Predictive Capacity Planning Modeling with Tactical and Strategic Applications

by

Michael A. Zeppieri

B.S., Engineering Management
United States Military Academy, 1997

Submitted to the MIT Department of Civil and Environmental Engineering and to the Sloan School of Management in partial fulfillment of the requirements for the degrees of

Master of Science in Civil and Environmental Engineering
and
Master of Business Administration

In conjunction with the Leaders for Manufacturing Program at the
Massachusetts Institute of Technology
June, 2004

Signature of Author____
Department of Civil and Environmental Engineering
Sloan School of Management
May 7, 2004

Certified by____
Stephen C. Graves, Thesis Supervisor
Abraham J. Siegel Professor of Management Science & Engineering Systems

Certified by____
David Simchi-Levi, Thesis Supervisor
Professor of Civil and Environmental Engineering & Engineering Systems

Accepted by____
Margaret Andrews, Executive Director of Master's Program
Sloan School of Management

Accepted by____
Heidi Nepf, Chairman, Departmental Committee on Graduate Studies
Department of Civil and Environmental Engineering

Predictive Capacity Planning Modeling with Tactical and Strategic Applications

by

Michael A. Zeppieri

Submitted to the MIT Department of Civil and Environmental Engineering and to the Sloan School of Management on May 7, 2004 in Partial Fulfillment of the Requirements for the Degrees of Master of Science in Civil and Environmental Engineering and Master of Business Administration

Abstract

The focus of my internship was the development of a predictive capacity planning model to characterize the storage requirements and space utilization for Amazon's Campbellsville (SDF) Fulfillment Center (FC). Amazon currently has a functional model that serves the purpose of capacity planning, but they were looking for something that would provide more insight into their storage requirements against various demand forecasts and time horizons. With a more comprehensive model in place, Amazon would then have the ability to initiate studies, with the intent of optimizing the key parameters of their capacity flow

Amazon utilizes a fairly robust and complex software solution for allocating items to storage locations as it receives shipments from its network of manufacturers and distributors. Amazon designates its capacity storage areas as being Prime, Reserve or Random Stow. Prime storage locations are bins from which workers pick items to fulfill orders. Reserve storage consists of pallet locations from which workers replenish Prime bins. Random Stow is a special case form of storage not taken into consideration for the purposes of my internship. The algorithm that determines the capacity allocation for a particular item is driven by two key parameters.

The first parameter Amazon refers to as Days of Cover, which serves as a cycle and safety stock control variable. The maximum Days of Cover setting dictates the quantity of a particular item allocated to Prime locations, with any remaining items in the shipment being sent to Reserve. The minimum Days of Cover serves as the trigger resulting in a replenishment move from Reserve to Prime.

The second parameter, designated as Velocity to Bin Type Mapping, associates Prime bin locations to item demand in terms of outbound cubic volume per week. Amazon's Campbellsville facility has three tiers of Prime storage: Library Deep, Case Flow and Pallet Prime, with each tier representing a larger physical bin size in terms of capacity. The Velocity to Bin Type mapping parameter essentially sets ranges of demand to each of the bin types, such that items are sent to the bin type matching their demand.

Amazon's capacity constraints are seasonal, with the holiday season posing the greatest challenge in terms of having enough Prime locations to fulfill demand. Amazon currently has no means of predicting the effects on their capacity allocation when operators make adjustments to either Days of Cover or Velocity to Bin Type Mapping. While operators do have some degree of insight into the dynamics these parameters have on capacity allocation, they are unable to optimize capacity utilization. The challenge for my internship was to develop a model that would provide insight into the dynamics driving these system parameters.

The scope of the internship was to convert a demand prediction based on Sales figures into a forecast on capacity storage requirements. The focus of my work centered on providing the logic flow to perform this conversion, driven by a model built using Microsoft Excel and driven by a series of Visual Basic macros. My deliverable to Amazon and subsequent analysis resides entirely in the logic flow development process, and the functional capabilities provided by the model in its completed state.

Thesis Supervisor: Stephen C. Graves
Title: Abraham J. Siegel Professor of Management Science & Engineering Systems

Thesis Supervisor: David Simchi-Levi
Title: Professor of Civil and Environmental Engineering & Engineering Systems

Acknowledgements

I would like to express my gratitude to all who contributed to the success of both this internship and my time with MIT's Leaders for Manufacturing Program. The last two years, and in particular my time with Amazon, was an invaluable learning experience that I will take with me throughout the remainder of my career.

I wish to acknowledge the staff and faculty of the LFM Program, for helping to foster a truly unique and special learning environment. To Don Rosenfield, for being the guiding light of LFM, and for dedicating the time and effort necessary to ensure that each LFM student has the most rewarding experience possible. I also wish to extend my gratitude to Nancy Young and Patti Smith, for their patience and professionalism in coordinating all of our deliverables and administrative requirements while away from campus on internship.

I was extremely pleased in my choices for thesis advisors, Professors Stephen Graves and David Simchi-Levi. Both provided unbelievable resources in terms of facilitating my application of classroom principles in a real world environment. They always took time out of their extremely busy schedules to assist me when required, and both displayed a genuine interest and motivation towards helping me drive towards providing Amazon with the deliverables they expected. Both are an invaluable asset to MIT, and the LFM Program is fortunate to have access to such knowledgeable and professional academics.

I especially want to thank Jeff Wilke of Amazon for helping to develop a working relationship with LFM in providing internship opportunities for the program. His career path and successes as an LFM alumni are an inspiration in terms of what LFM graduates can hope to achieve, and his interest in my internship's progress was especially encouraging.

My thanks also go to Lou Usarzewicz of Amazon, my direct internship supervisor, for continually pushing me to the limits of my capabilities, and making the internship a sometimes humbling but always rewarding experience.

I am eternally grateful to Brent Beabout and Kristin Toth, both LFM alumni and current employees of Amazon. Kristin dedicated a great deal of time and effort towards helping me navigate Amazon's recruiting process, and Brent was always available to assist me in finding the resources I required within the Amazon network.

The staff and management at Amazon's Campbellsville Fulfillment Center were an absolutely amazing group of people to work with. Mike Roth was an invaluable part of the internship process, and I owe much of its success to his contributions. Mo Sayegh also contributed greatly in offering his expertise and knowledge throughout the development of the project. My gratitude also goes out to Rich Lane for taking me under his wing and setting the tone for what would remain a positive work experience with the Amazon personnel in Campbellsville.

To the workers at Campbellsville, for always being eager to give me their insight and input, and for making a stranger in a strange land feel welcome and at home. My thanks especially go out to the fulfillment team for driving me around the facility in their forklifts, patiently waiting as I measured nearly every bin in facility. I also wish to thank the human resources office for helping me with my move to Campbellsville.

I have to thank my classmates for the sole reason that they are the most amazing group of people I have ever had the pleasure of working with. The wealth of knowledge and experience they offer to the LFM Program is absolutely incredible, and many of them offered their guidance and feedback for helping me resolve some of the more difficult analytical aspects of my internship.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

The scope of my internship was to focus on the planning of storage capacity for one of Amazon's Fulfillment Centers (FC) in North America. This storage capacity represents the configuration of warehouse space into storage bins in which Amazon maintains their inventory for the items they sell on their website. Storage capacity in and of itself is not a significant constraint on their operations for a majority of the year. However, capacity becomes critical once they start ramping up into the holiday season, as their demand cycle is very much seasonal.

One of the many challenges of this internship is that the very nature of LFM tends to focus on manufacturing, with a heavy concentration of lean manufacturing principles and operations. While many of these concepts certainly apply to the realm of online retail, Amazon is not a manufacturer in the true sense of the word. Yet the logistical, warehouse and network flow challenges of running an online retail empire certainly mirrors those faced in the world of manufacturing.

The worst case scenario in terms of order fulfillment is to have an item in stock, have a customer place an order for that item, and then not be able to fulfill that order due to the item not being in an accessible location. Amazon refers to this scenario as **FUD**, fillable unfilled demand. In the world of online retail, delayed shipments to customers could potentially lead to lost customers if the order experience does not meet their expectations. While many factors play into this dynamic, effective capacity planning becomes crucial during the holiday season.

Amazon utilizes a fairly robust software solution for managing their capacity, driven by operator defined system parameters. Facility and operations managers set and change these parameters on a routine basis. I will revisit the specifics of these parameters later in the thesis. The scope of the internship developed as a result of the tactical concerns surrounding these parameters, with the introduction of strategic elements as well. In terms of dealing with capacity, the internship developed over two key phases, resulting in my building two deliverables for Amazon. Again, I will cover the specifics of these deliverables in greater detail later in the thesis. The foundation for my work with Amazon remained centered on the tactical and strategic concerns of capacity planning

1.1 Tactical Considerations

Operators control the parameters that drive capacity utilization. However, they set these parameters, or knobs of the system, largely through intuition. There was therefore a need to develop a toolset or model that would enable operators to predict the effects that changes in these parameters would have on their capacity usage. The question of having enough capacity given a certain demand prediction is a concern that is highly relevant to a FC operator. Having the ability to predict the capacity usage prior to making changes to the system parameters would provide them with an invaluable tactical capability. Similarly, having such a model in place would work in conjunction with parallel efforts to better understand and ultimately optimize these system parameters.

FC managers analyze and sometimes change the system parameters on a weekly basis. Under the current framework, these parameters are universal in that they apply to Amazon's full range of available inventory. It is interesting to note that at the current time Amazon does not utilize different parameters for items exhibiting different demand behavior. Amazon would gain a fairly significant tactical capability in having the means to understand the dynamics of these demand parameters considering the impact of demand on how Amazon allocates its storage capacity.

1.2 Strategic Considerations

There was also a strategic concern from those dealing with capacity planning at Amazon's corporate headquarters in Seattle. Their concern was largely long term planning in terms of evaluating Amazon's entire capacity resources. At the strategic level, the question became one of determining if Amazon currently has enough FCs, or the right kind of storage infrastructure at each FC, to handle future demand predictions. With their operations expanding at a phenomenal rate, there is a genuine concern that they may run out of capacity to meet these future demand predictions. This introduces the cost/benefit analysis of building new facilities versus finding a way to better utilize the capacity currently in their network.

Amazon already retains a toolset that deals with these strategic concerns in terms of capacity planning. The demand forecast report that Amazon currently utilizes, known as the **S&OP**, translates aggregate sales figures by Product Line into gross number of outbound units. The current capacity planning model essentially utilizes the S&OP data and translates it into a rough order of magnitude prediction for capacity requirements. While this model is certainly very well developed from an analytical perspective, it lacks an intuitive user interface that makes it very applicable beyond usage by those involved in developing it. There was truly a need to build upon the concept of the model already in place, but with a more user friendly interface and comprehensive analytical framework. Initial discussions also revealed a need for the capability to add features at a later date and interface the new model with other studies and efforts developed in parallel to the capacity model.

1.3 Thesis Outline

The internship developed over two key phases, and while the actually deliverables for each were quite different, they do tie together in that the former did set the groundwork for the latter.

Chapter 2 will give a brief introduction to the key terminology and parameters that define Amazon's capacity infrastructure. As I will be using these terms quite freely throughout the thesis, it is advisable for readers to become familiar with this groundwork.

Chapter 3 will explore my initial approach to the project scope, as defined primarily at the operator level. I will explore the lessons learned from my data mapping experience, and reveal those areas of further study exposed as a result of my analysis of said data.

In Chapter 4, I will discuss how the model scope changed as a result of discussions that better aligned the expectations of both operators and planners. I will then take the reader through the development of the model in establishing the logic flow.

As the model is driven entirely by macros and a robust set of Visual Basic code, Chapter 5 will discuss the architecture I developed to translate the model logic into executable code.

Chapter 6 will take the reader through the final model, explaining how the macros work and operate in relation to one another. It will also serve to demonstrate the transition from developing a model to creating an actual deliverable, with considerations given to user interface execution. This will culminate with application of the model using real world data, the first trial run utilized as a benchmark for validating the model. This trial run helped to expose some of the limitations of the model while also illustrating its wide range of applications. In this chapter, I will discuss some of the mistakes made in developing the model, and go into some detail on an alternative approach to the one produced while I was on internship.

Chapter 7 will discuss in greater detail the functionality of the model, and its potential for application to other areas of interest and fields of study within Amazon.

For Chapter 8, I will discuss the next logical steps for the model's development. These are initiatives that either Amazon can assume internally, or assign to further LFM interns. The true value of the model is in its application to other analytical efforts. This chapter will also serve as a conclusion to the thesis, with reflections on leadership, culture and other elements external to the actual analytical work done on the model. This will hopefully serve as a frame of reference for planning future LFM internships, allowing LFM students to get the most out of their internship experience through Amazon.

Chapter 2: Amazon Terminology

Before continuing a discussion on the specifics of my internship, it is necessary to familiarize the reader to the nuances and terminology of how Amazon manages their FCs from a capacity perspective. These next few pages will therefore take the reader through the learning experience journey I went through during the first few weeks of my internship. Those already familiar with Amazon's operations may choose to skip this section, although it could also be a helpful exercise to understand the perspective of an outsider describing Amazon's structure.

My internship focused on a very specific component of Amazon's supply chain, and as such I will limit my discussion to the boundaries of the system I analyzed. The boundaries of this system includes how Amazon receives items from manufacturers and distributors, manages the storage of those items and ultimately sends those items down the system flow to fulfill an order.

2.1 Amazon Fulfillment Centers

I spent the majority of my time working at one of Amazon's FCs, the term used for the network of warehouse they operate to receive, store and ship merchandise. The interesting thing to note about Amazon's FCs is that they represent a wide range of tactical capabilities. Many of Amazon's FCs are enormous complexes that contain the latest in warehouse management technologies. Product mix in each of their FCs can range anywhere from a specialized mix of specific items to a large mix of nearly all of their website offerings. Together, the FCs operate as a cohesive network, providing Amazon with the ability to ship items to customers across North America in line with their corporate strategy for customer satisfaction.

The FC I worked at was highly automated and contained a fairly broad mix of Amazon's product offerings. The truly amazing aspect of Amazon is that each of their FCs is unique in terms of its infrastructure and operations, while maintaining a foundation of base competencies common to all the FCs.

Amazon differs considerably from traditional brick and mortar retail chains on several different levels. First, the customer facing side of online retail does not reside with the actual items for sale. Rather then browse a series of aisles and shelves, Amazon's customers browse a website to select items for purchase. The advantage of this is that from a capacity perspective, they can conceivably offer as many items as reasonably possible through a web interface. The challenge this creates is that each of Amazon's FCs must work both independently and in unison to provide the network necessary to meet customer expectations, ultimately tying into Amazon's corporate strategy.

An FC is a self contained warehouse that maintains the entire array of processes that ultimately contribute to order fulfillment. The FC has receiving docks at which Amazon receives its daily shipments of items from distributors, vendors, manufacturers, publishers and every other source for their item mix. Once passing through the receiving docks, operators process these items and prepare them for their appropriate storage locations. Forklifts of various sizes and shapes then transport the items to their destinations, usually arranged on pallets.

During the order fulfillment process, items move from the FC storage areas through the warehouse over a vast array of conveyor belts. At the far end of this conveyor system, a fairly advanced automated system sorts items into their appropriate customer orders. Once complete, a customer order is processed for shipping.

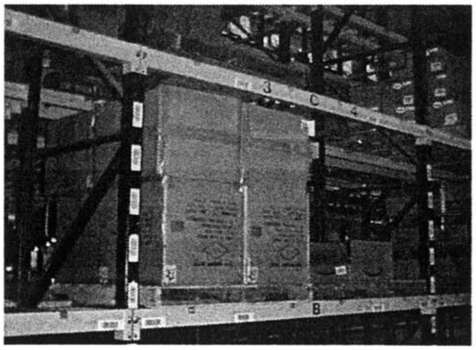
2.2 Storage Types

Considering the range of their product mix, Amazon maintains a fairly simple but effective range of storage types to meet their tactical needs. Storage falls into one of three categories within their FCs: **Prime**, **Reserve** and **Random Stow**. Amazon refers to a unit of storage as a **bin**, with each bin holding a unique item. They refer to each unique item as an **ASIN**: Amazon Stock Identification Number. The ASIN assignment is a bar coded numerical designator that identifies a unique item, similar to an SKU.

The bin system is very straightforward in design. Each bin, regardless of its application, contains a serial number sticker with a bar code. Whenever an employee adds or removes an item from a bin, they scan the bin and the item. This allows Amazon to track all of their inventory with a high degree of fidelity and accuracy.

Prime Bins – These are the bins from which workers actually pull an item to fulfill an order. Prime bins are arranged in an aisle configuration, similar to what one might see in a brick and mortar retail store. They are usually several rows high, linear in configuration, and extend down the length of the FC. The typical configuration is for two rows of prime bins to face inward on a conveyor belt, or located in close proximity to a conveyor belt. Amazon consolidates its Prime storage in multi-leveled structures known as **Pick Mods**. Prime bins fall into one of three categories: **Library Deep**, **Case Flow** and **Pallet**.

Figure 1: Reserve Bins



Reserve Bins - Amazon utilizes these bins essentially to store the majority of their capacity. Reserve bins are in a pallet configuration, built within a metal framework such that each bin can hold one pallet. Reserve Bins exist along aisles that extend down the length of the facility, and are several rows high. Amazon arranges these Reserve aisles along traffic routes such that forklifts can access the items stored in each bin. The ground level row of Reserve storage almost always serves as Pallet Prime, the only exception in what is otherwise Reserve rows. Each pallet contains one and only one ASIN, although it is possible for a particular ASIN to occupy more then one bin. Workers replenish Prime Bins from these Reserve locations, and I will discuss the parameter that triggers a replenishment move shortly.

Random Stow Bins

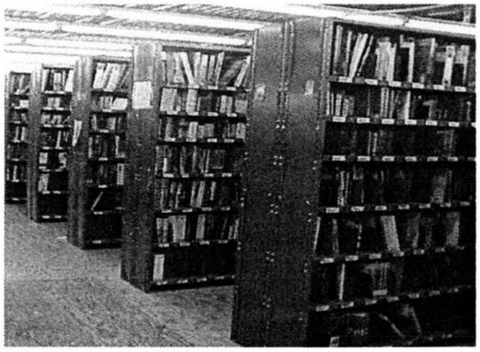
This type of storage is a holdover from when Amazon dealt primarily in books. When Amazon receives a shipment from certain publishers and distributors, the cases are often a mix of different ASINs. The same holds true for DVDs, music CDs and other small sized items. While many of the items in these mixed cases may already exist in other locations, Amazon does not consolidate like ASINs. Instead, they utilize a random storage system in a bookshelf configuration. Like the other bin types, each shelf is split up into numbered bins. The key distinction with Random Stow bins is that instead of the system software designating a specific bin for the item to reside in, stockers simply move down the bookshelf rows with these mixed cases and scan items into any random bin. While Random Stow bins will not factor into my model, the reason that Amazon does not utilize a random storage system for all of its items is that it has a greater potential for user error. Amazon uses a variety of tactics to reduce user error in Random Stow, such as color coding bin rows and instructing operators not to store like ASINs in adjacent bins. However, it is not feasible to implement such a storage hierarchy across the FC for Amazon's entire product offering, particularly when you start contending with larger sized ASINs.

2.3 Prime Bin Types

The majority of my thesis focused on gaining an understanding of the balance between the different Prime bin types, and as such it is necessary to further describe them.

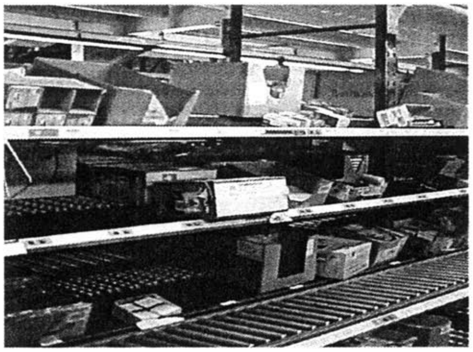
Library Deep bins represent the smallest of the prime bin locations, both in terms of size and total number of available bins. They are metal shelving systems, similar in size and dimension to bookshelves. Workers replenish and pick from the front of these bins. Each bin contains only one ASINs worth of items.

Figure 2: Library Deep Bins



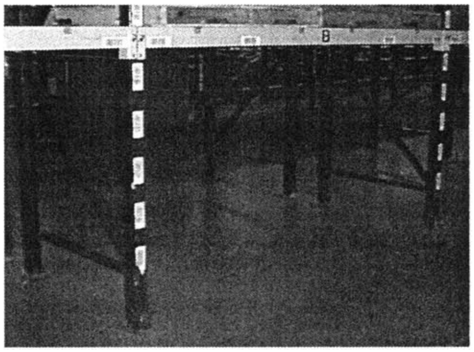
Case Flow bins represent the middle sized Prime bins, and there is a great deal of variability in their dimensions. I will discuss the significance of this variability later in the thesis. Case Flow bins are a metal frame shelving system, with the bottom of each shelf containing rollers for items to slide across. Workers replenish these bins from the rear. As is clearly shown in the photo, Amazon stores larger ASINs as individual items and smaller ASINs in the cases in which they are received.

Figure 3: Case Flow Bins



Pallet bins are the largest of the available Prime bin types. These bin types are familiar to anyone who has ever worked or been in a traditional storage warehouse. Pallet bins are just that, one pallet of homogenous items. Workers pick and replenish from the front of the Pallet bin locations. The picture to the right represents a typical Pallet Prime location in that it is the bottom row ground level of what is otherwise a Reserve stack. Pallet Prime bins also exist in several of the Pick Mods.

Figure 4: Pallet Bins



There also exists a new variation to the Pallet Prime bin type, a Double Pallet Prime bin. This bin is two pallets deep, with the forward bin serving as a pickable location, and the rear bin essentially serving as a pallet of Reserve storage located behind the bin from which operators pick items to fulfill orders.

2.4 The Picking Process

Workers move down the Prime bin aisles and pick items in sequence based on Amazon's extremely sophisticated order fulfillment software. The scanner each worker carries informs them of which bin to move to and how many items to pick from the bin. They then scan the items into a **tote**, a plastic receptacle that workers fill and then send down the conveyor. Once they fill a tote, workers simply scan a new one and continue moving down the aisle picking items. Totes, bins and items all contain bar codes such that Amazon can track item flow from bins to totes, and ultimately to order fulfillment.

Totes hold a random number of items, and do not represent a completed customer order. Each tote holds items for any number of customer orders. At the picking stage of order fulfillment, items are not sorted by customer order. Workers have a series of guidelines on how to best fill each tote, optimizing the use of each tote while staying within the dimensional constraints of the conveyor system.

Workers follow a pick path in that they are constantly forward moving. The system software works such that it notifies workers which bins to pick items from and how many, always in sequence in terms of bins. Workers usually make one lap of a particular Pick Mod floor, move to another floor of the Pick Mod or are notified to move to another Pick Mod. Picking managers can monitor workers and their picking rates, and adjust pick paths accordingly to meet hourly and daily picking goals.

2.5 Product Lines and Sort Types

Amazon consolidates like items in their **Product Lines**. From a web interface perspective, it makes sense that customers should be able to find like items within the same series of web pages. For instance, the Kitchens Product Line contains everything from silverware to appliances. For brick and mortar stores, it makes similar sense to consolidate items by application or customer association. However, for Amazon, their capacity system is not set up such that Product Lines are consolidated as they are on the web page. It is quite possible to have a Kitchen ASIN residing in a bin next to a bin full of books, DVDs, toys or clothing.

This creates an interesting topic for discussion, as consolidating like items is necessary from a web interface perspective but not from a storage perspective. Taking Kitchens again, the cubic dimensions of ASINs within this Product Line vary quite significantly. However this is not true for all Product Lines, as Music CDs, DVDs and other such ASINs do not exhibit nearly as much variance.

Sort Type is a subset of the Product Lines in that the ASINs in each Product Line fall into one of three sort categories: **Sortable, Full Case** and **Non Conveyable**. How the items move and are processed through the receiving docks depends largely on their designated Sort Type. I will now familiarize the reader with each of these Sort Types:

Sortable – Sortable items represent cases of an item that, although the same product, perhaps represent a range of different colors or schemes. This is especially common for Toys, although Amazon receives ASINs for other Product Lines that exhibit similar behavior. Let's say for instance Amazon receives a case of teddy bears. Each teddy bear may be exactly the same except for the fact that half may be black and half may be brown. Through the web interface, customers can select which of the two colors they wish to purchase. Amazon must therefore break down this case, consolidating like colored teddy bears. While the item in and of itself is the same, each color variation receives a different ASIN assignment, and is therefore stored separately. With the exception of Random Stow bins, Amazon always stores homogenous items in the same bin.

Full Case – Like the example above, Full Case items are when Amazon receives a case of items, and that case is already homogenous in terms of its contents. For instance, Amazon may receive one or several cases, perhaps even a pallet load, of one particular book, movie, CD or DVD. They therefore do not have to breakdown this case, but can instead process it for direct movement to its bin assignment.

Non Conveyable – Abbreviated as Non Con, these are items that are simply too large to move or transport on Amazon's conveyor system. An example of a Non Con item is a plasma television. While these items reside in bins the same as other ASINs, you will most always find them in Pallet locations. Rather then travel through the conveyor system, workers manually move these items to the order fulfillment phase of Amazon's supply chain.

2.6 System Parameters

I have made mention of the two system parameters that Amazon utilizes to monitor and control their capacity. I will now explain these parameters in greater detail, known as **Days of Cover** and **Velocity to Bin Type Mapping**.

Before continuing it is necessary to explain the term **Velocity**. Velocity is Amazon's measure of demand for a particular item, and it measures the amount of cubic volume a particular ASIN moves in a given period of time. For the purposes of the model development, I utilized forecasted Velocity predictions as opposed to historical data. Each ASIN has cubic dimensions of (length) x (width) x (height). Velocity simply measures the outbound flow of cube for an ASIN, usually over the span of a week.

Days of Cover – This parameter represents both a safety and cycle stock parameter for Amazon's inventory management. It actually exists as two values, the Min Days of Cover and the Max Days of Cover. The Max Days of Cover determines the split of inventory between Prime and Reserve storage. When Amazon receives a shipment of items, they will move enough of that item to Prime bins such that it meets the Max Days of Cover in terms of demand. The remainder of the shipment is sent to Reserve storage. Depending on the size of the shipment for a particular ASIN and the Max Days of Cover setting, these items could occupy one or several Prime bins. Similarly, the remainder of the shipment could occupy one or several Reserve bins. In some instances, an ASIN will receive an entirely new bin assignment if that particular ASIN does not already reside in storage, or the bins that ASIN already occupies are full. In other instances, Amazon will consolidate these items in a bin that the particular ASIN already occupies.

Over the course of time, the number of items in a Prime bin will deplete. When the Prime bin inventory depletes such that there is only enough to meet the Min Days of Cover worth of demand, it triggers a replenishment move from Reserve to Prime storage. The replenishment move will fill the bin back to its Max Days of Cover inventory.

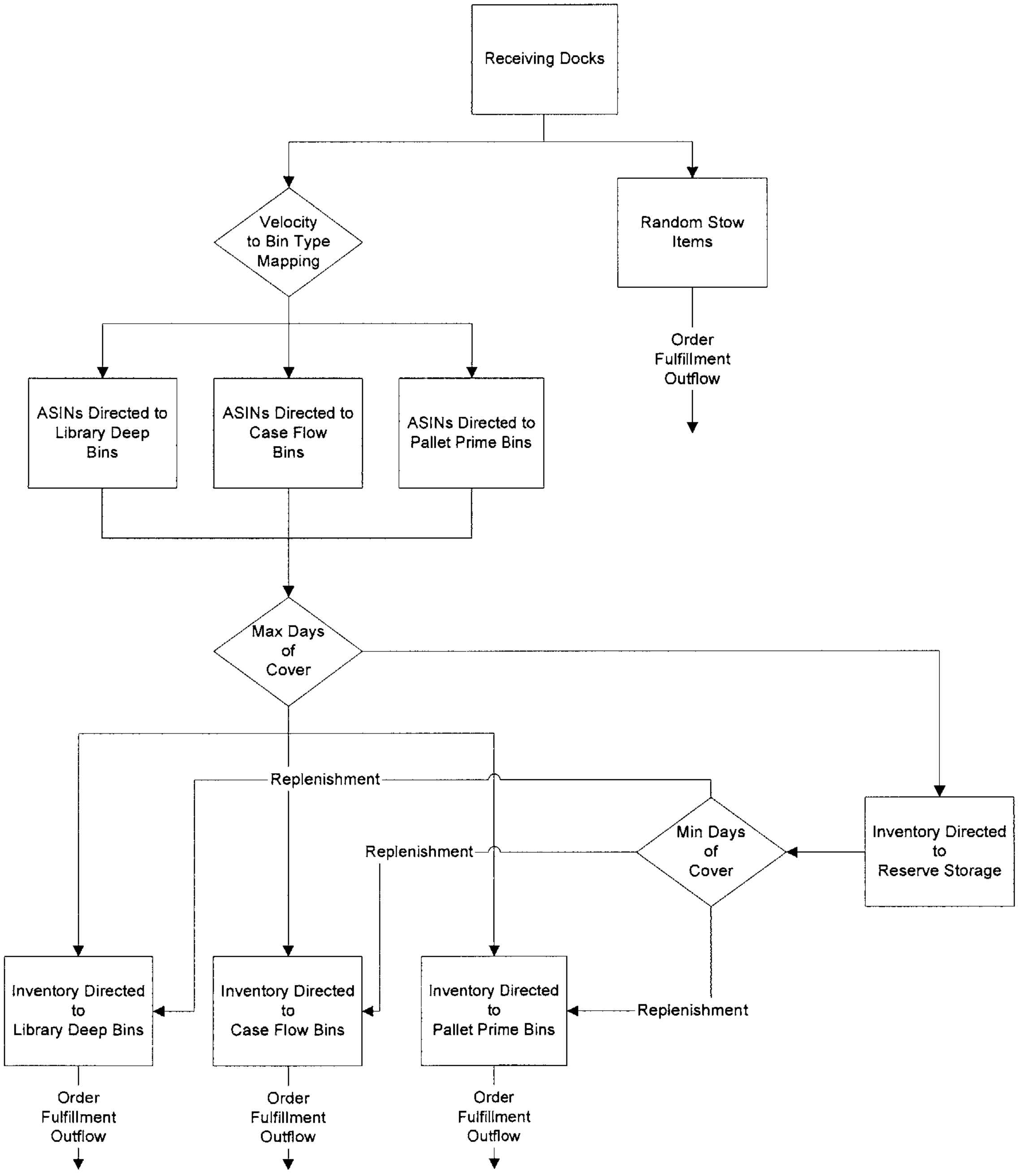
Therefore, the amount of inventory that varies between the Max and Min Days of Cover serves as a cycle stock for the system, with the Min Days of Cover inventory serving as the reorder point that triggers the replenishment move of inventory from Reserve to Prime. The amount of inventory maintained in the Prime bins based on the Min Days of Cover essentially acts as a safety stock for the Prime bins. For the purposes of our discussion, we will represent Days of Cover as the average between the Min and Max Days of Cover. This will adequately represent the average amount of coverage in a particular Prime bin over the course of its cycle stock depletion and subsequent replenishment.

Velocity to Bin Type Mapping – With this parameter, Amazon operators guide particular ASINs to a specific Prime Bin type depending on their Velocity. Operators will set Velocity ranges for each of the Prime bin types, and items that fall within those ranges are automatically routed to that particular bin type. So for instance any ASINs exhibiting a velocity of 0 to 1000 in^3/week are set to Library Deep bins; 1000 to 2000 in^3/week are sent to Case Flow; and everything greater then 2000 in^3/week are sent to Pallet Prime bins. This concept may seem a bit vague at this stage of the thesis, but I will expand upon its significance once I start getting into a discussion on how I developed the model.

2.7 Fitting it All Together

As a frame of reference, the following diagram should help the reader better understand how the parameters fit into the framework of Amazon's system as defined in this chapter:

Figure 5: Amazon Receiving and Capacity Allocation Reference Diagram


Receiving Docks
Velocity to Bin Type Mapping
Random Stow Items
Order Fulfillment Outflow
ASINs Directed to Library Deep Bins
ASINs Directed to Case Flow Bins
ASINs Directed to Pallet Prime Bins
Max Days of Cover
Replenishment
Replenishment
Min Days of Cover
Inventory Directed to Reserve Storage
Inventory Directed to Library Deep Bins
Inventory Directed to Case Flow Bins
Inventory Directed to Pallet Prime Bins
Replenishment
Order Fulfillment Outflow
Order Fulfillment Outflow
Order Fulfillment Outflow

Chapter 3: Initial Approach to the Model

The initial approach to the model deviated quite significantly from the final product. When I arrived at my FC assignment, the management team there was already working on a data mapping of the entire facility. The motivating factors driving this effort was a need to verify the virtual database, to ensure that it represented the capacity they actually had on hand in the facility.

One of the operators at the FC was already working on this data mapping effort, and the initial scope of the project was to continue and improve upon his work. The expectation at this time was focused quite narrowly on the tactical benefits of having such a data mapping on hand. My concept of the project scope was that the data collected through mapping the facility would serve as the groundwork for my analysis and thesis.

3.1 A Bin Mapping Approach

The operators at the FC to which I was assigned already had a data mapping model in development. From the standpoint of their day to day responsibilities, there was a vested interest in having an up to data mapping of their bin capacity. This essentially entailed the process of going row by row in the facility and measuring bins, also taking note of their type and functional use. As later chapters in the thesis will reveal, this was a somewhat limited approach to providing a robust solution for dealing with capacity, but at this stage, we had yet to articulate the scope of the project to take into consideration other factors. While I will not spend a great deal of time discussing this initial approach, it is relevant to document this work as it represents the focus of my efforts for the first two months of my internship, with later work partially influenced by these findings.

3.2 Initial Findings

One of the benefits of initiating the internship with this approach is that it gave me a more comprehensive understanding of Amazon's capacity snapshot. I was able to make several observations during the two months it took to develop the capacity mapping spreadsheets of the FC.

1. **Bin Utilization** – One issue that became readily apparent to me is the efficiency by which Amazon utilizes their capacity. On more then one occasion, I would observe a fairly large bin almost entirely empty with the exception of a few items. Even more striking is that this dynamic would remain over the course of several days. As previously explained, Days of Cover settings trigger replenishment moves, and one can assume that the bins I observed were either close to their replenishment point, or setup to completely deplete. During the summer months, this was not as much of a concern, as capacity was plentiful. However, it did beg the question of the importance of bin utilization as Amazon strains their capacity profile during the holiday season.

   There is definitely a tradeoff between the number of replenishment moves Amazon is capable of supporting and bin utilization. After raising this issue with my contacts in

Seattle, I learned that other teams were already focusing on this issue, particularly in the realm of space management. While this was certainly an intriguing problem to evaluate, because others were already focused on it, the scope of my internship would shift to other efforts.

2. **Bin Variability** – Another interesting dynamic is the actual configuration of the bins. While the next section will deal with this issue in greater detail, the key point to take away is the physical construction of the bins. Amazon's bins exhibit a great deal of variability in size within a particular bin type. What is more striking is that this variability is largely arbitrary, and not designed as a constructive input to their capacity management system.

   The resources required to reconfigure the bin construction in any FC is far more prohibitive then the benefits Amazon might gain in eliminating this variation. However, at the strategic level, when taking into consideration the option of building new FCs, bin variability becomes highly relevant. There is some value in conducting a study to determine the ideal dimensions for each bin type such that Amazon can optimize their bin utilization.

   This also raises the issue of the actual bin types currently in service. As I mentioned previously, Amazon utilizes a three tiered bin size system for their capacity management in Prime storage. The question does remain as to the functionality of this tiered system, also suggesting that Amazon requires the capability to evaluate "what if" scenarios in introducing new bin type configurations to the current slating of available bin types.

3.3 Bin Variability

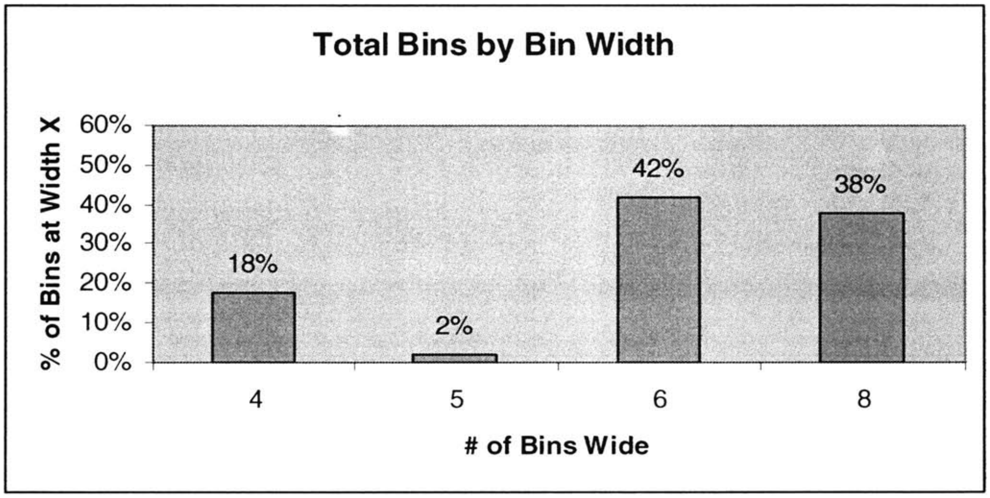
The variability in each of the bin types is driven by entirely different factors, and as such, each requires their own explanation:

Library Deep – There is very little variation in this bin type as they are structurally very similar to bookshelves. The depth and width of these bins remain constant across the entirety of the FC. The only variation exists in height, as the lower two bins are marginally taller then the upper two bins. While this variability does not seem to serve a purpose, it is insignificant and therefore negligible for the purposes of this discussion.

Pallet - Pallet bins, whether they are Prime or Reserve, represent an interesting dilemma in terms of variability. Regardless of the size of the bin, the actual utilization of the bin is limited to the amount of inventory you can physically fit on a pallet. There is also a safety factor for pallets such that operators cannot stack items beyond a certain height on the pallet. Pallet locations also require a safety buffer around the pallet such that forklift operators have room to maneuver in placing and removing pallets from the bin locations. It is interesting to note that in my inquiries, forklift operators and section managers were unable to agree on the exact measurements of these safety factors. I would therefore recommend that Amazon conduct a study to standardize these measurements, and use that as the frame of reference to determine the ideal size for Pallet bin locations. The current setup does exhibit some degree of arbitrary variability, but the significance of this becomes more relevant once you determine the space automatically lost to the safety factors and buffers previously discussed. The utilization of Pallet locations therefore becomes a comparison of the ratio of space automatically lost to these safety buffers and the physical limitations of what you can fit on a pallet.

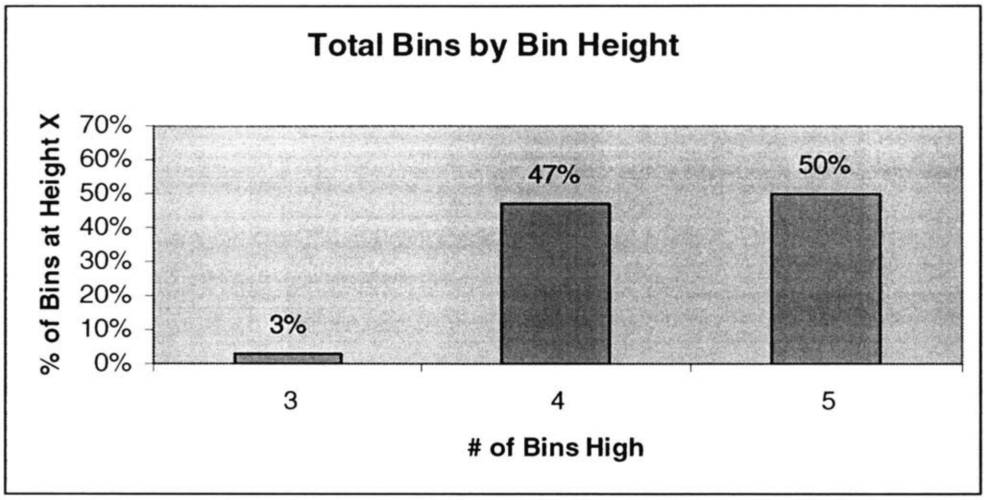
Case Flow - Case Flow bins require additional discussion as it is the only bin type that exhibits a high degree of variability. The following plots will give us some insight on how to best aggregate Case Flow bins into a more workable series of types. Due to concerns for shielding proprietary information, the data sets driving these charts are either normalized or masked. However, without the specific data point values embedded in the charts, we can still gain an understanding of the bin variability dilemma through graphical representation.

Figure 6: Bin Variability, Total Bins by Bin Width



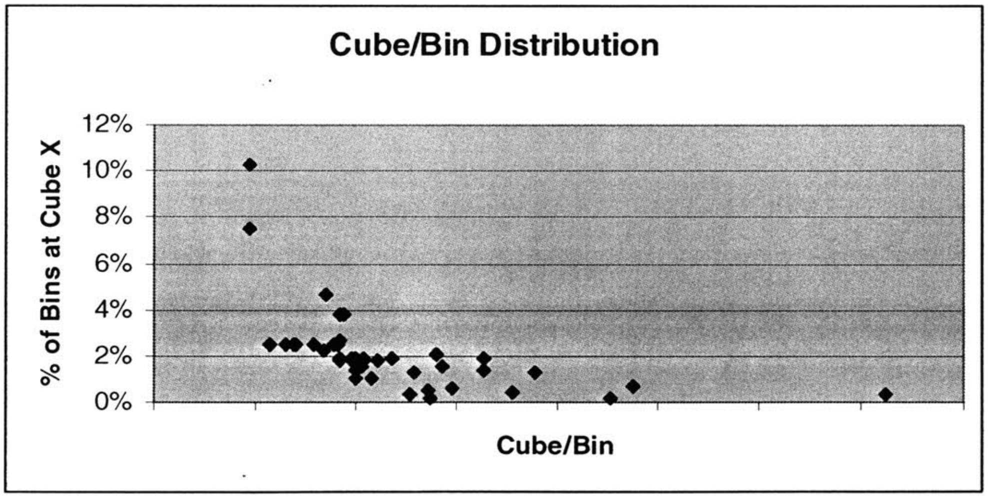
The first chart shows total number of bins by bin width. A Case Flow rack can be 4, 5, 6 or 8 bins wide. Something else to take into consideration is that for each of these widths, Case Flow bins also exhibit a high degree of variation in their height. Case Flow bins can take on a range of variable heights, which is illustrated in the next chart.

Figure 7: Bin Variability, Total Bins by Bin Height



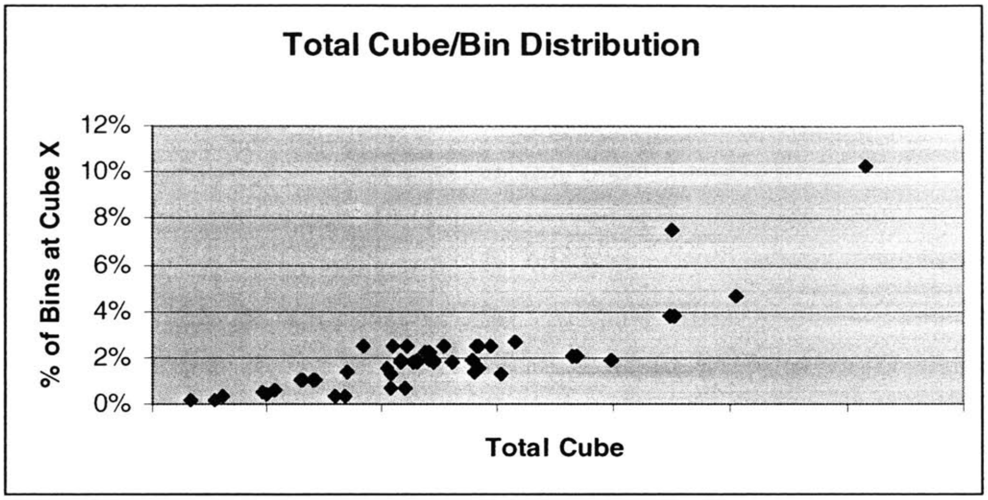
This second chart illustrates that Case Flow bins can also assume three different height configurations. So it is possible for a Case Flow rack of 4 bins wide to be 3, 4 or 5 bins high. It is this variable combination of bin height and width that causes the dilemma in contending with Case Flow bins. Unlike the other bin types, utilizing an average cube dimensional assumption will introduce a significant amount of error into the model.

Figure 8: Bin Variability, Cube/Bin Distribution



This next chart illustrates just how significant this variability is. What this chart shows is the total number of bins by a particular bin dimensional configuration. Each point on this plot represents a potential different Case Flow bin type. However, the distribution of these bin types as a function of just how many there are of that particular type shows that it may be possible to aggregate like dimensions.

Figure 9: Bin Variability, Total Cube/Bin Distribution



This final plot is similar to the previous, except that it captures the total available cube presented by each dimensional combination. The previous plot captured the cube/bin distribution. This plot provides additional insight as it shows the total cube represented and total number of bins providing said cube.

These charts are not very helpful in truly understanding the variability in Case Flow bins, so the next analysis I conducted was to determine the average cube per configuration, and its standard deviation. As it turns out, we have a total of eight variations on the configuration of the possible 12 combinations. For proprietary information purposes, I cannot show the bin count or cube dimension data values driving this analysis. However, the data points tabulated below represent the Standard Deviation in each Case Flow bin configuration as a percentage of the total Average Cube per Bin.

Table 1: Case Flow Deviation as a % of Cube

Rows High	Bins Wide	Deviation % of Cube
3	4	23%
4	4	16%
4	6	30%
4	8	21%
5	4	35%
5	5	39%
5	6	26%
5	8	28%

I do not believe any of the variability is negligible for each of the possible Case Flow configurations. For the purposes of the model, I decided to use a representative average cubic dimension based on the aggregation of bin configurations. However, the variation is significant enough to warrant further discussion.

Another helpful study is to determine what is driving the deviation. Is it primarily variation in the height, depth or width dimensions of each bin configuration? The following table captures the driving dimension for each configuration:

Table 2: Sources of Case Flow Deviation

Rows High	Bins Wide	Average Width (X)	Width Deviation	Average Height (Y)	Height Deviation	Average Depth (Z)	Depth Deviation
3	4	23.63	0	24.88	5.61	102.00	0
4	4	18.56	0	17.17	2.83	73.00	0
4	6	15.88	0	18.43	4.62	81.24	11.22
4	8	10.50	0	16.94	1.98	78.88	7.75
5	4	23.88	0	14.17	3.51	80.17	8.95
5	5	18.90	0	14.50	5.66	102.00	0
5	6	15.80	0.067	13.93	3.69	102.00	0
5	8	11.94	0	14.58	3.23	80.17	8.00

As this table shows, height and depth are driving the variation in the bin dimensions. We can account for the height deviation in that for a particular configuration, the bin frames are not spaced symmetrically. This creates deviation in the bin height as the actual shelves are not spaced evenly along the vertical frame, with this variability being entirely arbitrary. As for the depth variability, in some Pick Mods, but not all, the bottom row of bins is longer then the rows above it. So for instance a 4-6 configuration, the bottom row is longer then the other three rows

3.4 Limitations of the Initial Approach

It should be fairly obvious at this point that while the initial approach was certainly a useful exercise, it has limited applicability. A meeting in Seattle at the end of July confirmed that from the strategic perspective, I would have to develop a more robust capacity solution. A follow-up meeting between operators at the FC and decision makers from Seattle allowed everyone to better define their expectations and requirements such that the model I delivered to Amazon began to take shape.

My interaction for the first few weeks was fairly exclusive to the operators at the FC to which I was assigned. Their concept of my contributions was more on an optimization of bin size and utilization. While the data mapping approach was certainly the correct solution for conducting that study, strategic planners in Seattle were more interested in the dynamics for Days of Cover and Velocity to Bin Type Mapping.

The visit to Seattle in July was quite useful in that I learned of other teams within Amazon who were focusing on the bin size concerns of the FC operators. Passing on my data mapping spreadsheets to those teams assisted in their efforts, and led to conversations and meetings that better coordinated the different studies going on within Amazon. The need to develop a solution that could integrate with these other efforts also became readily apparent.

For those interested in seeing snapshots of my bin mapping effort, I included an example of each layer of my deliverable to Amazon in Appendix A. This effort may prove to be a valuable template for LFM students faced with similar data mapping or organizational efforts, particularly those that have a physical layout component in addition to the actual data set.

Chapter 4: Developing the Model

Certain aspects of the model were somewhat self explanatory in their relevance to the development process. It was extremely important to all involved with the project that the final model be as automated and user friendly as possible. Far too often, developers will provide a model that is highly sophisticated, yet remains on a server somewhere because the interface is not particularly intuitive. As Amazon currently has several studies going on that run parallel to the scope of my project, it became necessary to ensure that the model have some degree of scalability. Above all else, the most important feature of the model was to provide operators with a predictive degree of control over setting Days of Cover and Velocity to Bin Type Mapping. The challenge of the model development therefore became to isolate these two variables, taking into consideration the logic flow that extends from their settings.

The model development process began in earnest after a meeting in late August that consisted of my thesis advisors, FC managers and the internship supervisor. It was extremely important that we get both operators at the FC level and planners at the corporate level on board for supporting the model approach. The conclusion we came to was that it would be possible to develop an approach that would serve as a functional tool to the FC operators, and have applicability to studies and efforts across the entire breadth of Amazon's operations.

4.1 System Inputs and Outputs

One of the challenges of developing this model was the limitations of Amazon's demand forecasting techniques. Given the seasonality of their demand cycles, and the fact that they derive the S&OP from Sales figures, it became difficult to translate fairly abstract demand data into bin usage requirements.

To utilize Days of Cover and Velocity to Bin Type Mapping as variables in the model, we needed to discuss demand in terms of outbound cubic volume per week. The S&OP reflects demand as outbound units per week. I therefore had to develop a scheme for translating the S&OP forecast into a Velocity distribution forecast from which I could then determine the bin usage requirements.

I identified the boundaries of this analytical framework, separating the inputs and outputs into several key categories:

Sort Data – Character field by which to sort the data into workable subsets.
Source Data – Constants or system inputs that remain unchanged for each iteration of the model, but that may change from one iteration to the next.
Decision Variables – The variables that drive the model calculation flow.

Table 3: Sort Data, Source Data and Decision Variables

Sort Data	
Product Lines	The Product Line categories into which Amazon organizes their ASIN inventory.
Sort Type	The category subset that classifies ASINs within each Product Line.
Bin Types	The current bin types present in the FC for a given iteration of the model.
Source Data	
# of ASINs	The number of unique ASINs within each Product Line and Sort Type
AVG Cube/Unit	The average cubic dimensions of items by Product Line and Sort Type.
Standard Deviation	The Standard Deviation around the AVG Cube/Unit
S&OP Outbound Demand	The number of outbound units per week by Product Line and Sort Type. This is the demand figure driving the model
S&OP Inventory	The total number of units per week in the FC by Product Line and Sort Type.
Total # Bins	The total number of each bin type in the FC.
Cube/Bin	The average size of each bin by bin type. For the purposes of this model, I neglected the variability considerations previously mentioned, and utilized the average size of each bin type for the model.
Decision Variables	
Min Velocity	The minimum range of the Velocity to Bin Type Mapping setting; set by bin type.
Max Velocity	The maximum range of the Velocity to Bin Type Mapping setting; set by bin type.
Days of Cover	The AVG Days of Cover setting by Product Line.

At this point, I started to develop a feel for the model calculation flow. My model would have to take into consideration all of the variables identified in the previous table, and determine the number of bins by bin type that each Product Line/Sort Type would require given the Days of Cover and Velocity to Bin Type Mapping settings for that iteration. The value to this approach is that it would allow operators to modify both the Days of Cover and Velocity to Bin Type Mapping settings for each iteration, thereby exposing the dynamics of each in terms of how they effect and drive bin usage requirements.

4.2 Key Variable Relationships

There were two parallel efforts that initiated once I got a feel for the calculation flow for the model. One was the architecture and approach to the model, which I will cover in the next chapter. The other issue was translating the calculation flow into actual relationships between the variables. Going back to the discussion on S&OP, there still remained the challenge of converting a demand forecast in terms of outbound units per week into an outbound Velocity forecast in terms of cubic space per week. For the purposes of the model, I decided to use in^3/week as the unit for Velocity. The conversion at its most basic level works as follows:

$$(\text{AVG Cube } (\text{in}^3) \,/\, \text{ASIN unit})\ (\text{outbound ASIN units} \,/\, \text{week}) = \text{outbound in}^3 \,/\, \text{week}$$

When you are dealing with historical data, or specific information about the behavior of each of the Product Lines, you can very easily extract the AVG Cube/ASIN unit and outbound ASIN units/week down to the specific ASIN level. However, the S&OP demand forecasts do not provide the fidelity for this level of detail, and it therefore became necessary to find a way to replicate the behavior of each of the Product Lines using forecasted distributions.

For the AVG Cube/ASIN unit, it required that I find a way to replicate the size variation in ASIN units down to the Product Line/Sort Type. Similarly for the outbound ASIN units/week, I had to replicate the outbound demand down to the Product Line/Sort Type level. I decided that we can normalize the former distribution and utilize a Pareto Distribution to replicate the latter. Combining these two distributions together would then provide groupings of hypothetical ASINs and their respective outbound Velocity, down to the Product Line/Sort Type level. Subsequent sections in this chapter will cover the specifics of how those calculations work.

With a distribution for outbound Velocity, I could then sort and filter the hypothetical ASIN clusters to their bin types based on the Velocity to Bin Type Mapping settings for the model. Similarly, the AVG Days of Cover would enable me to determine the cube directed to Prime storage. With these two pieces of information in place, I could finally calculate a prediction for the bin usage by bin type given the S&OP outbound forecasts. With the basic logic in place, the next step was to map these relationships and then articulate the specifics of the calculations. The remainder of this chapter will cover those topics.

At this point it is necessary to introduce the concept of distribution slices. To replicate the behavior of these hypothetical ASINs, we will cut the distributions into manageable intervals, or **slices**, to conduct the analysis. Users will be able to set the number of slices for each distribution, with more slices adding greater fidelity with the tradeoff being that the model will run slower.

4.3 The Calculation Flow

Having identified the basic calculation flow, it was then appropriate to plot the logic flow in the form of a chart. While this logic flow chart went through several iterations and changes over the course of the model's development, the diagram below reflects the logic as it appears in the final model.

If you go to Appendix B, you can find the previous iterations to the logic flow diagrams. One thing you will notice is that the interpretation of how to implement the distribution approach changed significantly. This was largely because working with the distributions posed certain challenges that did not expose themselves until I actually began attempting to code the logic.

Another obvious exclusion from previous iterations is the removal of the Pipe Drag Coefficient for calculating the number of available bins by bin type. Throughout the development of the model, we recognized that certain dynamics would result in the model giving somewhat different output then observed in real life. The challenge was to recognize or somehow implement some of these dynamics, but the problem was that many of the studies going on in parallel to my model were still attempting to understand these dynamics. In designing the model, I left certain placeholders in the code such that further studies would enable Amazon to introduce coefficients that could replicate these other dynamics, such as seasonality, pipe drag and bin utilization.

For the purposes of this model, we decided to develop it open ended enough such that users could implement these dynamics at a later time, and I focused on keeping the first version of my model as simple as possible. Later sections of the thesis will cover these assumptions and dynamics in greater detail.

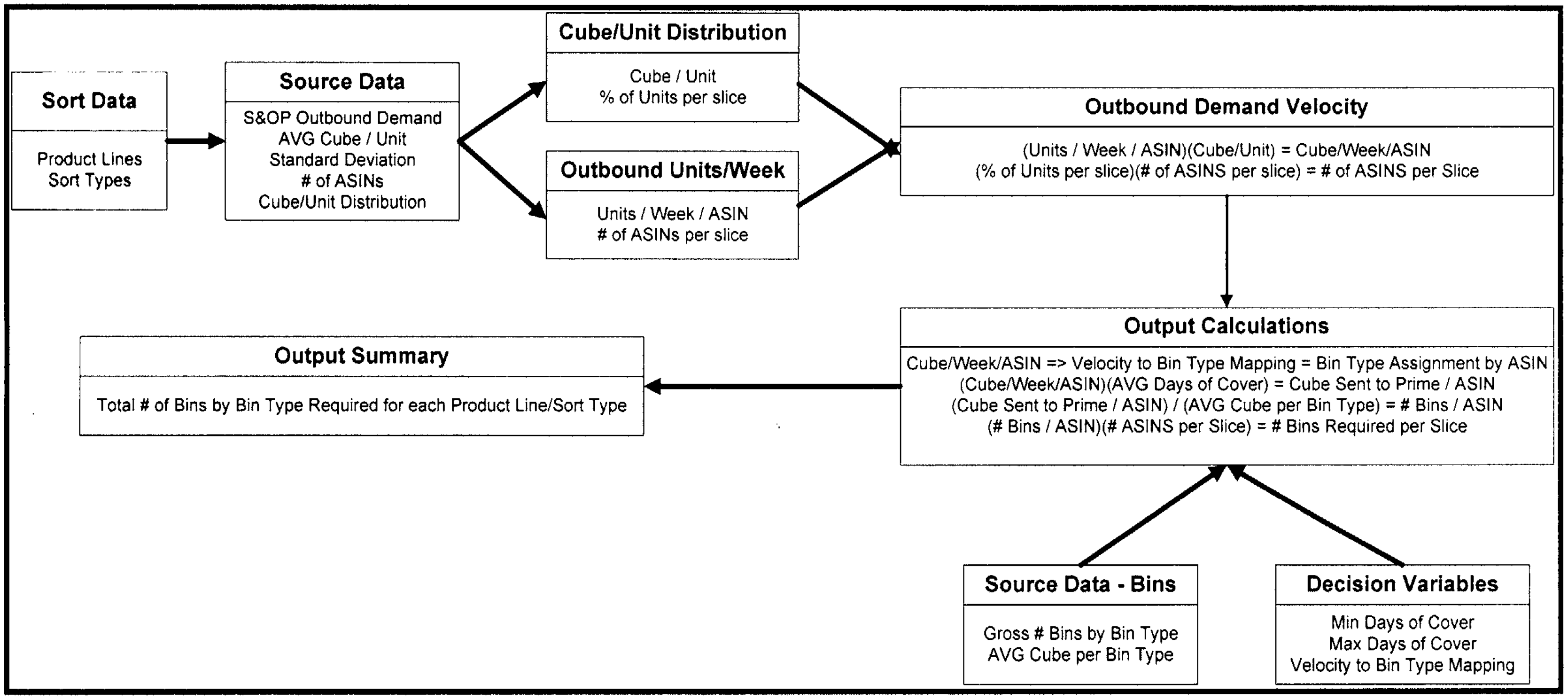

Sort Data
Product Lines
Sort Types
Source Data
S&OP Outbound Demand
AVG Cube / Unit
Standard Deviation
# of ASINs
Cube/Unit Distribution
Cube/Unit Distribution
Cube / Unit
% of Units per slice
Outbound Units/Week
Units / Week / ASIN
# of ASINs per slice
Outbound Demand Velocity
(Units / Week / ASIN)(Cube/Unit) = Cube/Week/ASIN
(% of Units per slice)(# of ASINS per slice) = # of ASINS per Slice
Output Calculations
Cube/Week/ASIN => Velocity to Bin Type Mapping = Bin Type Assignment by ASIN
(Cube/Week/ASIN)(AVG Days of Cover) = Cube Sent to Prime / ASIN
(Cube Sent to Prime / ASIN) / (AVG Cube per Bin Type) = # Bins / ASIN
(# Bins / ASIN)(# ASINS per Slice) = # Bins Required per Slice
Output Summary
Total # of Bins by Bin Type Required for each Product Line/Sort Type
Source Data - Bins
Gross # Bins by Bin Type
AVG Cube per Bin Type
Decision Variables
Min Days of Cover
Max Days of Cover
Velocity to Bin Type Mapping

Figure 10: Model Logic Flow

4.4 Normalizing the Cube/Unit

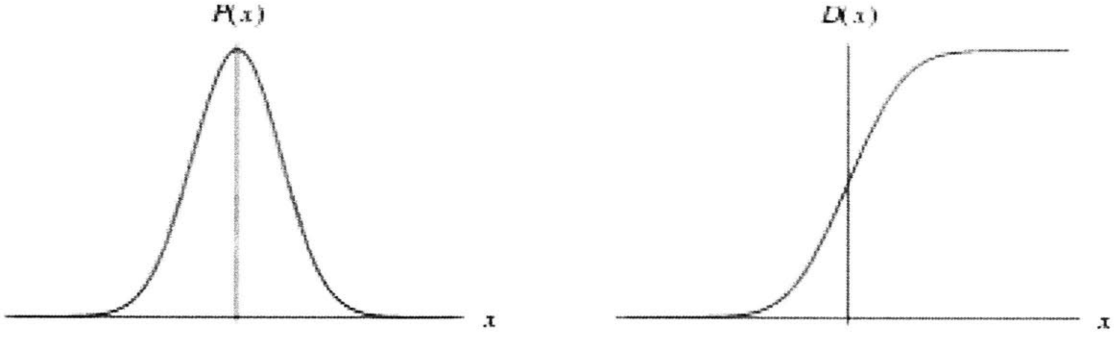
One of the two critical components for the model's approach is representing the Cube/Unit distribution for each of the Product Lines/Sort Types. When dealing with individual units for a particular ASIN, we find there are two scenarios that drive the size of the unit as reflected in the capacity allocation software. For adequately large sized items, the cubic dimensions of each individual item are the physical dimensions of its actual packaging. Examples of these items include printers, computer monitors, large appliances and other electronics. Amazon receives smaller items, such as books, CDs, DVDs and video games, in cases. In many instances, these items remain in the cases in which they were received when moved to either Prime or Reserve bins.

I decided that the best way to replicate the Cube/Unit size distribution across different Product Lines/Sort Types was through a Normal Distribution. For some Product Lines, items are of nearly equivalent size, and the standard deviation around the mean cubic dimension is rather small. For other Product Lines, variation in size of ASINs is quite pronounced, but we can still represent this variability by normalizing it and placing less weight on the outliers.

"We define the normal distribution in a variate x with a mean μ and variance σ^2 has probability function:" [1]

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)}$$

Figure 11: Standard Normal Distribution [1]



"The so-called 'standard normal distribution' is given by taking $\mu = 0$ and $\sigma^2 = 1$ in a general normal distribution. An arbitrary normal distribution can be converted to a standard normal distribution by changing variables to $Z = (x-\mu)/\sigma$, so $dz = dx/\sigma$, yielding:" [1]

[1] Weisstein, Eric W. *Normal Distribution*. From MathWorld – A Wolfram Web Resource. http://mathworld.wolfram.com/NormalDistribution.html.

$$P(x)dx = \frac{1}{\sqrt{2\pi}} e^{-z^2/2} dz$$

Model users will provide four key inputs to drive this distribution:

AVG Cube/Unit – This is the mean or average size for items within a particular Product Line/Sort Type. While the items in a particular Product Line/Sort Type may change over the span of a few weeks or months as Amazon introduces newer ASINs to the mix, in the short term, this number remains relatively constant. Users can therefore derive this number from whatever the current value is for each Product Line/Sort Type.
Standard Deviation – This is simply the deviation around the AVG Cube/Unit value that users input. Again, users can derive this value from the current data for each Product Line/Sort Type.
# of Slices – As discussed previously, slices represent the number of intervals the user wishes to split the distribution into. The higher the number, the slower the model will run, but increasing the number of slices also increases the fidelity in the calculations.
# of Deviations – This represents the number of Standard Deviations around the AVG Cube/Unit that the user wishes to run the model for. In cases where a Product Line/Sort Type exhibits fairly close aggregation around the mean with a few outliers, controlling the # of Deviations essentially allows the user to cut off outliers that could skew the calculations.

Having defined the parameters for application of the Normal Distribution, I will now walk the reader through a sample of the calculations to illustrate the logic I later implemented in the model. The numerical examples that I will use for the remainder of this chapter represent the actual values I used in the test case trial run for the model. These examples, while demonstrating how the model logic works, will also build towards the discussion of the lessons learned from the first actual validation run of the model. Let us assume the following inputs from the user:

Table 4: Normal Distribution Inputs

AVG Cube/Unit (in^3)	Standard Deviation	# of Slices	# of Deviations
466.56	51.84	5	3

Running the distribution for 3 Standard Deviations around the mean, we are applying the Normal Distribution for z values within the range of -3 to 3. Since we are splitting this interval into five slices, the corresponding cumulative distribution values are reflected as follows:

Table 5: Normal Distribution Output

P(z)	Cube/Unit (in^3)
0.201	423
0.400	453
0.600	480
0.799	510
0.999	622

The logic for these calculations works as follows. Dividing the range of z values into 5 slices, we utilize the NormSDist(z) Excel worksheet function to determine the corresponding MinP(z) and MaxP(z) at z equals -3 and 3 respectively. Splitting the interval between MinP(z) and MaxP(z) into 5 slices, we now have the P(z) values as recorded in the table above. We then convert the P(z) values into Cube/Unit data points using the NormInv(P(z), mean, standard deviation) Excel function.

The table above represents the five slices that will represent the Cube/Unit distribution for this particular Product Line/Sort Type. The entries in each row of the above table represent the probability P(z) that the Cube/Unit is less then or equal to the given value. For instance, taking the first row of the table, the probability that the Cube/Unit is less then or equal to 423 in^3 is .201.

However, we also have to consider that I will later combine this distribution with the Demand Velocity distribution. For that calculation, we want each of the slices to have the same weight, and we want to focus on particular intervals rather then the cumulative distribution. In doing this, I can also place less weight on outliers, as the distribution will more adequately represent aggregation of units around the mean. The converted data values that I will apply in later calculations are as follows:

Table 6: Normalized Output by Slice

P(z)	Cube/Unit (in^3)
.2	367
.2	438
.2	467
.2	495
.2	566

In making this conversion, I have created a discrete probability distribution from the previous distribution. In this case, each row represents the probability P(z) that the Unit/Cube is a specific value. Taking the first row of the table, we assume that the Cube/Unit is 367 in^3 with a probability of .2.

We are concerned about outliers, and as such, we decided to set the P(z) intervals as equal sizes. Taking into account the user setting of 5 slices, this corresponds to slice intervals of 20%. Utilizing the NormInv Excel function once again, we convert each of the P(z) intervals into discrete Cube/Unit data points that will carry forward in the model for later calculations. The Cube/Unit distribution as reflected in this table will now represent the dimensional behavior of this particular Product Line/Sort Type.

4.5 Demand Velocity and the Pareto Distribution

For the Demand Velocity calculations, I needed to apply a distribution that would replicate the demand behavior of each Product Line/Sort Type in terms of outbound units/week. Application of this distribution required that users could match the distribution to the specific behavior of each Product Line/Sort Type. There was also a requirement to model demand behavior at various time horizons, based on historical demand behavior, short term behavior or forecasted demand behavior.

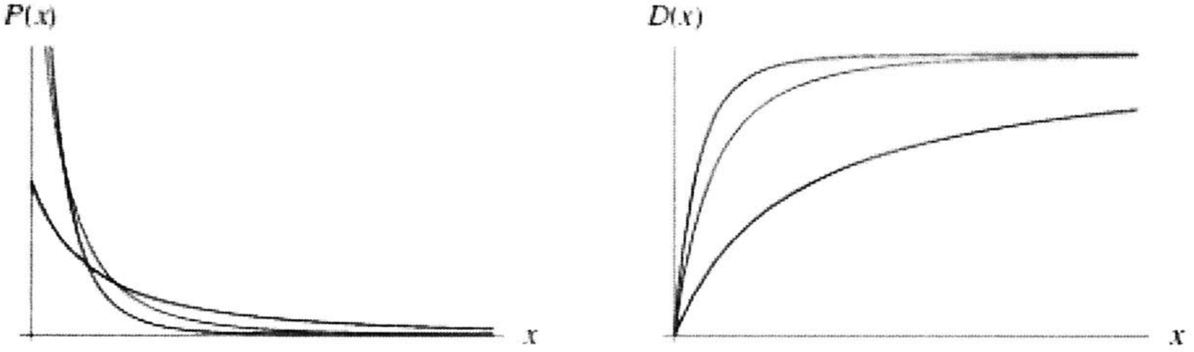
The Pareto distribution seemed to be the logical match given the demand behavior of Amazon's Product Lines. The concept of the Pareto is that a small percentage of a population accounts for the majority of that population's behavior. Translated to Amazon, this essentially describes the behavior of a small percentage of a Product Line's ASINs making up the greatest percentage of its outbound demand flow, or fast movers.

We define the Pareto Distribution as "the distribution with probability and density function:" [2]

$$D(x) = 1 - \left(\frac{k}{x}\right)^{a}$$

$$P(x) = \frac{ak^{a}}{x^{a+1}}$$

Figure 12: Pareto Distribution [2]



The positive Pareto constants *a* and *k* essentially drive the distribution, defined over the range $x \geq k$. The function $P(x)$ goes to zero for the condition $x = k$, and goes to one as x approaches infinity.

[2] Weisstein, Eric W. *Pareto Distribution.*" From Mathworld-A Wolfram Web Resource. http://mathworld.wolfram.com/ParetoDistribution.html.

The challenge of applying the Pareto distribution to this model is that users will define the behavior of the distribution, and the model will then derive the constants *a* and *k* to replicate the desired demand behavior. With the Pareto constants determined, the model will then utilize the Pareto function to generate the demand distribution based on the number of ASINs and number of individual units. To maintain simplicity for the model, we do not want to distract users with calculations that would export them from the model. From a design perspective, it would hinder the flow of the model if users have to conduct the Pareto Distribution calculations manually. Keeping in mind the operators who will ultimately use the model, it became necessary to automate this calculation rather then force the user to conduct the calculation manually and then import the results back into the model. It therefore became essential to find a way to automate the calculation of the Pareto constants. In order to solve for the Pareto constants, we need to rewrite the Pareto equation as follows:

$$f(x) = 1 - \left(\frac{k}{k + x} \right)^{a}$$

Where f(x) represents the cumulative percentage of total demand that is accounted for by x% of the items, assuming that the items are sorted in order of increasing demand, . We now want to create two breakpoints using this equation, allowing us to then solve for the constants a and k that satisfy the solution for both breakpoints:

$$f(x) = 1 - \left(\frac{k_1}{k_1 + x} \right)^{a} = A \qquad f(x) = 1 - \left(\frac{k_2}{k_2 + x} \right)^{a} = B$$

In terms of the demand behavior for a particular Product Line/Sort Type, I decided to utilize the classic Pareto Principle. The Pareto Principle, discovered by management thinker Joseph Duran, and named after the Italian economist Vilfredo Pareto, finds application in a variety of data sets that include sales productivity and organizational performance. For the purposes of this model, I will use the Pareto Principle to replicate demand behavior. "The Pareto Principle (also known as the *80-20 Rule*, the *law of the vital few* and the *principle of factor sparsity*) states that for many phenomena 80% of consequences stem from 20% of the causes. Moreover, among those 'top 20' it is also the case that 80% of consequences result from 20% of causes, and so on." [3]

Table 7: Pareto Principle 80/20 Rule

% of ASINs	% of Demand Volume
20%	80%
40%	15%
40%	5%

[3] *Pareto Principle.* From Wikipedia, the Free Encyclopedia. http://en.wikipedia.org/wiki/Pareto_principle.

Translating this into two cumulative breakpoints, we find that the first 20% of the ASINs account for 80% of the Demand Volume, and the first 60% of the ASINs account for 95% of the Demand Volume.

$$f(.20) = 1 - \left(\frac{k_1}{k_1 + .20} \right)^a = .80 \qquad f(.60) = 1 - \left(\frac{k_2}{k_2 + .60} \right)^a = .95$$

Rewriting the two breakpoints to solve for k_1 and k_2, we now have:

$$k_1 = \frac{.20(1 - .80)^{1/a}}{1 - (1 - .80)^{1/a}} \qquad k_2 = \frac{.60(1 - .95)^{1/a}}{1 - (1 - .95)^{1/a}}$$

The approach for utilizing these two equations is to evaluate k_1 and k_2 at various values for a, with the intent of minimizing the value $/k_1$-$k_2/$. The approach for doing this is to establish a range of values for a, and goal seek the solution in which $k_1 \approx k_2$. For this particular case, we find that the solution settles on a = 2.681 and k = .243.

One note of warning in the Pareto is that $f(x)$ never approaches a cumulative value of 100%. However, the distribution does approach one such that we can force the tail end of the distribution to 100% without skewing the constraints of the Pareto behavior.

Having established an approach for automating the calculation of the Pareto constants, I will now shift the discussion to application of the Pareto Distribution within the context of the model. Like the Normal Distribution, users will be prompted for several key inputs:

% of ASINs – These are the breakpoints that represent the ASIN population for the distribution. Please note that these are not the cumulative percentages, but rather discrete breakpoints.
% of Demand Volume – These are the demand volume percentages that correspond to the % of ASIN breakpoints previously defined by the user. Again, these are not cumulative values.
# of ASINs – This represents the # of ASINs currently in a particular Product Line/Sort Type. Like the Normal Distribution, since we do not expect this value to change in the short term, users can simply extract its value from the current real world data.
# of Slices – Serves the same function as previous discussions of slices.

I will now walk the reader through a sample of the calculations, using the following user inputs:

Table 8: Pareto Distribution Inputs

% of ASINs	% of Demand Volume	# of ASINs	# of Slices
20%	80%	7,468	10
40%	15%		
40	5%		

We will also assume an Outbound Demand of 32,545 units/week for this Product Line/Sort Type.

From the Pareto breakpoints, we again find that the cumulative breakpoints equate to the first 20% of the ASINs accounting for 80% of the Demand Volume, and the first 60% of the ASINs accounting for 95% of the Demand Volume. From my previous discussion on the Pareto constants, we know that a = 2.681 and k = .243.

Evaluating the Pareto function $f(x)$ over the 10 slices, we essentially are evaluating the function in intervals of .1, allowing us to calculate the corresponding cumulative percentage breakdowns. Each of the slices therefore accounts for 10% of total demand. We can then determine the # of ASINs and corresponding Demand Volume in terms of units/week/ASIN, making sure to convert the cumulative percentages into discrete intervals.

# of ASINS in Slice = (% of ASINs in slice)(# of ASINs)
Units/Week/ASIN = (Outbound Demand)(Slice Size) / (# of ASINs in Slice)

The logic flow of the calculation is such that we determine the intervals, using the slice approach to define the Cumulative % of Demand. This will then allow us to calculate the Cumulative % of ASINs in each respective slice. The conversion results in our determining the # of ASINs in each slice, and the outbound Demand Velocity of Units/Week per ASIN. These last two values replicate the demand behavior of this particular Product Line/Sort Type, and will carry forward in the model calculation flow.

The following table summarizes the evaluation of this logic:

Table 9: Pareto Distribution Outputs

Cumulative % of Demand	Constant a	Constant k	Cumulative % of ASINs	# of ASINs in Slice	Units/Week per ASIN
0.1	2.681	0.243	0.010	73	45
0.2			0.021	85	38
0.3			0.035	101	32
0.4			0.051	123	27
0.5			0.072	155	21
0.6			0.099	204	16
0.7			0.138	290	11
0.8			0.200	465	7
0.9			0.331	977	3
1			1.000	4997	1

The significance of each row in the Velocity Distribution table is that it gives us the # of hypothetical ASINs for a respective slice, and the demand for each ASIN in that slice in terms of units/week. So taking the first slice of the table, we have a hypothetical slice of

73 ASINs, with each ASIN exhibiting a demand of 45 units/week. This holds true for each of the remaining rows in the table, with each row representing one slice of hypothetical ASINs.

4.6 Calculating Cubic Demand

The next step in the calculation flow was to convert the Unit/Cube and Units/Week Demand Velocity into a Demand Velocity in terms of in^3/week. With the Normal and Pareto Distributions now in place, it simply became a matter of combining the two distributions together. The number of slices, or breakpoints, for this Velocity Distribution is directly correlated to the number of slices chosen by the user for the previous two distributions. In the distribution examples outlined in the last two sections, we had 5 slices for the Normal Distribution and 10 slices for the Pareto Distribution. This translates into 50 slices for the final Velocity Distribution that will carry forward in the model.

Recall from the previous two sections that we identified the two source distributions as having the following range of values:

Table 10: Pareto and Normal Distribution Outputs

Pareto Distribution	
Units/Week per ASIN	# of ASINs in Slice
45	73
38	85
32	101
27	123
21	155
16	204
11	290
7	465
3	977
1	4997

Normal Distribution	
Slice Cube/Unit	Cube/Unit Probability
367	0.2
438	0.2
467	0.2
495	0.2
566	0.2

When I combine the two distributions together, it results in a Demand Velocity Distribution of 50 lines of data. For the sake of simplicity, I will only show the first 10 lines of data to illustrate how the calculation flow works:

Table 11: Velocity Distribution Output

Velocity Distribution	
Velocity per ASIN Cube/Week	# of ASINs in Velocity
16409	15
19593	15
20857	15
22122	15
25305	15
14080	17
16811	17
17897	17
18982	17
21713	17

The first five rows of data correspond to the first row of the Pareto Distribution combined with every row in the Normal Distribution to generate the Velocity Distribution. Similarly, the next sequence of five rows corresponds to the second row of the Pareto Distribution converted into the Velocity distribution using every row in the Normal Distribution. The reasoning behind this is that we need to account for every behavior possibility in terms of the Unit/Cube dimension distribution and the Units/Week demand distribution.

At this point in the model development, we begin to understand the simplicity of the model design. By keeping the calculations relatively simple, it allows for the development of a suitable Velocity Distribution for a particular Product Line/Sort Type.

The actual calculations work as follows:

Velocity per ASIN (Cube/Week) = (Units/Week/ASIN)(Slice Cube/Unit)
# of ASINs at Velocity = (# of ASINs in Slice)(Cube/Unit Probability)

For instance, from the Pareto Distribution, the first line of data tells us that we would expect to see 73 ASINs with a demand of 45 units/week. To convert this data point into a Cube/Unit Velocity Distribution, we conduct the conversion with each line of the Normal Distribution. The first row of the Normal Distribution tells us that 20% of the ASINs in this Product Line/Sort Type have a cube/unit dimension of 367 in^3. We would therefore expect that:

(.2)(73 ASINs) $\cong$ 15 ASINs with a corresponding Demand Velocity of:
(367 in^3/unit)(45 units/week) $\cong$ 16409 in^3/week

These calculations represent the first row in the table for the Velocity Distribution. The rows that follow utilize the same logic and calculations for each of the subsequent slices.

4.7 Determining Bin Usage

It is at this phase in the analysis that we start to take into consideration Days of Cover and Velocity to Bin Type Mapping. We will be driving the actual bin assignments and numerical requirements from the Demand Velocity distribution generated in the previous section.

Since we now have the Cubic Demand Velocity in terms of in^3/week, we can associate each slice of ASINs in the distribution to a particular bin type. We do this through the Velocity to Bin Type Mapping. This is where we start to see the effect that this parameter has on bin allocation. Shifting the Velocity to Bin Type Mapping settings ultimately determines what range of Demand Velocities are assigned to which bin type. Therefore, shifting the Velocity settings for each bin type will have a significant effect on the number of bins utilized for each iteration of the model. This is not so much a calculation as it is a sort procedure for assigning each Demand Velocity slice to its appropriate bin type based on the parameter settings.

The next calculation is where we see the significance of Days of Cover. To determine the amount of cube sent to Prime, we multiply the Days of Cover by the Demand Velocity in each respective slice. The calculation is quite simply:

Cube Sent to Prime per ASIN = (Velocity/ASIN/Week)(Days of Cover)(1 week / 7 days)

This value essentially dictates the split between the amount of cube sent to Prime and the amount sent to Reserve. However, for the purposes of this model, we are only concerned with the amount sent to Prime.

Since we know the bin type this slice is now assigned to, and the amount of cube sent to each respective bin type by ASIN slice, we can determine the number of bins required per ASIN. This is calculated as follows:

Number of Bins per ASIN = (Cube Sent to Prime per ASIN)/(Bin Type Dimensions)

We are not dealing with whole numbers, so it is possible to calculate a fraction of a bin in this formulation. We therefore round up, as it is possible for the system software to allocate items to a bin but not fill it completely. However, regardless of whether or not the bin is completely filled, we cannot use that bin for another ASIN.

The slices consist of more then one ASIN, so we must now convert the Number of Bins per ASIN into the total number of bins required for each slice. Since we determined the number of ASINs per slice in calculating the Demand Velocity distribution, the calculation is as follows:

Number of Bins per Slice = (Number of Bins per ASIN)(# of ASINs in Velocity)

Again we have to round up since it does not make intuitive sense to talk about a fraction of an ASIN. There was some discussion as to where the round up should take place in the model. The question posed was does it make more sense to conduct the round up at the ASIN level or at the slice level. We understand that rounding up in either case, since the very framework of the model is dealing with slices of hypothetical ASINs rather then specific ASINs, does introduce some error. However, I decided that the error this would introduce into the model was negligible. To allow further analysis of this tradeoff, the model documents the roundup both at the ASIN and slice level.

Now that we have the bin requirements down to the ASIN slice level for each of the Product Line/Sort Types, the final piece of the puzzle is to simply aggregate the total bin requirements by Product Line/Sort Type and by bin type. This is the final phase of the model, and where the model proves its usefulness. In the event that the model determines that the FC requires more bins of a particular type then is currently available in the FC, this allows the user to change either the Days of Cover settings or Velocity to Bin Type Mapping Settings to shift capacity usage such that available capacity matches the demand requirements for the FC.

Taking the same ten lines from the Demand Velocity table, the following table illustrates the final calculations as they will appear in the model. The flow of the final calculations should become more obvious with the data provided in this table:

Table 12: Bin Type Assignment and Resource Requirements

Velocity Range Bin Assignment	Cube Sent to Prime per ASIN	Number of Bins per ASIN	Actual Bins Required per ASIN	Bin Utilization	Number of Bins per Slice	Actual Bins Required per Slice
Case Flow	19925	1.026	2.000	0.513	29.120	30.000
Case Flow	23791	1.225	2.000	0.612	29.120	30.000
Case Flow	25327	1.304	2.000	0.652	29.120	30.000
Case Flow	26862	1.383	2.000	0.691	29.120	30.000
Case Flow	30728	1.582	2.000	0.791	29.120	30.000
Case Flow	17097	0.880	1.000	0.880	16.969	17.000
Case Flow	20414	1.051	2.000	0.525	33.938	34.000
Case Flow	21731	1.119	2.000	0.559	33.938	34.000
Case Flow	23049	1.187	2.000	0.593	33.938	34.000
Case Flow	26366	1.357	2.000	0.679	33.938	34.000

The first row of the table tells us that the ASINs of the first Demand Velocity slice will reside in Case Flow bins. The Days of Cover calculation allows us to determine that approximately 19925 in^3 of inventory will be sent to Prime Case Flow bins from this shipment. Using the average dimensions of a Case Flow bin, this translates into a little over one Prime bin for an ASIN in this slice. As mentioned earlier, the roundup factor translates this into an actual bin requirement of two bins. Because we are allotting two bins for this ASIN, but are only partially filling the second bin, it results in a low Bin Utilization for this particular ASIN. Finally, we convert the number of Case Flow bins required to reflect the total number of ASINs in the slice, which comes out to a little over

29 bins. Again taking into account the roundup factor, this results in the first slice of ASINs requiring a total of 30 Case Flow bins.

4.8 Utilization

Amazon uses two perspectives when dealing with how well they use capacity. The first of these perspectives is called **Packing Factor**. This term refers to how efficiently Amazon is able to use a particular bin type. Previously, I discussed how for each bin type, a certain amount of space is lost due to safety buffers and other space restrictions. There is also the element of fitting items and cases into the bins. The cases and items that Amazon receives come in a wide variety of shapes and sizes, none of which are optimized to the bin sizes they use. Packing factor is a measure of the ratio of fillable space to total available space for a particular bin type.

For the purposes of the model, we did not include Packing Factor in the calculations. However, it would be quite simple to introduce a Packing Factor coefficient into the model. For each of the bin types, we recognize its size by the AVG cubic dimensions that represent each particular bin type. Multiplying this value by a Packing Factor reduction coefficient would essentially reduce the bin type dimensions to reflect the average amount of space Amazon can effectively use for each bin type. However, this would require further studies into the average Packing Factor for each bin type, and a more precise definition for how Amazon wishes to consider Packing Factor in the future.

Bin **Utilization** serves as a measure of how efficiently Amazon is using the available fillable space within a bin. I included a rough calculation for utilization in the model output. In my discussion of the model calculations, I made mention of the fact that I round up when the model determines that a particular bin type requires a fraction of a bin. In dealing with a total bin count, this round up is essential. However, we can use the fractional bin requirements to gain some perspective on utilization. There is also a time component to utilization, because as a bin depletes, the utilization of the space in that bin decreases. The utilization of a particular bin is at its lowest point just prior to replenishment.

Say for example a particular ASIN slice requires 1.23 Library Deep bins. The model will round this up to two bins. However, the second of those two bins is not completely filled, and therefore is not being utilized completely. The utilization for that second bin works out to the fractional portion of the bin required, or .23.

4.9 Model Assumptions

Development of the model required a couple of key assumptions, some of which I already covered in previous sections of the thesis. The assumption that is the greatest leap of faith, and therefore the Achilles Heel of the model, is that it assumes a perfect world. When I say perfect world, I mean that the model assumes that the FC is subject to stable or stationary demand such that it exhibits steady state behavior. While this is a reasonable assumption for the off season months, the seasonality of the holiday season introduces dynamics such that the FC is not in steady state.

There is a time and flow component to Amazon's operations in that for any given week, there are items already occupying bins, items flowing into the FC and items flowing out of the FC to fulfill orders. The model assumes that every bin is available, and it assigns ASIN slices to bin types based on that assumption. I am therefore assuming that the Cube/ASIN is independent of the demand per ASIN.

You can examine this scenario in one of two ways. If the model is providing bin requirements in an ideal world, then Amazon should strive to configure their FCs such that the bins available and the bins required match up perfectly. However, it is somewhat cost restrictive to reconfigure an entire FC so as to change the distribution of bin types.

The other perspective on the model output is that it helps operators better understand the dynamics of the parameters driving capacity usage. The model is static in its current form, but multiple iterations of the model, over a period of weeks and months, provides a picture that will ultimately allow Amazon to optimize these parameters.

Another assumption driving the model is representing the Min and Max Days of Cover as the average of the two. Again there is a flow component to the amount of cycle stock present in a particular bin over its pendulum swing of depletion and replenishment. However, utilizing basic supply chain logic dictates that the average of the two is a suitable assumption for representing this cycle stock parameter.

Chapter 5: Coding the Model

With the model logic calculation flow in place, the next phase was to actually code the model. Initially discussions pointed towards utilizing a database approach to the model. However, early experiment in Microsoft Access proved to lack the intuitive interface I was really hoping to achieve. My preference and level of comfort was more towards utilizing Microsoft Excel, but there are limitations to what you can do with it as a toolset.

One of the things I wanted to avoid was forcing the user to cut, copy and paste data from one table into the next to perform their calculations. The vulnerabilities and opportunities for mistakes on the part of the user are greater using this approach. It became obvious after designing several different approaches to the model that the best solution would be to use Excel calculations automated and driven by Visual Basic coding.

Not having a background or any experience with Visual Basic coding, I relied heavily on two primary references for translating the model logic into executable code. These references served more as a guideline to understanding Visual Basic, and I did not lift any specific code from these sources. I therefore do not have these references cited in the thesis text, but I did include them in my list of references at the end of the thesis.

We validated the model logic shortly after the Midstream Review, so coding began in earnest in early October. The challenging aspect of this part of the internship was not so much designing the model architecture, but rather the limitation of my never coding in Visual Basic before. After a two week crash course in Visual Basic, I was up and running, with the results of those efforts summarized in this chapter. This chapter will focus more on architecture and logic as opposed to actual coding.

5.1 Model Design and Approach

As the previous Chapter demonstrated, the calculations for this model build upon one another in a series of steps. It therefore made sense to design the model such that it would take the user through these steps in a user friendly and intuitive flow. It also became apparent to me that from a troubleshooting perspective, it made sense to allow the user to choose either to view each of these steps in succession, or run all of the steps automatically, thereby taking the user to the final output.

I decided to break each series of calculations into its own procedure. The problem this created was that many procedures have to call on the same data, so I had to devise a way to make certain sets of data available to all of the procedures. There was also the problem of certain data sets consisting of numerous values, leaning the model application to the usage of arrays. This became easily resolved through the use of global variable arrays. I also found that I was using a lot of the same formatting code repeatedly in the model. To streamline the code a bit, I also created a series of formatting procedures that would enable me to utilize the same series of formatting code without having to duplicate it for each of the actual model procedures. The model is split into a series of worksheets and macros on each of the respective worksheets. The following table summarizes the function and purpose of each worksheet and macro:

Table 13: Visual Basic Design Functions

Model Worksheet Functions	
Product Lines	Serves as the input sheet on which users can designate what Product Lines and Sort Types they wish to run the model for. This is part of the open ended design of the model that allows users to customize each iteration of the model. Users can choose to run all Product Lines for every Sort Type, one Product Line for one Sort Type, or any Product Line and Sort Type combination they desire to run an iteration for
Cube Inputs	This serves as the input screen for all of the variables discussed in the model calculation flow. The input sheet will only generate for the Product Line and Sort Type choices the user makes on the "Product Lines" worksheet. The inputs included on this page are as follows: **Days of Cover:** Min Days of Cover, Max Days of Cover, AVG Days of Cover. **Pareto Inputs:** % of ASINs, % of Demand Volume, # of ASINs, # of Slices. **Cube/Unit Inputs:** AVG Cube/Unit, Standard Deviation, # of Slices, # of Deviations. **S&OP:** Outbound Units, Inventory Units **Bin Types:** Bin Types, Total # of Bins by Type, AVG Bin Dimensions by Type, Velocity to Bin Type Mapping Range, Days of Cover Note that Days of Cover is mentioned twice in this list. This is because you can associate Days of Cover either by Product Line/Sort Type or by bin type. The current configuration has Days of Cover associated by Product Line/Sort Type. However, I included a toggle as a placeholder to correspond to current discussion at Amazon about the more accurate association for Days of Cover.
Normal	Displays the results of the Normal Distribution of the Cube/Unit.
Pareto	Displays the results of the Pareto Distribution for the Demand Velocity in units/week.
Demand	Displays the results of integrating the Normal and Pareto Distributions to create the Demand Velocity Distribution in cube/week.
Output	Displays the results for determining the bin assignment for each ASIN slice and the number of each respective bin type required for each ASIN slice.
Summary	Displays the total number of bins by bin type required for each of the respective Product Lines/Sort Types.

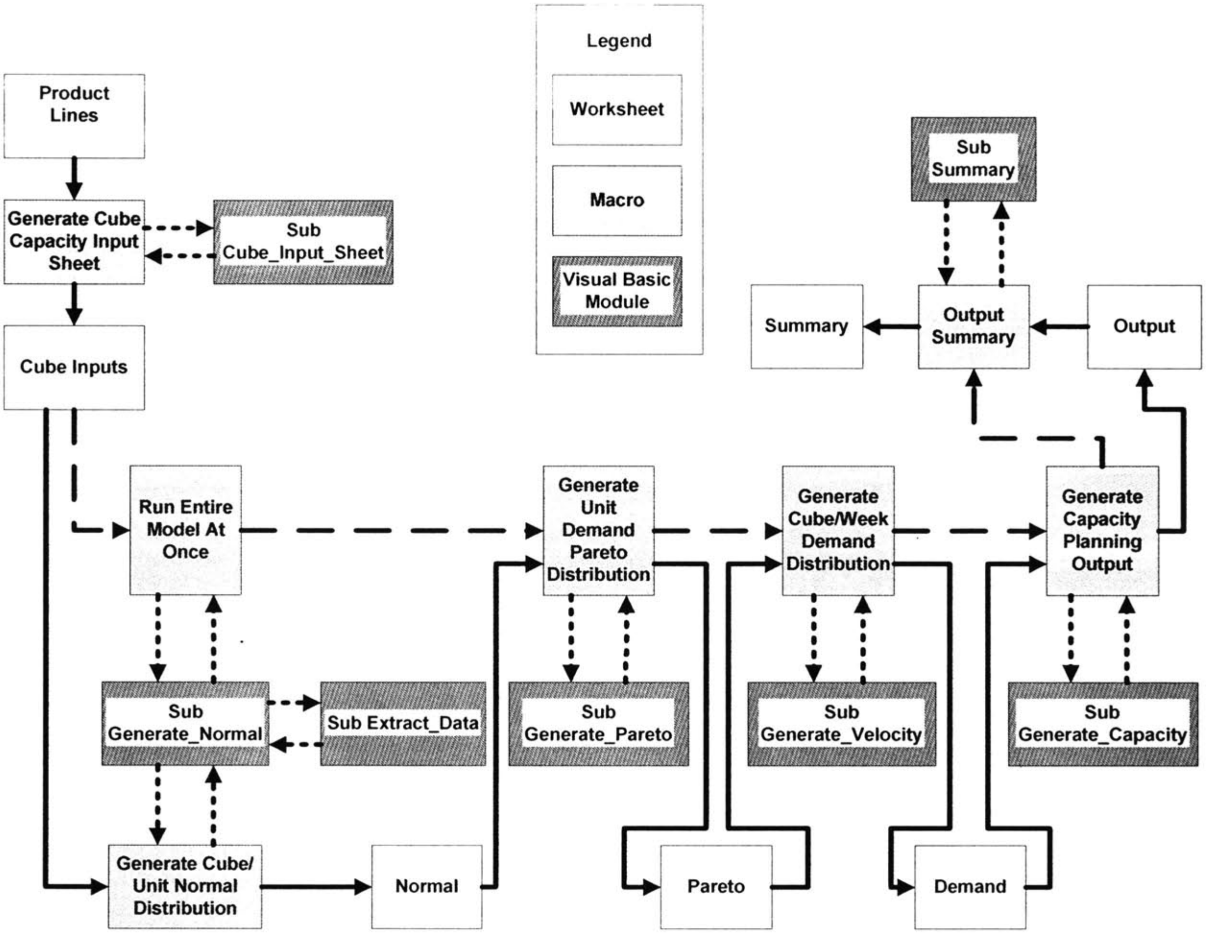
Macro Functions	
Generate Cube Capacity Input Sheet	Generates the “Cube Inputs” input sheet based on the Product Line and Sort Type choices made by the user on the “Product Lines” worksheet.
Run Entire Model at Once	This macro essentially runs all of the remaining macros at once, immediately taking the user to the final “Summary” worksheet.
Generate Cube/Unit Normal Distribution	Extracts the user inputs from the “Cube Inputs” worksheet and places them into global variable arrays. This macro also generates the Normal Distribution displayed on the “Normal” worksheet.
Generate Unit Demand Pareto Distribution	Calls upon the global variable arrays to generate the Pareto Distribution output displayed on the “Pareto” worksheet.
Generate Cube/Week Demand Distribution	Integrates the Normal and Pareto distributions, and then displays the final Demand Velocity Distribution in terms of cube/week on the “Demand” worksheet.
Generate Capacity Planning Output	This macro essentially serves as a sort filter. It calculates the bin type assignment and number of bins required by slice for each of the ASIN slices and their respective demands. It then takes these calculations and displays them on the “Output” worksheet.
Output Summary	Analyzes the results on the “Output” worksheet and then sums total number of bins required by bin type for each of the respective Product Lines. It then takes the results and displays them on the “Summary” worksheet.

The modular design of the model isolates each of the calculations such that it would be very easy for someone to extract that logic for application in other models or efforts. The open ended design of the model code also allows other programmers to easily add features to the model. This is one of the benefits of using global variable arrays and a procedure approach to the model. Some of the features we discussed adding to the model would simply consist of using the global variable arrays in other calculations. It would also be quite easy to add inputs or global variable arrays to introduce other factors once the dynamics of the model are exposed over repeated use.

5.2 System Architecture

Taking all of the macros and worksheets into consideration, it is important that the reader understand how all of these properties of the model fit together into one design architecture. This understanding is of particular use for those readers who will go on to design models themselves. An essential step to the process is developing a road map for the logic flow, but from the perspective of a programmer. The following diagram essentially shows how all of the pieces fit together into one system architecture:

Figure 13: Model Architecture



The actual thesis will not cover how to use the model, as this document is not meant to serve as a user manual. For those interested in taking a walkthrough of the model with screen shots, Appendix C contains snapshots of the trial run of the model in greater detail.

Chapter 6: The Trial Run

The discussion thus far has focused on developing the model logic, without much consideration given to determining whether or not this model logic was in fact valid. As this section will prove, I made one critical error in developing the model. The focus after Midstream became trying to code the model and have it operational by the end of the internship. I completed the coding in early November, and then shifted to validating the model. The problem this created in that focusing solely on making the coding aspect of the model operational, I shelved further analysis to ensure the distributions I chose were in fact the most appropriate.

One of the limitations to running validation trials for the model was accessing the required data. While Amazon maintains a vast database of historical data, the interface for running sorts on this data is quite complex and not particularly intuitive. For LFM students that intern with Amazon this coming year, it is essential that you learn how to utilize this database system from the beginning of the internship.

By November, Amazon had already entered into its holiday season crunch, and receiving assistance for learning how to navigate the databases was difficult to attain. A lesson learned on my part was that I did not aggressively pursue learning how to navigate Amazon's data warehouse from the beginning of my internship Once I left the FC and returned to Boston, data support for validating the model became nonexistent as I was no longer tied to the Amazon network. The take away for other LFM students and future Amazon interns is conducting resource planning from day one such that you can better anticipate and prepare for data requirements that are inevitably part of internship closure.

I did not validate the assumptions for the Normal and Pareto Distributions during the model development. In designing the model, I simply assumed the distributions would work and incorporated them into the model. As we fill soon find with the Pareto Distribution, this was not a valid assumption. For future reference, a better approach would have been to validate the distributions, explore different options for incorporation into the model and then initiate the designing and coding the model.

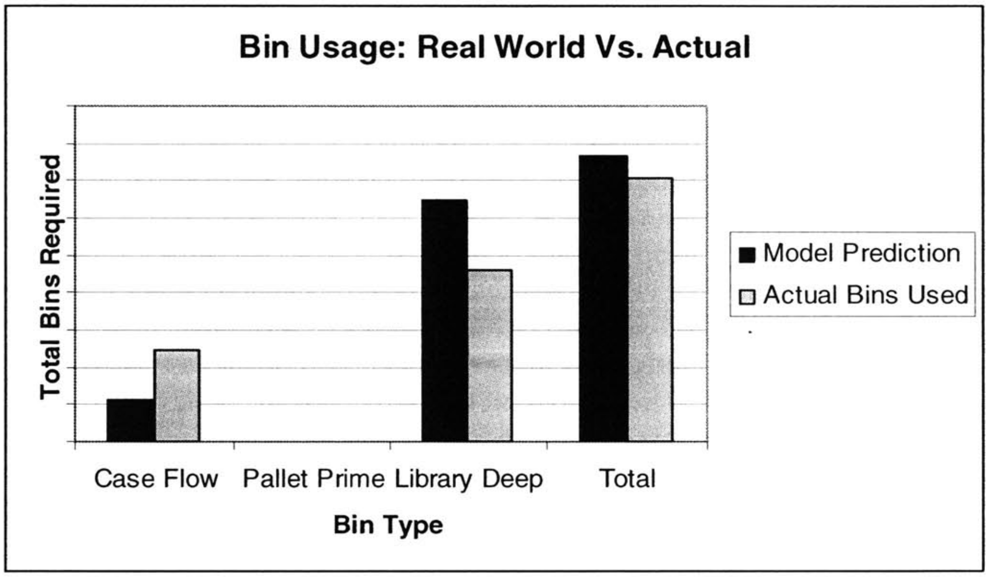
6.1 Results of the Test Case

Fortunately I was able to get enough data to run the model for one trial week. While this is not enough to validate the model, it does demonstrate that the model does in fact work as designed. The week in question was during the holiday crunch, so this was not the ideal scenario for which to run the model. However, as the next section will illustrate, running the model at this time did help to expose some of the negative dynamics in Amazon's operations that the model will ultimately help them eliminate.

For the purposes of this discussion, we will focus on the trial run of the test case. This test case represents the numbers utilized in demonstrating the model calculations in previous sections. I will focus on the results of this trial run, and the implications these results have on the model:

The trial run represents the behavior of one Product Line/Sort Type analyzed for the first week of December. This consisted of extracting the required inputs for the model from the specific characteristics of this particular Product Line/Sort Type. I also received data summarizing the bin capacity utilization for this particular Product Line/Sort Type for the week in question. The goal of the trial run was to run the model using the appropriate input data, and then compare the projected bin requirements as determined by the model against the actual bin usage for this Product Line/Sort Type for the week in question. The results of the trial run were as follows:

Figure 14: Trial Run Results



It was necessary for proprietary reasons to shield the total number of bins by type that the model predicted and real world bin usage. However, it is important to note that the model was under on Case Flow bins by a total of 1328 bins and was over on Library Deep bins by a total of 1889 bins.

What this trial run illustrates is that the model over estimated on Library Deep bins and under estimated on Case Flow bins. However, further exploration of this discrepancy led to the discovery that it is most likely explained by dynamics at play during the holiday season and not a failure in the model logic. While these results would seem discouraging at first, a better understanding of why the numbers did not match are actually encouraging for validating the functionality of the model.

6.2 Other Dynamics at Play

I already explained that one of the limitations of the model is that it assumes a perfect world, and this trial run quite effectively illustrates this. During the holiday season, operators know through intuition and past experience that certain bin types will become critical in future weeks. They therefore lock certain bins, essentially reserving them for future use. The other dynamic at play is bin spillover. If the allocation software wants to send an ASIN to Library Deep bins, but there aren't any Library Deep bins available, it will automatically send that ASIN to the next higher bin size, in this example Case Flow bins. For the week in question, there was no Library Deep storage remaining in the FC. A visual inspection of Library Deep storage for that week yielded that every bin was in fact already full. Operators sometimes lock bins, preventing their usage in anticipation for upcoming demand. At this point in the holiday season, locking was not an issue.

This dynamic quite accurately allows us to explain why the model was over on Library Deep bins and under on Case Flow bins. It also brings to light the fact that further iterations of the model can include coefficients to account for these dynamics. We already identified Packing Factor and Utilization as two dynamics that may require adjustment coefficients. Seasonal Spillover is a third dynamic we can now take into consideration.

Out of curiosity I adjusted the results of the trial run to account for spillover, and I discovered something quite interesting. If you look at the far right column of the plot, the total number of bins required in both the forecast and real world usage were nearly identical. While it would seem that summing Library Deep and Case Flow bins together is comparing apples and oranges, they are close enough in size that this total becomes somewhat relevant.

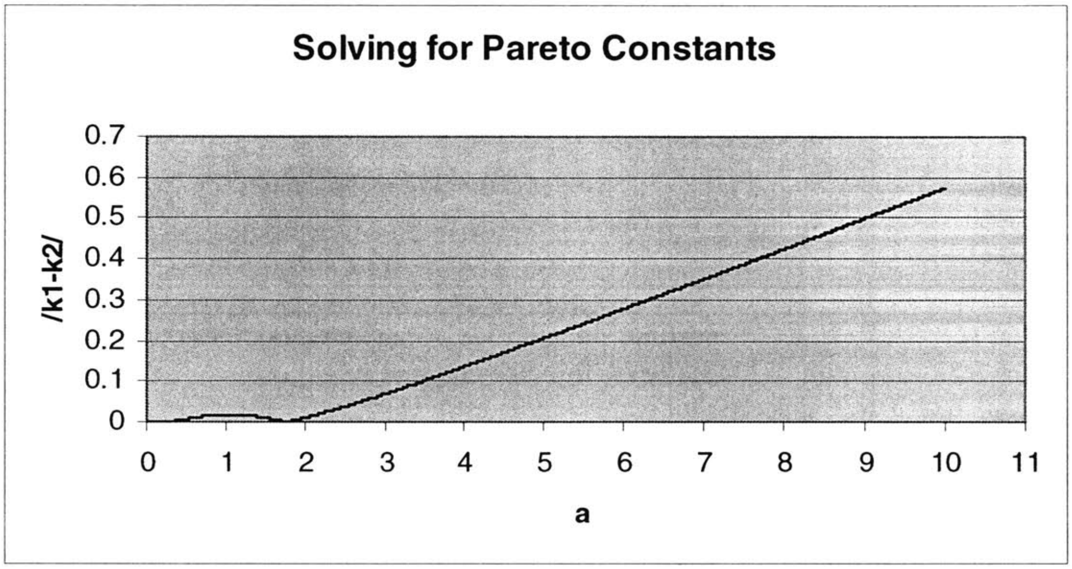
Let us assume for the moment that the model was accurate in predicting "x" Library Deep bins. Originally, the model predicted "y" Library Deep bins, which is an overage of 1889 bins. Again because of proprietary protection, I cannot expose the numbers used for this calculation. But when I convert the overage in Library Deep bins into Case Flow bins, and scale both bin counts accordingly, I find that on Case Flow bins I am off by less then 500 bins. I simply did a quick cube conversion, but the overflow cube also is distributed over an unknown number of ASINs. However, this quick calculation, while not enough to validate the model, certainly was encouraging for the trial run. In terms of order of magnitude, the model did seem to be working. Assuming the seasonality dynamic, I find that the model does an accurate job of replicating bin usage within Amazon's storage capacity network.

6.3 The Limitations of the Pareto Distribution

This trial run was not enough to validate the model. Without the data support to run further trials of the model, I began to test the limits of the model by focusing on the distributions. Further exploration of the Normal Distribution assumption did not reveal any critical flaws in the model logic. While I recognized that certain Product Line/Sort Types may not exhibit Normal Distributive behavior, I also recognized that it would be possible to split Product Lines into size categories such that this assumption would still be applicable.

Further exploration of the Pareto Distribution proved problematic. For the entire development phase of the model, we assumed that the Product Lines would exhibit 80/20 Pareto behavior. However, I also realized that I would need to test the model for certain ranges of behavior that deviate from the 80/20 test case. This is where the model began to unravel. From our previous discussion, we know that the Pareto constants do solve for the 80/20 test case:

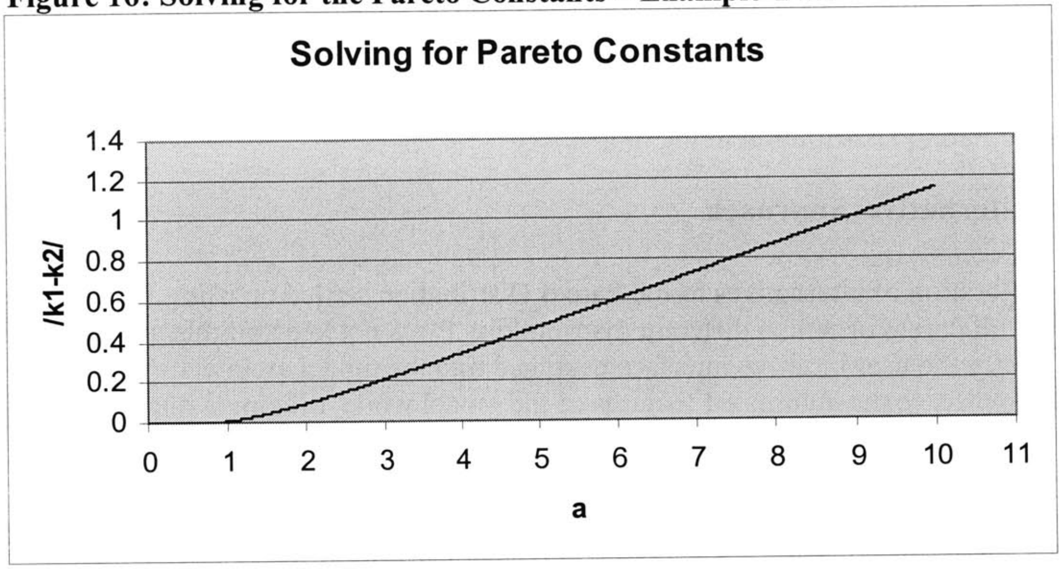
Figure 15: Solving for the Pareto Constants – Example One



As you can see from this plot, the Pareto constants do solve for the range of $a>1$. What this means is that there is a distinct solution for a, assuming that a must assume a value greater then 1. Visually, the distinct solution occurs between 1.5 and 2, where the plot of $/k1\text{-}k2/$ hits its minimum value, approaching zero. Since we defined the solution for a occurring when $k1 \cong k2$, this trial run of the Pareto logic has a distinct solution. However, running other test runs would prove that this is not always the case.

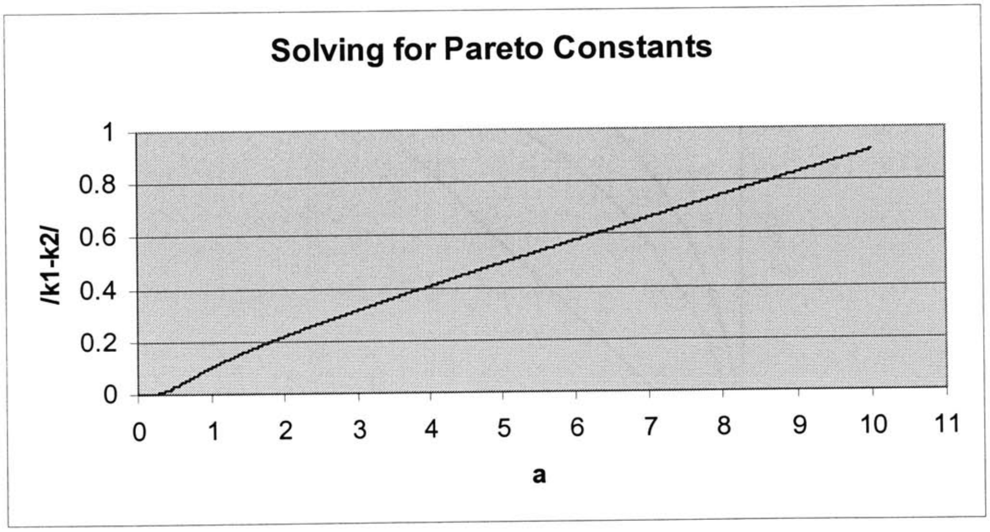
Let us see what happens when you run a Pareto Distribution of 10% of the ASINs accounting for 80% of the Demand Volume and 60% of the ASINs accounting for 95% of the Demand Volume. In this scenario, the Pareto constants never achieve a solution in the range $a>1$. If you were to continue the plot for values of $a<1$, the value of $/k1\text{-}k2/$ continues to decrease, driving closer and closer to zero.

Figure 16: Solving for the Pareto Constants – Example Two



Similar exploration of other feasible demand distributions would yield even more bizarre results. Let us assume that 30% of the ASINs account for 70% of the Demand Volume and that 50% of the ASINs account for 95% of the Demand Volume. In this case, there is no solution for the Pareto constants either:

Figure 17: Solving for the Pareto Constants – Example Three



Running various trials of different distributions demonstrated that the Pareto logic is not globally applicable to any feasible demand configuration. Discussion about this problem resulted in our coming to the conclusion that perhaps these inputs were in violation of true Pareto behavior, therefore explaining why the goal seeking algorithm could not find a solution for the Pareto constants.

The intent of showing these examples is to demonstrate that the Pareto Distribution was not as robust an approach as we initially assumed. We discovered scenarios in which the user could set distribution slices that would prevent the model from solving for the Pareto constants *a* and *k*. This was not acceptable for maintaining the automated design of the model logic, and these results made it fairly obvious that the Pareto distribution was perhaps not the best choice for this model. I then began to explore other alternatives to replace the Pareto Distribution in the model.

6.4 An Alternative Approach

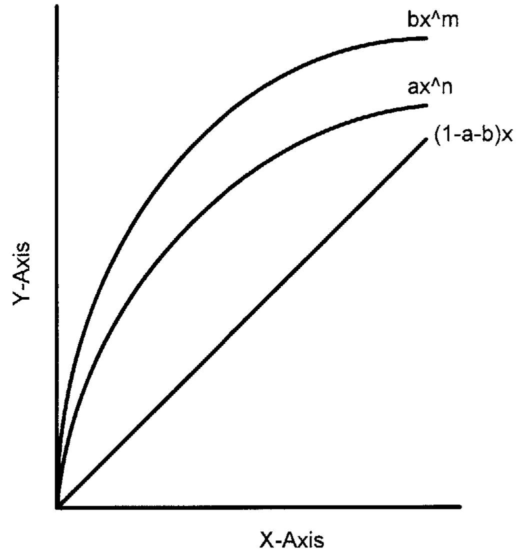
Initial exploration of alternatives to the Pareto Distribution settled on using lookup tables, either for the Pareto or with a different distribution. I was not keen on this approach as it would nullify the user intuitive interface designed into the model architecture. I still wanted to preserve the automated features of the model while utilizing a distribution that could handle all feasible demand distributions exhibited by the Product Lines.

We decided to explore a solution of the following form:

$$f(x) = (1 - a - b)x + ax^n + bx^m$$

Graphically, the combination of the three terms in the equation looks something like this:

Figure 18: Alternative Approach to the Pareto



The application for this approach is to prompt the user to enter in different values for the powers *n* and *m* to approximate the Pareto behavior we sought to achieve in the previous approach. This will allow the model to solve for the constants *a* and *b*, thereby matching the desired demand behavior. With the powers and constants determined, the model can then utilize *f*(*x*) as applied in the previous model architecture. Of course implementing this solution will result in some changes to the model, which I will now discuss.

We began development of this alternative approach during the course of the Spring semester, and due to time constraints, full implementation and coding of this approach was not feasible. However, the one limitation of the model in its current state is the lack of flexibility in the Pareto Distribution for replicating any conceivable demand behavior. A few trials will demonstrate that this alternative approach is a far more robust solution, and can adequately replace the Pareto Distribution in the current model logic. The modular design of the model should prove to be quite valuable in making this replacement.

Let us once again assume a typical cumulative Pareto distribution, where the first 20% of the ASINs account for 80% of the Demand Volume, and the first 60% of the ASINs account for 95% of the Demand Volume. We wish to solve for the constants *a* and *b*, by manipulating the powers *m* and *n*.

Let $w = (.8 - .2)/(-.2 + (.2)^m)$
Let $x = (-.2 + (.2)^n) / (-.2 + (.2)^m)$
Let $y = w - ((.96 - .6) / (-.6 + (.6)^m))$

Let $z = x - ((-.6 + (.6)^n) / (-.6 + (.6)^m))$
Let $a = w - bx$
Let $b = \max(0, y/z)$

The following table summarizes several trial runs, all of which exhibited the desired behavior. Based on this analysis, it is safe to conclude that this approach is more robust and a better fit for the model:

Table 14: Alternative Approach Validation

Trial	First Point		Second Point		M	n	a	b
1	20%	80%	60%	95%	.95	.50	-20	0
2	20%	80%	60%	95%	.80	.50	-6.76	4.51
3	20%	80%	60%	95%	.6	.50	-8.80	8.86
4	10%	80%	50%	95%	.95	.50	-38.0	5.38
5	10%	80%	50%	95%	.8	.50	-12.4	6.6

If you look at trials 1, 2 and 3, we notice that there is more then one solution that satisfies the behavior we wish to emulate. Similarly trials 4 and 5 show that for distribution points for which the Pareto Distribution could not find a solution, this approach does have a solution. While it may be of some concern that a particular demand distribution does not have one unique solution, this adds a level of flexibility that was lacking with the Pareto Distribution approach. By altering *m* and *n*, the user can essentially define which result best matches the desired demand behavior. Of course this introduces a trial and error aspect to the model, but one that Amazon can still implement into the model logic in its current flow.

Chapter 7: Model Applications

I have made brief mention of the applicability of this model to Amazon's operations throughout the course of the thesis, but in this section, I will explore these applications in greater detail. The intent of this model from its initial design phase was to deliver to Amazon a robust capacity planning model that would provide insight on each and every applicable dynamic. The model in its current state does have the potential to do just that, and represents phase one of what I hope will become an ongoing effort within Amazon.

7.1 Introducing New Bin Types

Going back to the lessons learned from the data mapping of the FC, it raised the question of the functionality inherent to Amazon's current bin configuration. The current slating of three Prime bin types may not be the most appropriate for optimizing their capacity usage. Similarly, there is some interest in perhaps exploring the introduction of new bin types, either as intermediate capacity alternatives within the current bin framework, or something entirely new.

The one constraint on this discussion is the ability for Amazon to completely reconfigure their FCs as they currently exist. However, given Amazon's growth and expansion, it is reasonable to expect that they will construct new FCs in the future. One capability they require is the ability to determine what those FCs should look like. While there are many other dynamics and consideration competing for attention in the development of any new FC, capacity allocation and planning holds a crucial role in this discussion.

The power of this model is that it enables planners to play "what if" scenarios with the current FC configurations. By introducing new bin types, or adjusting the dimensions of the current bin type configuration, planners can run the model over multiple iterations and optimize bin configuration against Packing Factor and Utilization. This is a capability that Amazon currently does not have. While the model is somewhat limited in its ability to fully answer this question, it certainly provides the foundation upon which to conduct further studies on this topic.

7.2 Capacity Forecasts

At the tactical level, this model provides the greatest value. As operators begin to plan for the holiday season, allocation of capacity becomes critical. As the trial case shows, operators often have to utilize overrides to the system in managing their capacity. However, this is largely a patchwork approach to the problem, and violates the design and intent of Amazon's capacity planning software.

I envision that operators can take the demand forecasts for an upcoming holiday season and use the model to determine when and what kind of bin type they will run out of. This will enable operators to either adjust their system parameter settings ahead of time, understanding the ripple effect that those changes will have in the coming weeks and months. It will also allow planners to better plan what overrides they may have to implement, and shift inventory accordingly.

7.3 The Replenishment TradeOff

You cannot really discuss capacity planning without also understanding the labor required to support a certain capacity strategy. Days of Cover triggers replenishment moves, and there is definitely a tradeoff between having too much inventory residing in Prime storage and the amount of labor required to support a more aggressive replenishment strategy.

We did not consider replenishment in the model, but this is the type of add on to the model that is quite sensible. It would not be terribly difficult to add a worksheet that factors Days of Cover such that it calculates the number of replenishment moves required to support a particular iteration of the model. This will ensure that operators do not adjust their Days of Cover settings such that it exceeds the bounds of what they can feasibly support in replenishment moves.

7.4 Days of Cover and Velocity to Bin Type Mapping

The ideal solution to the model is that it ultimately drives towards optimizing the Days of Cover and Velocity to Bin Type Mapping settings for the FC. For an optimization to work, it would have to take into consideration Packing Factor, Utilization, Replenishment and any other dynamic that the settings of these parameters infringe upon. The current version of the model has placeholders for many of these dynamics, understanding that implementing these factors into the logic flow will only make the model more robust and comprehensive in its applicability.

In the short term, I already discussed how Amazon can account for these dynamics through the use of adjustment coefficients. However, one would expect that Amazon will ultimately revise this model, taking into consideration the efforts by other analysts within their network.

The one limitation of this model is that I developed it in isolation from many of the other individuals and teams working on similar issues. The final product and value of the model could have been infinitely better if built within a team environment where all of the stakeholders in the analysis could brainstorm the solution. I will cover this issue in greater detail when I discuss leadership and culture.

Similarly, Amazon sets its Days of Cover and Velocity to Bin Type Mapping settings globally for an entire FC. However, the discussion thus far should make it somewhat obvious that each Product Line/Sort Type can potentially exhibit quite different behavior, as do each of the respective Bin Types. What the model does allow for is to set these parameters specific to the Product Line/Sort Type and Bin Type.

Over multiple iterations, this brings to light how Amazon works with Product Lines in its current configuration. While it is sensible to aggregate like items by Product Line for presentation on their website, it may not make sense to aggregate items by Product Line when taking into consideration storage allocation. What this model will allow Amazon to do is perhaps store items based on their demand, regardless of their respective Product Line. The model will also allow Amazon to gain a better understanding of how to associate demand ranges to each respective Bin Type.

Chapter 8: Recommendations

This Chapter is not so much tied to the analytical aspects of the model, but rather discusses the actual internship experience. While many of these observations were documented in my exit interviews, I feel they will serve as a useful reference for Amazon so that they may improve the internship experience for future LFM students. Similarly, LFM students may find some value in reading this section, as it will make them aware of the types of challenges they will face on their internships, regardless of the partner company they are assigned to.

8.1 Further Validation of the Model

There is no doubt that the model will not be of much use unless those involved in its development can build confidence in its accuracy. The only way Amazon can achieve this is to run further iterations of the model against real world data, as I did in the trial run.

I would have liked to have done this while on internship, but the data support required to run these iterations was not there. Regardless, what I perceive as the best approach to validating the model is to run it for the entire FC at different time stamps during the year. I would expect that a complete FC run of the model over the course of the holiday season and also at the lull in the summer months would be most appropriate. Validating the model in this regard will also help to expose the dynamics and nuances in FC operations that may cause the model to deviate from what happened on the ground in real life. I would also suggest running these trials for more then one FC to expose dynamics that may be present in one but not the other.

8.2 Opportunities for Further Intern Study

An encouraging aspect of the work I did for Amazon is that it exposed the possibility for a myriad of other projects that future LFM students could assume. Below is a list of areas of study that I think would be most beneficial:

Model Validation – Validating the model and running it against numerous test cases is quite a substantial amount of work. I have no doubt that Amazon could structure an internship around acquiring the data required to validate the model and then running trials as discussed previously. The value in doing this is that with multiple trials in place, it will begin to expose the dynamics of the system parameters. From this study, Amazon can then initiate discussion on the next phase of the model, which would be more of an optimization approach. With the appropriate number of test cases available for study and comparison, this would be a worthwhile exercise and a necessary step in continuing the capacity planning discussion

Bin Type – There are currently studies going on to determine how to best optimize Packing Factor and Utilization within the bin types. I think this study would be of great interest and value to other LFM students.

Replenishment – I identified the tradeoff between capacity and replenishment as an area for future study and investigation. Developing the logic and calculations that could plug into my model is another suitable area of study for LFM interns.

8.3 Leadership Observations

One observation I made about Amazon's corporate structure is the disparity in the perspectives of operators at the FC and strategic planners and analysts in Seattle. Managers and leaders at the FC level are of course more focused on day to day operations. Similarly, many of the managers and analysts at the corporate level do not have a great deal of familiarity with the specifics of FC operations, yet they are making decisions or conducting studies that would require a better appreciation of an FC perspective.

The differing perspectives became more readily apparent in our attempts to define the model scope. My initial work at the FC focused primarily on the tactical needs of the FC, but subsequent meetings with my internship supervisor broadened the scope to include the strategic concerns in Seattle. It is understandable that the focus of these two realms is quite different given the nature of their immediate concerns, but it also poses a certain leadership and culture dilemma.

Specifically, it is difficult to recommend what Amazon should do to address this cultural issue, and perhaps it is an aspect of their culture they do not wish to change. I think it would be helpful from a communication perspective if analysts, programmers and other planners at the strategic level gain a better understanding of the concerns and challenges faced at the FC level. It is one thing to read about these challenges or analyze data on it, and quite another to experience and face these challenges at a personal level. I think a rotation program such that all incoming leaders and managers experience operations at the FC level is the most reasonable and appropriate solution.

Similarly, FC operators need to gain an appreciation for the long term planning concerns coming from Seattle. I did observe that Amazon maintains a fairly robust and frequent series of conference calls and meetings to coordinate their efforts, but more interactions between FC operators and the corporate leadership would help to strengthen this working relationship.

This is not to say that I observed this working relationship as being dysfunctional or broken. On the contrary, the professionalism at every level of Amazon's leadership chain was quite impressive. But Amazon's evolution as a company will reside in its ability to target and optimize the dynamics that define its operations, similar to the work I did on my internship. Focusing on the cultural barriers that hinder this process is ultimately to the benefit of the entire company, and will facilitate for the more efficient sharing of information and coordination of efforts.

8.4 Conclusions

The value of this internship is that it enabled me to extract lessons learned, specific to the actual analytical work of the project scope and more general to the actual internship process. As such, my conclusions and recommendations fall into three categories: those specific to the LFM Program, lessons learned for Amazon, and things I learned for myself.

LFM Lessons Learned – After several discussions with my thesis advisors on this topic, we came to the conclusion that the LFM Program does not provide instruction for the realm of complex model development and system design. One of the limitations to our field of study is that it is still very much a developing field. During my undergraduate studies, our field of engineering fell under the title of Systems Engineering. Yet other institutions named similar programs as Industrial Engineering, Manufacturing Engineering and Operations Research, among others. While the title assigned to our field of study is not particularly important, the academic toolsets taught to students is highly relevant. If you examine more traditional branches of engineering, such as Civil or Mechanical, you will find that academic instruction in those fields have a fairly robust range of toolsets for implementing and managing a project. While those toolsets certainly exist in our field, they are not as intrinsically linked to the academic process. The LFM Program does an outstanding job of teaching the basic toolsets and concepts, through problem sets and case study analysis. Yet the process of fitting various toolsets together into a functional model is something not currently part of the LFM curriculum. Similarly, the academic principles required for such endeavors transcend the study of supply chain, network flow, transportation and other such concepts. Database design, data mining, project planning, coding logic and other such tools are critical to managing tasks that cut across various levels of an organization's operations. As such, it would be in the best interest of LFM students if the program eliminated some of the redundancy in teaching fundamental concepts, and instead introduced a series of courses that cover the more advanced dimensions of model design and development.

Amazon Lessons Learned – The value of my deliverable to Amazon is not so much in the specific model I developed, but rather in the ripples my work created throughout their organization. I mentioned previously that I did not perceive a great deal of communication and interaction between the various operational and analytical teams working in Seattle. While it is sometimes difficult to fully integrate an intern into an organization's operations, I observed that many of the engineering teams at Amazon also tend to operate in isolation from one another. In coordinating the resources required for my project, many of these teams had the opportunity to interact with one another. I do not think that this happens on a frequent enough basis such that Amazon can fully leverage the talent in its organization. From a project planning and leadership perspective, Amazon needs to take a few key steps to target this issue:

1. **Focus Clarity** – Amazon needs to identify the main areas of effort or focus that will be of the most benefit to their company. Their management and leadership need to not only identify these critical tasks, but also designate ownership of those tasks. It is then the responsibility of the task managers to identify all of the players

linked to their respective task. While it may seem like common sense to do so, it becomes important that all of those involved in the project meet on a routine basis to share information, coordinate efforts and benchmark task progress. This more structured environment may seem foreign to Amazon's software culture, but it is essential that the company adopt such a structure if it wishes to evolve as a high performance organization.

2. **Standardize Toolsets** – Another dynamic I observed is that the various engineering teams not only work in isolation from one another, but they all utilize different toolsets for their analysis. The problem this creates is that the work done by one team is rarely intuitive or of much use to other teams. This was the primary issue with Amazon's current capacity planning model in that the engineer who developed it was the only person who fully understands how to utilize it. While standardization may place some limitations on project engineers in the short term, the long term benefits is better coordination and integration of parallel efforts. The results of my model demonstrated the implications these findings had on many of the other concurrent studies throughout Amazon's operations. A standardized toolset could conceivably allow better sharing of information between different engineering teams, and allow one team to more easily port the data from another study into their own efforts.

3. **Tactical Vs. Strategic** – Operators at the FC level do not share the same perspective or focus as managers and analysts at the corporate level. To a certain extent, these differing perspectives are inevitable. However, I did observe a lack of communication and understanding between the tactical and strategic levels of Amazon's operations. The common attitude I encountered at the FC level was that corporate does not understand the challenges at the operator level. Similarly, I witnessed a similar attitude at the corporate level that FC operators do not always take into account the big picture. There is definitely a need for both cultures within the organization to better understand the concerns and focus of their counterpart. A training program or similar environment of information sharing could conceivably amend this problem.

Personal Lessons Learned – The analytical focus of my internship was highly rewarding, but the lessons I will take away from my experience reside more in the realm of project and resource planning:

1. **Understanding the Problem** – In tackling future projects, a hard lesson I learned on internship is the necessity to clearly understand and define the problem from day one. I was not nearly aggressive enough in gaining a better understanding of the project scope, and this mistake set a tone that resonated throughout the remainder of my internship. Projects certainly evolve and change over time, and it is not uncommon for the scope or focus to assume a whole new direction. However, a clear understanding of the initial problem makes such transitions far smoother and easier to implement.

2. **Resource Planning** – My undergraduate background is partially in Civil Engineering, yet I failed to utilize one of the most valuable toolsets from that engineering discipline. There is no reason why I should not have completed a Gantt and Pert analysis on my project in the initial few weeks. The value of such studies is that it establishes a timeline for the project and allows managers to identify the critical path. Similarly, it would have made resource requirements that emerged later in my internship more evident in the beginning. I was not able to validate my model due to a lack of data in the latter phases of my internship. Identifying this resource requirement back in June certainly could have enabled me to better plan and network these data requirements far earlier in the internship.

3. **Reporting** – Initially, I was very aggressive about documenting my work and providing status reports to my thesis advisors and internship supervisor. However, as the internship moved forward, I became increasingly complacent in providing these reports. It takes a concerted effort and discipline to maintain such a reporting scheme, but the benefits of this project management approach far outweigh the additional effort.

My understanding is that Amazon is currently in the process of implementing the model I developed into their operations. This is certainly encouraging in terms of defining the success of the internship. Yet some phases of the internship were more successful then others. From an analytical perspective, I learned a great deal about my strengths and limitations. The analytical structure of the model itself is quite robust, and from a design perspective, should prove to be of great value to Amazon. However, my comments also expose some of the project planning failings and limitations of my internship experience.

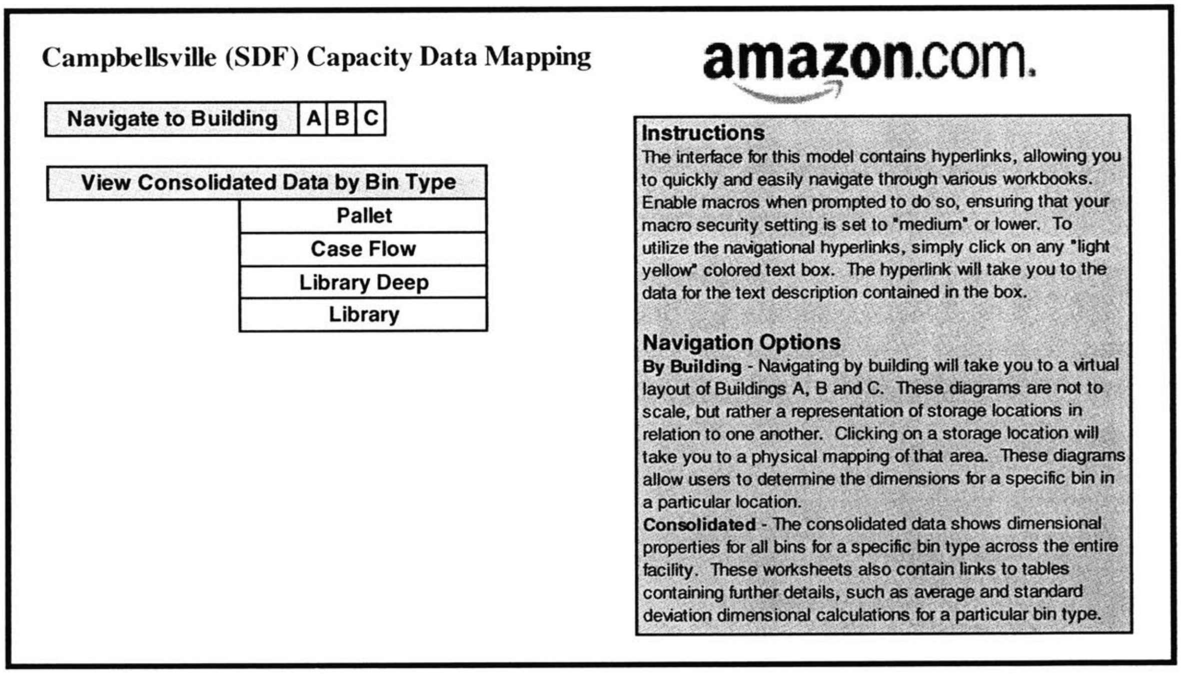
This document should serve as a helpful reference for anyone interested in gaining an understanding of model development and implementation, to include resource and timeline planning. My hope is that future LFM students, other academics in the MIT community and Amazon operators will be able to apply my work and findings such that it serves as a useful template for their own efforts.

Appendix A – Data Mapping Tables

As mentioned previously, my first deliverable to Amazon was a complete data mapping of the FC to which I was assigned. While I must shield the specific numbers for proprietary reasons, the actual design of data mapping effort may be of interest to some readers.

A.1 Data Mapping Start Page

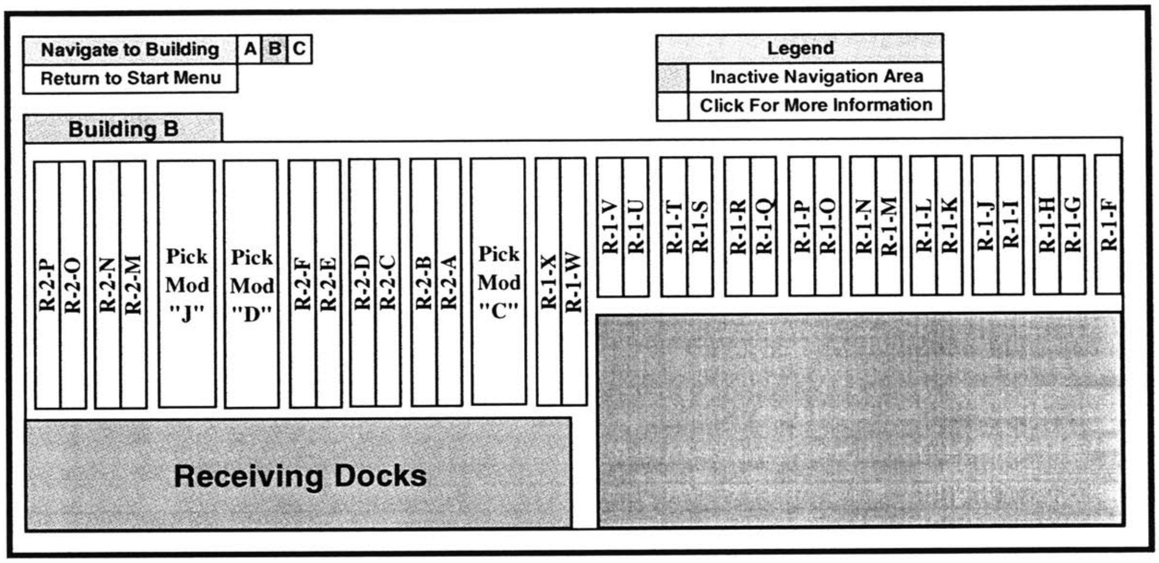
The navigation of the data mapping deliverable is driven entirely by macros. The design is such that users can choose to navigate to any specific building, or view consolidated data by bin type. I constructed the charts containing the analysis on bin variability from the consolidated data for the FC.


Campbellsville (SDF) Capacity Data Mapping
amazon.com.
Navigate to Building
A
B
C
View Consolidated Data by Bin Type
Pallet
Case Flow
Library Deep
Library

Instructions
The interface for this model contains hyperlinks, allowing you to quickly and easily navigate through various workbooks. Enable macros when prompted to do so, ensuring that your macro security setting is set to "medium" or lower. To utilize the navigational hyperlinks, simply click on any "light yellow" colored text box. The hyperlink will take you to the data for the text description contained in the box.

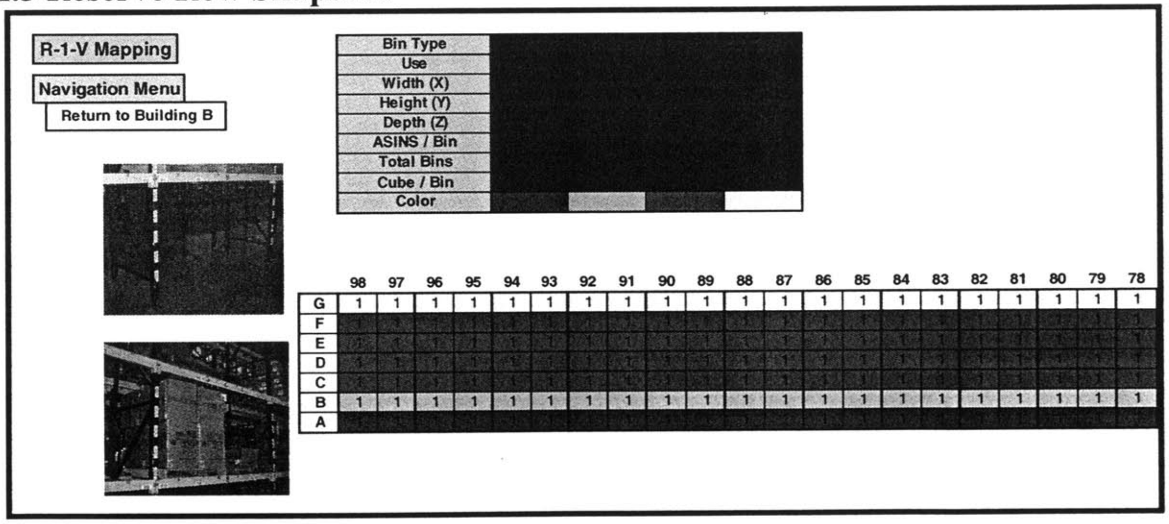
Navigation Options
By Building - Navigating by building will take you to a virtual layout of Buildings A, B and C. These diagrams are not to scale, but rather a representation of storage locations in relation to one another. Clicking on a storage location will take you to a physical mapping of that area. These diagrams allow users to determine the dimensions for a specific bin in a particular location.
Consolidated - The consolidated data shows dimensional properties for all bins for a specific bin type across the entire facility. These worksheets also contain links to tables containing further details, such as average and standard deviation dimensional calculations for a particular bin type.

A.2 Building Navigation Menu



The FC to which I was assigned was split into one of several buildings. Each respective building consisted of a mix of Reserve racks and Pick Mod bin storage. Users can click on each of the storage areas in the above diagram to see a specific layout of that storage area down to the bin level. For instance, if a user clicks on R-1-V, which is a reserve rack, they would be taken to the following screen:

A.3 Reserve Row Snapshot



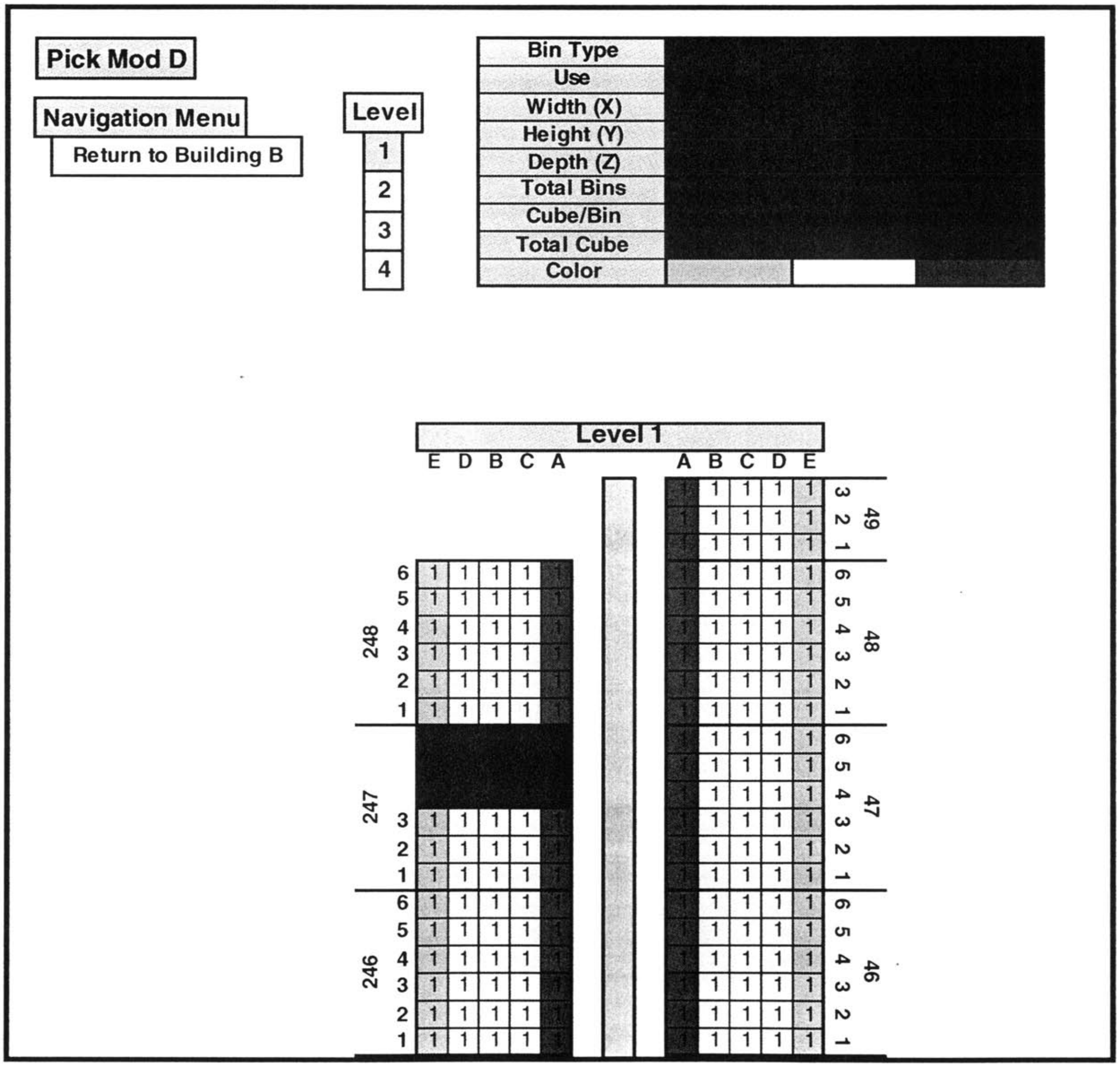
Bin Type	[redacted]			
Use	[redacted]			
Width (X)	[redacted]			
Height (Y)	[redacted]			
Depth (Z)	[redacted]			
ASINS / Bin	[redacted]			
Total Bins	[redacted]			
Cube / Bin	[redacted]			
Color	■	▨	■	□

	98	97	96	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	79	78
G	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
F	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
E	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
D	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
B	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
A	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

This interface provides all of the relevant information for this specific Reserve rack in the FC. Users have navigation options for returning to the Building diagram, or one level higher. While the actual data is currently blacked out, each storage area data mapping contains a table that summarizes the storage type and dimensional characteristics of that specific area. Similarly, the actual bin diagram would continue well across the right hand side of the page. I provided a snapshot so that readers can gain an appreciation of the data mapping design, but the complexity and scope of the actual deliverable is far more extensive then illustrated here.

Going back to the building diagram, if a user were to click on Pick Mod "D", they would see the following:

A.4. Pick Mod Snapshot



Bin Type	[redacted]
Use	[redacted]
Width (X)	[redacted]
Height (Y)	[redacted]
Depth (Z)	[redacted]
Total Bins	[redacted]
Cube/Bin	[redacted]
Total Cube	[redacted]
Color	▨ □ ■

		E	D	B	C	A
248	6	1	1	1	1	1
	5	1	1	1	1	1
	4	1	1	1	1	1
	3	1	1	1	1	1
	2	1	1	1	1	1
	1	1	1	1	1	1
247	3	1	1	1	1	1
	2	1	1	1	1	1
	1	1	1	1	1	1
246	6	1	1	1	1	1
	5	1	1	1	1	1
	4	1	1	1	1	1
	3	1	1	1	1	1
	2	1	1	1	1	1
	1	1	1	1	1	1

A	B	C	D	E		
1	1	1	1	1	3	49
1	1	1	1	1	2	
1	1	1	1	1	1	
1	1	1	1	1	6	48
1	1	1	1	1	5	
1	1	1	1	1	4	
1	1	1	1	1	3	
1	1	1	1	1	2	
1	1	1	1	1	1	
1	1	1	1	1	6	47
1	1	1	1	1	5	
1	1	1	1	1	4	
1	1	1	1	1	3	
1	1	1	1	1	2	
1	1	1	1	1	1	
1	1	1	1	1	6	46
1	1	1	1	1	5	
1	1	1	1	1	4	
1	1	1	1	1	3	
1	1	1	1	1	2	
1	1	1	1	1	1	

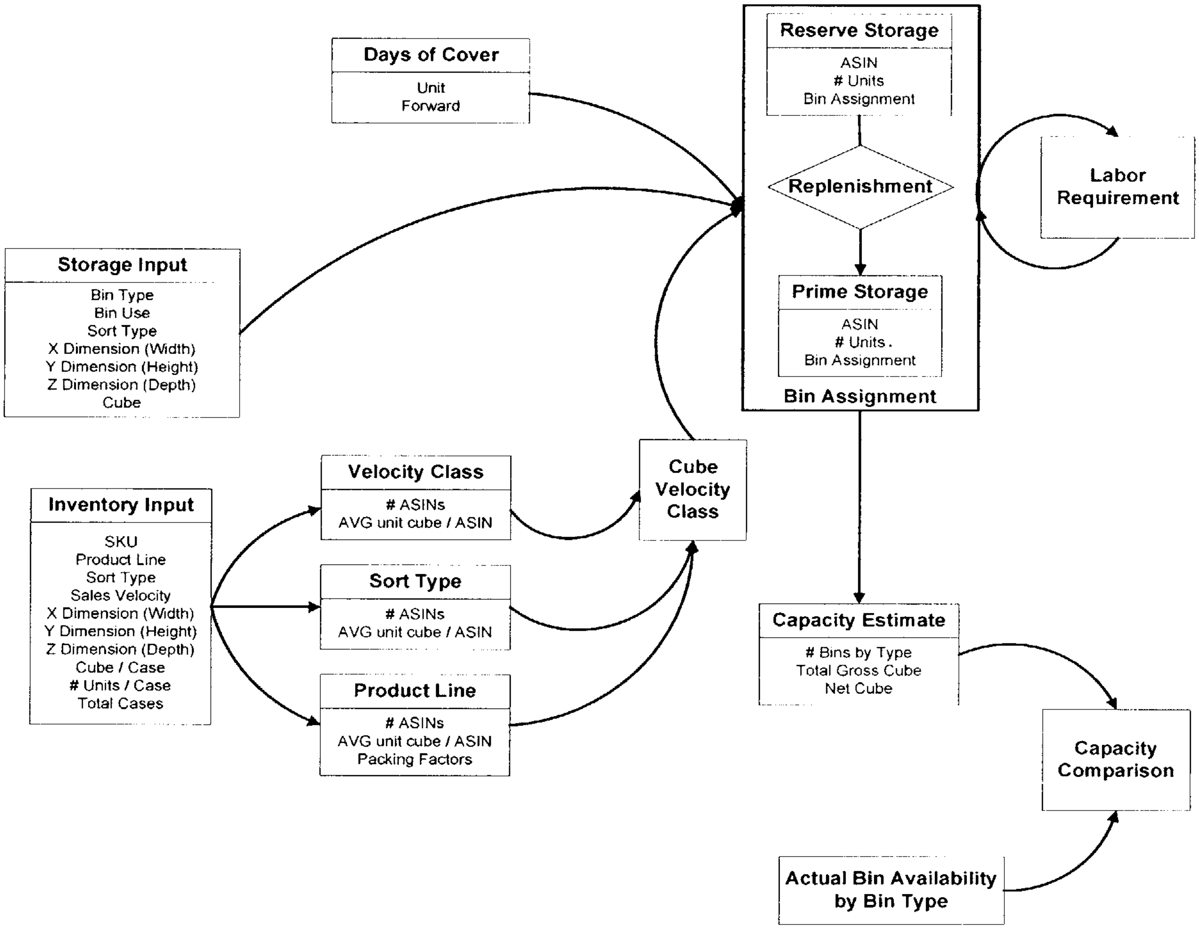
The format for the Pick Mods is very much like the Reserve rows, with one key exception. Pick Mods are several leveled structures, with Prime storage residing on each of the levels. Therefore, users have the flexibility to navigate between levels from the data mapping interface. Like the Reserve rows, the actual Pick Mod continues well beyond the limitations of the snapshot provided above, extending down the bottom of the diagram.

I cannot show snapshots of the consolidated data as it is too specific to Amazon's infrastructure. What a user would see on the consolidated sheets is a breakdown by storage area of the information embedded in each respective building of the FC.

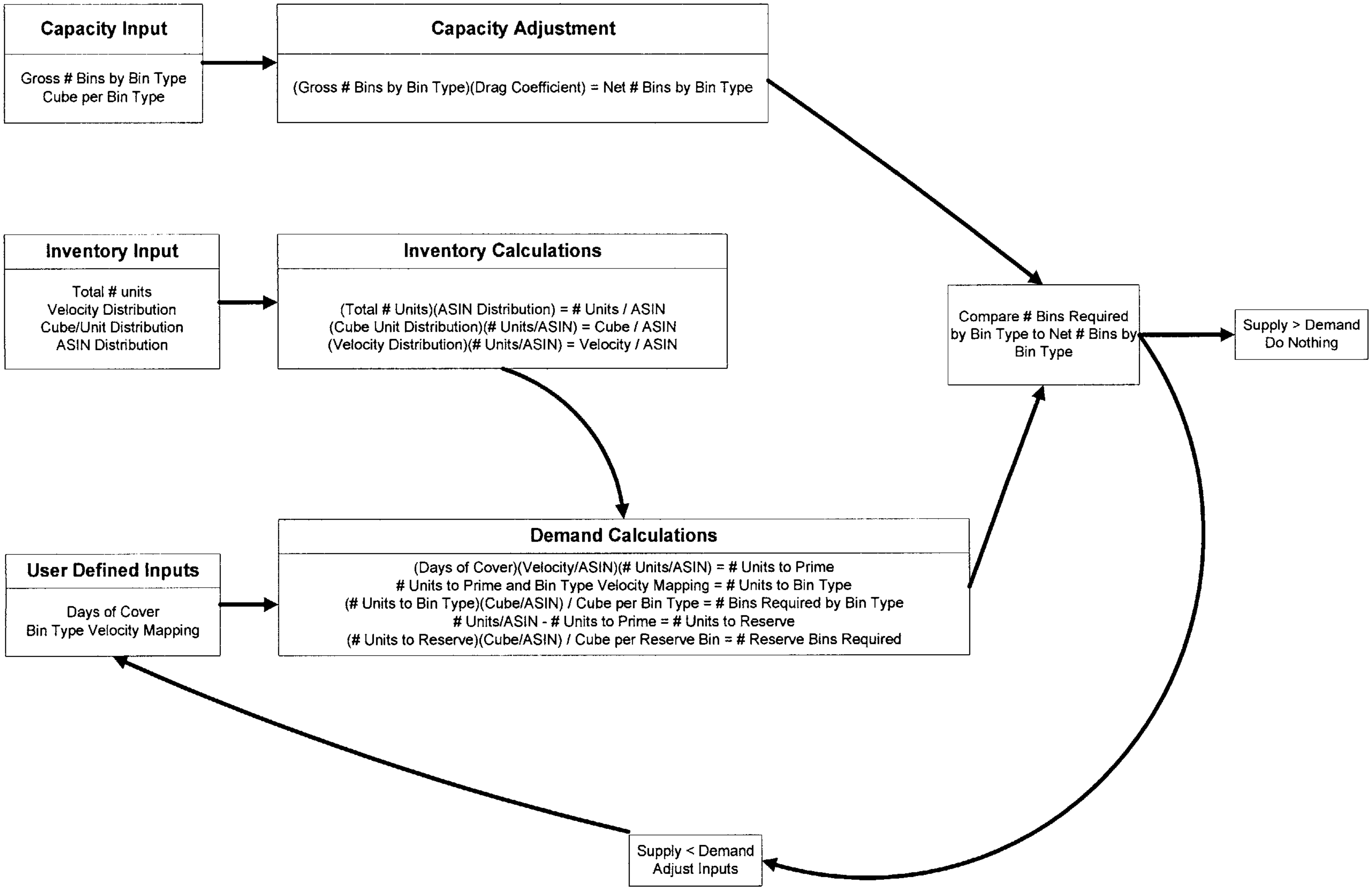
Appendix B – Model Logic Development

The model development went through several logic and flow chart iterations before settling on the logic flow implemented into the model. This Appendix contains the various versions of the model logic flow to document the iterative nature of the model's development. For those who wish to take the time to do so, it is interesting to examine the changes in the model logic from one iteration to the next.

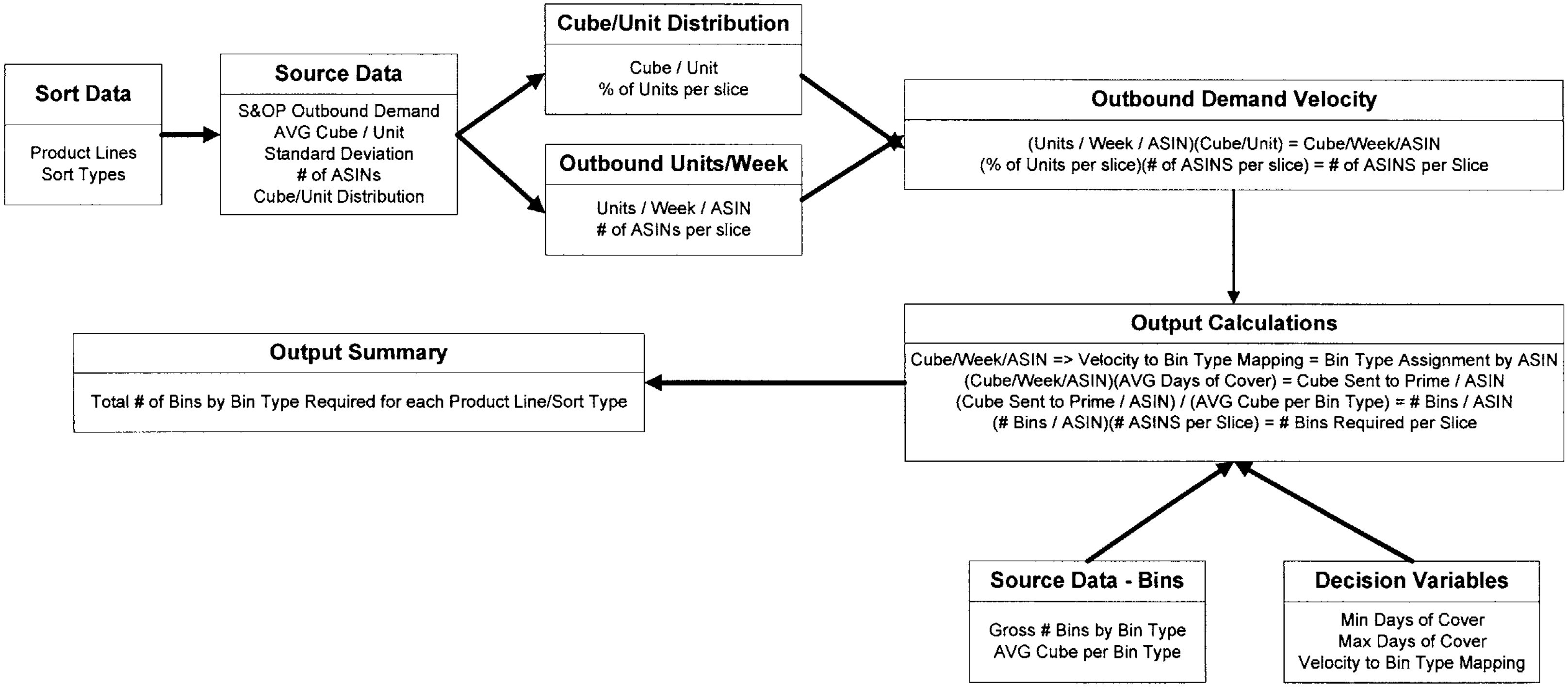
B.1 Model Flow Logic – Version 1



B.2 Model Flow Logic – Version 2

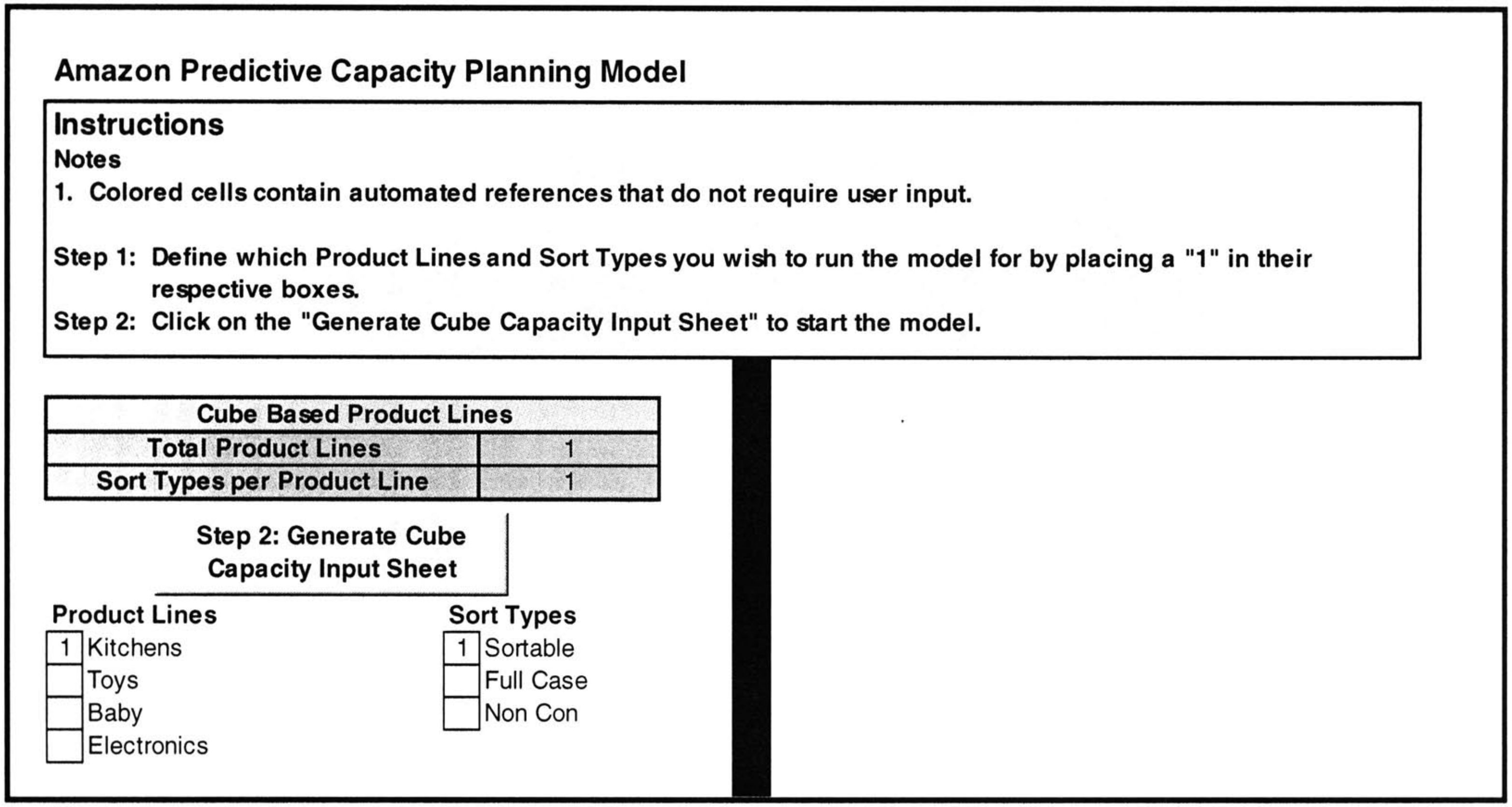

Capacity Input
Gross # Bins by Bin Type
Cube per Bin Type
Capacity Adjustment
(Gross # Bins by Bin Type)(Drag Coefficient) = Net # Bins by Bin Type
Inventory Input
Total # units
Velocity Distribution
Cube/Unit Distribution
ASIN Distribution
Inventory Calculations
(Total # Units)(ASIN Distribution) = # Units / ASIN
(Cube Unit Distribution)(# Units/ASIN) = Cube / ASIN
(Velocity Distribution)(# Units/ASIN) = Velocity / ASIN
Compare # Bins Required by Bin Type to Net # Bins by Bin Type
Supply > Demand
Do Nothing
User Defined Inputs
Days of Cover
Bin Type Velocity Mapping
Demand Calculations
(Days of Cover)(Velocity/ASIN)(# Units/ASIN) = # Units to Prime
# Units to Prime and Bin Type Velocity Mapping = # Units to Bin Type
(# Units to Bin Type)(Cube/ASIN) / Cube per Bin Type = # Bins Required by Bin Type
# Units/ASIN - # Units to Prime = # Units to Reserve
(# Units to Reserve)(Cube/ASIN) / Cube per Reserve Bin = # Reserve Bins Required
Supply < Demand
Adjust Inputs

B.3 Model Flow Logic – Version 3


Sort Data
Product Lines
Sort Types
Source Data
S&OP Outbound Demand
AVG Cube / Unit
Standard Deviation
# of ASINs
Cube/Unit Distribution
Cube/Unit Distribution
Cube / Unit
% of Units per slice
Outbound Units/Week
Units / Week / ASIN
# of ASINs per slice
Outbound Demand Velocity
(Units / Week / ASIN)(Cube/Unit) = Cube/Week/ASIN
(% of Units per slice)(# of ASINS per slice) = # of ASINS per Slice
Output Calculations
Cube/Week/ASIN => Velocity to Bin Type Mapping = Bin Type Assignment by ASIN
(Cube/Week/ASIN)(AVG Days of Cover) = Cube Sent to Prime / ASIN
(Cube Sent to Prime / ASIN) / (AVG Cube per Bin Type) = # Bins / ASIN
(# Bins / ASIN)(# ASINS per Slice) = # Bins Required per Slice
Output Summary
Total # of Bins by Bin Type Required for each Product Line/Sort Type
Source Data - Bins
Gross # Bins by Bin Type
AVG Cube per Bin Type
Decision Variables
Min Days of Cover
Max Days of Cover
Velocity to Bin Type Mapping

Appendix C – Model Screenshots of the Trial Run

C.1 Product Line and Sort Type Inputs



Amazon Predictive Capacity Planning Model

Instructions

Notes

1. Colored cells contain automated references that do not require user input.

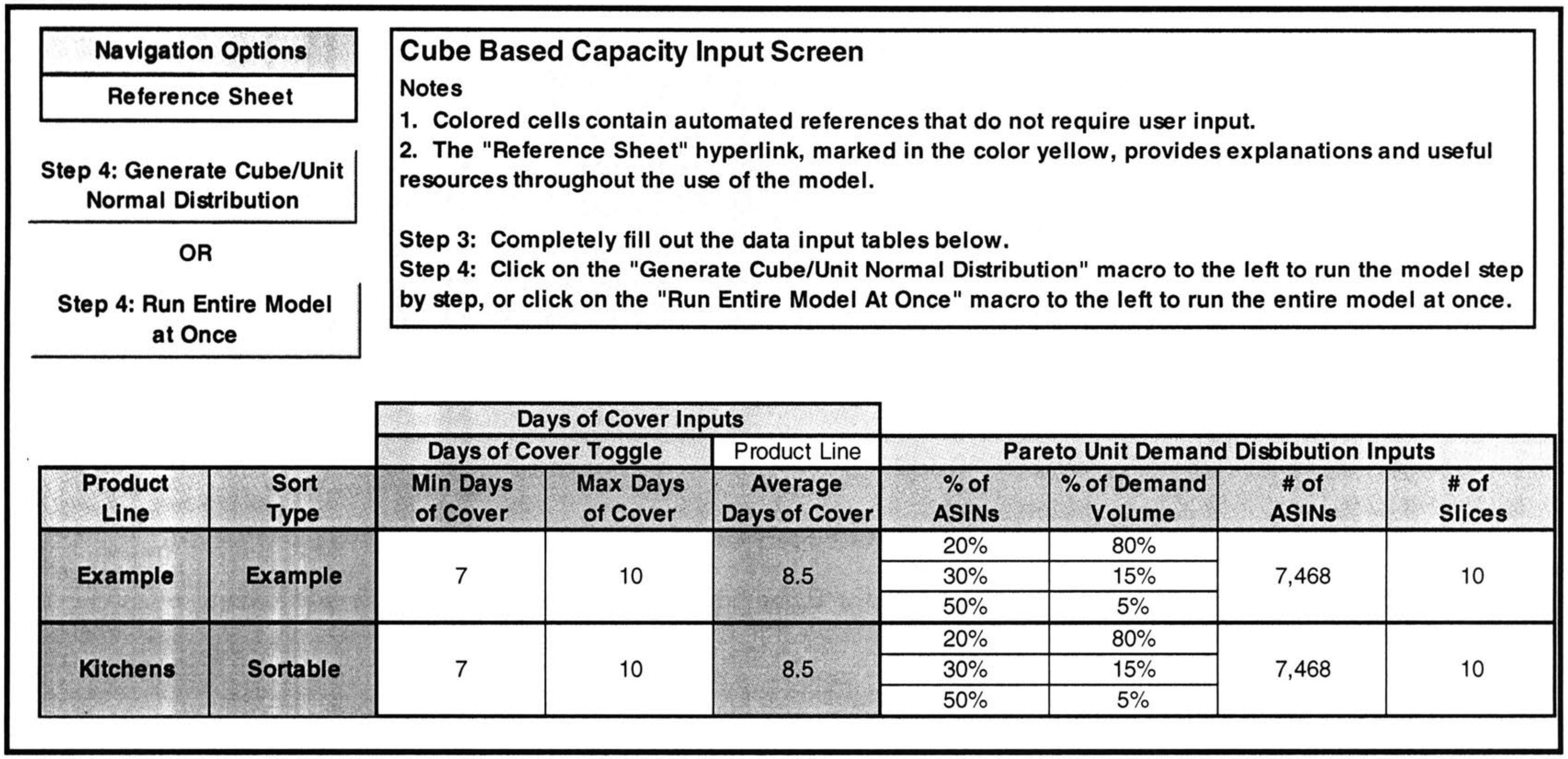
Step 1: Define which Product Lines and Sort Types you wish to run the model for by placing a "1" in their respective boxes.

Step 2: Click on the "Generate Cube Capacity Input Sheet" to start the model.

Cube Based Product Lines	
Total Product Lines	1
Sort Types per Product Line	1

Product Lines		Sort Types	
1	Kitchens	1	Sortable
□	Toys	□	Full Case
□	Baby	□	Non Con
□	Electronics		

C.2 Cube Capacity Input Sheet – Part 1



Navigation Options

Reference Sheet

Step 4: Generate Cube/Unit Normal Distribution

OR

Step 4: Run Entire Model at Once

Cube Based Capacity Input Screen

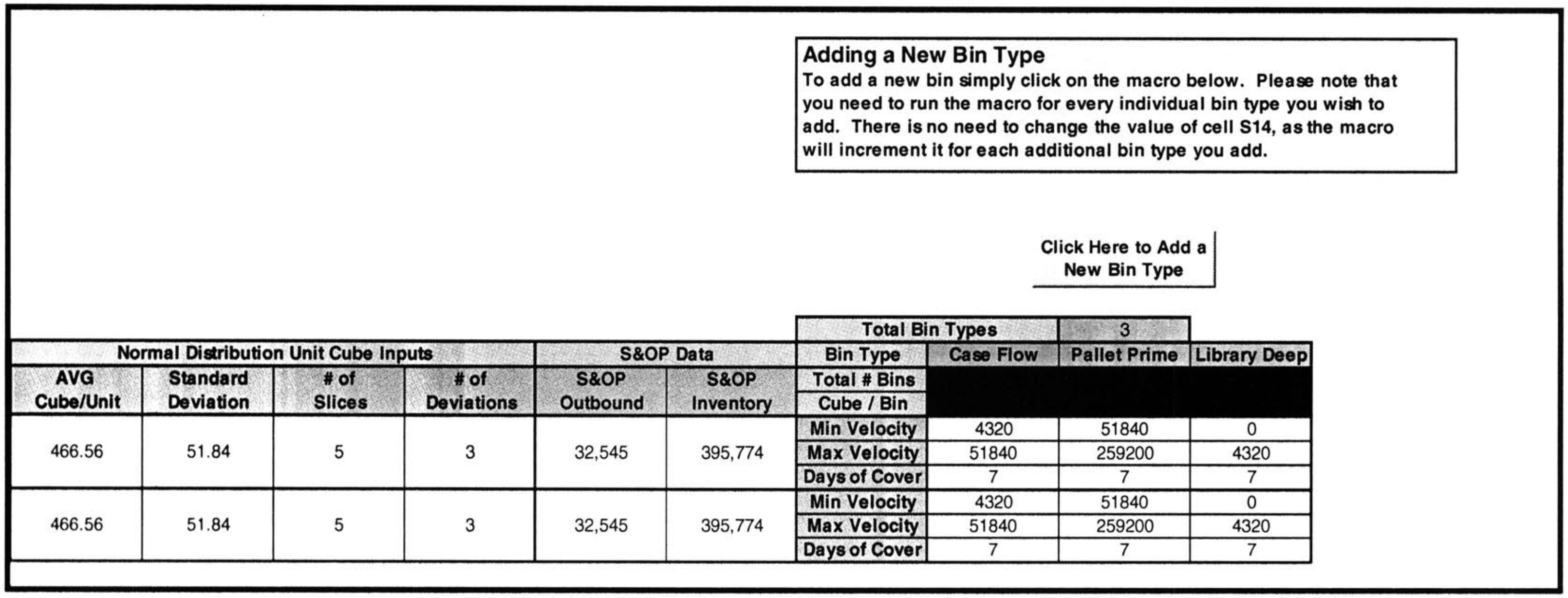
Notes

1. Colored cells contain automated references that do not require user input.
2. The "Reference Sheet" hyperlink, marked in the color yellow, provides explanations and useful resources throughout the use of the model.

Step 3: Completely fill out the data input tables below.
Step 4: Click on the "Generate Cube/Unit Normal Distribution" macro to the left to run the model step by step, or click on the "Run Entire Model At Once" macro to the left to run the entire model at once.

		Days of Cover Inputs						
		Days of Cover Toggle		Product Line	Pareto Unit Demand Disbibution Inputs			
Product Line	Sort Type	Min Days of Cover	Max Days of Cover	Average Days of Cover	% of ASINs	% of Demand Volume	# of ASINs	# of Slices
Example	Example	7	10	8.5	20%	80%	7,468	10
					30%	15%		
					50%	5%		
Kitchens	Sortable	7	10	8.5	20%	80%	7,468	10
					30%	15%		
					50%	5%		

C.3 Cube Capacity Input Sheet – Part 2



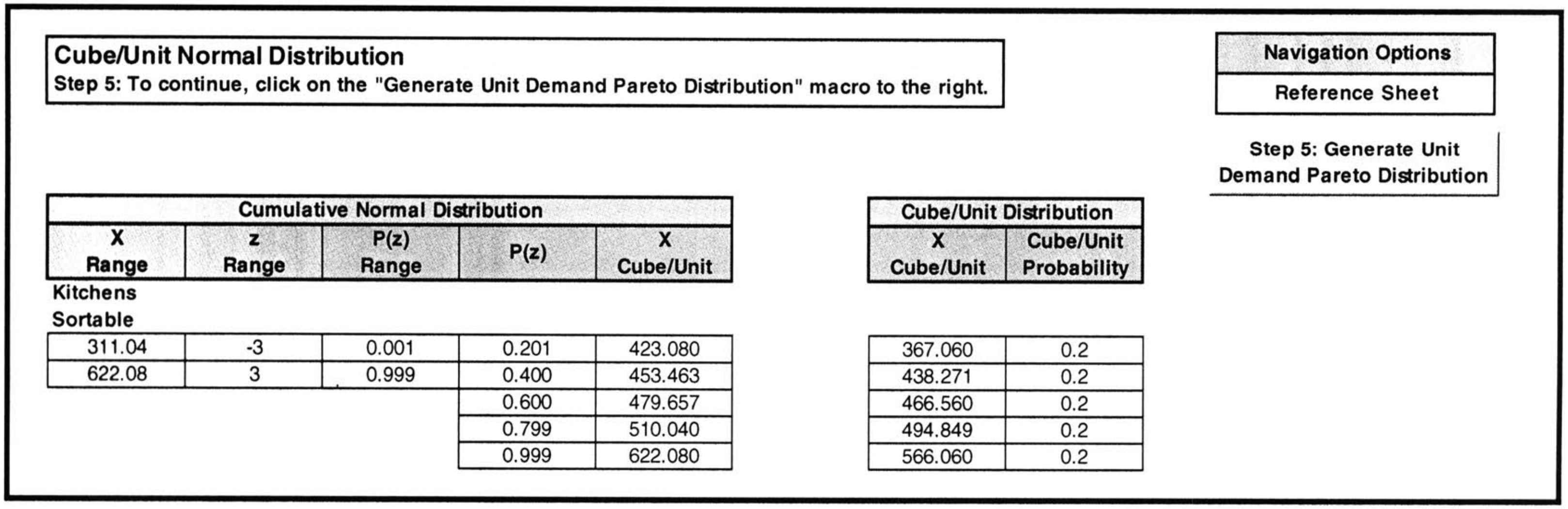
Adding a New Bin Type

To add a new bin simply click on the macro below. Please note that you need to run the macro for every individual bin type you wish to add. There is no need to change the value of cell S14, as the macro will increment it for each additional bin type you add.

Normal Distribution Unit Cube Inputs				S&OP Data	
AVG Cube/Unit	Standard Deviation	# of Slices	# of Deviations	S&OP Outbound	S&OP Inventory
466.56	51.84	5	3	32,545	395,774
466.56	51.84	5	3	32,545	395,774

Total Bin Types		3	
Bin Type	Case Flow	Pallet Prime	Library Deep
Total # Bins	[redacted]	[redacted]	[redacted]
Cube / Bin	[redacted]	[redacted]	[redacted]
Min Velocity	4320	51840	0
Max Velocity	51840	259200	4320
Days of Cover	7	7	7
Min Velocity	4320	51840	0
Max Velocity	51840	259200	4320
Days of Cover	7	7	7

C.4 Cube/Unit Normal Distribution



Cube/Unit Normal Distribution

Step 5: To continue, click on the "Generate Unit Demand Pareto Distribution" macro to the right.

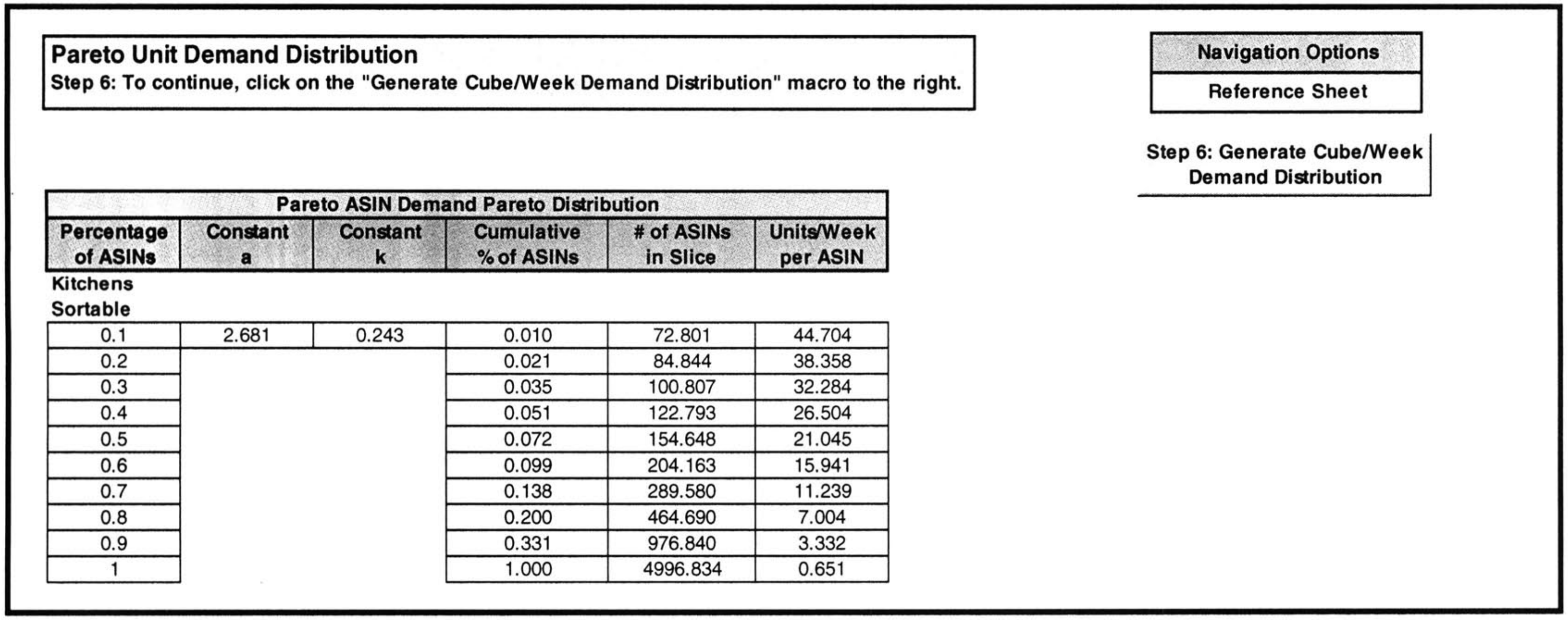
Navigation Options
Reference Sheet

Step 5: Generate Unit Demand Pareto Distribution

Cumulative Normal Distribution				
X Range	z Range	P(z) Range	P(z)	X Cube/Unit
Kitchens				
Sortable				
311.04	-3	0.001	0.201	423.080
622.08	3	0.999	0.400	453.463
			0.600	479.657
			0.799	510.040
			0.999	622.080

Cube/Unit Distribution	
X Cube/Unit	Cube/Unit Probability
367.060	0.2
438.271	0.2
466.560	0.2
494.849	0.2
566.060	0.2

C.5 Pareto Unit Demand Distribution



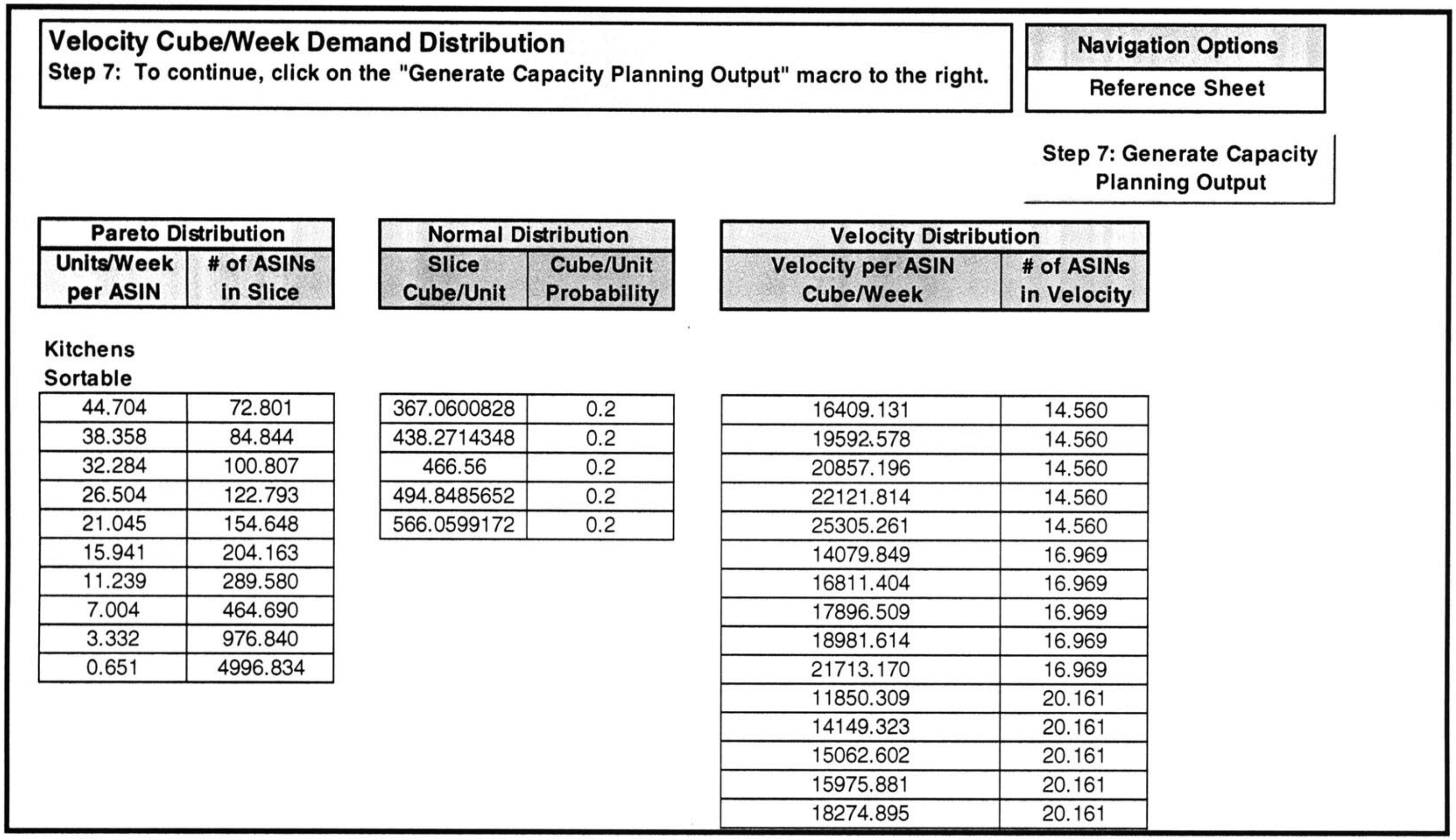
Pareto Unit Demand Distribution

Step 6: To continue, click on the "Generate Cube/Week Demand Distribution" macro to the right.

Pareto ASIN Demand Pareto Distribution

Percentage of ASINs	Constant a	Constant k	Cumulative % of ASINs	# of ASINs in Slice	Units/Week per ASIN
Kitchens Sortable					
0.1	2.681	0.243	0.010	72.801	44.704
0.2			0.021	84.844	38.358
0.3			0.035	100.807	32.284
0.4			0.051	122.793	26.504
0.5			0.072	154.648	21.045
0.6			0.099	204.163	15.941
0.7			0.138	289.580	11.239
0.8			0.200	464.690	7.004
0.9			0.331	976.840	3.332
1			1.000	4996.834	0.651

C.6 Velocity Cube/Week Demand Distribution



Velocity Cube/Week Demand Distribution

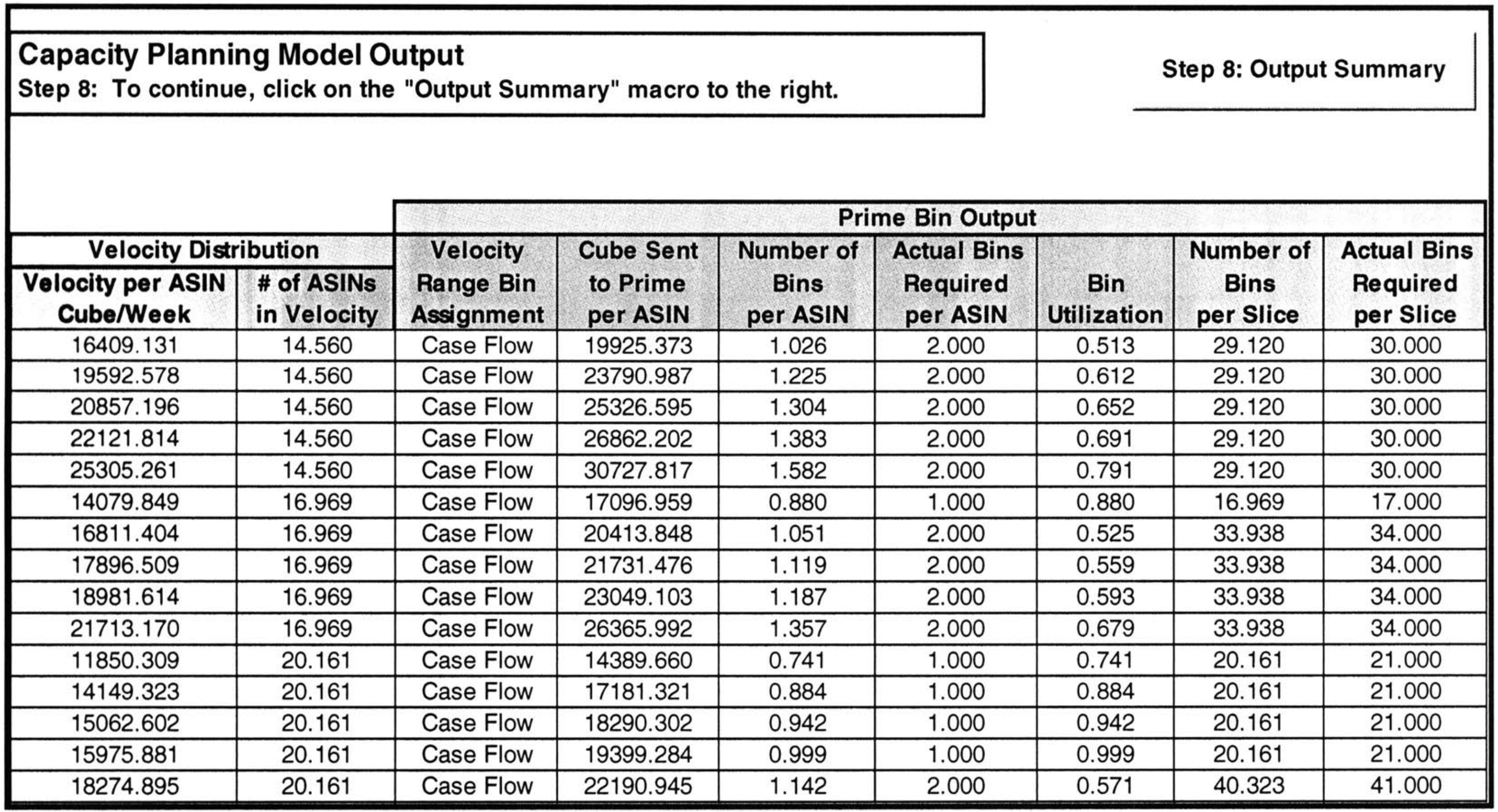
Step 7: To continue, click on the "Generate Capacity Planning Output" macro to the right.

Pareto Distribution	
Units/Week per ASIN	# of ASINs in Slice
Kitchens Sortable	
44.704	72.801
38.358	84.844
32.284	100.807
26.504	122.793
21.045	154.648
15.941	204.163
11.239	289.580
7.004	464.690
3.332	976.840
0.651	4996.834

Normal Distribution	
Slice Cube/Unit	Cube/Unit Probability
367.0600828	0.2
438.2714348	0.2
466.56	0.2
494.8485652	0.2
566.0599172	0.2

Velocity Distribution	
Velocity per ASIN Cube/Week	# of ASINs in Velocity
16409.131	14.560
19592.578	14.560
20857.196	14.560
22121.814	14.560
25305.261	14.560
14079.849	16.969
16811.404	16.969
17896.509	16.969
18981.614	16.969
21713.170	16.969
11850.309	20.161
14149.323	20.161
15062.602	20.161
15975.881	20.161
18274.895	20.161

C.7 Capacity Planning Model Output



Capacity Planning Model Output

Step 8: To continue, click on the "Output Summary" macro to the right.

Velocity Distribution		Prime Bin Output						
Velocity per ASIN Cube/Week	# of ASINs in Velocity	Velocity Range Bin Assignment	Cube Sent to Prime per ASIN	Number of Bins per ASIN	Actual Bins Required per ASIN	Bin Utilization	Number of Bins per Slice	Actual Bins Required per Slice
16409.131	14.560	Case Flow	19925.373	1.026	2.000	0.513	29.120	30.000
19592.578	14.560	Case Flow	23790.987	1.225	2.000	0.612	29.120	30.000
20857.196	14.560	Case Flow	25326.595	1.304	2.000	0.652	29.120	30.000
22121.814	14.560	Case Flow	26862.202	1.383	2.000	0.691	29.120	30.000
25305.261	14.560	Case Flow	30727.817	1.582	2.000	0.791	29.120	30.000
14079.849	16.969	Case Flow	17096.959	0.880	1.000	0.880	16.969	17.000
16811.404	16.969	Case Flow	20413.848	1.051	2.000	0.525	33.938	34.000
17896.509	16.969	Case Flow	21731.476	1.119	2.000	0.559	33.938	34.000
18981.614	16.969	Case Flow	23049.103	1.187	2.000	0.593	33.938	34.000
21713.170	16.969	Case Flow	26365.992	1.357	2.000	0.679	33.938	34.000
11850.309	20.161	Case Flow	14389.660	0.741	1.000	0.741	20.161	21.000
14149.323	20.161	Case Flow	17181.321	0.884	1.000	0.884	20.161	21.000
15062.602	20.161	Case Flow	18290.302	0.942	1.000	0.942	20.161	21.000
15975.881	20.161	Case Flow	19399.284	0.999	1.000	0.999	20.161	21.000
18274.895	20.161	Case Flow	22190.945	1.142	2.000	0.571	40.323	41.000

References

Pareto Principle. From Wikipedia, the Free Encyclopedia.
http://en.wikipedia.org/wiki/Pareto_principle.

Sheriff, Paul. *Paul Sheriff Teaches Visual Basic 6.* Que Publishing: Indianapolis, Indiana, 1999.

Walkenbach, John. *Microsoft® Excel 2000 Power Programming with VBA.* Hungry Minds, Inc.: New York, NY, 1999.

Weisstein, Eric W. *Normal Distribution.* From MathWorld – A Wolfram Web Resource.
http://mathworld.wolfram.com/NormalDistribution.html.

Weisstein, Eric W. *Pareto Distribution.*" From Mathworld-A Wolfram Web Resource.
http://mathworld.wolfram.com/ParetoDistribution.html.

1:36-45