.Supply_Chain.Main_Frame.EventController.SimTime;
--   @.OrderLotAmount :=0;
--   @.LeadTime := 0;
    NextRow:=0;
    SKUlst.create;
```

```
   Name := Class.Name;
   for j := 1 to .Models.Supply_Chain.DC_Information.yDim loop
      DCobj := .Models.Supply_Chain.DC_Information["Path",j];
       for i:=1 to Class.SKU_Info.yDim loop
         DCobj.DC_Inventory_table.CursorY := 1;
         if
DCobj.DC_Inventory_table.finden(`["ID",1]..`["ID",DCobj.DC_Inventory_table.yDi
m],Class.SKU_Info["SKU_ID",i]) then
             if
DCobj.DC_Inventory_table["Inventory",DCobj.DC_Inventory_table.CursorY] <=
DCobj.DC_Inventory_table["ReOrderPt",DCobj.DC_Inventory_table.CursorY] then
               NextRow:=  Class.Demand_Table.yDim + 1;
                Class.Demand_Table["SKU_ID",NextRow]:=
DCobj.DC_Inventory_table["ID",DCobj.DC_Inventory_table.CursorY];
               Class.Demand_Table["DC_Path",NextRow]:= DCobj;
                Class.Demand_Table["Qty",NextRow] :=
DCobj.DC_Inventory_table["OrderUptoQty",DCobj.DC_Inventory_table.CursorY]-
DCobj.DC_Inventory_table["Inventory",DCobj.DC_Inventory_table.CursorY];
                Processing_time:=
(Class.Demand_Table["Qty",NextRow]*(z_triangle(1,Class.SKU_Info["Avg_Time_T
o_Manfacture",i],Class.SKU_Info["Min_Time_To_Manfacture",i],Class.SKU_Info["
Max_Time_To_Manfacture",i])));

               Class.Demand_Table["Order_Processing_Time",NextRow]
:=Processing_time;
               --@.OrderLotAmount := @.OrderLotAmount +
Class.Demand_Table["Qty",NextRow];
                if i = 1 then
                  Maxtime := Processing_time;
               elseif Maxtime < Processing_time then
                  Maxtime := Processing_time;
               end;
             end;
         end;
      next;
   next;
   now := .Models.Supply_Chain.Main_Frame.EventController.SimTime;
   .Models.Supply_Chain.Supplier_Information.CursorY := 1;
    if
.Models.Supply_Chain.Supplier_Information.finden(`["Supplier_ID",1]..`["Supplier_
ID",.Models.Supply_Chain.Supplier_Information.yDim],Name) then
```

```
.Models.Supply_Chain.Supplier_Information["Lastevent",.Models.Supply_Chain.Sup
plier_Information.cursorY] := lastevent;

.Models.Supply_Chain.Supplier_Information["Maxtime",.Models.Supply_Chain.Sup
plier_Information.cursorY] := Maxtime;
   end;
   for j:=1 to .Models.Supply_Chain.Supplier_Information.yDim loop
      if .Models.Supply_Chain.Supplier_Information["Supplier_ID",j] /= Name and
.Models.Supply_Chain.Supplier_Information["Lastevent",j]=lastevent then
          if j=1 then
            Maxnow := .Models.Supply_Chain.Supplier_Information["Maxtime",1];
         elseif Maxnow < .Models.Supply_Chain.Supplier_Information["Maxtime",j]
then
             Maxnow := .Models.Supply_Chain.Supplier_Information["Maxtime",j]
         end;
      end;
   next;
   if Maxnow >= Maxtime then
      Maxtime := 0;
   else
       Maxtime := Maxtime - Maxnow;
     end;
      wait(time_to_num(Maxtime));
      Class.ProcessOrder(Class);
end;
```

Supplier order processing method:

```
(Class1 : object)
is
   j:integer;
   OrderQty:integer;
   DC_object:object;
   lead_time:time;
do
--   lead_time:= ProcessTime;
   for j := 1 to Class1.Demand_table.yDim loop
       DC_object := Class1.Demand_table["DC_Path",j];
      DC_object.DC_Inventory_table.CursorY := 1;
      OrderQty :=0;
```

```
        if
DC_object.DC_Inventory_table.finden(`["ID",1]..`["ID",DC_object.DC_Inventory_ta
ble.yDim], Class1.Demand_table["SKU_ID",j]) then

DC_object.DC_Inventory_table["Inventory",DC_object.DC_Inventory_table.Cursor
Y] :=
DC_object.DC_Inventory_table["Inventory",DC_object.DC_Inventory_table.Cursor
Y]+Class1.Demand_table["Qty",j];

DC_object.DC_Inventory_table["Leadtime",DC_object.DC_Inventory_table.CursorY
] := Class1.Demand_table["Order_Processing_Time",j];
:=.Models.Supply_Chain.Main_Frame.EventController.SimTime;
         end;
   next;
   Class1.Demand_table.delete;
end;
```

Add Distribution Center method:

```
(action : string)
is
   NextRow,x,y,j:integer;
   obj,obj1:object;


do
   x :=250;
   y := 125;
   inspect action
   when "Open" then
      @.setValue("DC_Name","");
      -- TODO: add code for the "Open" action here
   when "Apply" then
      if
.Models.Supply_Chain.DC_Information.finden(`["DC_ID",1]..`["DC_ID",.Models.S
upply_Chain.DC_Information.yDim],@.getValue("DC_Name")) then
          MessageBox("Distibution Center name already exists.Please enter a different
name.",1,1);
      elseif @.getValue("DC_Name") ="" then
         MessageBox("Distibution Center name is empty.Please enter a Distibution
Center name.",1,1);
      else
```

```
        obj :=.Models.Supply_Chain.DC.derive;
        obj.setName(@.getValue("DC_Name"));
        obj1
:=  obj.createObject(.Models.Supply_Chain.Main_Frame,x,y+.Models.Supply_Chain.
DC_Information.yDim*100);
        NextRow := .Models.Supply_Chain.DC_Information.yDim + 1;
        .Models.Supply_Chain.DC_Information["DC_ID",NextRow]:=@.getValue("DC
_Name") ;
        .Models.Supply_Chain.DC_Information["Path",NextRow]:= obj.RootFrame;
        obj.RootFrame.DC_Inventory_table.inheritFormat:= false;
       for j:=1 to .Models.Supply_Chain.SKU_Table.yDim loop
           obj.RootFrame.DC_Inventory_table["ID",obj.RootFrame.DC_Inventory_table
.yDim+1]:=.Models.Supply_Chain.SKU_Table["SKU_ID",j];
           obj.RootFrame.DC_Inventory_table["Inventory",obj.RootFrame.DC_Inventor
y_table.yDim] :=
obj.RootFrame.DC_Inventory_table[2,obj.RootFrame.DC_Inventory_table.yDim];
        next;
        obj.RootFrame.DC_Inventory_table.OpenDialog;
        .Models.Supply_Chain.Inventory_data["DC_ID",.Models.Supply_Chain.Invento
ry_data.yDim+1]:=@.getValue("DC_Name");
        .Models.Supply_Chain.Inventory_data["Path",.Models.Supply_Chain.Inventory
_data.yDim]:=obj.RootFrame;
        if .Models.Supply_Chain.Supplier_Information.yDim /= 0 then
           for j:=1 to .Models.Supply_Chain.Supplier_Information.yDim loop
              .Models.Supply_Chain.Main_Frame.drawLine(2,450,(125+((j-
1)*100))+10,250,y+((.Models.Supply_Chain.DC_Information.yDim-
1)*100)+10,130,2,0);
           next;
        end;
       end;
    when "Close" then
        -- TODO: add code for the "Close" action here
    end;
end;
```

Add Store method:

```
(action : string)
is
    lst,SKUlst,SelectedSKUlst:list;
    NextRow,x,y,count,n,j,locy:integer;
```

```
   obj,obj1: object;

do

   x :=50;
   y := 125;
   inspect action

   when "Open" then
   @.setValue("Store_Id","");
   lst.create;
   @.setList("DCDropdownlist",lst);
--   SKUlst.create;
--   SelectedSKUlst.create;
--   @.SelectedSKUlst.delete;
   --    @.setList("Selected_SKUlst",@.SelectedSKUlst);

      for j := 1 to .Models.Supply_Chain.DC_Information.yDim loop
         lst.append(.Models.Supply_Chain.DC_Information["DC_ID",j]);
      next;
      @.setList("DCDropdownlist",lst);
      -- TODO: add code for the "Open" action here
   when "Apply" then
      if
.Models.Supply_Chain.Store_Information.finden(`["Store_ID",1]..`["Store_ID",.Mod
els.Supply_Chain.Store_Information.yDim],@.getValue("Store_Id")) then
         MessageBox("Store name already exists.Please enter a different name.",1,1);
      elseif @.getValue("Store_Id") ="" then
         MessageBox("Store name is empty.Please enter a Store name.",1,1);
      else
       NextRow:= .Models.Supply_Chain.Store_Information.yDim + 1;
       obj := .Models.Supply_Chain.Store.derive;
       obj.setName(@.getValue("Store_Id"));
       y:= y+(.Models.Supply_Chain.Store_Information.yDim*100);
       obj1 :=  obj.createObject(.Models.Supply_Chain.Main_Frame,x,y);
       n := @.getIndex("DCDropdownlist");
       locy := 125;
       .Models.Supply_Chain.Main_Frame.drawLine(1,x,y+10,250,locy+((n-
1)*100)+10,166,2,0);

       .Models.Supply_Chain.Store_Information["Store_ID",NextRow]:=
@.getValue("Store_Id");
       .Models.Supply_Chain.Store_Information["DC_ID",NextRow]:=
@.getValue("DCDropdownlist");
       .Models.Supply_Chain.Store_Information["Path",NextRow]:= obj.RootFrame;

       obj.RootFrame.Inventory_table.inheritFormat:= false;
       for j:=1 to .Models.Supply_Chain.SKU_Table.yDim loop
          obj.RootFrame.Inventory_table["ID",obj.RootFrame.Inventory_table.yDim+1
]:=.Models.Supply_Chain.SKU_Table["SKU_ID",j];
       --
    obj.RootFrame.Inventory_table["Inventory",obj.RootFrame.DC_Inventory_table.y
Dim] :=
obj.RootFrame.DC_Inventory_table[2,obj.RootFrame.DC_Inventory_table.yDim];
       next;
       obj.RootFrame.Inventory_table.OpenDialog;
       -- TODO: add code for the "Apply" action here
      end;
   when "Close" then
       -- TODO: add code for the "Close" action here


   end;

end;
```

Add Supplier method:

```
(action : string)
is
   lst,SKUlst,SelectedSKUlst,Updatelst:list;
   NextRow,x,y,count,n,j,no:integer;
   obj,obj1: object;
   SelectedSKU,supplier_name,previous_action : string;
do
   x :=450;
   y := 125;
   inspect action
   when "Open" then
   @.setValue("Supplier_Id","");
   lst.create;
   --SKUlst.create;
   @.SKUlst.delete;
   @.SelectedSKUlst.delete;
   @.setList("Selected_SKUlst",@.SelectedSKUlst);
```

```
        for j := 1 to .Models.Supply_Chain.SKU_Table.yDim loop
          .Models.Supply_Chain.Supplier_Information.CursorY := 1;
          if .Models.Supply_Chain.SKU_Table["Selected",j] = true  then
                -- do nothing
          else
            @.SKUlst.append(.Models.Supply_Chain.SKU_Table["SKU_ID",j]);
          end;
        next;
        @.setList("SKU_list",@.SKUlst);
        if @.SKUlst.Dim = 0 then
          MessageBox("You don't have any Inventory items to select",1,1);
          @.closeDialog;
        end;

    --    previous_action := "Open";

    when "Apply" then
      if @.getValue("Supplier_Id") ="" then
         MessageBox("Supplier name is empty.Please enter a Supplier name.",1,1);

      elseif
.Models.Supply_Chain.Supplier_Information.finden(`["Supplier_ID",1]..`["Supplier_
ID",.Models.Supply_Chain.Supplier_Information.yDim],@.getValue("Supplier_Id"))
then
    --    @.setValue("ErrorMsg","Supplier name already exist.Please enter a different
name");
        MessageBox("Supplier name already exists.Please enter a different name.",1,1);
        @.SKUlst.delete;
        @.SelectedSKUlst.delete;
        @.setList("Selected_SKUlst",@.SelectedSKUlst);

        for j := 1 to .Models.Supply_Chain.SKU_Table.yDim loop
          --.Models.Supply_Chain.Supplier_Information.CursorY := 1;
          if .Models.Supply_Chain.SKU_Table["Selected",j] = true  then
                -- do nothing
          else
            @.SKUlst.append(.Models.Supply_Chain.SKU_Table["SKU_ID",j]);
          end;
        next;
        @.setList("SKU_list",@.SKUlst);


      elseif @.SelectedSKUlst.Dim=0 then
```

```
       MessageBox("Please select SKUs",1,1);

     else
        obj := .Models.Supply_Chain.Supplier.derive;
        obj.setName(@.getValue("Supplier_Id"));
        no :=0;
        supplier_name := "";
   /*    for j:=1 to .Models.Supply_Chain.Supplier_Information.yDim loop

           if (supplier_name /=
.Models.Supply_Chain.Supplier_Information["Supplier_ID",j]) then
              no :=no+1;
              supplier_name :=
.Models.Supply_Chain.Supplier_Information["Supplier_ID",j];
           end;
        next;
           */
        obj1
:=  obj.createObject(.Models.Supply_Chain.Main_Frame,x,y+(.Models.Supply_Chai
n.Supplier_Information.yDim*100));
   --    n := @.getIndex("DCDropdownlist");
        for j:=1 to .Models.Supply_Chain.DC_Information.yDim loop
           .Models.Supply_Chain.Main_Frame.drawLine(2,x,(y+(.Models.Supply_Chai
n.Supplier_Information.yDim*100))+10,250,125+((j-1)*100)+10,130,2,0);
        next;
        obj.RootFrame.SKU_Info.inheritFormat:= false;
   --      obj.RootFrame.SKU_Info.OpenDialog;
        for j:=1 to @.SelectedSKUlst.Dim loop
        NextRow:= obj.RootFrame.SKU_Info.yDim + 1;
   --    obj.RootFrame.SKU_Info["Supplier_ID",NextRow]:=
@.getValue("Supplier_Id");
        obj.RootFrame.SKU_Info["SKU_ID",obj.RootFrame.SKU_Info.yDim +
1]:=@.SelectedSKUlst.read(j) ;
   --    obj.RootFrame.SKU_Info["SKU_ID",obj.RootFrame.SKU_Info.yDim +
1]:="S" ;
        .Models.Supply_Chain.SKU_Table.CursorY := 1;
        if    .Models.Supply_Chain.SKU_Table.finden(`["SKU_ID",1]..`["SKU_ID",.Mo
dels.Supply_Chain.SKU_Table.yDim],@.SelectedSKUlst.read(j)) then
           obj.RootFrame.SKU_Info["Avg_Time_To_Manfacture",obj.RootFrame.SKU_
Info.yDim]:=.Models.Supply_Chain.SKU_Table["Avg_Time_To_Manfacture",.Mode
ls.Supply_Chain.SKU_Table.CursorY];
           obj.RootFrame.SKU_Info["Min_Time_To_Manfacture",obj.RootFrame.SKU
_Info.yDim]:=.Models.Supply_Chain.SKU_Table["Min_Time_To_Manfacture",.Mod
els.Supply_Chain.SKU_Table.CursorY];
            obj.RootFrame.SKU_Info["Max_Time_To_Manfacture",obj.RootFrame.SKU
_Info.yDim]:=.Models.Supply_Chain.SKU_Table["Max_Time_To_Manfacture",.Mo
dels.Supply_Chain.SKU_Table.CursorY];
        end;
        .Models.Supply_Chain.SKU_Table["Selected",.Models.Supply_Chain.SKU_Tab
le.CursorY]:= true;
        next;
        .Models.Supply_Chain.Supplier_Information["Supplier_ID",.Models.Supply_Ch
ain.Supplier_Information.yDim+1] := @.getValue("Supplier_Id");
        .Models.Supply_Chain.Supplier_Information["Path",.Models.Supply_Chain.Sup
plier_Information.yDim]     := obj.RootFrame;

    --    action := "Close";
    end;
        -- TODO: add code for the "Apply" action here
        -- TODO: add code for the "Apply" action here
    when "Close" then
        -- TODO: add code for the "Close" action here
    when "Add_SKU" then
        Updatelst.create;
        SelectedSKU :=@.getValue("SKU_list");
        @.SelectedSKUlst.append(SelectedSKU);
        @.setList("Selected_SKUlst",@.SelectedSKUlst);
        for j:=1 to @.SKUlst.Dim loop
        if @.SKUlst.read(j) = SelectedSKU then
        --    if @.SKUlst.read(j) = SelectedSKU and
@.SelectedSKUlst.find(@.SKUlst.read(j))then
                --do nothing
        else
            if @.SelectedSKUlst.find(@.SKUlst.read(j))then
                --do nothing
            else
                Updatelst.append(@.SKUlst.read(j));
            end;
        end;
        next;
        @.setList("SKU_list",Updatelst);
        Updatelst.delete;
    end;
end;
```

Experiment Manager Control Method

```
(localExp:integer)
is
   j,i: integer;
   DCobj:object;
do
   print "CONFIG Experiment ",localExp;
   for j := 1 to .Models.Supply_Chain.DC_Information.yDim loop
      DCobj := .Models.Supply_Chain.DC_Information["Path",j];
      for i:=1 to DCobj.DC_Inventory_table.yDim loop
         DCobj.DC_Inventory_table["ReOrderPt",i] :=
ceil(DCobj.DC_Inventory_table["InitialInventory",i]*.Models.Supply_Chain.Main_F
rame.Experiment_DC_Reorder_ratio);
      DCobj.DC_Inventory_table["OrderUptoQty",i] :=
ceil(DCobj.DC_Inventory_table["InitialInventory",i]*.Models.Supply_Chain.Main_F
rame.Experiment_DC_OrderUpTo_ratio);
      next;
   next;

end;-- of the method
```