NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

COMPARING NAVAL-INVENTORY FILL RATES FROM COMPOSITE AND REPAIR-AND-PROCURE SUPPLY SYSTEMS THROUGH DISCRETE EVENT SIMULATION

by

Brian H. Hays

June 2019

Thesis Advisor:	Emily M. Craparo
Co-Advisor:	Javier Salmeron
Second Reader:	Kevin J. Maher

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** June 2019	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE COMPARING NAVAL-INVENTORY FILL RATES FROM COMPOSITE AND REPAIR-AND-PROCURE SUPPLY SYSTEMS THROUGH DISCRETE EVENT SIMULATION	**5. FUNDING NUMBERS**
6. AUTHOR(S) Brian H. Hays	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Naval Supply Systems Command Weapon Systems Support currently uses Distributed Requirement Planning (DRP) to generate repair and procurement quantity recommendations for Supply Class IX repair parts. In an effort to provide an optimization-based method for making wholesale inventory decisions, a mixed integer linear optimization program termed Wholesale Inventory Optimization Model (WIOM) was developed as an alternative to existing tools. In its current form, however, WIOM does not explicitly consider the repair cycle for those national item identification numbers that may be either procured or repaired.

The thesis develops a next-event, discrete-event inventory simulation model, which we term Comparative Replenishment Stream Simulation (COMPRESS) to compare the effects of having two supply streams with differing lead times versus what we commonly see in a single supplier stream. With WIOM strictly considering a "composite" supply stream, the determination needs to be made if the disparity between inventory systems that utilize either a composite or repair-and-procure supply stream is great enough to warrant a WIOM reformulation. We select fill rate as the performance metric to compare these inventory systems. Through COMPRESS we find that there is considerable difference in the resulting fill rates. Additionally, we find that as the variability in demand and lead time increases, the difference in the resulting fill rate values increases as well.

14. SUBJECT TERMS inventory, simulation, event graphs, reorder point, fill rate, back order, Wholesale Inventory Optimization Model	**15. NUMBER OF PAGES** 79
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

COMPARING NAVAL-INVENTORY FILL RATES FROM COMPOSITE AND REPAIR-AND-PROCURE SUPPLY SYSTEMS THROUGH DISCRETE EVENT SIMULATION

Brian H. Hays
Lieutenant Commander, United States Navy
BS, University of North Texas, 2002
MS, University of North Texas, 2003

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
June 2019

Approved by: Emily M. Craparo
Advisor

Javier Salmeron
Co-Advisor

Kevin J. Maher
Second Reader

W. Matthew Carlyle
Chair, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Naval Supply Systems Command Weapon Systems Support currently uses Distributed Requirement Planning (DRP) to generate repair and procurement quantity recommendations for Supply Class IX repair parts. In an effort to provide an optimization-based method for making wholesale inventory decisions, a mixed integer linear optimization program termed Wholesale Inventory Optimization Model (WIOM) was developed as an alternative to existing tools. In its current form, however, WIOM does not explicitly consider the repair cycle for those national item identification numbers that may be either procured or repaired.

The thesis develops a next-event, discrete-event inventory simulation model, which we term Comparative Replenishment Stream Simulation (COMPRESS) to compare the effects of having two supply streams with differing lead times versus what we commonly see in a single supplier stream. With WIOM strictly considering a "composite" supply stream, the determination needs to be made if the disparity between inventory systems that utilize either a composite or repair-and-procure supply stream is great enough to warrant a WIOM reformulation. We select fill rate as the performance metric to compare these inventory systems. Through COMPRESS, we find that there is considerable difference in the resulting fill rates. Additionally, we find that as the variability in demand and lead time increases, the difference in the resulting fill rate values increases as well.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

CNO	Chief of Naval Operations
COMPRESS	Comparative Replenishment Stream Simulation
CRR	carcass return rate
DES	discrete-event simulation
DLA	U.S. Defense Logistics Agency
DLR	depot-level reparable
DRP	Distribution Requirements Planning
IDS	Individual Demand Schedule
IP	inventory position
LT	lead time
NAVSUP	Naval Supply Systems Command
NAVSUP WSS	Naval Supply Systems Command Weapon Systems Support
NIIN	national item identification number
PLT	procurement lead time
PNG	Peak Policy and the Next Generation Inventory Model
Q	order quantity
RTAT	repair turn-around time
s	order point *or* reorder point
SCM	Supply Chain Management
SPO	Service Planning and Optimization
SR	survivability rate
WIOM	Wholesale Inventory Optimization Model

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Naval material supply chain management (SCM) is the collection of processes that result in Navy customers receiving the parts and materials they need, when and where they need them, anywhere in the world (Naval Supply Systems Command Weapon Systems Support [NAVSUP WSS] 2018). "[SCM] is [Naval Supply Systems Command's (NAVSUP's)] largest product [and] service in terms of resources invested with over 3,000 civilian, military and contractor personnel involved, $21 billion of inventory on hand and an annual material budget of over $3.5 billion. It covers over 430,000 Supply Class IX repair part line items for which the [NAVSUP WSS] is responsible" (NAVSUP WSS 2018). Unfortunately, due to warehousing and budget restrictions, NAVSUP WSS is unable to guarantee any item will be always available and must place an emphasis on prioritizing the procurement of items based on criticality of need.

NAVSUP WSS currently uses Distributed Requirement Planning (DRP) to generate repair and procurement quantity recommendations for Supply Class IX repair parts based on calculating total shortages (Ellis 2018). DRP uses a heuristic approach towards this problem, and in an effort to provide an optimization-based method for making wholesale inventory decisions, Salmerón and Craparo (2017) developed a mixed integer linear optimization program termed Wholesale Inventory Optimization Model (WIOM) as an alternative to existing tools that provide these values.

In its current form, WIOM does not explicitly consider the repair cycle. For those national item identification numbers that may be either procured or repaired, depending on the number of carcasses available, NAVSUP WSS has thus far provided an "expected" lead time that is calculated using the probabilities that the item will be repaired or procured with the average lead times resulting from these two separate "streams" of acquisition (i.e., repaired or procured). However, WIOM's modeling framework could, in principle, be adapted in order to explicitly represent each of these events.

The thesis develops a next-event, time-advance inventory discrete event simulation model, which we term Comparative Replenishment Stream Simulation (COMPRESS) in

order to compare the effects of having two supply streams with differing lead times versus what we commonly see in a single supplier system. With WIOM strictly considering the latter, the determination needs to be made if the disparity between the two systems is great enough to warrant reformulation of WIOM.

Using COMPRESS, we select fill rate as the performance metric to compare inventory systems that utilize either a composite or repair-and-procure supply streams under various experimental designs. These experimental designs include combinations of historical or parametrically derived demands along with deterministic or normally distributed lead times to form experimental-design comparison groups. From our fill rates, we are able to calculate for a given item i, mean simulated fill-rate differences ($\overline{x^i}$s) and absolute differences ($\overline{y^i}$s) along with corresponding standard deviations (σ_{x^i}s and σ_{y^i}s, respectively).

This thesis gains several key insights. First, a high fraction of $\overline{x^i}$s and $\overline{y^i}$s exhibit a considerable amount of "difference" regardless of experimental-design comparison group. While the experimental-design comparison group that utilizes historical demands and normally distributed lead times has the highest $\overline{x^i}$ and $\overline{y^i}$ fractions equaling zero among all the comparison groups, these fractions of $\overline{x^i}$s and y^is are only 29% and 25%, respectively. Second, we notice a similar increasing trend in dispersion levels as the variability increases within the inventory system as we see in the $\overline{x^i}$ and y^i analysis. While the comparison group we previously note additionally exhibits the least amount of dispersion, 46% of the σ_{y^i} values and 61% of the σ_{x^i} values are greater than 0.1. Third, we further study the means in conjunction with the respective standard deviations. Taking the parametric demand with variable LT experimental-design comparison group as the extreme of our study, the increased proportions of this group's $\overline{x^i}$ vs. σ_{x^i}s and $\overline{y^i}$ vs. σ_{y^i}s at higher range values further indicate considerable variability within our data. Finally, we select a single item and use the Wilcoxon ranked-sum test to perform a formal statistical analysis comparing the item's resulting composite and repair-and-procure fill rates. After performing 600 replications each, the resulting p-value is 0.00753.

References

Ellis D (2018) Presentation on DRP Functionality provided to Javier Salmerón and Emily Craparo via personal communication, January 18.

Naval Supply Systems Command Weapon Systems Support (2018) Navy Supply Chain Management. Accessed August 23, 2018, https://www.navsup.navy.mil/public/navsup/wss/nscm/.

Salmerón J, Craparo E (2017) *Wholesale Inventory Optimization Model*, Release - 4.0. Project deliverable, Naval Postgraduate School, Monterey, CA.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my team of advisors, Professors Craparo and Salmerón. I truly appreciate their guidance and patience. And my second reader, Kevin Maher CDR, SC, USN (ret.), for his mentorship in and out of the classroom.

Finally, I cannot begin to express my gratitude to my wife and daughters. Without their love and encouragement, I would not have been able to complete this.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

Naval material supply chain management (SCM) is the collection of processes that result in Navy customers receiving the parts and materials they need, when and where they need them, anywhere in the world (Naval Supply Systems Command Weapon Systems Support [NAVSUP WSS] 2018). "[SCM] is [Naval Supply Systems Command's (NAVSUP's)] largest product [and] service in terms of resources invested with over 3,000 civilian, military and contractor personnel involved, $21 billion of inventory on hand and an annual material budget of over $3.5 billion. It covers over 430,000 Supply Class IX repair part line items for which the [NAVSUP WSS] is responsible" (NAVSUP WSS 2018). The Supply Class IX category includes, "Repair parts and components to include kits, assemblies, and subassemblies (repairable or non-repairable) required for maintenance support of all equipment" (NAVSUP 2015, p. 2–10). Unfortunately, due to warehousing and budget restrictions, NAVSUP WSS is unable to guarantee that each item will be always available and must place an emphasis on prioritizing item procurement on criticality of need.

Supply Class IX repair parts, known as depot-level repairables (DLR), possess the material control codes E, G, H, Q and X and warrant extra oversight in their inventory management due to their relative cost. These assigned codes are a single alphabetic character that serves to separate parts into manageable groups. According to the Chief of Naval Operations (CNO) (2017), "all DLRs will be managed under a one-for-one reorder policy with requisitions being limited to a quantity of one each" (p. 2). This means that for each requisition, the customer must submit the broken item (carcass) to the appropriate repair organization for evaluation, and if feasible, its subsequent repair. Upon completion of repair, the DLR reenters the supply system as a Ready-for-Issue asset to fulfill future requisitions.

This DLR inventory management process is analogous to the one a person might encounter during the replacement of an automobile's alternator. During the sale, the auto part dealership offers a price discount in exchange for the customer's faulty alternator. The auto part dealership will in turn send this faulty alternator to a repair facility for refurbishment. Once refurbishment is complete, the alternator will go back to the auto part dealership where it is becomes available for customer purchase.

The number of assets in a DLR's rotational pool does not stay constant over time. Two events may cause an item's exit from the rotational pool rather than undergoing repair. First, the item may be lost or completely destroyed. Second, even if the item is returned for repair, it may be so badly damaged as to be unrepairable. NAVSUP WSS uses the terms carcass return rate (*CRR*) and survivability rate (*SR*) in their Distribution Requirements Planning (DRP) model to recommend new-asset procurement quantities for compensating those losses (Ellis 2018a). NAVSUP WSS defines CRR as the probability that a carcass is returned to the appropriate repair facility, while SR is the probability that the repair facility is capable of restoring an item to a functional status, given it was returned (Ellis 2108a).

NAVSUP WSS currently uses DRP to generate repair and procurement quantity recommendations for Supply Class IX repair parts based on calculating total shortages (Ellis 2018a). DRP starts by estimating the projected inventory quantity for a given item i with the following equation:

$$\textit{Projected Inventory}_i = \textit{Supply}_i - \textit{Demand}_i, \tag{1}$$

where $\textit{Supply}_i$ is the sum of all on-hand Ready-for-Issue assets, the number of assets on order, and the number of carcasses in the repair system, and $\textit{Demand}_i$ is the sum of an item's forecasted demand during procurement lead time (*PLT*, i.e., the sum of the time required to administratively generate a procurement action, produce the item, and ship to the customer), unfilled customer demands, and items beyond physical repair (Ellis 2018a). We use the term "order" to denote the process of bringing inventory quantities into an inventory system, whereas "demand" removes inventory quantities from the system. When $\textit{Projected Inventory}_i$ reaches a "point of need" (where $\textit{Projected Inventory}_i$ breaches either zero or its safety stock level), DRP estimates the given item i 's total shortage with the following equation:

$$\textit{Total Shortage}_i = \textit{Projected Inventory}_i - \textit{Saftey Stock Level}_i, \tag{2}$$

where *Projected Inventory*$_i$ is the estimation from Equation (1) for an item, and *Safety Stock Level*$_i$ is the item's safety stock level (Ellis 2018a). Tersine (1994) defines safety stock as "extra inventory kept on hand as a cushion against stockouts due to random perturbations of nature or the environment" (p. 206). In other words, organizations put safety stock in place to serve as a buffer in the event of unplanned demands attempting to prevent on-hand quantities from declining to zero and causing a condition known as a stockout. More formally, Silver et al. (1998, p. 234) define safety stock as, "the average level of the net stock (on hand minus backorders) before a replenishment arrives."

Finally, DRP uses the item's point of need in conjunction with its target receipt date, order quantity duration (duration the order quantity is expected to last), and PLT to identify the time to generate the repair or procurement quantity recommendation.

DRP uses a heuristic approach that requires, as an input, the reorder point, planned minimum safety stock level, or both for each item. An item's reorder point (also known as order point) represents the inventory level that initiates an ordering action for inventory replenishment. In an effort to provide an optimization-based method for making wholesale inventory decisions, Salmerón and Craparo (2017) developed a mixed integer linear optimization program termed, Wholesale Inventory Optimization Model (WIOM), as an alternative to existing tools that provide these values. As a tool,

> WIOM comprises a series of mathematical optimization models, algorithms, and their computational implementations, which [NAVSUP WSS planners use] to help guide wholesale inventory decisions. Given certain assumptions and data inputs, WIOM calculates optimal reorder points [and safety levels] for tens of thousands of national item identification numbers (NIIN) in the maritime and aviation areas. (Salmerón and Craparo 2017, p. 1)

In its current form, WIOM does not explicitly consider the repair cycle. For those NIINs that may be either procured or repaired, depending on the number of carcasses available, NAVSUP WSS has thus far provided an "expected" lead time (LT) that is calculated using a weighted average of procurement and repair lead times. However, WIOM's modeling framework could, in principle, be adapted for explicitly representing these events. This thesis' goal determines if there exists a discernable performance difference between the current model and one which incorporates the repair cycle. If performance difference is significant, then it would warrant a WIOM reformulation.

B. LITERATURE REVIEW

In this section, we provide some relevant literature review for our study.

1. Inventory Management

"The control and maintenance of inventories of physical goods is a problem common to all enterprises in any sector of a given economy" (Hadley and Whitin 1963, p. 1). Each enterprise has its own set of inventory requirements that presents unique management problems. Subdividing the maintenance of inventories into systems of retail, wholesale, and manufacturing aids in defining the magnitude of complexity an enterprise experiences.

"[Wholesale] systems comprise organizations that purchase large quantities of manufactured goods for distribution to retail systems" (Tersine 1994, p. 5). This type of inventory management focuses on providing its customers products at an acceptable service level while maintaining inventories at distributed networks of storage centers at the lowest possible cost. Service level is defined as the number of stockouts over a given period, and it represents an organization's ability to meet their customers' demands. In the case of the U.S. military, organizations like the U.S. Defense Logistics Agency (DLA) and NAVSUP WSS play the wholesale inventory manager's role while individual units act as the retail systems. There are many different models designed to aid the inventory management process, and one example is the reorder-point, order-quantity (s, Q) model.

In the (s, Q) model, the reorder point (s) defines a value of the inventory position (IP) where a system initiates a procurement action. For a given item i, we calculate its IP as:

$$IP_i = OH_i + OO_i - BO_i, \tag{3}$$

where OH_i is the item's current on-hand inventory level, OO_i is the item's current on-order amount, and BO_i is the current number of backorders for the item. The order quantity (Q) represents the procurement amount of each order. Just as in calculating s, there exist multiple approaches to establish the Q. Financial considerations are commonly used as a guiding principle, and in the case of wholesale inventory systems, managers have to account for costs ranging from purchase to storage and maintenance fees.

As Salmerón and Craparo (2017) note, WIOM uses the (s,Q) system to model NAVSUP WSS's wholesale inventory. NAVSUP WSS, however, employs business rules that deviate from this system significantly. NAVSUP WSS does not explicitly calculate Q values for the procurement actions themselves. They rely on Navy Enterprise Resource Planning's rules in deciding the procurement amount and then attempt to maximize effectiveness with an appropriate reorder point. Secondly, NAVSUP WSS "tries to 'lump' its repair recommendations into quarterly buckets, so the system generates one consolidated purchase request per quarter. This process is further complicated because a significant proportion of the items are organically repaired with the workload forecast done over six-month periods" (Ellis 2018b).

There are numerous metrics for measuring an inventory system's performance, and the Department of Defense considers fill rate a primary metric in measuring a supply chain's reliability at the wholesale level (Department of Defense 2017). Fill rate is defined as the fraction of demands, f_i, met using on-hand inventory for a given item i. Silver et al. (1998, p.253-274) approximate its calculation with the following equation:

$$1 - f_i = \frac{1}{Q_i}\int_{s_i}^{\infty}(x - s_i) f_{X_i}(x)dx, \tag{4}$$

where X_i is the item's demand probability distribution (with density function $f_{X_i}(x)$, if continuous, or, if discrete, an analogous probability mass function); Q_i, is a pre-calculated order quantity for the item; and s_i, the reorder point based on IP. Salmerón and Craparo (2017) refine Equation (4) to better approximate fill rate in cases where the expected LT demand value exceeds the order quantity; that is, where multiple simultaneous orders are expected.

2. Discrete-Event Simulation of Inventory Systems

"Discrete-event simulation [(DES)] concerns the modeling of a system as it evolves over time by a representation in which the state variables change instantaneously at separate points in time" (Law 2007, p. 6). This limits the system's state variables to update only at times when an event occurs. Law (2007, p.9) defines a system state as "the collection of state variables necessary to describe the system at a particular time." Additionally, Law defines an event as "an instantaneous occurrence that may change the state of the system" (Law 2007, p. 6). Although this system state updating process can greatly reduce the time requirement for running a simulation, it does not allow for continuous changes in the state variables.

Tako and Robinson (2012) note that 86 out of 127 papers (68%) between the years 1996 and 2006 which use modeling to study logistics and supply chains employ DES. These figures speak to DES's many practical applications in investigating the various aspects of inventory systems, and to gain an understanding of their applicability in inventory analysis, we review several examples: Vidalakis et al. (2013) use DES to gain insight into how demand fluctuations affect LT and cost efficiency in the construction industry. Their findings reveal that during low demand conditions, these fluctuations affect LTs significantly while increases in demand have a negatively exponential impact on cost efficiency as inventory costs increase. Caliguire (2009) establishes a model for studying the AIM-9 Sidewinder Missile repair process line to reduce repair cycle times and thus improve replenishment rates. The study's significant findings include identifying the process in the repair cycle that leads to the greatest reduction in mean cycle time and the benefits of reorganizing the current workforce into an optimal configuration. The most relevant example comes from Bachman et al. (2016). Using simulation, they compare the software package (a combination of inventory solutions Peak Policy and the Next Generation Inventory Model [PNG]) that DLA currently uses against other inventory forecasting methods for items with infrequent and high-variable demand. As they note, these are the majority of the hardware items that DLA stocks. Through their research, they conclude that PNG outperforms all the other inventory control methods. PNG performs so well that, since its implementation, DLA has saved nearly $400 million per year, improved its customer service while reducing buyer workload, and has not experienced an increase in inventory levels.

3. Previous WIOM Simulation Studies

Roth (2016) uses a DES tool termed Comparative Inventory Simulation to contrast three different methods for selecting reorder points. Roth's first method entails simply calculating the reorder point based on the mean and the LT demand's standard deviation. His second method is a contractor-provided tool called Service Planning and Optimization (SPO) that NAVSUP WSS was using at the time. SPO is considered a "black-box" program. NAVSUP WSS does not possess the proprietary rights for accessing SPO's model or algorithms (Roth 2016). His last method is the mixed integer linear optimization model WIOM developed by Salmerón and Craparo (2017).

Running simulations over five types of material, Roth evaluates 24 total cases and observes WIOM-derived reorder points providing the highest fill rate in 22 of the 24 cases. Secondly, he notes that WIOM's performance in estimating fill rates is largely dependent upon the probability distribution used for each NIIN. Fits under Poisson distributions are generally accurate, whereas fits under the gamma distribution are consistently less accurate. Lastly, he observes simulated backorder lengths being up to 50% shorter when using WIOM versus the other methods.

After the aforementioned thesis by Roth, NAVSUP WSS developed an interest in adding an additional WIOM feature known as "Persistence." Persistence enables WIOM to preserve the previous solutions' values therefore reducing what we know as churn (the change in solutions between model runs). With this new feature addition, Teter (2018) conducts 15 different experiments using his Comparative Optimized Results Simulation model to determine whether varying persistence and WIOM's running periodicity has any detrimental effects on simulated fill rates. Unlike Roth, Teter uses 4.5 years of historical demand data that NAVSUP WSS provides to create the study's demand signals. This method removes a major assumption the researcher must make about which probability distribution the demands follow. From his observations, Teter concludes that WIOM produces the best results when run on a quarterly basis. Optimal solutions appear to “expire” with time passage and changing demand patterns. Secondly, he notes that the persistence parameter addition has little impact on fill-rate performance while reducing churn by up to 99%.

C. OBJECTIVES AND SCOPE

Our research objective is to conduct a DES to compare the effects of having two supply streams with differing LTs versus what we commonly see in a single supplier system. With WIOM strictly considering the latter, we need to determine whether the disparity between the two systems is great enough to warrant a WIOM reformulation. In this study, we compare the fill rates resulting from weighted composite and dual-stream supply systems under various conditions.

The thesis develops a next-event, time-advance inventory DES model, which we term Comparative Replenishment Stream Simulation (COMPRESS). COMPRESS incorporates the Simplekit modeling tool developed by Sanchez and Oliver (2018) and is capable of simulating various inventory systems. Although COMPRESS possesses this functionality, we only consider the relative information from historical, maritime, non-nuclear repairables for use as input in our simulations. We limit ourselves to these test cases to focus our research’s scope and effectively cover our objective.

II. METHODOLOGY AND DATA

A. METHODOLOGY

In this section, we introduce our various methodology and data elements.

1. Background and Rationale for Design Choice

Extending a DES design example Law (2007) offers, we examine a simulation that compares alternative ordering policies for a company that carries a single product. Assume the company uses a periodic-review, lost-sale inventory model, and would like to estimate the correct inventory level to stock each month during a given time period. Although they only carry a single product, the magnitude of random occurrences the company could potentially witness complicates finding a definite solution for this problem. These random perturbations range from all aspects of customer demands to variations in order arrival times.

Law (2007) outlines the computer program's organization and logic that he uses for conducting the simulation. Event "type" assignment serves as the basis for organizing the program into a logical order. Event types indicate to the program the priority at which an event's evaluation occurs. That is, if the simulation schedules multiple events for the same time, evaluation of the event with the lowest type occurs first. As shown in Table 1, there are four types in this simulation. In event-type order, the first is order arrival from the supplier. This event represents the due-in receipt quantities that the inventory system requires to maintain stock levels for meeting future demands. The second is customer demand, which introduces two variation sources into the model. First, demands arrive independently at sporadic intervals. Secondly, each demand's quantity potentially varies as well. The third type is simulation termination. This stops the program from progressing further and collects the final system state for analysis. The final type is IP evaluation with possible ordering. On a monthly basis, the program evaluates the IP to see if the current inventory level requires replenishment. If the IP is at or below the reorder point, the program schedules an order arrival.

Table 1. Event Description and Event Type. Adapted from Law (2007).

Event Description	Event Type
Arrival of order from supplier	1
Customer demand	2
Simulation termination	3
Inventory position evaluation	4

As shown in Figure 1, event graphs are a convenient means to depict a simulation's flow and the relationships between events. Extending on Törn's (1981) idea of modifying Petri net graphs to aid in coding and simulation model design, Schruben (1983) proposes a method for representing event-oriented systems graphically. In this approach, nodes represent events while directed arcs depict the relationships between them (Law 2007). We think of relationships as the effects one event has on another, and in this example, we see instances of events scheduling other events, events scheduling themselves, and the passing of information for use during subsequent evaluations. Through Buss' (1996) review and tutorial on event graphs, we see that the lowercase t's with subscripts represent the delay an event assigns to another event's evaluation. In this problem, there is both deterministic and random delay assignment. Next, we notice that not all event evaluations cause the automatic scheduling for the events to which they connect. As seen between the "evaluate IP" and "order" events, the condition of the IP being less than or equal to the reorder point is a requirement for the order event scheduling. Finally, each node indicates its effect on the state variables. In the case of customer demands, as the simulation generates a demand, it evaluates the generated demand against the on-hand quantity to see if it suffices. If it does, the simulation decrements the demand quantity from the on-hand quantity. It is important to note that this is a simple inventory system example which does not account for backorders. If a demand arrives and the on-hand quantity is less than the demand quantity but greater than zero, the system will create a partial fill and decrement the on-hand inventory to zero. If there is zero on hand, then the system considers this a loss sale with no further action.

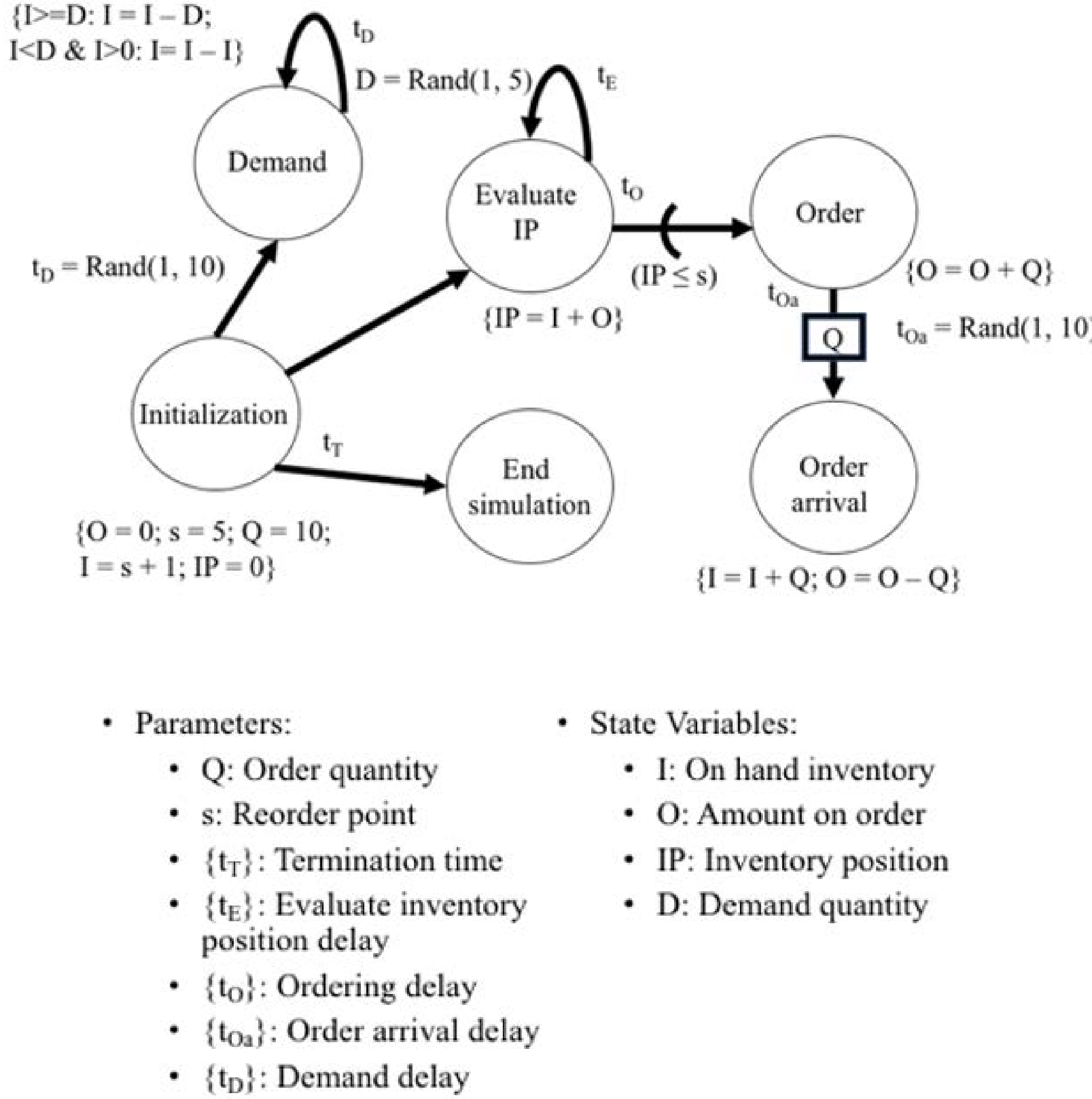


Figure 1. DES Example Event Graph. Adapted from Law (2007).

A simulation model is deterministic if it "does not contain any probabilistic (i.e., random) components" (Law 2007, p. 6). This simulation type will produce the same results each time a researcher performs it for a given input data set. That is, the researcher observes no variation in the output. It is often beneficial to start studies under deterministic conditions to gain an insight into the relationships between the input and output. From these relationships, researchers can develop an understanding of how variation in the input parameters translates to changes in the output with the potential benefit of identifying those with the greatest effect.

Stochastic simulation models conversely have “at least some random input components” (Law 2007, p. 6). A researcher conducts this simulation type in an attempt to capture the randomness as seen in nature. It is significant to note that, since the inputs into these models are random, they give rise to an output that is also random. It is with this randomness that the researcher must exercise caution because although the results can lead to an understanding of all possible outcomes, they are in fact just a representation “of the true characteristics of the model” (Law 2007, p. 6) and do not constitute a definite answer for a problem.

COMPRESS incorporates both deterministic and stochastic simulation models which we visually present using event graphs. Although we note that there are limitations to each model, we believe they provide useful insights into our research question.

2. Experiments and Assumptions

Simplekit is a Python-based, event-queueing toolkit that creates event routine instances with future time stamps and relative priorities, and then places them into a priority queueing system (Sanchez and Oliver 2018). It is important to note that due to the number of event routines, COMPRESS extends this concept and adds a queue-order placement parameter to improve fidelity. In determining an event’s time stamp, COMPRESS uses either deterministic or stochastic methods for scheduling the demand and order routines while executing all other events immediately. The demand and order routines serve as the basis for the variation in COMPRESS, and we design the following experimental combinations based on three factors:

- historical vs. parametrically-derived demands
- composite (one-stream) vs. “coin-flip” (repair-and-procure) ordering
- deterministic vs. normally-distributed LTs

“Composite” refers to a weighted single-stream supply system (combination of procurement and repair, as is currently used in WIOM), and “coin-flip” refers to two supply streams (repair or procure, which occur with certain probability).

As Teter (2018) notes, NAVSUP WSS only provides 4.5 years of historical data. This fact limits those simulations using historical demands to this length of time. On the other hand, we are capable of extending those simulations that parametrically derive their demands out to a greater duration. For these simulations, we choose a five-year duration.

To carry out our simulations in a simple environment, we employ several assumptions in COMPRESS. First, all repair orders are for an order quantity of 1. Second, the historical demands represent a steady-state system and the simulations using them do not require a “warm-up” period. Third, COMPRESS bases its procure-or-repair decision on a “coin flip” where we define the probabilities of the procure and repair decisions as follows:

$$ProcureProbability = 1 - CRR \times SR \tag{5}$$

$$RepairProbability = CRR \times SR \tag{6}$$

The assumption is that this coin flip approximates the actual process (where COMPRESS places a repair order when a repairable carcass is available). Since each demand generates a carcass that is repairable with probability $CRR \times SR$, the coin flip simulates the existence of such repairable carcass. (A detailed simulation of DRP tracking individual carcasses is beyond this research’s scope.) Fourth, the SPO data files (Craparo 2018) we use in our research accurately reflect the NIINs’ historical information. Lastly, when we select stochastic LTs, COMPRESS models both procure and repair LTs as normally distributed.

B. DATA

To run COMPRESS, we start by generating an input parameter set. We either interactively enter this set into an Excel workbook, draw it from historical requisition data in combination with the WIOM (or SPO) solution files, or use the WIOM (or SPO) solution files by themselves. In the remainder of this document, we use only inputs from SPO, but a researcher could perform the analysis with calculated reorder points from WIOM or any other tool.

1. Interactive Parameter Input

Interactively entering input parameters provides us with a method to design experiments for studying the effects the input has on the output. That is, we are capable of observing which input parameters cause the greatest effect during a simulation's course under a given set of conditions.

2. Historical Requisition Data with SPO Solution Files

As Figure 2 depicts, we start COMPRESS's input generation process by organizing and combining the files that contain the requisition data from fiscal years 2013 through 2017 (1 Oct 2012–30 Sep 2017) into a master requisition data frame. NAVSUP WSS (Ellis 2017) provides these files (in comma-separated value format) which represents all the requisitions received by them throughout this period. At this point, our data frame contains a number of individual requisitions that are not pertinent to our simulation. We reduce this number of requisitions by identifying and keeping only those that contain maritime, non-nuclear, repairable NIINs. Upon examining this requisition subset, we note that there are multiple instances of demand quantities greater than one. Although this is contrary to the CNO's (2017) one-for-one demand policy, COMPRESS is capable of performing simulations with demand quantities greater than one. Next, we group the requisition elements by NIIN (762 in total) and organize the demand quantity and Julian date into quarterly lists where they wait until we process the quarterly SPO solution files.

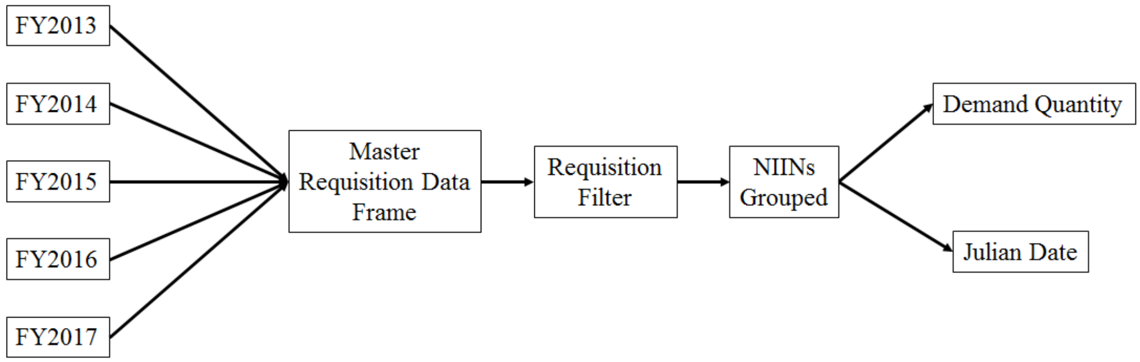


Figure 2. Historical Requisition Processing

After developing the quarterly lists, we combine the quarterly SPO solution files into a master SPO data frame that we filter by NIIN to obtain the remaining elements COMPRESS requires for a complete input set. Refer to Figure 3. These elements include: quarterly demand, reorder point, a composite order quantity consisting of procure and repair quantities, *PLT*, repair turn-around time (*RTAT*) (the amount of time it takes an item to complete the repair cycle), *PLT* sigma ("sigma" is used as an alternate term for standard deviation), *RTAT* sigma, *SR*, and *CRR* (which we organize into a list to coincide with the quarterly demand quantity, and Julian date list). Because this study's objective is to gain insight into how two replenishment streams affect fill rate, we must derive the procurement quantity from the composite order quantity element we find in our source files (repair is automatically set to 1 for each order quantity under our assumptions). Next, we merge this data with the NIINs' historical demand information into a single input line that we subsequently evaluate for completeness. If any element is missing, we purge this input line and continue with the next line's creation. Conversely, if the input line is complete, we write it to a master output file in comma-separated value format. Table 2 contains examples and element types of the final input.

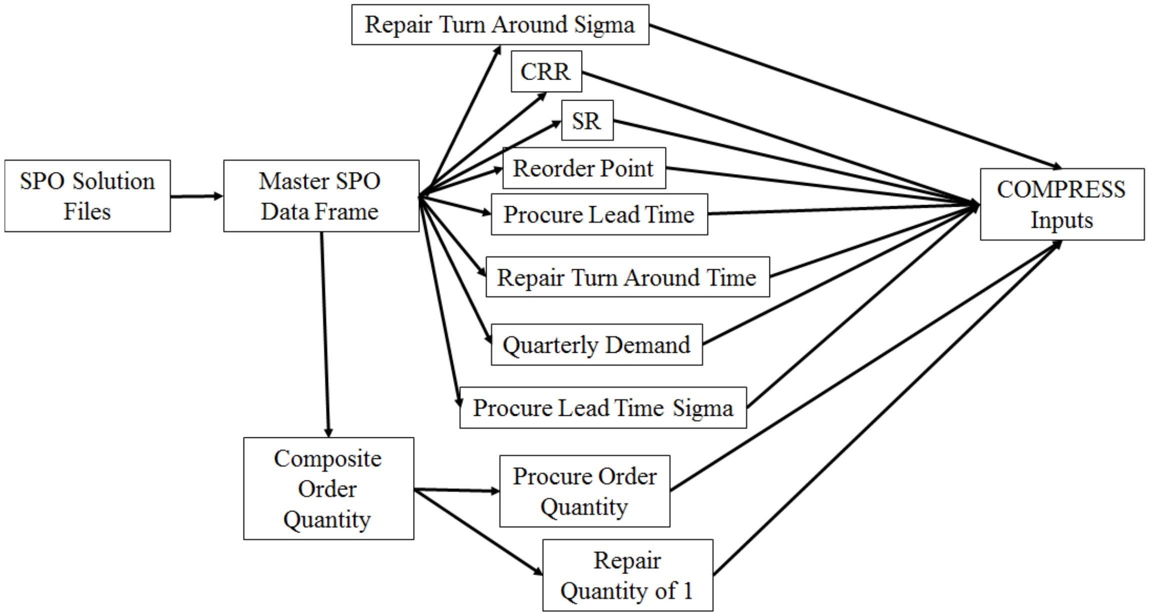


Figure 3. SPO File Processing

Table 2. COMPRESS Input Using Historical and SPO Data with Two Normally Distributed *LT*s

	Name	Example	Input Element Type[b]
1	NIIN	000011632	Character string
2	Demands	[[1, 1], [0], …, [1]]	List of integers by quarter
3	Reorder Point	[5, 3, 7, …, 5, 6]	List of integers
4	$Q_{Procure}$	[1, 1, 1, …, 1, 1]	List of integers
5	Q_{Repair}	[1, 1, 1, …, 1, 1]	List of integers
6	Target Stock Level	6	integer
7	Julian Dates[a]	[[25, 46], [0], …, [15]]	List of integers by quarter
8	*PLT*	[2.3, 2.2, …, 2.5, 2.3]	List of floats
9	*RTAT*	[1.3, 1.2, …, 1.5, 1.3]	List of floats
10	Procure Probability	[0.2, 0.2, …, 0.1, 0.3]	List of floats
11	Demand Distribution	0	Integer
12	Demand Sigma	[1.2, 1.3, …, 1.1, 1.3]	List of floats
13	*PLT* Distribution	2	Integer
14	*PLT* Sigma	[1.2, 1.3, …, 1.1, 1.3]	List of floats
15	*RTAT* Distribution	2	Integer
16	*RTAT* Sigma	[1.2, 1.3, …, 1.1, 1.3]	List of floats
17	Demand Cycles[c] per Year	4	Integer
18	Simulation Time	5	Integer
19	Deterministic Intervals	0	Integer
20	On Hand Starting Position	2	Integer
21	Demand Interarrival Distribution[d]	1	Integer
22	Display Event Log	0	Boolean
23	Display Graph	0	Boolean
24	Composite	0	Boolean

[a] Julian dates are formatted to coincide with day of the quarter in which COMPRESS is currently operating.

[b] All integer or float values are non-negative.

[c] Demand Cycle is a set time interval containing demands as COMPRESS's configuration defines.

[d] See Table 3 and Table 4.

3. SPO Solution Files

To use the SPO solution files by themselves, the process is similar to the one we describe in Section II.B.2. The difference is that the data elements we draw out to generate the quarterly demand signals may differ. For these, we use a NIIN's mean demand, $\hat{\boldsymbol{x}}$, along with the associated standard deviation, $\hat{\sigma}$. As outlined by Salmerón and Craparo

(2017), the relationship between these terms determines the specific probability distribution that COMPRESS uses to generate random demands. Table 3 summarizes these relationships, while Table 4 contains the means for distribution parameter estimation.

Table 3. Distribution Selection Criteria.
Adapted from Salmerón and Craparo (2017).

	Name	Criteria	Exception
1	Deterministic	$\hat{\sigma}/\hat{x} < 0.01$	
2	Truncated Normal	$\hat{x} \geq 20$	$\hat{\sigma}/\hat{x} \geq 2.325$ use (6)
3	Poisson	$\hat{x} \leq 20$ and $0.9 \leq \hat{\sigma}^2/\hat{x} \leq 1.1$	
4	Negative Binomial	$\hat{x} \leq 20$ and $\hat{\sigma}^2/\hat{x} > 1.1$	Upon error use (6)[a]
5	Binomial	$\hat{x} \leq 20$ and $\hat{\sigma}^2/\hat{x} < 0.9$	Upon error use (6)[a]
6	Gamma	Listed Exceptions	

[a] Errors are defined in Salmerón and Craparo (2017).

Table 4. Probability Distributions and Estimators to Approximate Demand.
Source: Salmerón and Craparo (2017).

Prob. Dist.	Name (Parameters)	Mean	Variance	Variance/Mean	Parameter Estimation (Based on $\hat{x}, \hat{\sigma}$)
1	Poisson (λ)	λ	λ	1	$\lambda = \hat{x}$ or $\lambda = \hat{\sigma}^2$
2	Negative Binomial (r, p) (r = # failures to stop) (Generalized Neg. Bin.)	$rp / (1\text{-}p)$	$rp / (1\text{-}p)^2$	$1 / (1\text{-}p) > 1$	$\tilde{p} = 1 - \frac{\hat{x}}{\hat{\sigma}^2}$; $\tilde{r} = \frac{\hat{x}^2}{\hat{\sigma}^2 - \hat{x}}$
3	Normal (μ, σ)	μ	σ^2	σ^2 / μ	$\mu = \hat{x}$; $\sigma = \sqrt{\hat{\sigma}^2}$
4	Binomial (n, p)	np	$np(1\text{-}p)$	$1\text{-}p < 1$	$\tilde{p} = 1 - \frac{\hat{\sigma}^2}{\hat{x}}$; $\tilde{n} = \frac{\hat{x}^2}{\hat{x} - \hat{\sigma}^2}$
5	Gamma (k, θ) (k=shape, θ=scale)	$k\theta$	$k\theta^2$	θ	$k = \frac{\hat{x}^2}{\hat{\sigma}^2}$; $\theta = \frac{\hat{\sigma}^2}{\hat{x}}$
6	Deterministic (μ)	μ	0	0	$\mu = \hat{x}$

C. METAMODEL

As Figure 4 shows, our research incorporates COMPRESS into a larger program system to produce the output we desire.

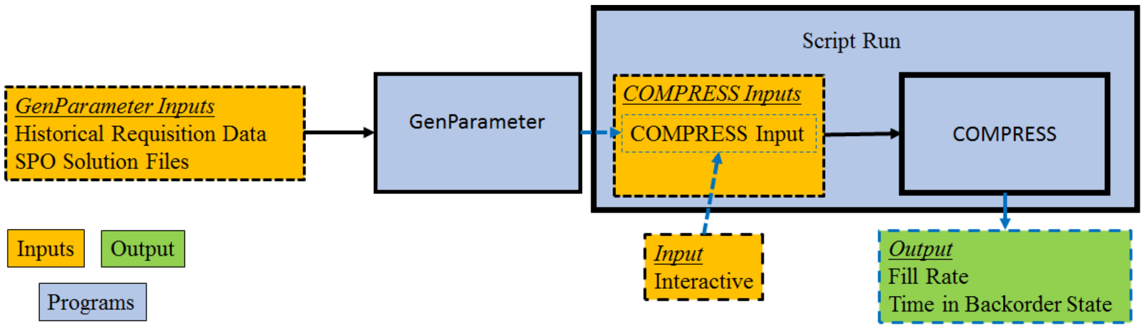


Figure 4. Metamodel Relationship

This metamodel (or model of models) includes three programs that vary in function and degree of complexity. From input generation to DES, each program takes in input and produces output for use as input by the next program in the schema. These programs include

- script run—takes COMPRESS's inputs and COMPRESS itself as arguments and runs the number of iterations the user defines
- GenParameter—creates COMPRESS inputs by
    - combining historical requisition data with the SPO solution files or
    - parametrically generating demands and combining with SPO solution files
- COMPRESS—as we note, our DES

As we outline in Section II.B, there are three ways we generate the input for COMPRESS. This in turn directs the path through our metamodel's network that we must follow for COMPRESS to achieve output. These specific paths include

- interactive—User directly enters input into COMPRESS
- historical data or parametric fit with SPO solution—GenParameter-COMPRESS

D. COMPRESS DEVELOPMENT

In the following, we discuss the various aspects of COMPRESS along with its development.

1. Overview

According to Law, DES using a next-event, time-advance approach frequently incorporates the following components:

> *System state*: The collection of state variables necessary to describe the system at a particular time
>
> *Simulation clock*: A variable giving the value of simulated time
>
> *Event list*: A list containing the next time when each type of event will occur
>
> *Statistical counters*: Variables used for storing statistical information about system performance
>
> *Initialization routine*: A subprogram to initialize the simulation model at time 0
>
> *Timing routine*: A subprogram that determines the next event from the event list and then advances the simulation clock to when the time of that event is to occur
>
> *Event routine* [event]: A subprogram that updates the system state when a particular type of event occurs (there is one event routine for each event type). (Law 2007, p. 9)

COMPRESS is a class-based, object-oriented, Python program that incorporates all of these components with focus on event routines that model either a composite or two-supply stream inventory system. In describing COMPRESS's development, we break the discussion into three parts. First, we describe Simplekit and examine how it coordinates COMPRESS's various logic aspects. Second, we describe the event routines in COMPRESS's inventory system with an examination of the actions they perform. Finally, we conclude our discussion with presenting select event graphs demonstrating COMPRESS's versatility in representing various inventory system configurations.

According to Sanchez and Oliver (2018), Simplekit is a discrete-event modeling tool that bases its logic on Schruben's Event Graphs. Utilizing the Priority Queue class in the Python Queue module as an event list, Simplekit provides the methods by which COMPRESS initializes the pending queue, sets simulation time to zero, tracks simulation time, adds and removes events from the queue, stores and retrieves event notice info, progresses forward in time, and terminates. Logically, the event queue itself provides the motive force by which the simulation moves forward in time. While the event queue is not empty, a Priority Queue method evaluates the events' attributes to find and remove the next event. We define a method as a function that performs a specific task on the object it belongs to while the attributes provide the distinguishing characteristics by which the method evaluates the events. The attributes belonging to each event include "delay," "event type" (or priority), and "numerical order" of the events placement in the queue. As COMPRESS's events perform their actions, the Priority Queue method matches the system time to the event's delay attribute and thus moves our simulation forward in time.

Eight events comprise the intricate network of COMPRESS's inventory system. Table 5 provides a brief event type description for each event. Using an input parameter set to shape the simulation's operating conditions, COMPRESS schedules events, evaluates the system state, makes operating decisions and updates the state variables until it reaches simulation termination. In Figure 5, we provide a brief example of the event log for a fictional NIIN's DES that demonstrates the evaluation of COMPRESS's inventory system's events along with how the state variables potentially fluctuate over time. In Figure 6, we present a graphical representation of the change in state variable's quantities with respect to simulation time during the DES.

Next, we describe each event in detail.

Table 5. COMPRESS' Inventory System's Events, Event Descriptions, and Event Types

Event	Event Description	Event Type
Demand Cycle	Generates the number of demands per cycle	1
Individual Demand Cycle	Distributes number of demands throughout the cycle	2
Fill Demand	Fills demands from on hand inventory	3
Back Order	Creates backorder if demand is greater than on hand inventory	4
Update IP	Order arrival processing	5
Evaluate IP	Places an order for the amount *Q* to arrive at LT	6
Order	Updates the IP	7
Order Arrival	Evaluates the IP for placing an order	8

```
NIIN, Demands, Full Orders Processed, Partial Orders Processed, Stockout Count, Backorder Count, Time in Backorder, Service Level, Fill Rate
11-231-3423, 57, 12, 0, 45, 45, 1443.038932041306, 0.15000000000000002, 0.21052631578947367
Time:     2270   Event: Init             Onhand Inventory:  4   Back Orders:  0   On Order:  0   Inventory Position:  0
Time:     2270   Event: UpdateInvPos     Onhand Inventory:  4   Back Orders:  0   On Order:  0   Inventory Position:  4
Time:     2270   Event: EvaluateInvPos   Onhand Inventory:  4   Back Orders:  0   On Order:  0   Inventory Position:  4
Time:     2271   Event: DemandCycle      Onhand Inventory:  4   Back Orders:  0   On Order:  0   Inventory Position:  4
Time:     2271   Event: IndDMDSched      Onhand Inventory:  4   Back Orders:  0   On Order:  0   Inventory Position:  4
Time:     2276   Event: FillDemand       Onhand Inventory:  3   Back Orders:  0   On Order:  0   Inventory Position:  4
Time:     2276   Event: UpdateInvPos     Onhand Inventory:  3   Back Orders:  0   On Order:  0   Inventory Position:  3
Time:     2276   Event: EvaluateInvPos   Onhand Inventory:  3   Back Orders:  0   On Order:  0   Inventory Position:  3
Time:     2276   Event: Order2           Onhand Inventory:  3   Back Orders:  0   On Order:  3   Inventory Position:  3
Time:     2276   Event: UpdateInvPos     Onhand Inventory:  3   Back Orders:  0   On Order:  3   Inventory Position:  6
Time:     2276   Event: EvaluateInvPos   Onhand Inventory:  3   Back Orders:  0   On Order:  3   Inventory Position:  6
Time:     2284   Event: FillDemand       Onhand Inventory:  2   Back Orders:  0   On Order:  3   Inventory Position:  6
Time:     2284   Event: UpdateInvPos     Onhand Inventory:  2   Back Orders:  0   On Order:  3   Inventory Position:  5
Time:     2284   Event: EvaluateInvPos   Onhand Inventory:  2   Back Orders:  0   On Order:  3   Inventory Position:  5
Time:     2306   Event: FillDemand       Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  5
Time:     2306   Event: UpdateInvPos     Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  4
Time:     2306   Event: EvaluateInvPos   Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  4
Time:     2362   Event: DemandCycle      Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  4
Time:     2362   Event: IndDMDSched      Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  4
Time:     2363   Event: FillDemand       Onhand Inventory:  0   Back Orders:  0   On Order:  3   Inventory Position:  4
Time:     2363   Event: UpdateInvPos     Onhand Inventory:  0   Back Orders:  0   On Order:  3   Inventory Position:  3
Time:     2363   Event: EvaluateInvPos   Onhand Inventory:  0   Back Orders:  0   On Order:  3   Inventory Position:  3
Time:     2363   Event: Order2           Onhand Inventory:  0   Back Orders:  0   On Order:  6   Inventory Position:  3
Time:     2363   Event: UpdateInvPos     Onhand Inventory:  0   Back Orders:  0   On Order:  6   Inventory Position:  6
Time:     2363   Event: EvaluateInvPos   Onhand Inventory:  0   Back Orders:  0   On Order:  6   Inventory Position:  6
Time:     3007   Event: OrderArrival     Onhand Inventory:  3   Back Orders:  0   On Order:  3   Inventory Position:  6
Time:     3007   Event: UpdateInvPos     Onhand Inventory:  3   Back Orders:  0   On Order:  3   Inventory Position:  6
Time:     3007   Event: EvaluateInvPos   Onhand Inventory:  3   Back Orders:  0   On Order:  3   Inventory Position:  6
Time:     3013   Event: FillDemand       Onhand Inventory:  2   Back Orders:  0   On Order:  3   Inventory Position:  6
Time:     3013   Event: UpdateInvPos     Onhand Inventory:  2   Back Orders:  0   On Order:  3   Inventory Position:  5
Time:     3013   Event: EvaluateInvPos   Onhand Inventory:  2   Back Orders:  0   On Order:  3   Inventory Position:  5
Time:     3023   Event: FillDemand       Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  5
Time:     3023   Event: UpdateInvPos     Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  4
Time:     3023   Event: EvaluateInvPos   Onhand Inventory:  1   Back Orders:  0   On Order:  3   Inventory Position:  4
```

Figure 5. A Sample Event Log for a Fictional NIIN where $\{s = 3; Q_{Procure} = 3; Q_{Repair} = 3; PLT = 1; RTAT = 1\}$

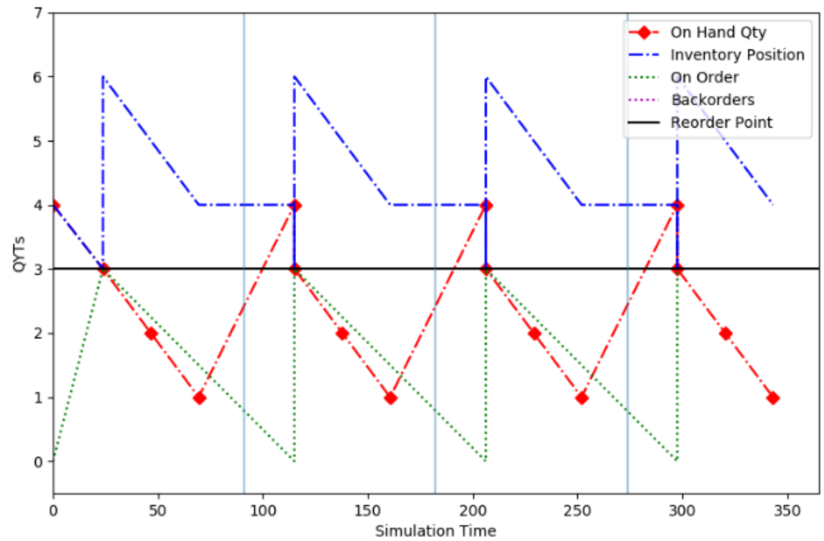


Figure 6. State Variable Quantities for Fictional NIIN DES with Respect to Simulation Time

2. Event Demand Cycle

COMPRESS progresses through a simulation in terms of demand cycles, and for DESs using historical information with SPO solution files, we define the cycle length in quarters. As the new cycle starts, Demand Cycle generates the number of demands that the inventory system observes during this period before it self-schedules the start of the next cycle. Depending on how we initialize COMPRESS, Demand Cycle performs one of four options with respect to the demand parameter. As we mention in Section II.B, three of these options include passing historical demands, demands the user defines, and demands Demand Cycle generates parametrically. The fourth option we introduce here is instances of zero demand. Whether from historical or parametric means, if the method Demand Cycle is using produces zero demands, then it automatically self-schedules to start the next cycle. Before terminating its routine, Demand Cycle's final action is to schedule and pass the cycle length to the event Individual Demand Schedule (IDS).

3. Event Individual Demand Schedule

IDS is responsible for scheduling the individual demands throughout the cycle depending on the demand types Demand Cycle passes. For demands that are historical, IDS normalizes the Julian dates to coincide with dates within the cycle. For demands that are deterministic, IDS is capable of using either of the following equations for demand placement in the cycle:

$$Step = Cycle\,Length\,/\left(Demands + 1\right), \tag{7}$$

$$Step = Cycle\,Length \big/ Demands, \tag{8}$$

where *Cycle Length* is the length of the cycle and *Demands* is the number of demands IDS receives from Demand Cycle. IDS uses *Step* to schedule demands at successive intervals throughout the cycle. Although IDS is capable of using either equation, we recommend using Equation (7) in lieu of (8) because of Equation (7)'s ability to evenly distribute the demands' interarrival times throughout the cycle. Finally, COMPRESS's last method distributes the demands randomly throughout the cycle utilizing the following formula:

$$Delay = a + u(b - a), \tag{9}$$

where *Delay* is the specific time when the demand occurs, *a* and *b* are the lower and upper bounds of the cycle period, respectively, and *u* is a random value generated from a uniform distribution between zero and one. IDS performs this operation for each demand that Demand Cycle passes. Unlike Demand Cycle, which schedules one instance for its following event, IDS schedules its follow-on event (Fill Demand) for each demand it distributes.

4. Event Fill Demand

During Fill Demand evaluation, the inventory system's current state presents two options. First, if the on-hand inventory is greater than or equal to the demand quantity, it will fill the demand, update the current on-hand inventory, and subsequently update the IP. Second, if the demand is greater than the on-hand inventory, Fill Demand will schedule and pass the demand quantity to the event Backorder.

5. Event Backorder

In complexity terms, Backorder is relatively simple in comparison to COMPRESS's other events. It updates the backorder count and controls the timer that tracks the accumulative time when the inventory system is in a backordered state. COMPRESS's backorder count is a queueing system that allows the number of backorders to increase or decrease as it creates or fills them. Backorder controls the backorder timer by evaluating the backorder count's relation to zero. If the count is equal to zero, Backorder preforms no action. If the count moves from zero to one, Backorder starts the timer. If the count is greater than zero and does not meet the previous condition, Backorder allows the timer to continue to run. We choose to track the time the system is in a backorder state to add an additional distinguishing characteristic beyond fill rates for each inventory system configuration. This additional characteristic enables us to judge each configuration's ability to manage items with relatively longer LTs. We now reach the point where Backorder schedules the event Update IP before it terminates.

6. Event Update IP

Backorder, Order, and Order Arrival possesses the ability to schedule Update IP to simulate an IP continuous review management process. These events potentially affect the state variables on which the inventory system bases the IP. It is Update IP's responsibility to ensure the IP is a current reflection of these state variables, and it uses Equation (3) for its calculation. Upon completion of this calculation, the event performs its final action and schedules the event Evaluate IP.

7. Event Evaluate IP

For COMPRESS's inventory system to place an order, it must first evaluate an item's IP against its s. For instances where $IP \leq s$ Evaluate IP schedules the event Order with no delay. Conversely, for instances where $IP > s$ Evaluate IP terminates with no further action.

8. Event Order

Upon the determination that the inventory system needs to place an order, Order generates an order utilizing one of two methods depending on the inventory system's configuration. If the system is set to the composite method, Order will pass a composite value of Q (a weighted average of the number of items in a procurement and the number of items in a repair order, as the SPO solution files define) while it schedules the event Order Arrival to either a deterministic or a normally distributed composite LT. COMPRESS uses the following equation to calculate *CompositeLT* for a given item *i*:

$$CompositeLT_i = PLT_i \times (1 - CRR_i \times SR_i) + RTAT_i \times (CRR_i \times SR_i). \tag{10}$$

Alternatively, if the system is set to a repair and procure method, Order first generates *u* for a given item *i* with the following:

$$u_i \overset{d}{=} Uniform(0,1). \tag{11}$$

Order then uses u_i to determine a random order choice, f_i, by utilizing the following:

$$f_i = \begin{cases} Repair & \text{if } u_i \leq SR_i \times CRR_i \\ Procure & \text{if } u_i > SR_i \times CRR_i \end{cases}. \tag{12}$$

Upon f_i 's determination, Order uses the following to calculate Q_i:

$$Q_i(f_i) = \begin{cases} 1 & \text{if } f_i = Repair \\ Max\left(1, \left[\dfrac{Q_{composite_i} - 1(SR_i \times CRR_i)}{1 - (SR_i \times CRR_i)}\right]\right) & \text{if } f_i = Procure \end{cases}, \tag{13}$$

where $Q_{Composite_i}$ is the item's composite Q value. Just as in Order's composite method, Order's repair and procure method additionally passes its Q value while scheduling the event Order Arrival with either a deterministic or normally distributed PLT or RTAT to coincide with the random order choice.

9. Event Order Arrival

Upon Order Arrival evaluation, one of three possible actions occurs, depending on the backorder count's state. In the first, the backorder count is equal to zero. In this instance, Order Arrival adds the Q value to the on-hand quantity and then schedules the event Update IP before terminating. The second and third action, on the other hand, occur when the backorder count is greater than zero. Depending on the Q value, it will either be enough to fill the outstanding backorders with the remaining Q value supplementing the on-hand quantity, or it will not be enough, and therefore, leave a backorder quantity remaining. In the instance where Q reduces the backorder count to zero, Order Arrival stops the backorder timer prior to scheduling Update IP and terminates. Conversely, if Q does not reduce the backorder count to zero, then Order Arrival only schedules Update IP before terminating.

We now present three event graphs to demonstrate the operating logic of select inventory system configurations COMPRESS is capable of using to conduct DESs. In our first example (Figure 7), we have a deterministic demand configuration that IDS schedules according to Equation (7) with a "coin-flip" order supply system that has deterministic LTs. In our next example (Figure 8), we change the first configuration to one that includes Demand Cycle parametrically generating the number of demands that IDS schedules according to Equation (9). In our final example (Figure 9), we change the previous configuration to include an ordering method that uses a composite Q value (as the SPO solution files define), and a composite LT from Equation (10) that Order calculates. Although these examples incorporate small configuration changes, they speak to COMPRESS's versatility and ability to conduct DES under various conditions.

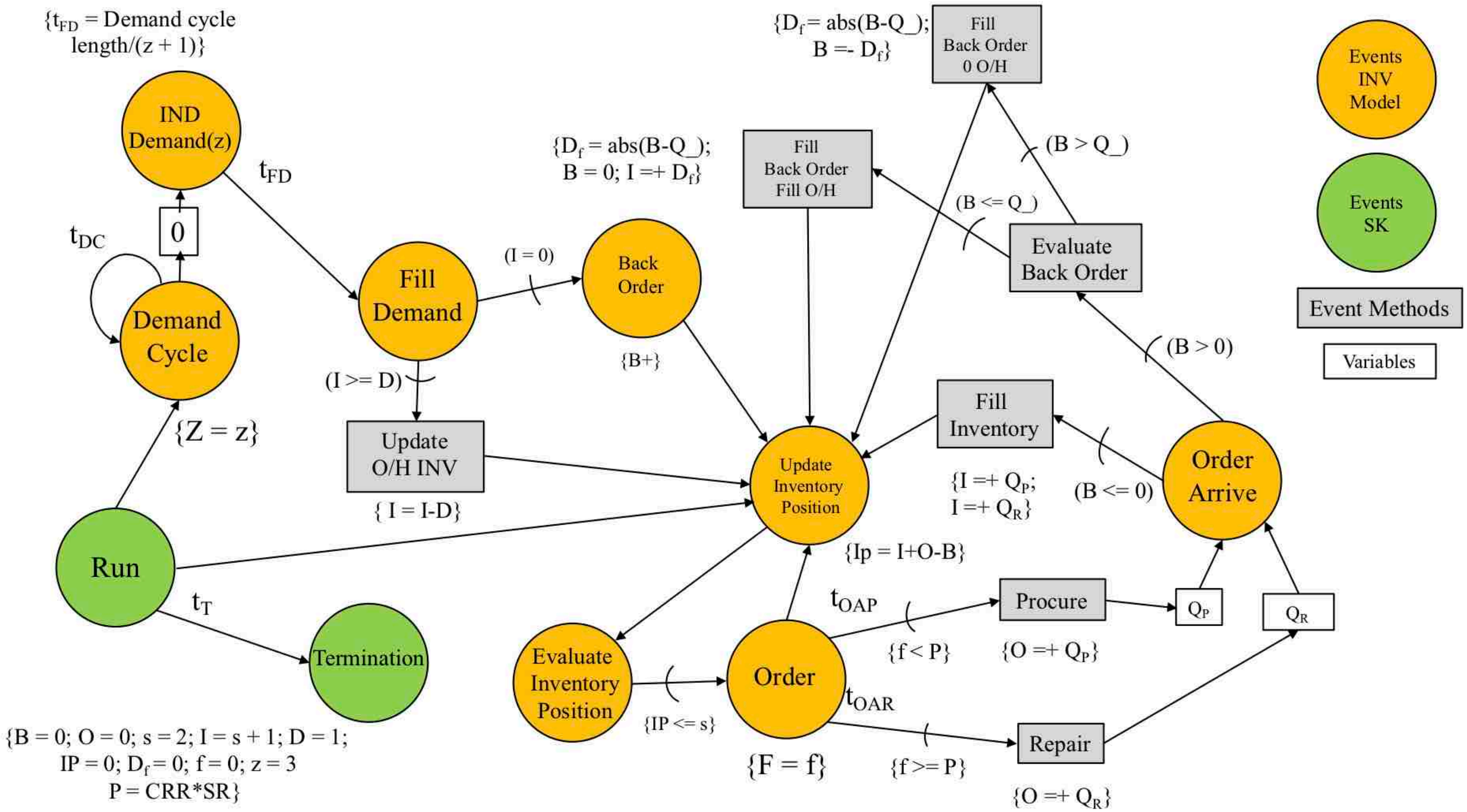


Figure 7. Event Graph for Inventory System with Deterministic Demands and Two Supply Streams

Figure 7 parameters and state variables are as follows:

- Parameters:
  - Q_P, Q_R: Order quantities
  - s: Reorder point
  - D: Demand quantity
  - z: Demands per cycle
  - t_{DC}: Demand cycle time delay
  - t_{OAP}: Procure order arrival delay
  - t_{OAR}: Repair order arrival delay
  - t_T: Termination time
  - t_{FD}: Fill demand delay
  - {f}: "Coin flip"; U(0, 1)
  - {D_f}: Difference between backorders and received order Qty
  - {P}: Ordering scheme probability
- State Variables:
  - I: On-hand inventory
  - O: Amount on order
  - B: Amount on backorder
  - IP: Inventory position

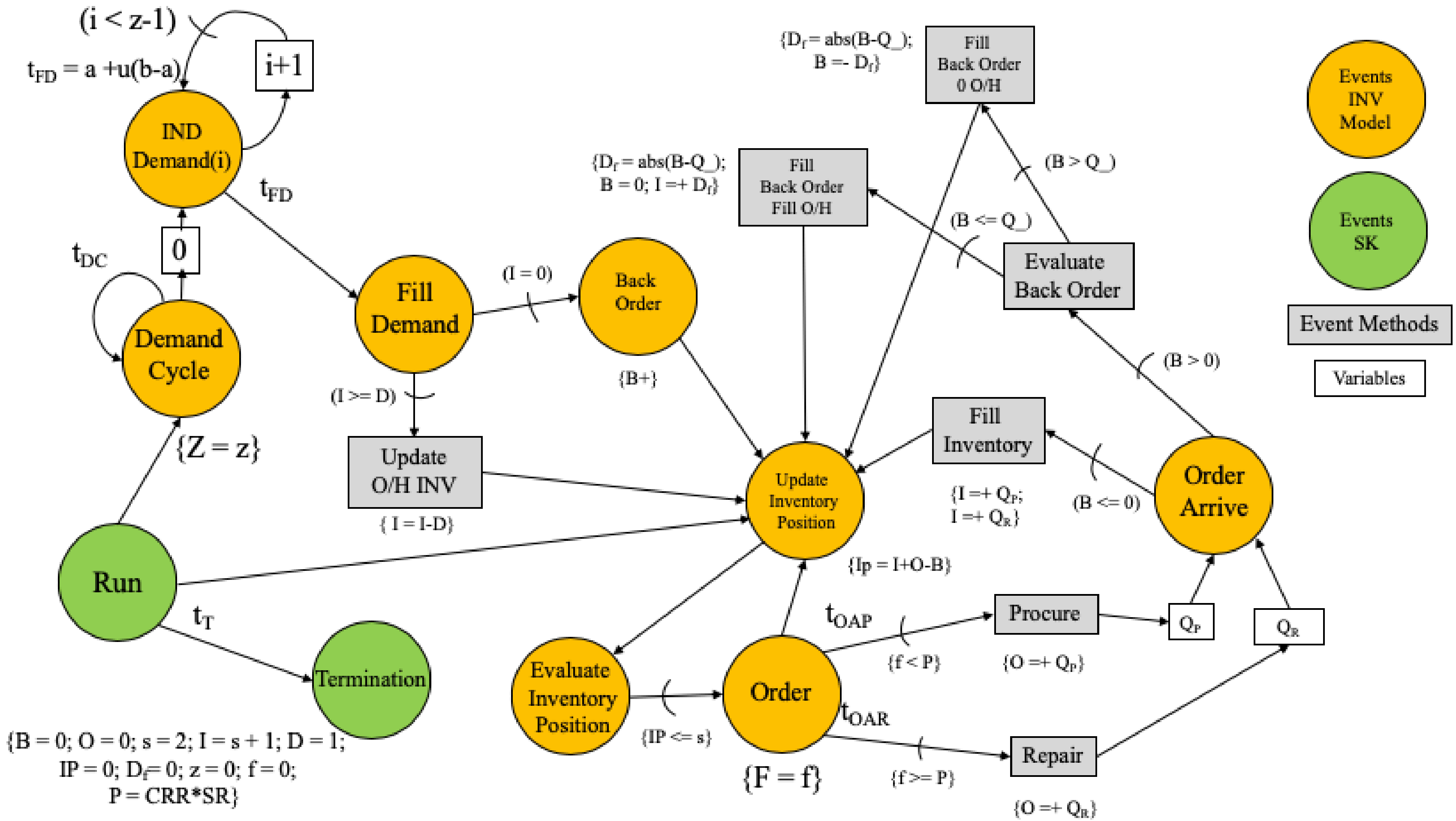


Figure 8. Event Graph for Inventory System with Variable Demands and Two Supply Streams

Figure 8 parameters and state variables are as follows:

- Parameters:
  - Q_P, Q_R: Order quantities
  - s: Reorder point
  - D: Demand quantity
  - t_{DC}: Demand cycle time delay
  - t_T: Termination time
  - $\{t_{FD}\}$: Fill demand delay
  - {f}: “Coin flip”; U(0, 1)
  - $\{D_f\}$: Difference between backorders and received order Qty
  - {P}: Ordering scheme probability
  - $\{t_{OAP}\}$: Procure order arrival delay
  - $\{t_{OAR}\}$: Repair order arrival delay
- State Variables:
  - I: On-hand inventory
  - O: Amount on order
  - B: Amount on backorder
  - IP: Inventory position
  - z: Demands per cycle (Based on Distribution)

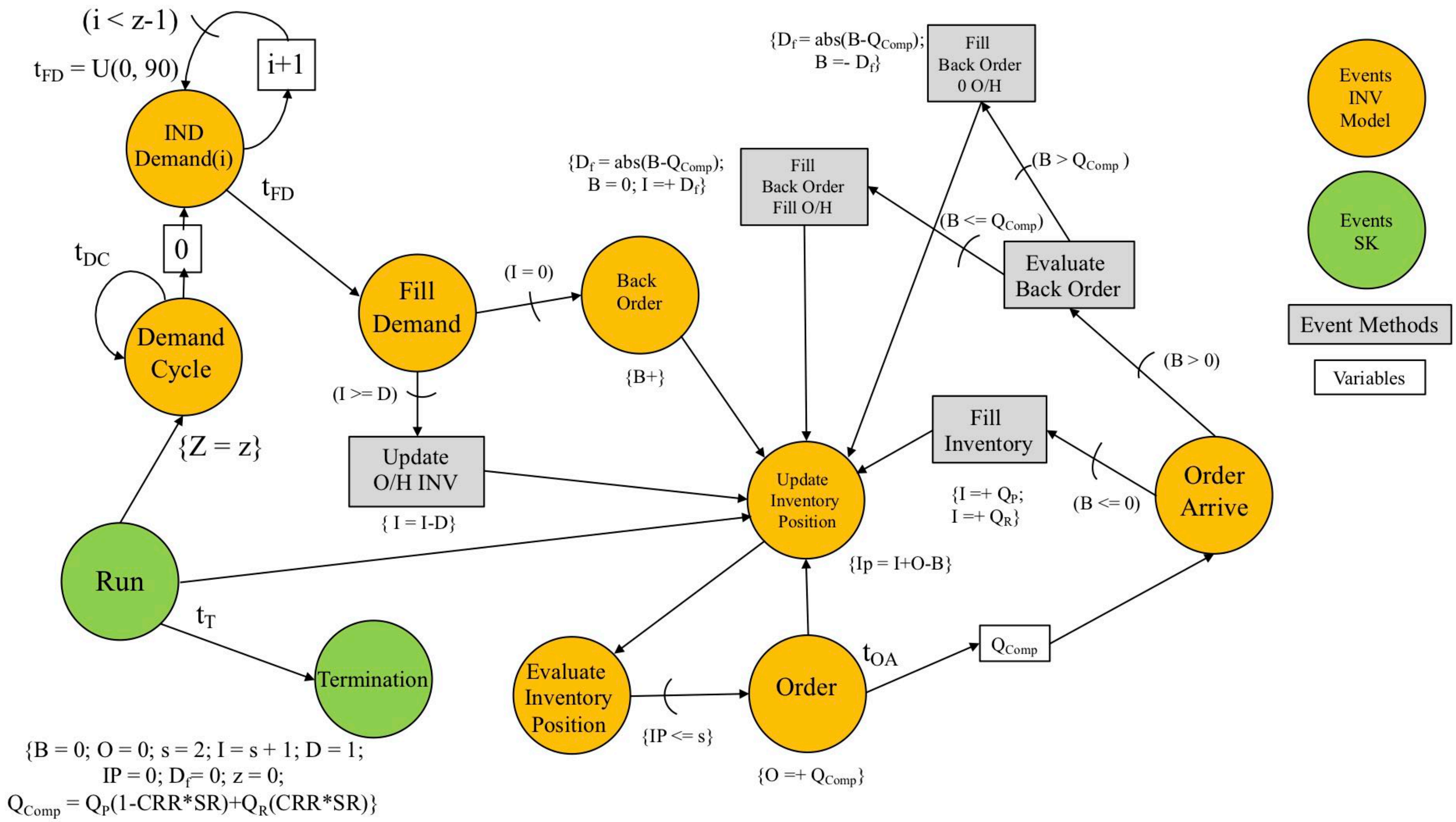


Figure 9. Event Graph for Inventory System with Variable Demands and a Composite Supply Streams

Figure 9 parameters and state variables are as follows:

- Parameters:
    - Q_{Comp}: Composite order quantities
    - s: Reorder point
    - D: Demand quantity
    - t_{DC}: Demand cycle time delay
    - t_T: Termination time
    - $\{t_{FD}\}$: Fill demand delay
    - $\{D_f\}$: Difference between backorders and received order Qty
    - $\{t_{OA}\}$: Order arrival delay
- State Variables:
    - I: On-hand inventory
    - O: Amount on order
    - B: Amount on backorder
    - IP: Inventory position
    - z: Demands per cycle (Based on Distribution)

E. COMPRESS OUTPUT

COMPRESS conveniently formats its output as comma-separated values for easy importation into any statistical software suite. This output (by item) includes the number of demands, full orders processed, partial orders processed, stockouts, backorders along with the total time the system is in a backordered state, and fill rate. COMPRESS calculates the fill rate for a given item i with the following formula:

$$\textit{\textbf{Fill Rate}}_i = 1 - \textit{\textbf{Number Stockouts}}_i \big/ \textit{\textbf{Number Demands}}_i\,, \tag{14}$$

where *Number Stockouts*$_i$ is the total number of stockouts of i and *Number Demands*$_i$ is the total number of demands for *i* during the DES.

III. ANALYSIS

A. EXPERIMENTAL DESIGN NOTATION

In our analysis, we use the notation in Table 6 to represent our experimental designs. Each experiment is coded with three digits indicating the demand type, LT, and supply stream type, respectively. For example, HD1 represents a design utilizing historical demands with a deterministic LT, and a composite supply stream.

Table 6. Experimental Design Notation

	First Digit	Second Digit	Third Digit
Design Element	Demand	LT	Supply Stream(s)
Choice	Historical (H); Parametric (P)	Deterministic (D); Variable (V)	Composite (1); Repair or Procure (2)

B. DISCRETE EVENT SIMULATIONS PER EXPERIMENTAL DESIGN

The reader should note that, in our analysis, we remove instances where the DES's parametrically generated demand quantity equals zero. Due to output design, COMPRESS automatically calculates a 100% fill rate for each of these instances, causing inflated values. Table 7 summarizes the resulting number of replications per each experimental design.

Table 7. Number of Replications per Experimental Design

Design	Number of Replications
HD1[a]	762
HD2[a]	762
HV1[b]	7,620
HV2[b]	7,620
PD1[b, c]	6,865
PD2[b, c]	6,861
PV1[b, c]	6,887
PV2[b, c]	6,895

[a] Due to deterministic demand and LT, we only perform one replication for each of the 762 NIINs.

[b] We perform ten replications per NIIN.

[c] Numbers are reduced from 7,620 due to zero generated demand quantities.

C. FILL RATE

We now present our simulated fill-rate analysis, and since the Department of Defense considers fill rate a primary metric (Department of Defense 2017), we will focus our efforts here while reminding the reader that further analysis can be completed on COMPRESS's other performance metrics if so desired. Tables 8 and 9 contain each experimental design's fill rate summary statistics across all NIINs for the number of replications in Table 7.

Table 8. COMPRESS Simulated Fill Rates with Historical Demands Summary

	Minimum Fill Rate	Maximum Fill Rate	Mean Fill Rate	Fill Rate Greater than or Equal to 85%[a]
HD1	0.00	1	0.7252	0.3858
HD2	0.00	1	0.7269	0.3832
HV1	0.00	1	0.6356	0.3391
HV2	0.00	1	0.6397	0.3478

[a] The fraction of simulated fill rates in the experimental design achieving at least 85%.

Table 9. COMPRESS Simulated Fill Rates with Parametric Demands Summary

	Minimum Fill Rate	Maximum Fill Rate	Mean Fill Rate	Fill Rate Greater than or Equal to 85%[a]
PD1	0.0025	1	0.7910	0.5286
PD2	0.0025	1	0.7850	0.5160
PV1	0.0020	1	0.6802	0.4054
PV2	0.0011	1	0.6818	0.4145

[a] The fraction of simulated fill rates in the experimental design achieving at least 85%.

Upon initial inspection of these simulated fill-rate values, a significant global difference does not appear to exist between inventory systems that use either a composite or repair-and-procure supply stream. Due to the limited insight this inspection provides towards our research objective, we choose to further our analysis by

- organizing our experimental designs into comparison groups (as Table 10 presents) that we link by NIIN
- indexing each experimental-design comparison group by NIIN
- creating composite and repair-and-procure supply system simulated fill-rate combinations under each NIIN subindex
- obtaining the difference between each aforementioned fill-rate combinations thus creating a new data set (*X*), sub-indexed by NIIN, for each experimental-design comparison group.

For a given item *i* with simulated fill-rates within a composite (*J*) and repair-and-procure (*K*) experimental-design comparison groups, we calculate the simulated fill-rate combination differences (x_{jk}^{i}) as follows:

$$x^{i}_{jk} = Fill\ Rate1^{i}_{j} - Fill\ Rate2^{i}_{k} \quad \forall i \in I, j \in J, k \in K, \tag{15}$$

where $Fill\ Rate1_j^i$ is an item's simulated fill rate while using a composite supply system, and $Fill\ Rate2_k^i$ is an item's simulated fill rate while using a repair-and-procure supply system. The reader should note that Equation (16) arranges each *I*'s x_{jk}^{i}values into a matrix which we term $\mathbf{X}^{i}$.

Table 10. Experimental-Design Comparison Groups

HD1-HD2
HV1-HV2
PD1-PD2
PV1-PV2

We start the next step in our analysis by examining the fraction of simulated fill-rate combination differences that are either zero, positive, or negative. As Table 11 summarizes, this sign comparison does not indicate the amount of dispersion within the data. It does, however, present the relative frequency of the combinations that have a zero difference, that is, instances where $Fill\ Rate1_j^i$ is equal to $Fill\ Rate2_k^i$. It is interesting to note that in the HV1-HV2 experimental-design comparison group, almost 50% of the simulated fill-rate combinations have a zero difference. Secondly, we notice that the positive and negative values appear to maintain an even distribution across each comparison group as the zero-difference fraction declines. Beyond these simple observations, we apply caution toward drawing any further conclusions with regards to these results. This is due to the fact that these results, like those in Tables 8 and 9, are still on a global scale. That is, they do not indicate if these patterns are consistent for each NIIN.

Table 11. Fraction of Zero, Positive, and Negative Simulated Fill-Rate Differences

	Zero	Positive[a]	Negative[b]
HD1 vs HD2	0.723	0.126	0.151
HV1 vs HV2	0.486	0.248	0.266
PD1 vs PD2	0.216	0.389	0.395
PV1 vs PV2	0.190	0.402	0.408

[a] Composite supply stream has a greater fill rate than repair-and-procure supply streams.

[b] Repair-and-procure supply streams have a greater fill rate than composite supply stream.

Now, we switch to a per-NIIN analysis and examine central tendency and deviation measurements that will provide better insights into NIINs whose fill rates using a composite supply stream differ from those using separate, repair-and-procure streams (we exclude experimental design groups HD1 and HD2 from the analysis beyond this point). We first begin by determining the absolute simulated fill-rate combination differences (y_{jk}^i) for a given item *i* within a comparison group as follows

$$\boldsymbol{y}_{jk}^{i} = \left|\boldsymbol{x}_{jk}^{i}\right| \quad \forall \boldsymbol{i} \in \boldsymbol{I}. \tag{16}$$

These y_{jk}^i values from Equation (17) are also arranged in a matrix, which we term $\mathbf{Y}^i$. Second, we compute the mean simulated fill-rate combination difference ($\overline{x^i}$) for a given item *i* within a comparison group with the following formula:

$$\overline{x^i} = \frac{1}{nm}\sum_{j=1}^{n}\sum_{k=1}^{m} x^i_{jk} \quad \forall i \in I, \tag{17}$$

where n and m are the number of rows and columns of $\mathbf{X}^i$, that is , $n = |J|$ and $m = |K|$. Last, we calculate the mean absolute simulated fill-rate combination differences ($\overline{y^i}$) for a given item i within a comparison group as:

$$\overline{y^i} = \frac{1}{nm}\sum_{j=1}^{n}\sum_{k=1}^{m} y^i_{jk} \quad \forall i \in I. \tag{18}$$

Tables 12 and 13 contain the summary statistics across all $\overline{x^i}$s and $\overline{y^i}$s, respectively.

Table 12. Mean Simulated Fill-Rate Combination Differences Summary

	Minimum	Maximum	Mean	Standard Deviation
HV1 vs HV2	-0.300	0.259	-0.004	0.059
PD1 vs PD2	-0.333	0.545	-0.003	0.080
PV1 vs PV2	-0.333	0.444	-0.002	0.106

Table 13. Mean Absolute Simulated Fill-Rate Combination Differences Summary

	Minimum	Maximum	Mean	Standard Deviation
HV1 vs HV2	0.00	0.454	0.098	0.083
PD1 vs PD2	0.00	0.545	0.167	0.078
PV1 vs PV2	0.00	0.454	0.230	0.096

Tables 14 and 15, respectively, display the fractions of the $\overline{x^i}$s and $\overline{y^i}$s falling within each tables' stated ranges. We find it interesting that, irrespective of design comparison group, at least 68% of the $\mathbf{X}^i$s' mean simulated fill-rate combination differences are between -0.1 and 0.1 while at least 73% of the $\mathbf{Y}^i$s' mean absolute simulated fill-rate combination differences are between 0 and 0.3.

Table 14. Fractions of $\overline{x^i}$s within Stated Mean Simulated Fill-Rate Combination Difference Ranges

	[-0.3, -0.2)	[-0.2, -0.1)	[-0.1, 0)	[0][a]	(0, 0.1]	(0.1, 0.2]	(0.2, 0.3]
HV1 vs HV2	0.0157	0.0289	0.3372	0.2887	0.3005	0.0262	0.0026
PD1 vs PD2	0.0160	0.0931	0.3869	0.0519	0.3604	0.0638	0.0173
PV1 vs PV2	0.0413	0.1171	0.2996	0.0386	0.3422	0.1211	0.0280

[a] Does not discern between all differences being "truly" 0, or the summation of differences equaling 0.

Table 15. Fractions of $\overline{y^i}$s within Stated Mean Absolute Simulated Fill-Rate Combination Difference Ranges

	[0]	(0, 0.1]	(0.1, 0.2]	(0.2, 0.3]	(0.3, 0.4]	(0.4, 0.5]	(0.5, 0.6]
HV1 vs HV2	0.2480	0.2795	0.3570	0.1037	0.0105	0.0013	0.0000
PD1 vs PD2	0.0465	0.1809	0.4215	0.3032	0.0452	0.0013	0.0013
PV1 vs PV2	0.0333	0.1011	0.2437	0.3675	0.2343	0.0200	0.0000

To complement the central tendency measurements, we now quantify the extent of deviation found within each $\mathbf{X}^i$ and $\mathbf{Y}^i$. Specifically, we wish to gain insight into how close $\mathbf{X}^i$'s and $\mathbf{Y}^i$'s values are collectively to their respective mean for a given item i. We choose standard deviation (σ) to quantify this measurement. That is, for a given item i within a comparison group we calculate $\mathbf{X}^i$'s standard deviation (σ_{x^i}) as:

$$\sigma_{x^i} = \sqrt{\frac{1}{nm}\sum_{j=1}^{n}\sum_{k=1}^{m}\left(x_{jk}^i - \overline{x}^i\right)^2} \quad \forall i \in I. \tag{19}$$

Similarly, we calculate $\mathbf{Y}^i$'s standard deviation (σ_{y^i}) as:

$$\sigma_{y^i} = \sqrt{\frac{1}{nm}\sum_{j=1}^{n}\sum_{k=1}^{m}\left(y_{jk}^i - \overline{y}^i\right)^2} \quad \forall i \in I. \tag{20}$$

Tables 16 and 17, respectively, contain the fractions of σ_{x^i}s and σ_{y^i}s falling within the tables' stated ranges. We notice from the information in these tables that how we derive the demands appears to have a more profound effect on dispersion than when we randomly generate LTs. Furthermore, as we expect, when we randomly generate both demand and LT, we find higher fractions of σ_{x^i}s and σ_{y^i}s within the table's upper standard deviation ranges. Beyond these observations, we believe we will obtain a more concise analysis through studying each $\overline{x^i}$ and $\overline{y^i}$ in conjunction with their respective observed σ_{x^i} and σ_{y^i}. We present this analysis in a per-comparison group format.

Table 16. Fractions of σ_{x^i}s within Stated Standard Deviation Ranges

	[0, 0.1]	(0.1, 0.2]	(0.2, 0.3]	(0.3, 0.4]	(0.4, 0.5]
HV1 vs HV2	0.3924	0.3752	0.1850	0.0459	0.0013
PD1 vs PD2	0.1360	0.3333	0.3507	0.1667	0.0013
PV1 vs PV2	0.0772	0.1345	0.3355	0.3328	0.1119

Table 17. Fractions of σ_{y^i}s within Stated Standard Deviation Ranges

	[0, 0.1]	(0.1, 0.2]	(0.2, 0.3]	(0.3, 0.4]	(0.4, 0.5]
HV1 vs HV2	0.5394	0.4133	0.0472	0.0000	0.0000
PD1 vs PD2	0.2373	0.5600	0.1867	0.0160	0.0000
PV1 vs PV2	0.1171	0.3749	0.4621	0.0413	0.0000

We normalize our ranges to the minimum and maximum observed mean and standard deviation values, split each resulting range into quarters, and combine the quarters further creating sixteen different summary groups. Table 18 contains the $\overline{x^i}$ vs. σ_{x^i} fractions within comparison group HV1 vs. HV2, falling within the stated ranges while Figure 10 contains this information's graphical presentation. The reader should note that, in the following mean versus standard deviation figures, each dot depicts the relationship between mean and standard deviation across either simulations $\mathbf{X}^i$ or $\mathbf{Y}^i$ for a given item i. From Table 18 and Figure 10 we notice that at least 65% of the $\overline{x^i}$ vs. σ_{x^i}s fall within the lower 50% of the standard deviation range with an average mean that is between -0.02 and 0.12.

Table 18. Comparison Group HV1 vs. HV2's Fraction of $\overline{x^i}$ vs. σ_{x^i}s within Stated Ranges

		Standard Deviation			
		[0, 0.12]	(0.12, 0.24]	(0.24, 0.35]	(0.35, 0.47]
Mean Simulated Fill-Rate Difference	[-0.3, -0.16]	0.0000	0.0092	0.0131	0.0000
	(-0.16, -0.02]	0.0564	0.1404	0.0381	0.0039
	(-0.02, 0.12]	0.4121	0.2415	0.0577	0.0066
	(0.12, 0.26]	0.0000	0.0014	0.0066	0.0000

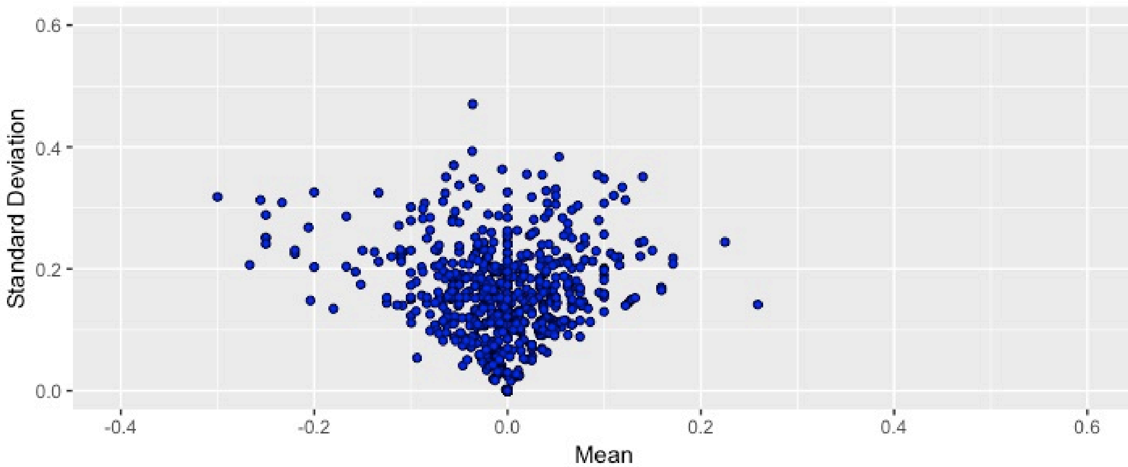


Figure 10. $\overline{x^i}$ vs. σ_{x^i}s within Comparison Group HV1 vs. HV2

Table 19 contains the fractions of $\overline{x^i}$ vs. σ_{x^i}s within comparison group PD1 vs. PD2 falling within the stated ranges while Figure 11 contains this information's graphical presentation. From Table 19 and Figure 11 we notice that at least 78% of the $\overline{x^i}$ vs. σ_{x^i}s fall within the (-0.11, 0.11] mean simulated fill-rate difference range while the addition of parametric demands seems to generate more dispersion within the data. That is, 26% of the $\overline{x^i}$ vs. σ_{x^i}s have a standard deviation within the (0.23, 0.34] range.

Table 19. Comparison Group PD1 vs. PD2's Fraction of $\overline{x^i}$ vs. σ_{x^i}s within Stated Ranges

		Standard Deviation			
		[0, 0.11]	(0.11, 0.23]	(0.23, 0.34]	(0.34, 0.46]
Mean Simulated Fill-Rate Difference	[-0.33, -0.11]	0.0000	0.0160	0.0573	0.0173
	(-0.11, 0.11]	0.1613	0.3707	0.2560	0.0440
	(0.11, 0.33]	0.0027	0.0240	0.0267	0.0160
	(0.33, 0.55]	0.0000	0.0013	0.0027	0.0040

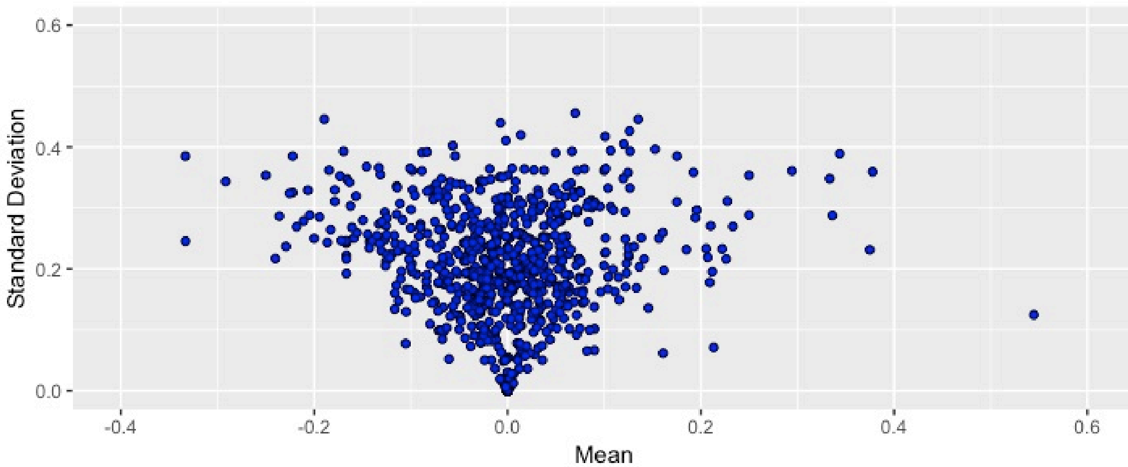


Figure 11. $\overline{x^i}$ vs. σ_{x^i}s within Comparison Group PD1 vs. PD2

Table 20 contains the fraction of $\overline{x^i}$ vs. σ_{x^i}s within comparison group PV1 vs. PV2 falling within the table's stated ranges while Figure 12 contains this information's graphical presentation. From Table 20 and Figure 12 we notice that only 36.4% of the $\overline{x^i}$ vs. σ_{x^i}s with a mean simulated fill-rate difference between -0.14 and 0.25 fall within the lower 50% of the standard deviation range. This finding is a stark contrast to the previous two comparison groups and can potentially signify an increase in the response variability between our two inventory systems as input variability increases. Only after increasing our encompassing standard deviation range by 25% do we witness about the same proportionality of the previous two comparison groups with 41% of the $\overline{x^i}$ vs. σ_{x^i}s falling into the (-0.14, 0.25] mean simulated fill-rate difference and (0.26, 0.39] standard deviation ranges.

Table 20. Comparison Group PV1 vs. PV2's Fraction of $\overline{x^i}$ vs. σ_{x^i}s within Stated Ranges

		Standard Deviation			
		[0, 0.13]	(0.13, 0.26]	(0.26, 0.39]	(0.39, 0.52]
Mean Simulated Fill-Rate Difference	[-.33, -0.14]	0.0000	0.0159	0.0426	0.0253
	(-0.14, 0.06]	0.0959	0.1997	0.2716	0.0786
	(0.06, 0.25]	0.0053	0.0626	0.1425	0.0426
	(0.25, 0.44]	0.0013	0.0013	0.0080	0.0040

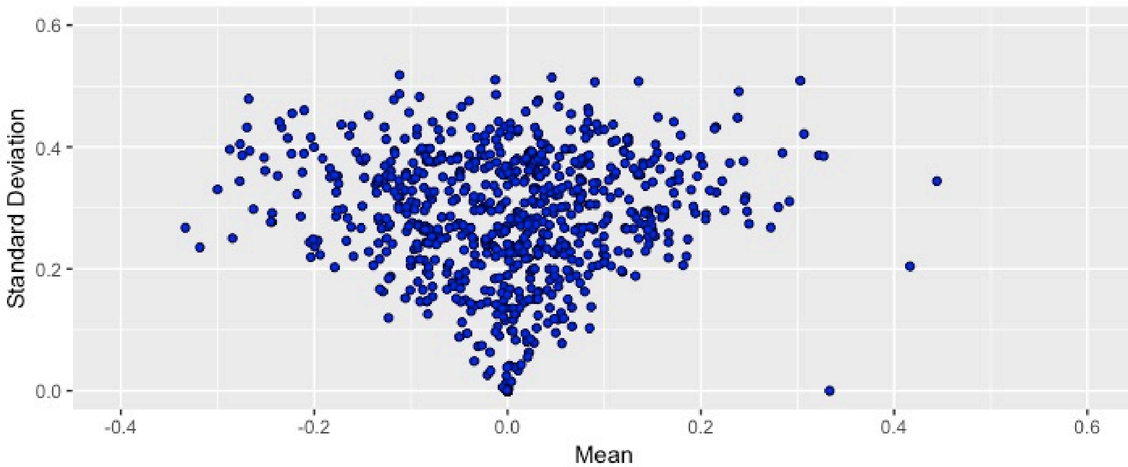


Figure 12. $\overline{x^i}$ vs. σ_{x^i}s within Comparison Group PV1 vs. PV2

Figure 13 provides an overlay of Figures 10, 11, and 12 to aid in comparing the three comparison groups.

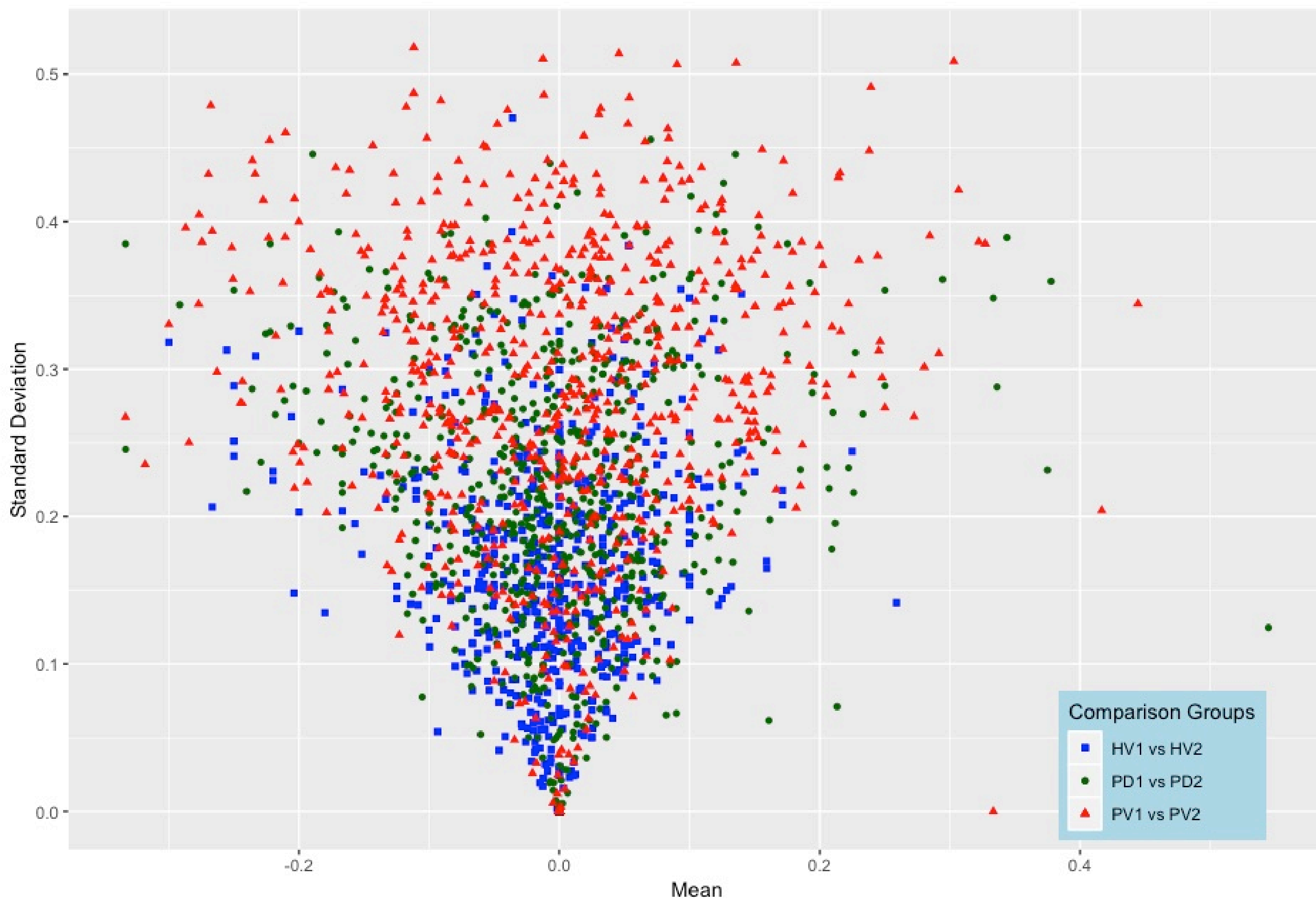


Figure 13. $\overline{x^i}$ vs. σ_{x^i}s Overlay

From Table 21 and Figure 14 we notice at least 74% of the $\overline{y^i}$ vs. σ_{y^i}s fall within the lower 50% of the ranges. That is 74% of the $\overline{y^i}$ vs. σ_{y^i}s have a mean absolute simulated fill-rate difference less than or equal to 0.23 with a standard deviation less than or equal to 0.13.

Table 21. Comparison Group HV1 vs. HV2's Fraction of $\overline{y^i}$ vs. σ_{y^i}s within Stated Ranges

		Standard Deviation			
		[0, 0.07]	(0.07, 0.13]	(0.13, 0.19]	(0.19, 0.26]
Mean Simulated Fill-Rate Difference	[0, 0.11]	0.3635	0.1942	0.0210	0.0026
	(0.11, 0.23]	0.0000	0.1850	0.1444	0.0118
	(0.23, 0.34]	0.0000	0.0013	0.0236	0.0105
	(0.34, 0.45]	0.0000	0.0000	0.0118	0.0276

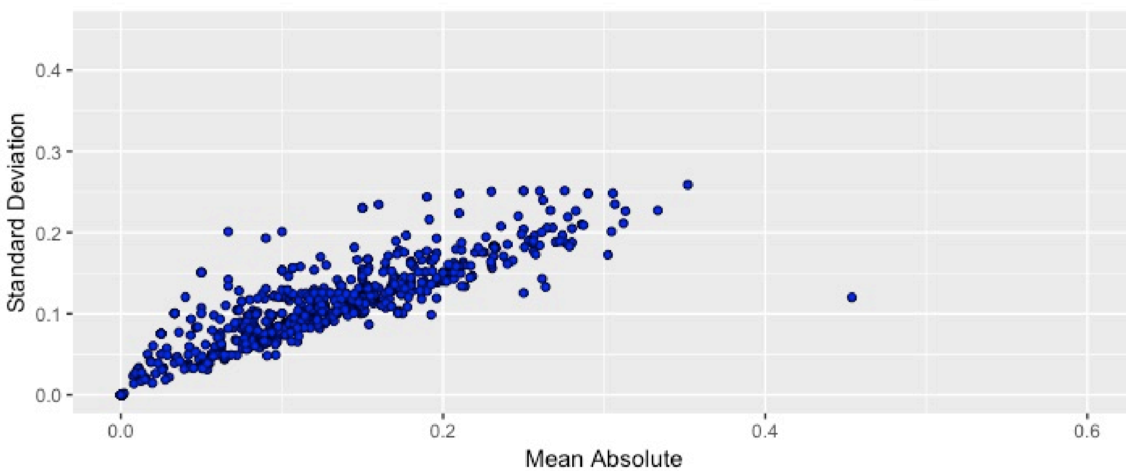


Figure 14. $\overline{y^i}$ vs. σ_{y^i}s within Comparison Group HV1 vs. HV2

Table 22 contains the fractional values of $\overline{y^i}$ vs. σ_{y^i}s within comparison group PD1 vs. PD2 falling within the stated ranges while Figure 15 contains this information's graphical presentation. From Table 22 and Figure 15 we notice at least 75% of the $\overline{y^i}$ vs. σ_{y^i}s fall within the lower 50% of the ranges. That is 75% of the $\overline{y^i}$ vs. σ_{y^i}s have a mean absolute simulated fill-rate difference less than or equal to 0.27 with a standard deviation less than or equal to 0.19. Proportionally, these results are similar to those found in the HV1 vs. HV2 comparison group analysis.

Table 22. Comparison Group PD1 vs. PD2's Fraction of $\overline{y^i}$ vs. σ_{y^i}s within Stated Ranges

		Standard Deviation			
		[0, 0.10]	(0.10, 0.19]	(0.19, 0.30]	(0.30, 0.39]
Mean	[0, 0.14]	0.2147	0.1267	0.0160	0.0000
Simulated	(0.14, 0.27]	0.0107	0.4133	0.1200	0.0080
Fill-Rate	(0.27, 0.41]	0.0000	0.0053	0.0706	0.0120
Difference	(0.41, 0.55]	0.0000	0.0013	0.0013	0.0000

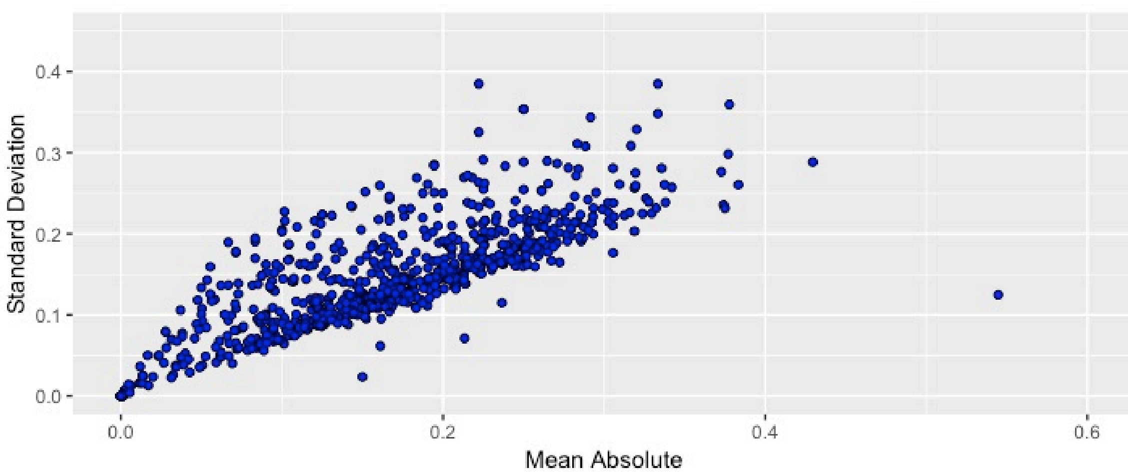


Figure 15. $\overline{y^i}$ vs. σ_{y^i}s within Comparison Group PD1 vs. PD2

Table 23 contains the fractional values of $\overline{y^i}$ vs. σ_{y^i}s within comparison group PV1 vs. PV2 falling within the stated ranges while Figure 16 contains this information's graphical presentation. From Table 23 and Figure 16 we notice only 35.4% of the $\overline{y^i}$ vs. σ_{y^i}s fall within the lower 50% of the ranges. This finding, like those for the $\overline{x^i}$ vs. σ_{x^i}s, is a stark contrast to the previous two comparison groups. Again, only after increasing our encompassing ranges by 25%, do we witness about the same proportionality of the previous two comparison groups with 31% of the $\overline{y^i}$ vs. σ_{y^i}s falling into the (0.23, 0.34] mean absolute simulated fill-rate difference and (0.19, 0.29] standard deviation ranges.

Table 23. Comparison Group PV1 vs. PV2's Fraction of $\overline{y^i}$ vs. σ_{y^i}s within Stated Ranges

		Standard Deviation			
		[0, 0.10]	(0.10, 0.19]	(0.19, 0.29]	(0.29, 0.38]
Mean Simulated Fill-Rate Difference	[0, 0.11]	0.0932	0.0479	0.0173	0.0000
	(0.11, 0.23]	0.0159	0.1970	0.0932	0.0066
	(0.23, 0.34]	0.0013	0.0812	0.3089	0.0266
	(0.34, 0.45]	0.0000	0.0000	0.0812	0.0293

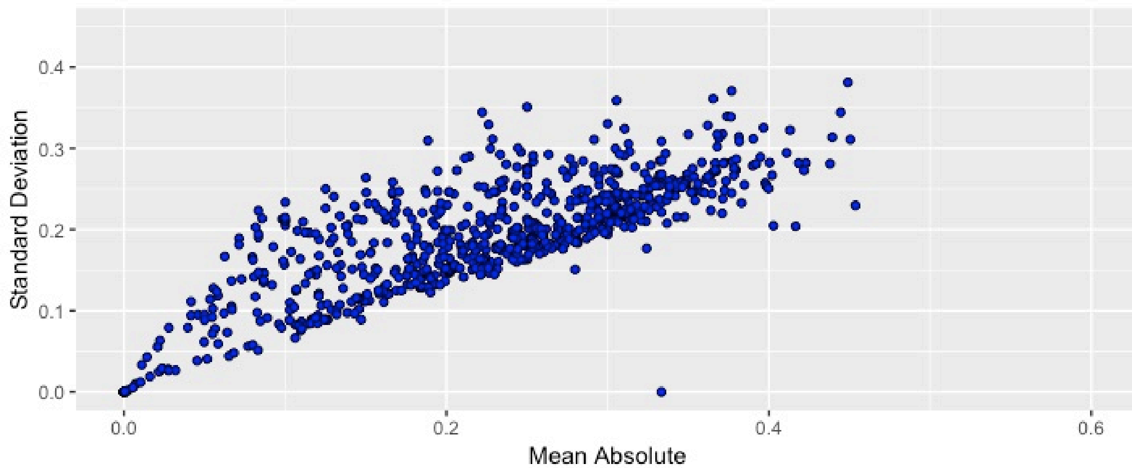


Figure 16. $\overline{y^i}$ vs. σ_{y^i}s within Comparison Group PV1 vs. PV2

As before, Figure 17 provides an overlay of Figures 14, 15, and 16.

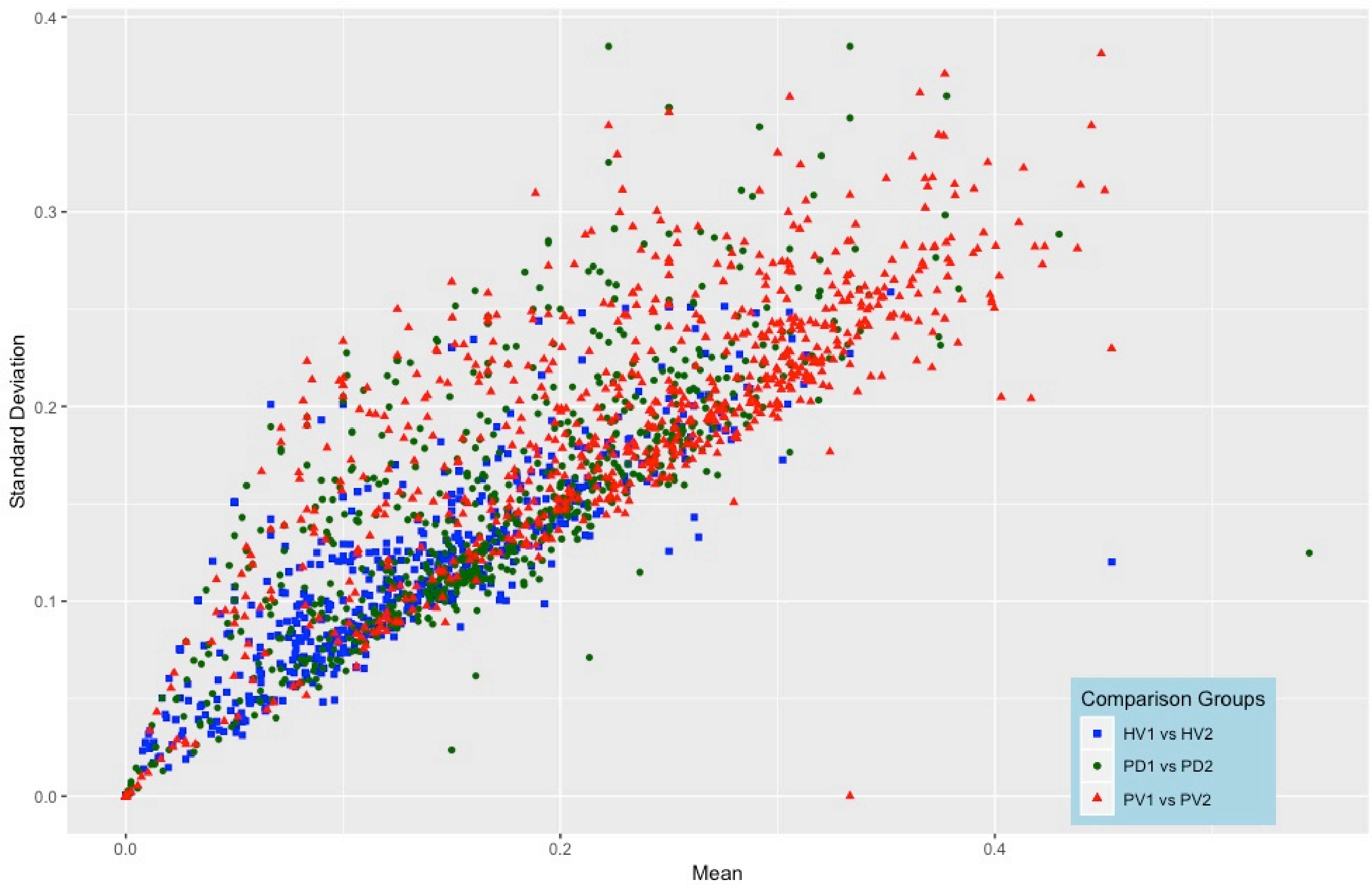
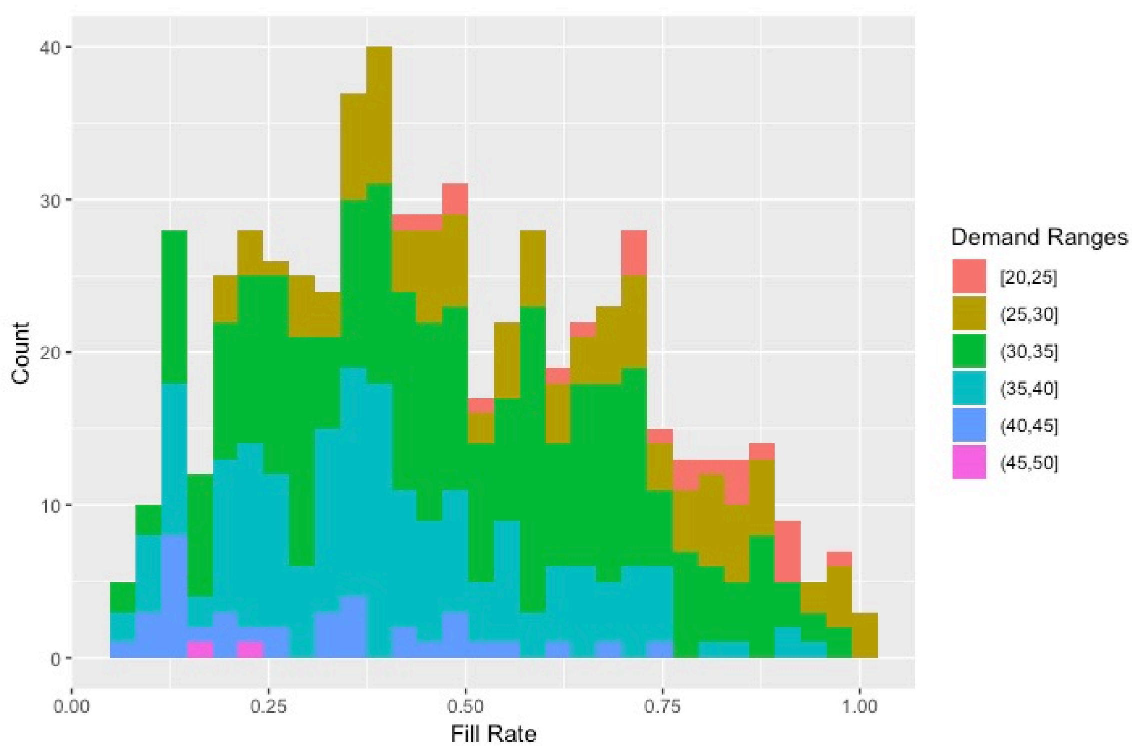


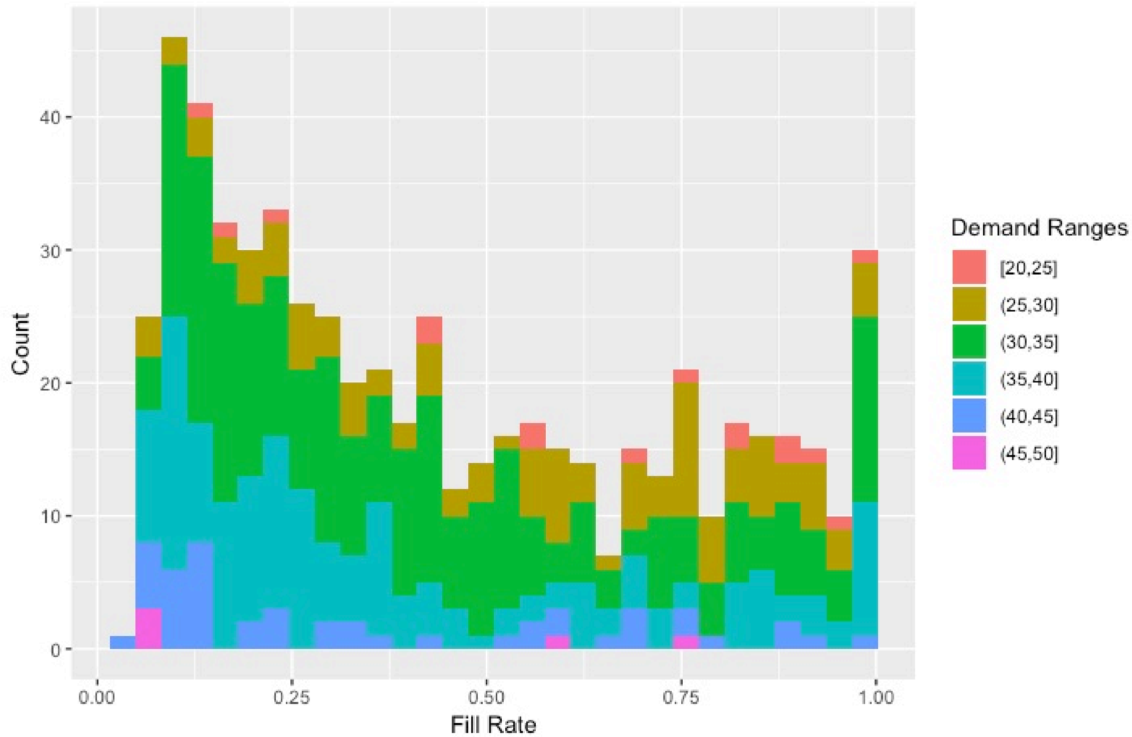
Figure 17. $\overline{y^i}$ vs. σ_{y^i}s Overlay

Finally, we choose to perform a formal statistical test on the fill-rate data COMPRESS generates for a select NIIN. This test can, and should, be repeated for each NIIN within this study, but we leave this as a topic for follow-on research. First, we select NIIN 01–024-7849 as our case item. Under the PV1 vs PV2 experimental-design comparison group, 01–024-7849's simulated fill-rate combination differences possess a 3 to 2 $Fill\ Rate1_j$ to $Fill\ Rate2_k$ ratio (that is, the composite supply stream is more often able to attain higher fill-rates than the repair-and-procure supply streams), mean simulated fill-rate difference is 0.09243 with a 0.43754 standard deviation, and mean absolute simulated fill-rate difference is 0.33589 with a 0.29344 standard deviation. Second, with an average 489.02 second (8.15 minute) run time, we perform 600 replications (n_1 and n_2) for both experimental designs PV1 and PV2 producing $Fill\ Rate1_a$ and $Fill\ Rate2_b$ within sets A and B, respectively. Figures 18 and 19 display the resulting fill-rate data.



Generated demand information is included only to show demand frequency is proportionally similar for replications between experimental designs PV1 and PV2.

Figure 18. $Fill\ Rate1_a$ Count Data for 01–024-7849 Utilizing Experimental-Design PV1



Generated demand information is included only to show demand frequency is proportionally similar for replications between experimental designs PV1 and PV2.

Figure 19. *Fill Rate*2_b Count Data for 01–024-7849 Utilizing Experimental-Design PV2

Figure 20 provides an overlay of the fill-rate data within Figures 18 and 19 to aid in comparing the fill rates COMPRESS generates for experimental-designs PV1 and PV2.

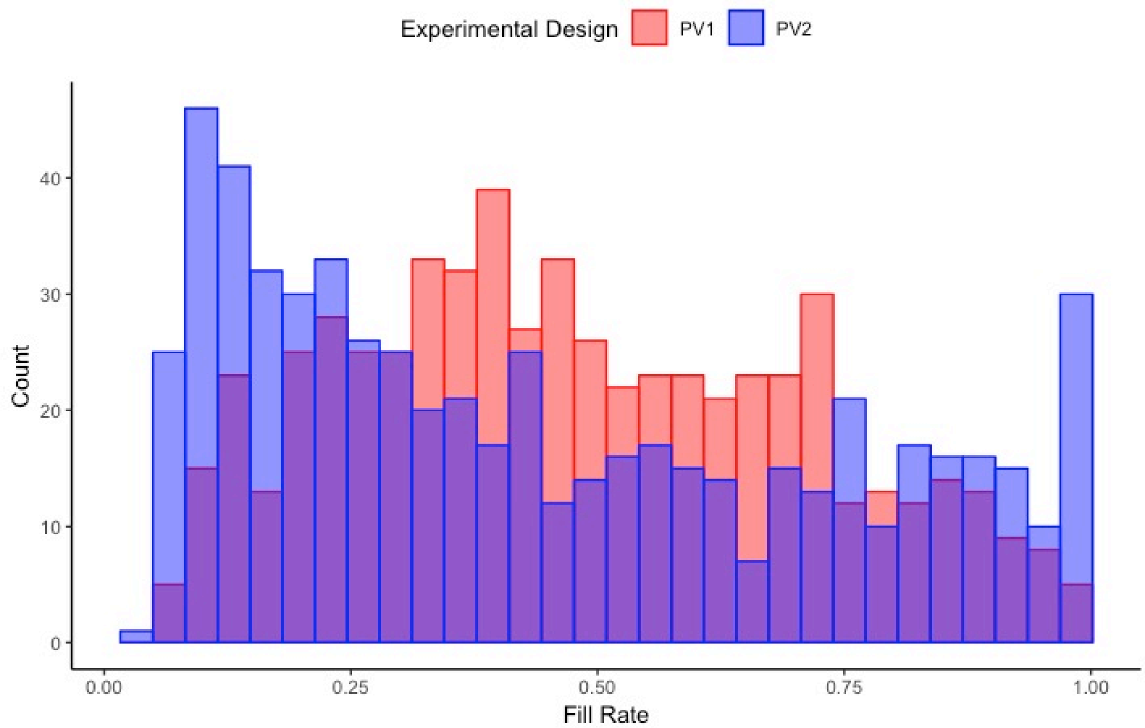


Figure 20. Fill-Rate Data COMPRESS Generates for Experimental-Designs PV1 and PV2 Overlay

Last, we choose the two-tailed version of the Wilcoxon ranked-sum test to determine if the fill-rate observations within sets A and B are from the same distribution. Under the assumptions of this test, our null and alternate hypothesis are as follows:

$$H_0 : \mathrm{P}\left(\mathit{Fill\ Rate}1_a > \mathit{Fill\ Rate}2_b\right) = \mathrm{P}\left(\mathit{Fill\ Rate}2_b > \mathit{Fill\ Rate}1_a\right), \text{ and}$$

$$H_a : \mathrm{P}\left(\mathit{Fill\ Rate}1_a > \mathit{Fill\ Rate}2_b\right) \neq \mathrm{P}\left(\mathit{Fill\ Rate}2_b > \mathit{Fill\ Rate}1_a\right), \text{ respectively.}$$

The median fill-rate value for set A is 0.4516 while set B's value is 0.3767. At the 0.05 significance level, the distributions within sets A and B differ significantly (Wilcoxon test statistic (W) = 200220, $n_1 = n_2 = 600$, p-value = 0.00753), and we reject our null hypothesis in favor of the alternative.

IV. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

This thesis develops a new DES model, COMPRESS, in order generate key inventory performance metrics under various operating configurations. Our research objective is to determine if there is a discernable difference in the fill rates between weighted composite and dual-stream supply systems under various operating conditions. In carrying out this study, we develop four experimental-design comparison groups.

We first note that a high fraction of the mean simulated fill-rate differences ($\overline{x^i}$s) and absolute differences ($\overline{y^i}$s) exhibit a considerable amount of "difference" regardless of design comparison group. While the HV1 vs. HV2 comparison group has the highest $\overline{x^i}$ and $\overline{y^i}$ fractions equaling zero among all the comparison groups, these fractions of $\overline{x^i}$s and $\overline{y^i}$s are only 29% and 25%, respectively. As we previously note in Table 14, the fractions where $\overline{x^i}$ equals zero does not discern between all differences being "truly" zero, or the summation of the differences equaling zero. The fraction of $\overline{y^i}$s, however, is a better indicator of this, and leads us to conclude that under conditions of deterministic demand and variable lead-times, 75% of the NIINs have some level of measurable difference between the fill rates obtained from inventory systems using either a composite or repair-and-procure stream(s). The PD1 vs. PD2 comparison group has the next highest zero $\overline{x^i}$ and $\overline{y^i}$ fractions at 5% each while the PV1 vs. PV2 comparison group's zero $\overline{x^i}$ and $\overline{y^i}$ fractions are 4% and 3%, respectively. This trend indicates that the differences between the fill rates increases as two inventory systems' variability increases.

Second, we use standard deviation to note the amount of dispersion within each $\mathbf{X}^i$ and $\mathbf{Y}^i$. Coincidentally, we notice a similar increasing trend in dispersion levels as the variability increases within the inventory system as previously seen in the $\overline{x^i}$ and $\overline{y^i}$ analysis. While the HV1 vs. HV2 comparison group exhibits the least amount of dispersion among the three groups, 46% of the σ_{y^i} values and 61% of the σ_{x^i} values are greater than 0.1. For this same standard deviation range, the fractions are 76%, 86%, 92%, and 88% for the PD1 vs. PD2 and PV1 vs. PV2 comparison groups, respectively. Even without taking our means into consideration, this indicates to us a considerable amount of variability within the $\mathbf{X}^i$s and $\mathbf{Y}^i$s. We would expect the opposite to be true if the fill rates from the two inventory systems were the same.

Third, we further study the means in conjunction with the respective standard deviations. Taking the PV1 vs. PV2 comparison group (parametric demand with variable LT and either composite or repair-procure supply stream, respectively) as the extreme of our study, we treat it as the limiting factor and a basis to draw our conclusions. The increased proportions of $\overline{x^i}$ vs. σ_{x^i}s and $\overline{y^i}$ vs. σ_{y^i}s at higher ranges further indicate considerable variability in our data. Furthermore, with the 31% fraction of $\overline{y^i}$ vs. σ_{y^i}s as an example, there exists the potential of a $\mathbf{Y}^i$ that possesses a mean absolute simulated fill-rate difference of 0.34 with a resulting standard deviation range of 0.19 to 0.29.

Finally, we use the Wilcoxon ranked-sum test to perform a formal statistical analysis. Our null hypothesis states that there is an equal probability of a randomly selected observation from set *A* being greater than a randomly selected observation from set *B* as a randomly selected observation from set *B* being greater than a randomly selected observation from set *A*. At the 0.05 significance level, the resulting p-value is 0.00753, leading us to reject the null hypothesis in favor of the alternative.

B. RECOMMENDATIONS

There are numerous examples of how to continue or improve upon this research. Our first recommendation is to re-perform our study's research objective with a focus on items that possess lower *SR* and *CRR* values that do not lend themselves towards the repair cycle. Second, incorporate variable repair quantities. Third, select a smaller set of NIINs, increase the number of replications per NIIN, and perform a formal unpaired statistical test on the resulting fill-rate sets. Last, we have created a process that operates in a cyclic system where information blocks are loaded into a simulation and executed prior to moving onto the next block. It will not take much effort to convert this process into a system that randomly draws with replacement blocks to execute them. This could potentially increase the variability and each simulation run's duration.

LIST OF REFERENCES

Bachman TC, Williams PJ, Cheman KM, Curtis J, Carroll R (2016) PNG: Effective inventory control for items with highly variable demand. *INFORMS*, https://doi-org.libproxy.nps.edu/10.1287/inte.2015.0829.

Buss AH (1996) Modeling with event graphs. *Proceedings of the 1996 Winter Simulation Conference* (Coronado, CA), http://doi.ieeecomputersociety.org.libproxy.nps.edu/10.1109/WSC.1996.873273.

Caliguire TA (2009) Modeling the AIM-9 Sidewinder repair line through discrete event simulation. Master's thesis, Operations Research Department, Naval Postgraduate School, Monterey. http://hdl.handle.net/10945/4763.

Chief of Naval Operations (2017) Depot level repairable item management. OPNAV Instruction 4400.9D. Washington, DC, https://doni.documentservices.dla.mil/Directives/04000%20Logistical%20Support%20and%20Services/04-400%20Supply%20and%20Material%20Services/4400.9D.pdf.

Craparo E (2018) SPO data set provided to the author via email, April 2.

Department of Defense (2017) DoD Supply Chain material Management procedures: supply chain inventory reporting and metrics. DoD Manual 4140.01-vol. 10. Washington, DC, http://www.esd.whs.mil/Portals/54/Documents/DD/issuances/414001m/414001m_vol10.pdf?ver=2017-11-28-104533-080.

Ellis D (2017) Data files form NAVSUP WSS provided to Sean Teter via personal communication, October 18–19.

Ellis D (2018a) Presentation on DRP Functionality provided to Javier Salmerón and Emily Craparo via personal communication, January 18.

Ellis D (2018b) Information regarding DRP's repair recommendation process provided to the author via personal communication, July 18.

Hadley G, Whitin TM (1963) *Analysis of Inventory Systems* (Prentice-Hall, Englewood Cliffs, NJ).

Law AM (2007) *Simulation Modeling and Analysis*, 4th ed. (McGraw-Hill, New York, NY).

Naval Supply Systems Command (2015) Operational forces supply procedures. NAVSUP publication 485-vol 1. Mechanicsburg, PA,http://navybmr.com/study%20material/NAVSUP%20P485%20(REV%205)%20Vol%20I.pdf.

Naval Supply Systems Command Weapon Systems Support (2018) Navy Supply Chain Management. Accessed August 23, 2018, https://www.navsup.navy.mil/public/navsup/wss/nscm/.

Roth GF (2016) A simulation of alternatives for wholesale inventory replenishment. Master's thesis, Operations Research Department, Naval Postgraduate School, Monterey, CA. http://hdl.handle.net/10945/48587.

Salmerón J, Craparo E (2017) *Wholesale Inventory Optimization Model*, Release - 4.0. Project deliverable, Naval Postgraduate School, Monterey, CA.

Sanchez PJ, Oliver H (2018) Simplekit, version 3.2.0. Accessed August 26, 2018, https://gitlab.nps.edu/pjsanche/simplekit-python/blob/master/simplekit.py.

Schruben L (1983) Simulation modeling with event graphs. *Communications of the ACM* 26(11), https://doi-org.libproxy.nps.edu/10.1145/182.358460.

Silver E, Pyke D, Peterson R (1998) *Inventory Management and Production Planning and Scheduling*, 3rd ed. (John Wiley and Sons, Hoboken, NJ).

Tako AA, Robinson S (2011) The application of discrete event simulation and system dynamics in the logistics and supply chain context. *Decision Support Systems* 52(4), https://doi.org/10.1016/j.dss.2011.11.015.

Tersine RJ (1994) *Principles of Inventory and Materials Management*, 4th ed. (PTR Prentice Hall, Englewood Cliffs, NJ).

Teter SM (2018) Simulated operating concepts for WIOM implementation. Master's thesis, Operations Research Department, Naval Postgraduate School, Monterey, CA. http://hdl.handle.net/10945/58269.

Törn AA (1981) Simulation graphs: A general tool for modeling simulation designs. *Simulation* 37(6), https://doi-org.libproxy.nps.edu/10.1177%2F003754978103700604.

Vidalakis C, Tookey JE, Sommerville J (2013) Demand uncertainty in construction supply chains: a discrete event simulation study. *J. Oper. Res. Soc.* 64(8), http://dx.doi.org.libproxy.nps.edu/10.1057/jors.2012.156.

Under Secretary of Defense for Acquisition, Tech. and Logistics (2016) *Supply chain metrics guide*. (Washington, DC). https://www.acq.osd.mil/log/SCI/.policy_vault.html/Supply_Chain_Metrics_Guide_signed_3Mar2016.pdf.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California