User's Guide for BaseStockSim Software version 2.4

Lawrence V. Snyder
Department of Industrial and Systems Engineering
Lehigh University
Bethlehem, PA, USA

May 25, 2006

Contents

1 Introduction

The BaseStockSim software allows users to simulate multi-stage inventory systems with reliable or unreliable stages. Despite its name, BaseStockSim can accommodate several types of inventory policies, including base-stock, (R, Q), and (s, S) policies, as well as a modified (R, Q) policy in which the value of Q is chosen dynamically based on the values of the state variables. For base-stock policies, base-stock levels are set by the user explicitly but may also be interpreted using the "committed service time" notion of so-called guaranteed-service models (see, e.g., [2, 4, 8]).

A BaseStockSim model consists of one or more *stages*, each of which consists of both a processing function and an output buffer that stores inventory. A stage may represent a physical location (such as a manufacturing facility or warehouse), a processing activity (such as production, assembly, or testing), or an SKU (such as a raw material, component, or end product). The inventory in a stage's output buffer is referred to as *finished-goods inventory* because it consists of items that have completed processing at that stage. However, these items can function as raw materials for downstream stages. In this way, a BaseStockSim model can represent the distribution of goods through a physical network or the assembly of one or more end products in a bill-of-materials (BOM) network.

BaseStockSim can also be used to model systems that face disruptions. Each stage in the model can experience disruptions during which no processing can take place and no finished-goods inventory

can be removed. You can specify each stage's disruption probability or set it to 0 for reliable stages.[1]

BaseStockSim has been tested using Windows XP but is expected to run smoothly under other recent versions of Windows.

This guide assumes some basic knowledge of inventory theory on the part of the reader. For introductory reading on this topic, see [6, 7]. For more advanced reading, see [1, 10].

2 Getting Started

2.1 What's New in Version 2.4

BaseStockSim v2.4 has a number of features that were not available in the previous public release, v2.0. Here are some of the main new features:

- Stages have several additional input parameters that describe their behavior. For example, [Proc Capac] imposes a processing capacity; [Fixed Cost] represents a cost incurred for each order placed to the stage, regardless of order size; and [Recovery Prob] allows the disruption probability to depend on whether a disruption occurred in the previous period.
- If a stage has multiple supplier stages, the supplier stages can be configured so that the downstream stage needs one of *each* upstream product (an AND relationship) or so that it needs *any* one of the upstream products (an OR relationship), or some combination of the two. OR relationships allow a stage to have redundant suppliers, each with different costs; more expensive suppliers are used only when cheaper suppliers are disrupted or have reached their capacity.
- Multiple simulation runs can be performed in sequence using the new Batch Runs feature, which provides a flexible way to test many values of the parameters for each stage. You can customize the output reporting by choosing which performance measures (costs, inventory levels, etc.) and statistics (mean, SEM, min, max) to display.
- A new type of inventory policy, called "Order Quantity," is like an (R, Q) policy except that Q is calculated dynamically, based on a function rather than a constant. To formulate this function, you specify the coefficients for a number of state variables. You can use this function to model a wide range of ordering behaviors. For example, you can model stages that over- or under-weight their pipeline inventory when choosing order quantities (as in the well known beer game [9]) or stages that adjust their order quantities based on whether their supplier experienced a disruption or stockout in the previous period.
- You can now specify a warm-up period for the simulation; performance measures are only recorded once the warm-up period ends.

[1] The terms "disruption" and "failure" are used interchangeably in BaseStockSim and this guide.

The software has also been enhanced and cleaned up in a number of more minor ways. A full list of changes in this version is given in Section 8.4.

2.2 Installing the Software

To install the software, simply drop the application file (`BaseStockSim.exe`) into a folder on your hard drive. The file `BSSTVOF.ini` must be placed in the same folder. The sample model files (Section 7) and Batch Runs templates (Section 6.1.2) can go anywhere you like.

BaseStockSim runs under Windows XP. It probably also runs under other recent versions of Windows but has not been tested under anything other than XP.

2.3 Conventions in This Guide

The following typesetting conventions are used in this user's guide:

- Input parameters (data provided for each stage in the model) are indicated with square brackets: [Proc Time], [Dem Mean], etc.
- Performance measures (data reported based on the results of the simulation) are indicated with pointy brackets: ⟨Total Cost⟩, ⟨Inv Level⟩, etc.
- Buttons that you can click in BaseStockSim are written in boldface: **Run Simulation**, **Clear All**, etc.
- Filenames are written in typewriter font: `BaseStockSim.exe`, `commonpart3.txt`, etc.

3 The Underlying Model

3.1 Time

BaseStockSim is a periodic-review model, which means that everything happens in discrete type buckets rather than in continuous time. The demands, capacities, and other parameters each refer to quantities per period. Events occur in each period following a specific sequence of events (Section 3.5.2). The model runs for a finite number of periods, which you specify.

3.2 Stages

A BaseStockSim model consists of one or more *stages*, each of which consists of a processing function and an output buffer containing finished-goods inventory. Each stage operates as a make-to-stock

system, with the target stock level determined by the inventory policy you specify. Of course, if the target inventory level equals 0, then the stage operates as a make-to-order system.

Stages receive orders from their downstream *customers*, which include other stages in the model as well as external customers whose demands are generated randomly according to parameters that you specify. These are refereed to as *internal* and *external* customers, respectively.

Stages also place orders to their *suppliers* to replenish their inventory. Each stage is also a supplier to itself since some processing takes place at the stage before the inventory can be replenished. You specify the supplier-customer relationships in the model by linking stages to one another (see Section 4.1).

3.3 Lead Times

Lead times in the BaseStockSim model occur at the stages rather than on the links. You should think of these as processing times (in fact, that's what they're called in the software) rather than as transportation lead times.

The transportation lead times between stages are always equal to zero. However, if you want to model non-zero transportation lead times, you can simply add the transportation lead time to the processing time of the downstream stage. (In some cases, this trick does not work—for example, if a stage's suppliers each have different lead times, or if a WIP holding cost is charged on in-process inventory but not on in-transit inventory. In cases such as these, you can add dummy stages that represent the transportation activity and have processing times equal to the transportation lead time.)

3.4 The Demand Process

All events in BaseStockSim are ultimately triggered by demands; if no stages have a mean demand greater than 0, then there is nothing to simulate. Demands are generated randomly according to a normal distribution with parameters that you specify. You can set the standard deviation to 0 to generate deterministic demands.

BaseStockSim has a feature for truncating demands if they exceed a bound that you specify. In particular, you can specify a constant k (called [Dem Bound] in the Parameters list) so that the total demand in any τ consecutive periods is bounded by $\mu\tau + k\sigma\sqrt{\tau}$, where μ and σ are the mean and standard deviation of the single-period demand. If the demand generated in period t would violate this bound for the τ periods $t-\tau+1,\ldots,t-1,t$ for *any* value of τ, then the demand in period t is truncated to satisfy the bound. Demands are never truncated below the mean, only above. This truncation strategy is used in the literature on guaranteed-service models [2, 3, 4, 8].

To prevent demands from being truncated, simply set $k = \infty$ ([Dem Bound] = INF).

BaseStockSim only handles integer numbers of items, so demands are always assumed to be integer. The demand parameters (mean, standard deviation, and bound) may be given as real numbers, but the demand generated will be rounded to the nearest integer.

3.4.1 A Note About Demand Sizes

In general, as the number of items "in play" in each period increases, the run time of your simulation will also increase. Therefore, it is preferable to keep your demand sizes as small as possible. If your stages typically have demands greater than 100 or so, you may want to divide the demand means by a constant throughout your model, and divide the demand standard deviations, processing capacities, and so on accordingly. Such a change is analogous to modeling batches of items (e.g., cases, palettes) rather than individual items.

3.5 The Replenishment Process

At a high level, the replenishment process is straightforward and familiar. However, the mechanics of the system are a bit more subtle and will be described in detail in this section. In some cases, a thorough understanding of this process may be necessary to build and analyze an accurate model.

In each time period, a given stage observes demands from its customers (internal and external) and then places replenishment orders to its suppliers (including itself). The size of the order is based on the observed demand and the stage's inventory policy. (See Section 3.6.)

If the supplier stages have inventory available, those items are immediately removed from the suppliers' inventory and delivered to the stage that placed the order, which begins processing them. Those items are released from processing and placed in the stage's inventory buffer after a number of periods equal to the processing time have elapsed.

If the inventory at the supplier stages falls short of the stage's order quantity, the supplier inventories are depleted as much as possible, and the excess items are placed on a backorder list at the ordering stage. In the next period, the stage will again attempt to obtain items from its suppliers to process these backordered items (as well as any new demands). However, backordered items do not factor into the next period's order quantity since a replenishment order has already been placed to the suppliers for these items.

Thus, an order really consists of two parts: A replenishment *order* that triggers processing at the supplier stages, and a *withdrawal* of inventory from the supplier stages' buffers. These two parts are related but separate, and the quantity ordered from a given supplier in the first part may be different from the quantity withdrawn from that supplier at the second part. However, each replenishment order is tagged to the stage that triggered it, so that when that item is placed in the upstream stage's inventory buffer, it can only be used by the stage that originally ordered it.

This replenishment process may be somewhat counterintuitive if you are used to thinking about inventory systems with lead times that occur on the links. In those systems, an order is placed in one period and then arrives L periods later. In contrast, in BaseStockSim, when an item leaves a stage it arrives immediately at the downstream stage. However, this behavior is consistent with the notion of a make-to-stock system, in which customers withdraw demands from on-hand inventory to the extent possible, excess demands are backordered, and demands trigger replenishment orders that are added to finished-goods inventory a number of periods later. The tagging of replenishment items to the stage that ordered them ensures a first-come, first-served inventory policy at the upstream stage.

3.5.1 Example

To illustrate the replenishment process, consider a system with two stages that see customer demand (labeled 1 and 2) and a third stage (stage 3) that supplies both downstream stages. Stages 1 and 2 each see (deterministic) demand with a mean of 10 items per period. All stages follow base-stock policies.

Suppose that, in period 1, stage 3 has 15 items on hand. Stages 1 and 2 each experience a demand of 10 units and order 10 units from stage 3. Stage 1 orders first, so 10 units from 3's output buffer are immediately removed and transferred to stage 1. Processing of these 10 units begins at stage 1.

Next, stage 2 places an order for 10 units from stage 3. The 5 units remaining in 3's inventory buffer are removed and transferred to stage 2, and 5 more units are backordered.

At the end of this period, 20 units begin processing at stage 3 (the total replenishment order), 10 units begin processing at stage 1, and 5 units begin processing at stage 2 (since stage 3 had insufficient inventory).

In period 2, stages 1 and 2 again place orders for 10 units. The 20 units that begin processing at stage 3 are completed and added to its inventory buffer. Stage 1 takes its 10 units and begins processing. Stage 2 needs 15 units (10 for the current demand and 5 for the backordered units), but only 10 are available. The first 5 are used to clear the existing backorder while the remaining 5 are used to process new demands, and 5 new units are added to the (now-empty) backorder list.

In this example, stage 1 always meets its demand while stage 2 always has a backlog of 5 units. This is a bit surprising since stage 1 and 2 appear to be identical, but it is a natural consequence of the replenishment rule and the fact that stages with smaller indices have priority over those with larger indices.

Now suppose that, in period 2, stage 1 experiences a demand of 15 instead of 10. Although stage 3's output buffer contains 20 units, stage 1 is only allowed to take 10 of them, since only 10 are tagged to stage 1. This rule enforces the FCFS behavior; without it, stage 1 could "steal" items that were ordered by stage 2, preventing stage 2 from clearing its backlog.

3.5.2 Sequence of Events

To summarize, the sequence of events at a given stage in each period is as follows:

1. The stage receives orders from its downstream stages and its external customer, if any.
2. The stage begins places an order to its suppliers, if any, according to its inventory policy and parameters.
3. The stage begins production of a number of units equal to the size of the order it placed to its suppliers, if any. Production can only begin if the stage's supplier(s) have inventory available, since this inventory serves as raw materials for the stage's production process. If the stage's supplier(s) do not have sufficient inventory, excess production orders are queued until a future period when upstream raw materials are available. Inventory is released to downstream stages first-come, first-served.
4. Items whose production began the correct number of time periods earlier are released from production into inventory.
5. Demands (observed in step 1) are satisfied and/or backordered.
6. Holding and penalty costs are incurred.

3.6 Order Quantity

The order quantity in each period is determined by the demand faced by the stage in that period (including orders placed by customer stages) and by the stage's inventory policy.

Before discussing the types of inventory policies and the decision rule for choosing suppliers, we define a few terms used throughout this section. Each refers to an inventory measure at each stage, and each is a random state variable.

On-hand inventory (OH) The number of items in the stage's output inventory buffer.

Backorders (BO) The number of items that have been demanded by the stage's (internal and external) customers but have not yet been delivered.

In-process inventory (IPR) The number of items that have begun processing or are waiting to begin processing at the stage, as described in Step 3 in Section 3.5.2.

Inventory position (IP) Equal to on-hand inventory minus backorders plus in-process inventory:

$$IP = OH - BO + IP.$$

Inventory level (IL) Equal to on-hand inventory minus backorders:

$$IL = OH - BO.$$

3.6.1 Inventory Policies

BaseStockSim provides four inventory policies that stages can follow.

Base-Stock Policy In each period, the stage places an order of sufficient size to bring its inventory position (IP) up to a target IP, called the *base-stock level*. Assuming the base-stock level is stationary over time (as it is in BaseStockSim), this means that the stage always places an order whose size is equal to the total demand observed by the stage in that period.

(R, Q) **Policy** When the stage's IP hits or falls below the *reorder point*, denoted R, it places an order whose size is a multiple of the *order quantity*, Q, so that the resulting IP is between R and $R + Q$. In general this means placing an order of size Q, but if the demand in the period is larger than Q, the resulting order may equal $2Q$, $3Q$, etc.

(s, S) **Policy** When the stage's IP hits or falls below the *reorder point*, denoted s, it places an order of sufficient size to bring its IP to the *order-up-to level*, S. If $s = S - 1$, the policy is equivalent to a base-stock policy.

Order Quantity Policy This policy works like an (R, Q) policy except that the order quantity is not a constant (Q) but rather is set dynamically using the current values of the state variables. You specify the coefficients of a wide range of state variables and BaseStockSim uses this to compute the order quantity. You can also specify a reorder point so that an order is only triggered if the IP falls below this point. This type of policy can be used to model a number of ordering behaviors that do not fall under any of the other policies. More details about the parameters of the order quantity policy are given in Section 3.7.

3.6.2 Supplier-Selection Rule

Raw Materials and Suppliers Each stage may require one or more raw materials to produce an item. The stage's raw materials are determined by the [Product Index] field of the suppliers. BaseStockSim ranks the suppliers for each raw material in increasing order of their [Product Cost] field. [2]

For example, suppose that stage 1 has the following suppliers:

[2] *Note*: The [Fixed Cost] field is not used in this ranking. Suppliers are only ranked according to their per-unit costs.

Supplier	Product Index	Product Cost
2	2	0.50
3	2	0.40
4	3	0.75
5	2	0.60
6	3	0.90
7	4	1.20

Then stage 1 has three raw materials (with indices 2, 3, and 4). Raw material 2 has three suppliers (in ranked order, they are 3, 2, 5), raw material 3 has two suppliers (4, 6), and raw material 4 has one supplier (7).

As discussed in Section 3.5, an order consists of two parts: a replenishment order to the suppliers and a withdraw of inventory from the suppliers. We next describe the supplier-selection rule that governs each of these parts.

Orders Once the order quantity is determined, the stage places orders for its raw materials. For each raw material, it first tries to place an order with the top-ranked (i.e., cheapest) supplier. If that supplier is unavailable (as described below), it places an order to its second-ranked supplier, then its third-ranked supplier, and so on. If all suppliers for a given raw material are unavailable, any remaining items are ordered from the top-ranked supplier, even though that supplier is unavailable.

A supplier is considered "unavailable" if *either* it has failed in the current period *or* its total orders received in the current period (from the current stage or its other downstream customers) equal or exceed its processing capacity ([Proc Capac]).

To continue the example given above, suppose that the supplier stages have the following processing capacities and failure states:

Supplier	Product Index	Proc Capac	Failed State
2	2	50	Failed
3	2	30	Not Failed
4	3	60	Not Failed
5	2	60	Not Failed
6	3	60	Not Failed
7	4	50	Not Failed

Suppose stage 1 wishes to place an order of size 100. The quantities placed to each supplier are as follows:

- *Raw Material 2.* The top-ranked supplier is stage 3, which has a capacity of 30, so the stage orders 30 units from 3. The next-ranked supplier is stage 2, which has failed, so the

stage moves to the third-ranked supplier, stage 5. The capacity of stage 5 is 60, leaving an additional 10 units, which the stage orders from stage 3. Therefore, the stage orders 40 from 3, 0 from 2, and 60 from 5.

- *Raw Material 3.* The top-ranked supplier is stage 4, so the stage maxes out stage 4's capacity (60) and then orders the remaining 40 units from stage 6.
- *Raw Material 4.* Although stage 7's capacity is only 50, the stage must order all 100 units from stage 7 since it has no other suppliers for this raw material.

Raw Material Withdrawal Items are withdrawn from each raw material's suppliers in order of their ranking, skipping any suppliers that have failed. BaseStockSim first determines the total quantity of each raw material that *could* be withdrawn and then withdraws the *minimum* of these quantities (to avoid ordering raw materials that cannot yet be used because of stockouts of other raw materials).

To complete the example, suppose that the suppliers have the following quantities on-hand:

Supplier	Product Index	On-Hand Inventory
2	2	40
3	2	70
4	3	50
5	2	50
6	3	30
7	4	60

Stage 1 needs to withdraw 100 items for each raw material. Therefore, it could withdraw 70 items of raw material 2 from supplier 3 (its top-ranked supplier); it skips supplier 2 since it has failed; and it could withdraw the remaining 30 units from supplier 5, for a total of 100 units. For raw material 3, it could withdraw 50 units from supplier 4 and 30 units from supplier 6, for a total of 80. Finally, it could withdraw 60 units of raw material 4 from supplier 7, for a total of 60.

There is no reason to order more than 60 units of raw materials 2 or 3 since there are no corresponding units of raw material 4 to pair with them. Therefore, the stage orders 60 units of each raw material: 60 of 2 from stage 3; 50 of 3 from stage 4 and 10 of 3 from stage 6; and 60 of 4 from stage 7.

3.7 The Order Quantity Policy

Like the (R, Q) policy, the order quantity policy involves placing an order of size Q when the inventory position hits or falls below a reorder point, R. However, whereas Q is a constant in the (R, Q) policy, Q is a function of the random state variables in the order quantity policy. This

function consists of several terms, each of which has a coefficient (which you specify) times a state variable (which BaseStockSim calculates in each period).

In fact, the order quantity policy is general enough that you can use it to model a base-stock or (s, S) policy in which the order-up-to level is computed as a function rather than as a constant. For more details, see Section 3.7.2.

You specify the coefficients and other parameters of the order quantity function in the Parameters list (see Section 4.2). Each of these parameters has the prefix "OQ" to remind you that it is used only for the order quantity function. Most of the parameters are coefficients of terms in the function; these have the suffix "Cf" for "coefficient." A few other parameters are not coefficients. All coefficients and other parameters are described in Section 3.7.1 below.

3.7.1 Order Quantity Function

The order quantity function is given as follows. All state variables refer to stage i in period t unless otherwise specified. Numbers in parentheses refer to notes given after the function.

Order quantity placed by stage i in period t =
[OQ Constant]
+ [OQ Demand Cf] × demand (1)
+ [OQ Forecast Cf] × demand forecast for derived demand in period $t + 1$ (2)
+ [OQ IP Cf] × inventory position *after* demand is observed
+ [OQ IL Cf] × inventory level *after* demand is observed
+ [OQ In Process Cf] × # of items in process
+ [OQ About To Compl Cf] × # of items that will complete processing at stage i in period t
+ [OQ Upstr IP Cf] × inventory position at upstream stage *before* demand is observed (3,4)
+ [OQ Upstr IL Cf] × inventory level at upstream stage *before* demand is observed (3,4)
+ [OQ Upstr Consec SO Cf] × # of consecutive periods, through period $t - 1$, in which a stockout occurred at upstream stage (4)
+ [OQ Upstr Consec NonSO Cf] × # of consecutive periods, through period $t - 1$, in which a stockout did not occur at upstream stage (4)
+ [OQ Upstr SO Cf] × 1 if a stockout occurred at upstream stage in period $t - 1$, 0 otherwise (4)
+ [OQ Upstr NonSO Cf] × 1 if a stockout did not occur at upstream stage in period $t - 1$, ¿0 otherwise (4)
+ [OQ Disrupt Cf] × 1 if stage i failed in period $t - 1$, 0 otherwise
+ [OQ NonDisrupt Cf] × 1 if stage i did not fail in period $t - 1$, 0 otherwise
+ [OQ Upstr Disrupt Cf] × 1 if upstream stage failed in period $t - 1$, 0 otherwise
+ [OQ Upstr NonDisrupt Cf] × 1 if upstream stage did not fail in period $t - 1$, 0 otherwise

Notes:

1. Demand includes orders from internal customer stages as well as external demands.

2. Demand forecast for period $t+1$ is calculated using exponential smoothing as follows:

$$\hat{D}_t = [\text{OQ Smoothing Const}] \times D_t + (1 - [\text{OQ Smoothing Const}]) \times \hat{D}_{t-1},$$

where D_t is the actual demand in period t and $\hat{D}_t$ is the forecast for the demand in period $t+1$, calculated in period t.

3. IP and IL are calculated before demand is observed since demand at upstream stage equals order at stage i, which has not yet been calculated.
4. Terms involving upsream stages are generally intended for use with serial or distribution systems (in which each stage has at most one upstream neighbor), but they can also be used in more general systems. If stage i has more than one raw material and/or supplier, only the top-ranked supplier for the raw material with the lowest index is considered in calculating these state variables.

There are two ways to impose bounds on the resulting order quantity:

- Impose bounds on the order quantity itself. [OQ OQ LB] is the lower bound and [OQ OQ UB] is the upper bound. If the order quantity violates one of these bounds, it will be set equal to the violated bound. The order quantity must always be non-negative, so a lower bound of 0 is always implied.
- Impose bounds on the resulting inventory position, that is, on the current inventory position plus the order quantity. [OQ IP LB] is the lower bound and [OQ IP UB] is the upper bound. If the new IP violates one of these bounds, the order quantity will be adjusted so that the new IP equals the violated bound.

If you set the lower bounds to 0 and the upper bounds to ∞ (INF), no bounds are imposed.

Specify the reorder point in the [OQ Reorder Pt] field. No order will be placed if the inventory position is greater than the reorder point.

Finally, you can specify an initial inventory level in the [OQ Init IL] field. The stage will begin the simulation with [OQ Init IL] units on hand.

3.7.2 Using the Order Quantity Policy to Model Order-Up-To Policies

An order quantity policy can be used to model an order-up-to-type policy by simply subtracting the current inventory position from the order quantity—that is, by setting [OQ IP Cf] to -1. In this way, you can use an order quantity policy to model a base-stock policy in which the base-stock level is a function of the state variables or an (s, S) policy in which S is a function of the state variables.

In particular, set the values of the order quantity parameters according to your desired function, and then subtract -1 from [OQ IP Cf]. To model an (s, S) policy, set [OQ Reorder Pt] to s. To model a base-stock policy, set [OQ Reorder Pt] to ∞ (INF).

3.7.3 Examples

Here are a few examples of how an order quantity policy might be used.

Special Cases All three of the other types of inventory policies are special cases of the order quantity policy. You can model these policies by setting the order quantity parameters as follows:

	Parameter Setting		
Parameter	Base-Stock	(R, Q)	(s, S)
[OQ Constant]	Base-stock level	Q	S
[OQ Reorder Pt]	INF	R	s
[OQ IP Cf]	-1	0	-1

Set all other order quantity parameters to 0, except [OQ OQ UB] and [OQ IP UB], which you should set to INF.

Demand Forecasting Suppose the stage wishes to set the base-stock level in period t equal to $\hat{\mu}_t + z\sigma$, where $\hat{\mu}_t$ is a forecast of the demand in period $t+1$ obtained using exponential smoothing, σ is the standard deviation of demand, and z is a service level constant (perhaps obtained by solving a newsvendor problem). To model such a policy, set the parameters as follows:

Parameter	Parameter Setting
[OQ Constant]	$z\sigma$
[OQ Reorder Pt]	INF
[OQ IP Cf]	-1
[OQ Forecast Cf]	1
[OQ Smoothing Const]	Exponential smoothing constant

Panicky Ordering Suppose the stage acts "panicky" when its supplier experiences a stockout and complacent when it does not. For example, suppose that the stage generally follows an (R, Q) policy, but that it orders an extra 20 units in period t if the supplier had a stockout in period $t-1$, and that it under-orders by 5 units if the supplier did not have a stockout. Use the following parameter settings:

Parameter	Parameter Setting
[OQ Constant]	Q
[OQ Reorder Pt]	R
[OQ Upstr SO Cf]	20
[OQ NonSO Cf]	-5

3.8 Disruptions

The disruption state at each stage is governed a Markov process with two parameters: the *failure probability* (FP) and the *recovery probability* (RP). If the stage has not failed in the current period, than it will fail in the next period with probability FP and will not fail with probability $1 - FP$. Similarly, if the stage has failed in the current period, it will fail in the next period with probability $1 - RP$ and will not fail with probability RP.

If a stage has failed in a given period, it cannot process any work-in-progress. Therefore, if a stage has a processing time of T periods and fails a total of m times during the processing of a given item, that item takes $T + m$ periods to complete its processing.

In addition, no inventory can be withdrawn from the stage's output buffer during a disruption, as described in Section 3.6.2.

3.9 Initial Inventory

The number of units on-hand at each stage at the start of period 1 is determined by the inventory policy and parameters. In particular, the stage has an initial inventory equal to [Base Stock Level] for a base-stock policy, equal to [R] + [Q] for an (R, Q) policy, equal to [S] for an (s, S) policy, and equal to [OQ Init IL] for an order quantity policy.

4 Building and Running Models

4.1 Adding Stages

To add a new stage to the current model, click the **Add Stage** button. A new stage appears in the model display. You can position the stage elsewhere in the display by dragging it. (The position of the stage makes no difference to the simulation; it is for display purposes only.) If the simulation has already been run, it must be reset before adding new stages.

To establish stage i as a supplier for stage j, drag stage i onto stage j and release the mouse. Supplier stages must always have larger indices than customer stages (i.e., $i > j$). Unfortunately, there is currently no feature to delete or decouple stages. The model need not be connected (that is, it may have two or more disconnected parts).

The maximum number of stages in one model is 25.

4.2 Entering Parameters

Each stage has a number of associated parameters, which are listed in the "Parameter" list in the left-hand side of the main window. To enter or edit parameters for a stage, first click on the stage (the label above the value list reads "Stage i"), then enter or change the parameters in the value list. The new value is recorded when you press enter, move to another cell in the Parameter list, click on another stage, or click on another button or control on the form.

Each parameter has a certain type of allowable value (integer, real number, Boolean, etc.), and if you try to enter an invalid value, BaseStockSim will display an error message and require you to correct the entry before you can leave the cell. Alternately, you can press Esc or Ctrl-Z before leaving the cell to restore the entry to its previous value.

Demands are always assumed to be integer. The demand parameters (mean, standard deviation, and bound) may be given as real numbers, but the demand generated (after truncating) will be rounded to the nearest integer. For more information on demands and truncation, see Section 3.4.

Some parameters allow a value of ∞. To specify ∞ as the parameter's value, enter "INF". These parameters serve as upper bounds for various processes in the simulation. They are always used for comparison, not for calculations. These parameters default to INF when you create a new stage.

The parameters are as follows:

[Proc Time] The processing time (# of periods) at the stage.

[Proc Capac] The maximum number of items that can start processing at the stage in a given period. The capacity does not apply to all WIP at the stage, only to items that *begin* processing in a given period.

[Com Serv Time] The committed service time (# of periods) at the stage. This represents the number of periods within which the stage is obligated to deliver an order to its downstream stages or external customer. If the "Service Time >= CST" checkbox is checked, items are never delivered before their committed service time has elapsed; rather, they are held in inventory at the upstream stage until the CST has elapsed. For more on committed service times, see [3]. If the checkbox is unchecked, this parameter does not affect the simulation itself, but rather the stockout cost and service levels reported as output to the simulation.

[Dem Mean] The mean demand per period from an *external* customer at the stage. If the stage has no external customer, set this to 0.

[Dem StdDev] The standard deviation of demand per period from an *external* customer at the stage. If the stage has no external customer, set this to 0.

[Dem Bound (#SD)] The demand from an external customer in any τ consecutive periods is assumed to be bounded by $\mu\tau + k\sigma\sqrt{\tau}$, where μ and σ are the mean and standard deviation of

periodic demand. Any randomly generated demand exceeding this bound is truncated at the bound. If the stage has no external customer, this parameter has no effect. Typical values for k are 2 or 3, or INF to avoid setting any upper bound on the demand. For more on demands and truncation, see Section 3.4.

[**Product Index**] The index of the item produced at the stage. By setting this field to the same value for two or more stages, you are indicating that those stages all make the same item. Any stage that has these common stages as suppliers may order that item from *any* of the suppliers, rather than being required to order items from *every* supplier. For more information, see Section 3.6.2.

[**Product Cost**] The per-unit product cost for orders placed *to* the stage by downstream stages (but not external customers). If a stage has multiple suppliers with the same Product Index, the suppliers are ranked according to their Product Cost. Product costs are incurred when an item is delivered, not when it is ordered. For more information, see Section 3.6.2.

[**Fixed Cost**] The fixed cost per order placed *to* the stage by downstream stages (but not external customers). The fixed cost is incurred for each order, regardless of the size of the order. If a stage has multiple customer stages that place orders in a given period, the fixed cost is incurred multiple times. Fixed costs are incurred when an order is placed, not when it is delivered.

[**Holding Cost**] The holding cost per unit on hand at the end of a period.

[**WIP Holding Cost**] The holding cost per unit in process at the end of a period.

[**Stockout Cost**] The cost incurred when items take longer than their [Com Serv Time] to be delivered. If [Com Serv Time] = 0, the stockout cost is incurred whenever an demand is not met from stock. If [Per Unit Time] is true, the cost for one late item is equal to the number of periods late (delivery time – [Com Serv Time]) times the [Stockout Cost]; otherwise, it is equal to the [Stockout Cost].

[**Per Unit Time**] Indicates whether the stockout cost is charged per item per time period (if true) or simply per item (if false).

[**Failure Prob**] The probability that the stage will fail in the next period, given that it did not fail in the current period. See Section 3.8.

[**Recovery Prob**] The probability that the stage will not fail in the next period, given that it did fail in the current period. See Section 3.8.

[**Inv Policy**] The inventory policy followed by the stage. Select the inventory policy from the drop-down list: either BS (base-stock policy), (R, Q), (s, S), or OQ (order quantity policy). See Section 3.6.1 for more information on inventory policies.

[**Base Stock**] The base-stock level at the stage. If Inv Policy $\neq$ BS, this field is ignored.

[R, Q] The reorder point (R) and order quantity (Q) at the stage. If Inv Policy $\neq (R, Q)$, these fields are ignored.

[s, S] The reorder point (s) and base-stock level (S) at the stage. If Inv Policy $\neq (s, S)$, these fields are ignored.

[OQ Fields] These fields provide coefficients and other options for order quantity policies. For more details about these fields, see Section 3.7. If Inv Policy $\neq$ OQ, these fields are ignored.

4.3 Clearing, Saving, and Loading Models

The **Clear All** button clears the current model (as well as any simulation results). The **Save** and **Load** buttons save and load the model, respectively, including all parameters and the locations of stages in the model display.

(The **Reset Sim** button resets any simulation results but not the model itself—see Section 4.5.)

4.4 Running the Simulation

To run the simulation, press the **Run Simulation** button, first specifying the number of periods to simulate in the "# of Periods" field and the length of the warm-up period in the "Warm Up" field. You may also specify a seed for the pseudo-random number generator in the "RandSeed" field or set it to 0 to generate the seed using the system clock (i.e., randomly).

4.5 Resetting the Simulation

To reset the simulation (but keep the model intact), press the **Reset Sim** button. All performance measures and simulation histories will be deleted.

5 Viewing Simulation Results

Once the simulation has run to completion, you can watch it period-by-period using the "Instant Replay" trackbar, or view a summary of the statistics from the simulation, or view a period-by-period accounting of the simulation history. Each of these features is discussed next.

5.1 Instant Replay

During the instant replay, you can observe the state of the system in each period using the Performance Measures list (to the right of the Parameters list), as well as using the inventory and failure indicators in the model diagram. These are discussed next.

5.1.1 Performance Measures

When you click on a stage, the "Measures" value list will list several performance measures for the stage in the current period:

⟨**On Hand**⟩ The on-hand inventory (always non-negative—if there are backorders, ⟨On Hand⟩ = 0).

⟨**Backorders**⟩ The number of backorders (always non-negative—if there is inventory on hand, ⟨Backorders⟩ = 0).

⟨**On Order**⟩ The number of items on order to the stage's own production process. Some of these items may not have begun processing yet, since upstream suppliers may not have had sufficient inventory. Use ⟨In Process⟩ to determine the number of items that have begun processing.

⟨**In Process**⟩ The number of items that have begun processing at the stage but have not completed processing.

⟨**Inv Pos**⟩ The inventory position at the stage; equal to ⟨On Hand⟩ – ⟨Backorders⟩ + ⟨On Order⟩.

⟨**Demand**⟩ The demand from the stage's external customer.

⟨**Derived Demand**⟩ The sum of the demand from the stage's external customer and the demands from the stage's downstream customers, if any.

⟨**Order Qty**⟩ The size of the order, if any, that the stage placed to its upstream suppliers and to its own production process.

⟨**Failed**⟩ Equal to 1 if the stage failed in the current period and 0 otherwise.

⟨**Cycle Stock**⟩ The portion of the stage's base-stock level accounted for by cycle stock, computed as $\mu(SI+T-S)$, where μ is the mean demand seen by the stage (i.e., the stage's own external demand plus demand derived from downstream stages), SI is the maximum inbound CST, T is the processing time at the stage, and S is the stage's own CST. If the stage uses an inventory policy other than base-stock, this measure is not meaningful. [*This performance measure is static from period to period.*]

⟨**Safety Stock**⟩ The portion of the stage's base-stock level accounted for by safety stock, computed as the base-stock level minus ⟨Cycle Stock⟩. If the base-stock level was set optimally according to the guaranteed-service model, this should equal $k\sigma\sqrt{SI+T-S}$, where σ is the standard deviation of the demand seen by the stage (i.e., the stage's own external demand plus demand derived from downstream stages) and the other parameters are as described for ⟨Cycle Stock⟩. If the stage uses an inventory policy other than base-stock, this measure is not meaningful. [*This performance measure is static from period to period.*]

In addition, the costs for the period (order, holding, WIP holding, stockout, and total costs) are displayed below the Performance Measures list.

5.1.2 Inventory Indicators

Above and to the right of each stage is an inventory indicator that provides a graphical representation of the on-hand inventory and backorders in the current period. The gray box is sized vertically based on the "target" safety-stock level, which is defined in Section 5.1.1. (If the stage uses an inventory policy other than base-stock, the size of the gray box may not be meaningful.) The number below the gray box gives the actual inventory level, positive or negative, while the colored bar indicates the inventory level graphically:

- If there is *on-hand inventory* (i.e., the inventory level is positive), the bar indicates the amount of on-hand inventory. Any amount up to the target safety-stock level is indicated in green, while excess is indicated in blue.
- If there are *backorders* (i.e., the inventory level is negative), the bar falls below the gray inventory box and is red.

Note that negative target safety stocks are allowed, in which case the inventory bucket "points" downward. In this case, negative (actual) safety stocks are indicated in red, while positive safety stocks are indicated in blue.

5.1.3 Failure Indicators

If a stage has failed in a given period, it is drawn in red.

5.2 Simulation Statistics

After the simulation has completed, you can view a summary of the results by clicking the **Display Stats** button. The stats window displays the date and time the simulation was completed, the failure probability of each stage, several performance measures for each stage (as well as totals), and a histogram of service times.

You can copy the text in the this window to the clipboard by selecting it and pressing Ctrl+C, then paste it into a text editor.

5.2.1 Performance Measures

For each performance measure, the program reports the average, standard deviation, minimum, and maximum values of that statistic, as well as the number of observations. For most measures, the number of observations is the number of periods; exceptions are indicated below.

The performance measures included are:

⟨**On Hand**⟩ The on-hand inventory (always non-negative—if there are backorders, ⟨On Hand⟩ = 0).

⟨**Backorders**⟩ The number of backorders.

⟨**On Order**⟩ The number of items on order to the stage's production process.

⟨**In Process**⟩ The number of items that have begun processing at the stage but have not completed processing.

⟨**Order Cost**⟩ The sum of the fixed and per-unit costs incurred for orders in a given period.

⟨**Holding Cost**⟩ The holding cost incurred in a given period; equal to holding cost per item per period times ⟨On Hand⟩.

⟨**WIP Holding Cost**⟩ The WIP holding cost incurred in a given period; equal to WIP holding cost per item per period times ⟨In Process⟩.

⟨**Stockout Cost**⟩ The stockout cost incurred in a given period. See descriptions of [Stockout Cost] and [Per Unit Time] parameters in Section 4.2 for more information.

⟨**Total Cost**⟩ The sum of the holding, WIP holding, and stockout costs.

⟨**Demand**⟩ The demand generated by the stage's external customer in a given period. [*Obs = 0 if no external customer.*]

⟨**Derived Demand**⟩ The demand generated by the stage's external customer plus that generated by orders from downstream stages in a given period.

⟨**All From Stock**⟩ The percentage of periods in which all items were delivered from stock (i.e., with service time equal to 0). Equal to the type-1 service level.

⟨**From Stock**⟩ The percentage of items demanded over the course of the simulation that were delivered from stock (i.e., with service time equal to 0). Equal to the type-2 service level; also known as the *fill rate*. [*Obs = total number of items demanded.*]

⟨**Within CST**⟩ The percentage of items demanded over the course of the simulation that were delivered within the stage's CST. [*Obs = total number of items demanded.*]

⟨**Order Quantity**⟩ The size of the order placed to the stage's upstream suppliers and the stage's own production process during a given period.

⟨**Failed**⟩ The percentage of periods during which the stage failed.

Service Time Histogram The lower part of the statistics window gives a histogram of service times for each stage. The service time is the actual delivery time for a given unit—the time between when an order for a unit is placed and when the unit is received. It is computed for each item individually: If an order is placed for 10 units with 5 arriving immediately and 5 arriving in 1 time period, then 5 units are counted in as a 0-period service time and 5 are counted as a 1-period service time.

For a given row, the columns list, for each stage, the number of items whose service time equals the service time for that row, the corresponding percentage of items, and the cumulative percentage. The last column can therefore be viewed as an empirical cumulative distribution function of service times; for example, 60% of items are delivered in 0 periods, 80% in 1 period or less, 95% in 2 periods or less, and 100% in 3 periods or less.

5.3 Simulation History

After the simulation has completed, you can view a period-by-period account of the simulation by clicking the **Display History** button. The history window has two tabs, labeled "History" and "Order History."

The History tab displays, for each stage and period, the value of several performance measures, all of which are described in Section 5.2.1: ⟨Total Cost⟩, ⟨Demand⟩, ⟨Derived Demand⟩, ⟨Inventory Level⟩ (= ⟨On Hand⟩ – ⟨Backorders⟩), ⟨Order Quantity⟩, ⟨On Order⟩, ⟨In Process⟩, and ⟨Failed⟩.

The Order History displays the size of the orders placed by each stage in each period. The order sizes are broken down by raw material and, for each raw material, by supplier. In addition to the raw materials, the order quantity placed by a stage to itself (i.e., to its own production process) is given in the column labeled "Self."

6 Batch Runs

You may want to run the simulation multiple times, each time with a different value of some parameter. For example, you may wish to test the effect of varying the capacity of a given stage, or you may wish to find the optimal base-stock level at some stage by simulating the system using a range of values for that parameter and choosing the one with the lowest cost. BaseStockSim has a feature called "Batch Runs" designed for this purpose. To launch this feature, click the **Batch Runs** button in the main window.

You can specify values for your batch runs in two ways, one of which is simpler but less flexible than the other (Section 6.1). You can specify the performance measures that you would like BaseStockSim to report, and then click the **Go** button to start the simulation. BaseStockSim displays the results of the trials in the box at the bottom of the Batch Runs window.

Set the number of trials you wish to perform using the "# of Trials" field. BaseStockSim will simulate each set of parameter values multiple times according to the value of this field. Each trial will use the number of periods specified in the "# of Periods" field on the main window, with a warm-up period as specified by the "Warm Up" field (also on the main window).

If you click the **Write Model** button, BaseStockSim will write the parameters of each stage in the current model to the text box. This feature is useful for keeping a record of the current model as you simulate multiple models or perform multiple tests.

You can edit the text in the text box however you wish. You can also save it to a text file using the **Save Text** button, clear it using "Clear Text," or copy it to the clipboard by selecting part of all of the text and pressing Ctrl-C. You can then paste it from the clipboard into a text editor or spreadsheet program for further analysis.[3]

6.1 Specifying Parameter Values

You can specify the parameters that you want to use, and the values that you want to use for each parameter, in two ways, described next.

6.1.1 Testing Ranges of Values

If you choose the "Test Range(s) of Parameter Values" option, you can specify one or two parameters that you want to vary. For each parameter, specify the range of indices of the stages for which you wish to vary the parameter. You can only specify a range of consecutive indices. Next, specify the start and end values for the parameter, as well as the step size. If you specify multiple stages, the values of the parameters are set for all stages *together*.

For example, suppose you fill out the first row of the form as follows:

> Change value of **Base Stock Level** for stages **1** through **3** Values: **100** through **200** step **10**

Then BaseStockSim will simulate the system with the base-stock level for stages 1, 2, and 3 equal to 100; then with the base-stock level for these three stages equal to 110; then equal to 120; and so on up to 200. If you had set the number of trials to 10, then it would simulate the system 10 times for each value of the base-stock levels.

If you specify two parameters, then BaseStockSim simulates the system for every *combination* of the parameters. For example:

> Change value of **Base Stock Level** for stages **1** through **3** Values: **100** through **200** step **10**

[3] *Hint*: If you paste the results into Microsoft Excel, use the "Text to Columns" feature to convert the text from one column into multiple columns. Choose the "Delimited" option and then check "Space" as the delimiter.

Then value of **Holding Cost** for stages **5** through **8** Values: **2.0** through **2.4** step **0.1**

These settings will result in a total of $11 \times 5 = 55$ tests, since there are 11 values of Base Stock Level and 5 values of Holding Cost. If you set # of Trials = 10, then there will be 550 separate runs.

Hint: You can perform multiple trials of the existing model, without changing any parameters, by leaving both drop-downs blank (set to "[none]") and entering a number greater than 1 in "# of Trials".

6.1.2 Loading Parameters from a File

Alternately, you can load a file that specifies the parameters and values you wish to use. This is less convenient but much more flexible than using ranges of parameter values. To choose this option, select the "Load Parameter Values from File" radio button, then click **Browse** to find the file that contains your parameters and values.

The file must be organized as follows:

- The first line contains the number of parameters you want to specify values for, then a space or tab, then the number of values for each set of parameters. For example, if you want to specify the base-stock level for 3 stages as well as the capacity for 2 stages, and you want to specify 100 sets of values for these parameters, then the first line should read "5 100". The first value is 5 because you're setting values for the base-stock level for 3 stages and the capacity for 2.
- The second line contains the index of the parameter, then a space or tab, then the stage number for each parameter you want to specify values for. The parameter indices start at 0 and go consecutively according to the Parameter box in the main window: 0 = [Proc Time], 1 = [Proc Capac], 2 = [Com Serv Time], and so on. So, if you want to set the base-stock level for stages 2–4 and the capacity for stages 1 and 4, then the second line should read: "16 2 16 3 16 4 1 1 1 4"
- The remaining lines each contain the relevant parameter values, separated by spaces or tabs.

Included with the BaseStockSim software are two sample files to demonstrate this format. The file `commonpart3_batch.txt` contains data for performing batch runs on the `commonpart3.txt` sample model file, also provided with BaesStockSim (see Section 7). According to this file, BaseStockSim will test four pairs of the failure and repair probabilities at stage 3: (0.001, 0,1), (0.01, 0.2), (0.1, 0.5), and (0.2, 0.8). For each of these pairs, it will also test four values of the demand standard deviation at stage 1 (0, 1, 2, and 5), and for each of these, it will test three values of the base-stock level at stage 1 (20, 40, and 60). This gives a total of $4 \times 4 \times 3 = 48$ sets of values to test.

The file `batch_template.xls` is an Excel spreadsheet containing a template for specifying parameters for batch runs. The two leftmost columns list all of the parameters and their corresponding

indices for reference. The data go in the section contained inside the blue cells; this section can simply be copied and pasted into a text file.

6.2 Reporting Options

You have a fair amount of control over what performance measures are reported for each set of parameter values you test. To specify your preferences for these measures, click the **Options** button. In the box labeled "Standard Performance Measures", you can choose which performance measures you want to see; these are the same performance measures that are given in the Stats window when you run a single simulation.

For each performance measure, you can specify whether you want BaseStockSim to report the Mean, SEM, Min, or Max values of the trial means. That is, Mean refers to the mean of the trial means, Min refers to the minimum of the trial means, and Max refers to the maximum of the trial means. SEM, or standard error of the mean, is the standard deviation of the trial means. You can calculate an α confidence interval on a given performance measure as Mean $\pm$ $z_{\alpha/2}$ $\times$ SEM, where $z_{\alpha/2}$ is the $\alpha/2$ fractile of the standard normal distribution.

You can quickly check or uncheck all of the boxes (Mean, SEM, Min, Max) under a given performance measure by double-clicking the name of the performance measure.

By default, BaseStockSim reports each type of cost summed over all stages. However, you can tell BaseStockSim to report the costs by each individual stage, as well, by checking "By Stage". The non-cost measures are always reported by individual stages, so their "By Stage" box is always checked.

BaseStockSim can also report two other performance measures, listed in the box labeled "Other Performance Measures". The first, SD of Total Cost, reports the standard deviation of the total cost across all periods (excluding warm-up periods) in all trials of the test. This measure gives a sense of the total variability in costs. (It should not be used to calculate a confidence interval on the mean cost.)

The second other measure, "Mean of SD of Order Qty", reports the mean (across trials) of the standard deviation (across periods) of the order quantity at each stage. This is useful for checking for the presence of the bullwhip effect [5], in which the standard deviation of the order quantity, and not the mean, is the relevant performance measure.

Once you have chosen your options for the batch runs, you can save them as the default for future runs by clicking the **Save as Default** button. To restore the settings to the "factory defaults," click **Restore Defaults**.

7 Sample Models

Two sample model files come with the software. The first, `commonpart3.txt`, represents a system with two end products that share a common raw material. Product 1 is a slow-moving but critical product while product 2 is a fast-moving but inexpensive one. The other model, `serial3.txt`, represents a 3-echelon serial supply chain with a single product.

In addition, BaseStockSim comes with two files, based on `commonpart3`, that demonstrate the Batch Runs feature; see Section 6.1.2.

8 Other Stuff

8.1 Disclaimer

THE AUTHOR OF THIS SOFTWARE MAKES NO CLAIMS, EXPRESSED OR IMPLIED, ABOUT THE PERFORMANCE OF THIS SOFTWARE, INCLUDING, BUT NOT LIMITED TO, THE STABILITY OF THE SOFTWARE OR THE QUALITY OF THE SOLUTIONS IT RETURNS. The author shall not be held liable for any damage or injury that results from the use of this software, including, but not limited to, damage to computer software or hardware.

8.2 Bug Reports

Please report bugs via e-mail to `larry.snyder@lehigh.edu`.

8.3 Acknowledgments

Special thanks to my students Ying Rong and Jae-Bum Kim for their involuntary beta-testing of v2.4 (translation: for using a version that I hadn't adequately tested and reporting bugs to me as they encountered them).

8.4 Version History

Version 1.0 – 3/7/05

Initial release.

Version 1.1 – 4/6/05

New features:

- Service time histogram
- Stockout cost may be per item or per item per unit time
- Dynamic sizing of column widths in results screen based on width of largest value

Bug fixes:

- Stage's parameter now changes when user changes parameter in list and then clicks a new stage
- Better handling of inventory buckets for negative safety stock
- Better updating of performance measure list and simulation display after parameters have been changed
- Fixed crashing problem when new stage is added after simulation has been run

Version 1.2 – 4/27/05

New features:

- Added "Service Time >= CST" option, which causes the system to operate as a "guaranteed-service" system, meaning that no items are delivered until the CST (or longer) has elapsed. Orders are tagged to the requesting stage (or external supplier) and can only be delivered to that stage. (If this option is not chosen, items are made available to downstream stages or external customers as soon as their processing is completed, on a first-come first-served basis.)

Version 2.0 – 9/18/05

New features:

- Changed terminology to adhere to more standard terminology
- Stats report is now in formatted differently for better readability and contains more info (standard deviation, etc.)
- Added History Window, which gives period-by-period details of demand, order qty, etc., for each stage
- Added In Process stat, which tracks number of items of WIP, i.e., number of items actually in process (On Order stat tracks number of items on order to production process (and upstream suppliers), regardless of whether processing has begun on those items)
- New stats are now reported: derived demand, % of periods met all from stock (type 1 service level), in process

Bug fixes:

- Fixed logic of supplier nodes so that they now distribute available items to their customer stages in FCFS order
- Fixed WIP holding cost so that it is now computed based on In Process, not On Order

Version 2.1 – (internal release)

New features:

- Changed the behavior of stages during disruptions. In particular, if a stage is disrupted, it cannot supply items to its downstream stages (or do any processing).
- Added Proc Capac field: processing capacity. Stage can only begin processing this many items per period.
- Added Recovery Prob field: probability of no failure given that stage failed last period.
- Better loading and saving of model files. Now saves version number and loads according to software version that saved the file. If no version number exists in the file, assumes saved using v2.0.
- Added feature to allow multiple (i.e., backup) suppliers at a given stage.
  - Each stage has a new field called Product Index that indicates the item # supplied by the stage.
  - Each stage also has a new field called Product Cost that indicates the per-unit cost to purchase an item from the stage.
  - The program determines the raw materials required by a stage by examining the Product Index of the stage's upstream suppliers. For each raw material, the program sorts the suppliers in increasing order of cost (ties are broken arbitrarily).
  - In each period, a stage places orders to its cheapest supplier for each raw material, if that supplier has not failed. If it has failed, the stage uses the second-cheapest supplier for that raw material, etc. If all suppliers for a given raw material have failed, the stage places its order from the cheapest one. (There may be smarter order-allocation rules – for example, taking into account the likely durations of the disruptions, or the suppliers' inventory policies/CSTs – but these are not implemented here.) [See v2.4 notes for a change in this policy.]
  - Each stage draws raw materials as available from its suppliers, regardless of which suppliers have failed and which suppliers the stage placed orders with in this period. That is, if supplier #1 has failed but has inventory available, the stage may place an order with supplier #2 but draw inventory from supplier #1. In fact, the stage may draw inventory from more than one supplier for a given raw material, if multiple suppliers have inventory available.
  - Product Costs are incurred when items are delivered to the stage (rather than when order is placed).
- Added average cost summary on main page.
- Added feature for simulating the system repeatedly, with multiple values of a given parameter. Use **Batch Runs** button.
- Added "warm-up period" feature, which ignores data from the first N periods when calculating stats, where N is specified by the user.
- Stage is now highlighted visually with a double border when selected.
- Confirmation requested before clearing model.

Bug fixes:

- Fixed bug that caused floating point error in TMinMaxAvg:GetStdev when variance = 0. Occasionally variance evaluated to something very small but negative, causing an error when the square root was taken.
- History window now uses fixed-width font.
- Turned off word wrap in History Window and added scrollbars.
- Fixed bug that considered wrong stage's failure state when doing processing at a stage.
- Fixed bug that caused some items to be delivered before their CST even when "Service Time ¿= CST" was turned on.

- Resets stage label to "Stage 0" when model is cleared.

Version 2.2 – (internal release)

New features:

- "Batch Runs" form now reports SD of total cost (across all periods in all trials), as well as SEM. Also removed Min and Max of Mean.
- Added feature to save text to file in History form.
- Implemented order quantity policy (though some parameters were added in later versions). See Section 3.7 for more information.

Bug fixes:

- Order quantity stats (e.g., in Performance Measures list) used to equal sum of all order quantities placed to all suppliers. Now it equals the production quantity at the stage (i.e., the order quantity from a stage to itself), which should equal the number of units of *each* raw material ordered.

Version 2.3 – (internal release)

New features:

- Four additional parameters added to order quantity function: [OQ In Process Cf], [OQ Forecast Cf], [OQ Smoothing Const], and [OQ Constant].
- Added Fixed Cost field. Charged per order placed by downstream stages. NOTE: Fixed Cost field refers to orders placed BY DOWNSTREAM stages, not to orders placed by the STAGE ITSELF.
- Added warning message when user tries to run simulation if some stage has positive fixed cost but no downstream customer.
- Added warning message when loading file saved using earlier version of software and converting to current version.
- Added warning message if any stage with positive (derived) demand has zero capacity. (Previously, warned only if *every* stage has zero capacity.)
- If a stage has multiple suppliers for the same raw material, it can now place orders to more than one in a given period. The size of each order is limited by the supplier's processing capacity (even though processing amount does not necessarily equal order quantity in a given period). No orders are placed to failed suppliers, except that: If all suppliers are failed and/or maxed out, any remaining items are ordered from supplier #1 (the cheapest supplier).
- Added Order Qty History in History Form.
- Batch Runs form now allows user to load a set of parameter values from a text file. This gives more flexibility than the old method for entering parameters to test (which is also still available).
- Batch Runs form has a **Write Model** button that dumps the model in text format into the window. (The model written is the base model, before parameters are changed.)
- Batch Runs form has **Options** button that opens a new window in which user can choose which performance measures (and stats [mean, SEM, ...] for each) to display.

Version 2.4 – 5/24/06

New features:

- Better data checks before simulation is run. New warning messages if:
  - Some stage has demand mean ¿0 and ¡1.
  - Some stage with positive demand mean and SD has zero demand bound.
  - Some stage with positive product cost has no customer.

  Also better checking in Batch Runs form.
- Four new parameters for order quantity function: lower and upper bounds on order quantity, starting IL, and reorder point.
- Inventory Policy is now chosen using a drop-down, not an integer.
- Several fields can now be set to INF to represent infinity. These are: [Proc Capac], [Dem Bound], [OQ OQ UB], [OQ IP UB], and [OQ Reorder Pt]. In general these are fields that are only used as bounds, not for arithmetic. New stages default to INF for these fields.
- Created icon for application.
- Increased maximum allowable # of stages from 10 to 25.

Bug fixes:

- New stages are staggered so they don't appear on top of one another.
- If Dem Bound = INF, now doesn't do bound checking. (Before, it avoided the actual comparison, but it still looped through all periods, which was a huge waste of time.)
- Fixed bug in supplier-selection rule for order quantity.

References

[1] S. Axsäter. *Inventory Control*. Kluwer, Boston, 2000.

[2] S. C. Graves and S. P. Willems. Optimizing strategic safety stock placement in supply chains. *Manufacturing and Service Operations Management*, 2(1):68–83, 2000.

[3] S. C. Graves and S. P. Willems. Supply chain design: Safety stock placement and supply chain configuration. In A. G. de Kok and S. C. Graves, editors, *Supply Chain Management: Design, Coordination and Operation*, volume 11 of *Handbooks in Operations Research and Management Science*, chapter 3. North-Holland, Amsterdam, 2003.

[4] G. E. Kimball. General principles of inventory control. *Journal of Manufacturing and Operations Management*, 1:119–130, 1988.

[5] H. L. Lee, V. Padmanabhan, and S. Whang. Information distortion in a supply chain: The bullwhip effect. *Management Science*, 43(4):546–558, 1997.

[6] S. Nahmias. *Production and Operations Analysis*. McGraw-Hill/Irwin, 5th edition, 2005.

[7] E. A. Silver, D. F. Pyke, and R. Peterson. *Inventory Management and Production Planning and Scheduling*. Wiley, 3rd edition, 1998.

[8] K. F. Simpson, Jr. In-process inventories. *Operations Research*, 6(6):863–873, 1958.

[9] J. D. Sterman. Modeling managerial behavior: Misperceptions of feedback in a dynamic decision making experiment. *Management Science*, 35(3):321–339, 1989.

[10] P. H. Zipkin. *Foundations of Inventory Management*. Irwin/McGraw-Hill, 2000.