ANALYSIS AND RECOMMENDATIONS ON TICKET DATA SYSTEMS AND PROCUREMENT STOCKING SYSTEMS FOR THE OFFICE OF INFORMATION TECHNOLOGY AT THE UNIVERSITY OF IDAHO

LOGAN KEARNEY, LINDEN BECK, PETER HUNTER, GUNAVARDHAN REDDY

SEPTEMBER 25

TABLE OF CONTENTS

CONTENTS

PROBLEM DEFINITION AND PROJECT SCOPE

PURPOSE

The purpose of our OM439 project was to analyze and improve the inventory management system within the University of Idaho's Office of Information Technology (OIT). Specifically, our team focused on the procurement department's inability to predict when to reorder stock or how much inventory to keep on hand. This gap in process leads to frequent stockouts, longer ticket resolution times, and limited flexibility for Technology Solution Partners (TSPs) responding to customer needs. Our goal was to create a model to determine the optimal stock levels at procurement service efficiency. By analyzing ticket data and testing inventory scenarios in ProModel, we aimed to reduce downtime, improve forecasting, and enable smarter stocking strategies going forward.

OIT OVERVIEW

OIT is the University of Idaho's centralized IT department. It manages all technology-related services for faculty, and staff. Within OIT, there are two key groups involved in our project, the Procurement Department and the TSPs, or Technology Solution Partners. Procurement processes procurement tickets, manages inventory, and orders technology equipment for everyone at the university. TSPs are the customer facing team that responds to ticket submissions and resolves tech related issues. Both teams rely on a centralized ticketing system known as TDX or TeamDynamix.

PROJECT CHALLENGES

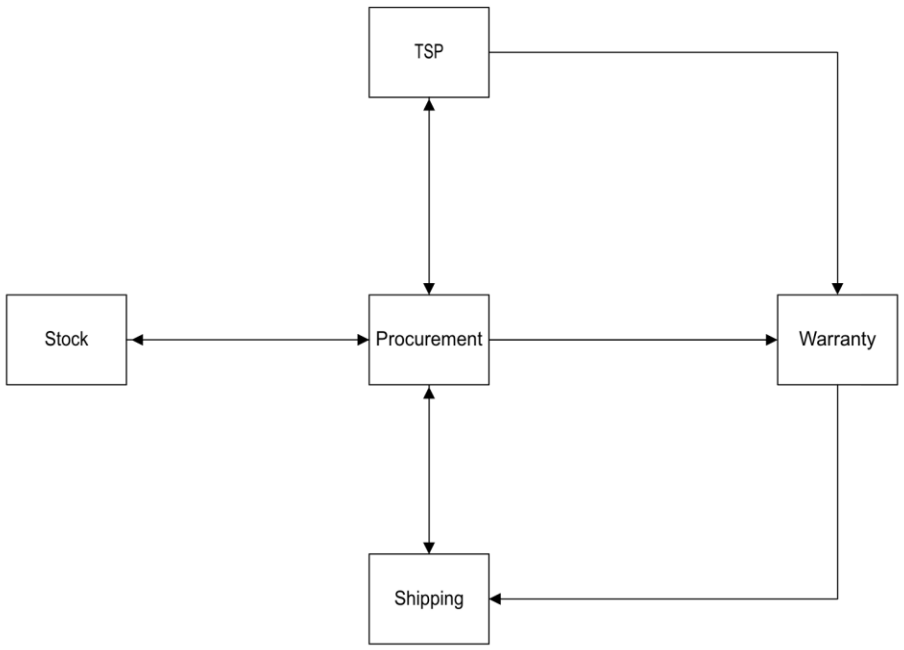
Throughout the project we encountered several challenges. Our scope was constantly changing and evolving throughout our process. Originally, we wanted to just predict when to restock and model campus using TSP regions. This was challenging because each ticket piece of the ticket that we needed to model this was not sorted or was missing valuable data. We were then presented with an option to compare 2 different models of stocking at OIT. Centralizing at procurement or implementing stock sites between Tsps. Another challenge we faced was the messy data that we were given. This took a lot of time to sort through what was useful and what was not due to the over 3000 tickets we received.

PROJECT SCOPE

The final scope of our project was to create a predictive model for re-stocking and optimizing inventory levels at procurement. In addition to this we wanted to analyze the ticket data and provide data visualization. Our objective was to reduce ticket resolution time and operational costs. Our core focus for this project became determining the optimum stock level that procurement should hold.

CURRENT OIT PROCESS FLOW

In the current process, when a ticket is submitted, the TSP checks whether the request is a warranty. If not, a purchase request is sent to Procurement. Procurement then either pulls from existing stock, contacts the vendor or if the item was damaged in shipping routed through the warranty process. Once the item is available, it's shipped to the TSP for final delivery to the customer. This centralized model has multiple routing paths and often leads to bottlenecks at procurement.



OUR MODEL

To solve this, we tested multiple stock-holding strategies within ProModel. These included keeping no stock, ordering everything on demand or holding large amounts of stock to ensure availability. The key question we addressed was, how much stock should we hold at Procurement to maximize efficiency?

KANBAN METHODOLOGY

As part of our inventory management strategy, we wanted to incorporate a Kanban methodology to help determine optimal stock levels sing JIT and replenishing only as needed. Although in theory it would be ideal to operate with zero inventory, that approach is not realistic for the OIT environment due to vendor delays, warranty routing, and variable ticket volume. However, Kanban still provides a structured, data-driven method for minimizing excess while ensuring necessary items are on hand when needed.

WHY THIS MATTERS

Overstocking ties up budget and storage space. The procurement location on campus doesn't have a lot of storage space in its office and it would not make sense to store excessive amounts of large monitors especially if there was no demand for them at the time. In addition to this, the understocking leads to long ticket fulfillment times and possible stockouts. Finding the optimal stock level is critical for preventing workflow bottlenecks in OIT. A more efficient inventory process improves responsiveness and ensures procurement can meet TSP needs more reliably. System wide improvements like this also support future growth and service expansion.

CURRENT PROCESS WEAKNESSES

Inconsistent stock levels

Stock levels are inconsistent because there is no formal method for tracking usage trends or predicting demand. Additionally, ticket volumes can vary by the time of the year and their system doesn't account for these demand spikes. The absence of a JIT system leases to manual decision making and guesswork.

Unknown re-order points

There is an absence of defined re-order points in the current process. Without these re-order points the procurement staff are left to make case-by-case decisions on when to re-order leading to inconsistent stock levels.

Longer ticket fulfillment times

Longer ticket fulfillment times are a direct result of the previous two weaknesses. This forces TSPs to have to wait on procurement to place orders from the vendors adding significant lead time to the overall ticket time.

Limited flexibility for TSP/procurement

TSPs lack flexibility because they rely on procurement for all items, even the most basic ones like an HDMI adapter. Procurement lacks flexibility because they must operate reactively without any forecasting tools.

Inconsistent ticket time estimation

Without visibility into stock levels and lead times there is inconsistent ticket time estimation.

OUR THEORY AND HYPOTHESIS

We theorized that using Kanban methodology would help reduce lead times and enable smarter re-stocking decisions. We also believed that low-cost, high-frequency items were ideal for stocking and that reducing stockouts would result in noticeable time savings across the ticket lifecycle. These could be items such as mice, keyboards, webcams, or HDMI cables.

PROCUREMENT TICKET DATA AND PROCESS

This section will cover the data sources used in our visualizations, analysis, and simulation. This process has been broken down into the following: ticket data process and ticket data analysis. Ticket data process will cover the ticketing system, and the data retrieved from procurement used in this report. It will also cover issues with the system and provide recommendations to remediate the problems found. The ticket data analysis section will cover the processes used to break down the ticket data into the final format used later in the data visualization and modeling of stock.

TICKET DATA PROCESS

Ticket data is collected by TDX and compiled into a report in the form of an excel sheet. This section will cover the process on the TDX side of the data process. This includes how it is collected and how that process influences the data available for analysis.

Data Types

To cover the types of data given for analysis, some terminology clarification must be done. The following are the four types of data that will be addressed in this paper:

USER DATA

User data is data that is input by an end user or employee that is not constrained. This can be seen in places like a text field, such as the ticket description field. Due to the freeform nature of this input, user data is considered unreliable information and is not suited for automation as its format is variable with unverifiable information.

CATALOGUE DATA

Catalogue data is information that is usually pulled from a dropdown or radio button set. It is generally a set series of options used to input the information in a consistent, monitored format.

GENERATED DATA

Users Input data to create fields. Generated data is locked in process and is not directly input by the user. While it is created by the computer it relies on other user or catalogue data fields as inputs so it has the potential for errors. It is preferred to those fields however as it is possible for error detection to occur as a standard process must be followed in the data generation so errors can be detected if they halt the processing step.

METRIC DATA

Metric data is generated automatically by TDX and consists of information about the state of the ticket. This data can be considered canonical and is good for baselining results as it can be trusted as valid and accurate data. These can be Ticket IDs, creation date of the ticket, age, etc.

As these datatypes progress from user data to metric data, they are considered more mature. This is due to the limitations they impose on the data input into the field, allowing for a better control over the possible data to analyze. As this ticketing system demonstrates, there are a variety of fields to be considered when creating a ticket template. The more mature fields are preferred, but any field can be used if the consequences of maintenance and data loss are addressed.

Ticket Data

In the context of this paper, ticket data is any of the fields of data given to us in the ticket report. There are additional pieces of ticket data that were unable to be utilized in this report, such as workflow information. While not a complete set of all available ticket data, the ticket data that was provided covers the key performance indicators and order information that was necessary to conduct this analysis and construct the stocking simulation.

Data that could have also influenced the design and analysis of the simulation such as stock numbers and order pallet size was not available for this analysis. These figures are not a part of the TDX system.

Report Data

The data report used in this project ran on March 11, 2025. It contained ticket data for procurement tickets for the fiscal years of FY23 and FY24. This was advised as the range to be used as it is expected to reflect current procurement procedures and practices.

It is recognized that depot is no longer part of the supply chain for computers, but this will not be factored into the report as workflow data to estimate depot cycle time was unavailable for analysis.

The following fields were included in the raw data for analysis:

Data Field	Data Type	Unit of Measure	Variable Type
Ticket ID	Metric Data	N/A	Integer
Asset Count	Generated Data	Count	Integer
Primary Responsibility	Catalogue Data	N/A	String
Description	User Data	N/A	String
Age	Metric Data	Days	Integer
Created By	Metric Data	N/A	String
Acct/Dept	Catalogue Data	N/A	String
Service Category	Metric Data	N/A	String
Source	Metric Data	N/A	String
Tags	Catalogue Data	N/A	String
Total Cost of Order	User Data	Dollars	Float
Vendor Order Description	Catalogue Data	N/A	String
Invoice Number	Catalogue Data	N/A	String
Vendor	Catalogue Data	N/A	String
Location	Catalogue Data	N/A	String

The breakdown and transformation of these fields for analysis will be covered in the ticket data analysis section.

As shown above, there are many values that are available for analytical purposes. While this report focused on building an analysis focused on VOD line-items, other reporting is possible with its built-in ticket ID centric format.

TICKET DATA ANALYSIS

Ticket data from TDX was broken down into usable information using excel and python scripts. This section will cover the processes used to put the data into a format useful for visualization and analysis.

PYTHON SCRIPTS

To facilitate the analysis of ticket data in a timely manner, a series of python scripts were written. They have been condensed into two major scripts that will be covered below. These are the primary filter and the order generator.

Primary Filter

The primary filter is used to break down the raw ticket data into a table for analysis. It uses the vendor order descriptions to create a list of line items for analysis. This is opposed to the raw ticket data which orders the data by ticket ID. The process of breaking down the VOD is covered in the vendor order description section of the paper.

The output file of the primary filter is the main source of data that is used in the later data visualizations and simulations. This file contains additional fields generated by the filter to aid in the analysis. A table of these fields and descriptions thereof is provided below.

Data Field	**Description**
TicketID	Ticket ID from ticket data
Item	Item in VOD line item (ex: T14 Gen 4). This value is pulled from a lookup table, if the filter for this item cannot be identified it is assigned a value of "N/A".
Quantity	Quantity of line items in the VOD to be purchased. If the primary filter is unable to determine quantity information, it will not create a line item in the output file.
Price	Price is the VOD line-item price in dollars. If the primary filter is unable to determine price information, it will not create a line item in the output file.
Category	Category is the type of line-item in the VOD such as laptops, desktops, or monitors. This is pulled from the same lookup table as the Item data field. If the item is unidentified then this field will as return as "N/A"
Age	Age of the ticket in days. This information is pulled from the ticket data associated with the VOD and is considered canonical as it is a metric ported directly from TDX.
Created	The date and time that a ticket was created. Like age, this is canonical as it is ported from TDX.
Vendor	Vendor is the same value as the data field from the raw ticket data. It was not directly utilized in our data analysis as the breakdown of the vendor information into its constituent components was infeasible for the time scope of this project. It is still provided as further analysis could potentially separate the data and utilize it in future reports.
Acct	This is the same data field as Acct/Dept in the raw ticket data. It is used for categorization of the data during analysis.
Dept	Dept is the abbreviation of the department name in the Acct/Dept field in the raw ticket data (ex: OIT, CNR, CLASS). It is used during analysis for the categorization of the data.
Weekday	String representation of the weekday of the creation date of the ticket.
Day	Integer day value of the creation date of the ticket.
Week	Integer week value of the creation date of the ticket.
Month	Integer month value of the creation date of the ticket.
Year	Integer year value of the creation date of the ticket.
WorkingDay	This field is not used for analysis, it was meant to represent the age of a ticket minus the weekends and holidays, but it was not fully implemented and is left in as legacy data.
ToStock	To Stock is a Yes/No Boolean field that indicates if the order is a stocking order. This information is pulled from a manually curated list that was compiled into a lookup table compared against ticket IDs.
Date	Day/Month/Year representation of the creation date of the ticket.

Lookup Tables

To retrieve and classify data from the raw ticket data using the primary filter, lookup tables are used to assign values. These are required to add classifications not included in the ticket data, such as device type (ex: laptop, monitor, cable). These had to be created manually and would not be suitable for use in a production environment unless they directly pulled information from a central catalogue.

Order Generator

Order generator is a python program used to generate a ticket order schedule for the ProModel simulation. Its output is only used in the ProModel simulation and will be covered in that section of the paper.

VENDOR ORDER DESCRIPTIONS (VODS)

VODs are a section of an order ticket that contains the description of items to be purchased. This includes the item, quantity, and price per item. A VOD can be constructed of multiple blocks of these line items to create a full order. VODs were the primary source of data for this analysis as they could be counted as canonical data. As nothing was ordered on a ticket that was not included in a VOD, it was safe to assume that it was a reliable source of information.

VODs have limitations as the primary source of information. They do not include other pieces of ticket metric information and have long item descriptors that come from the standard catalogue. VODs also do not contain information about the sourcing of the items, such as pulling from stock vs ordering. This means that even as a canonical source of information a VOD has limited use without the accompanying ticket information.

VOD PROCESSING STEPS

1. Section Identification

   The section identification step breaks the VOD into its constituent line items.

2. Separation

   Separation breaks the individual line item into the Quantity, Item, and Price sections.

3. Trimming

   This step removes unneeded characters from the separated sections.

4. Categorization

   The item information is compared to a lookup table of items and attributes such as device type and vendor are assigned as additional data fields.

5. Information Loading

   Non-VOD ticket data is loaded into the line item from the original ticket.

6. Export
   The assembled data is then added to a final output table and exported to excel.

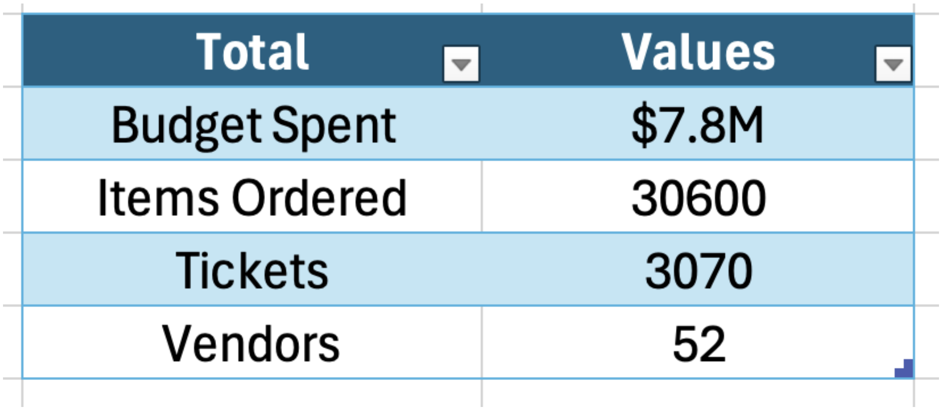
DATA REPORTING AND VISUALIZATION

DATA REPORTS

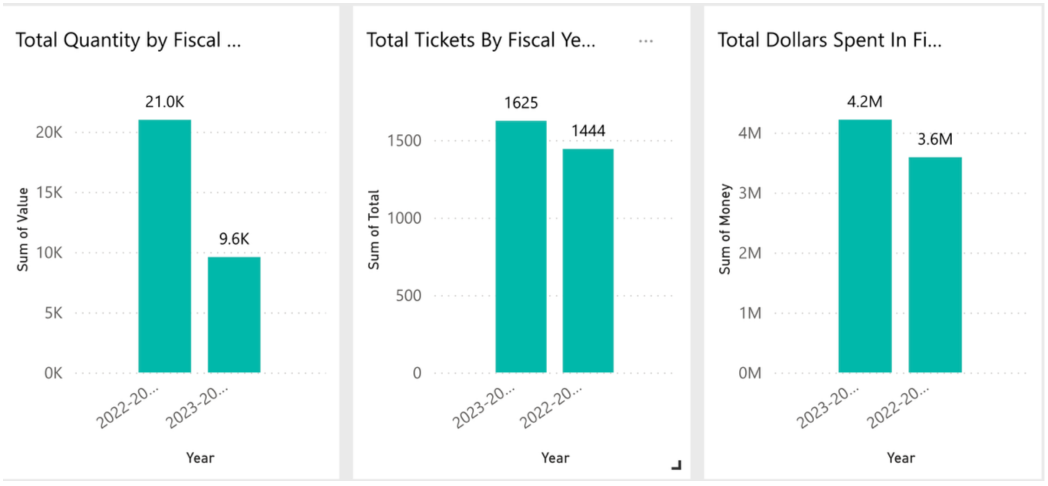
We have decided to visualize the data because we wanted to evaluate the performance and their spending on electronics so that we can run the simulation based in the problems and give IT leadership real time visibility of their performance for two fiscal years. From the data we got logan cleansed the raw data and re-structured 3400 tickets in new excel file.

We developed multiple reports from the data. These reports would give you the total budget spent on electronics and accessories, total tickets processed in these two fiscal years and averages of the IT procurements performance.

We visualized the reports in three ways. We calculated the averages individually for every single year and for both years separately.

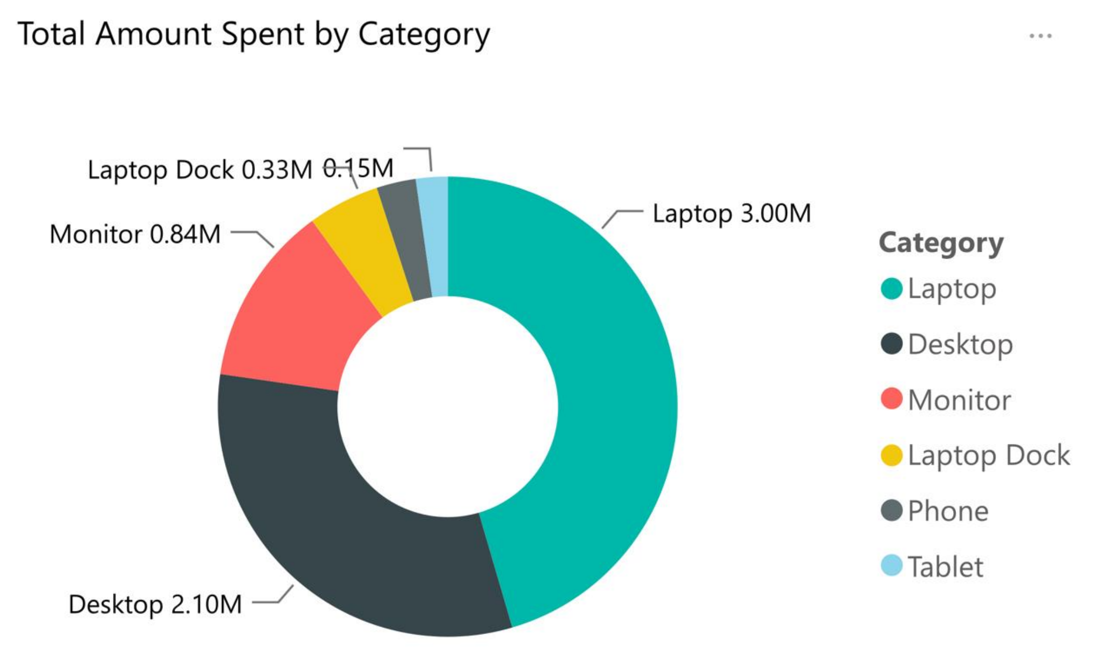


Total	Values
Budget Spent	$7.8M
Items Ordered	30600
Tickets	3070
Vendors	52



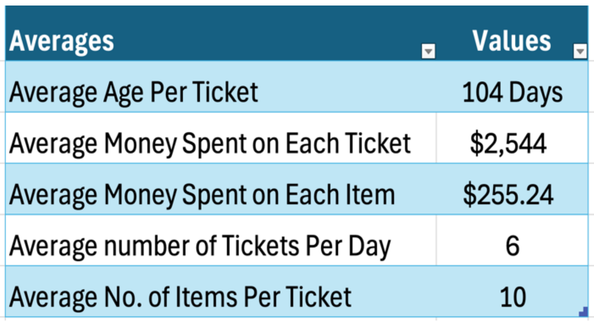
We calculated all these stats because it’s important to know that how the trends have changed in those two years. We have seen the total items ordered are decreased by 52% while the total dollars spent only decreased by 14%. From the above table and chart, we can say that the IT spent $7.8M dollars in total while spending $4.2M in 2022-2023 and

$3.6M in 2023-2024. The total distinct tickets are also decreased by 180 in the second fiscal year.



Out of $7.8M dollars $6M were spent just on three items which are Laptops, desktops and monitors. Laptops alone accounted for $3M dollars which says that IT must stock laptops depending on the demand.

PROBLEM:

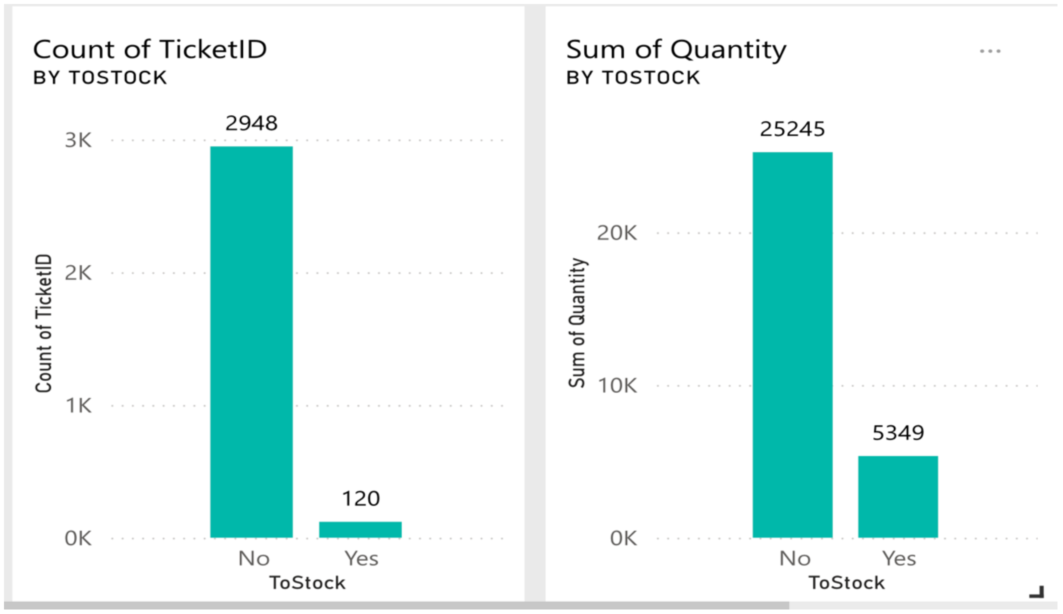


Averages	Values
Average Age Per Ticket	104 Days
Average Money Spent on Each Ticket	$2,544
Average Money Spent on Each Item	$255.24
Average number of Tickets Per Day	6
Average No. of Items Per Ticket	10

When you look at the data from above the main problem that IT is facing is delay in completing orders from customers. It’s almost taking the IT procurement department 104 days on an average to complete a ticket.

IT spends an average of $2500 per ticket and $255 per each item. On an average they processed 6 tickets each working day and must order 60 items on a single day.

ROOTCAUSE:



The chart explains why IT is taking almost 3 months 15 days for completing a ticket on average. IT doesn't have a good stocking method or inventory method. While they are processing 6 tickets on an average each day and must order 60 items the individual tickets that are used for purchasing stock is 120 out of 3040 and when you look at the ratio of total products ordered to stock and total products ordered for customers its 1:5 which low and should increase on LHS to decrease the processing time of tickets.

STOCKING IMPROVEMENT SIMULATION

CURRENT PROCESS

Currently OIT does not have a way to manage their inventory. When they run out of items they just order more. With our promodel simulation we set out to make a more efficient way for OIT to manage their inventory. We were unsure of how much inventory OIT had on hand, so we set up our simulation in two ways. We initially set up our simulation so that every time an order was placed it would be routed to the export/import location and get ordered there. Our second simulation was set up where there was a safety stock. Any time the simulation got below that set stock amount for each item, it was reordered.

PROMODEL SIMULATION

We decided to set up our promodel simulation with 5 locations. The locations that we chose were TSP, Procurement, stock, Import /export and Warranty. The TSP location represents the Person who receives the order and then sends it to procurement. In the real world, Stock is a Location inside of Procurement. We split them into two locations because we needed them to have different items that go in and out of them. Our Import and Export location was simple. It was where all the orders that had a warranty request went through. Our warranty location was also simple. We just had the orders go through the system then wait for the set time that we gave to the warranty variable. After that they went through the system and left at TSP.

We only used one entity in our simulation. We called this entity "Line_Item". There was only one entity but in our arrivals file we had it appear 600 times. This quantity of times was supposed to simulate a semester. The purpose of using only one entity was that it would keep our simulation simpler and it would also make it easier to build our arrivals file.

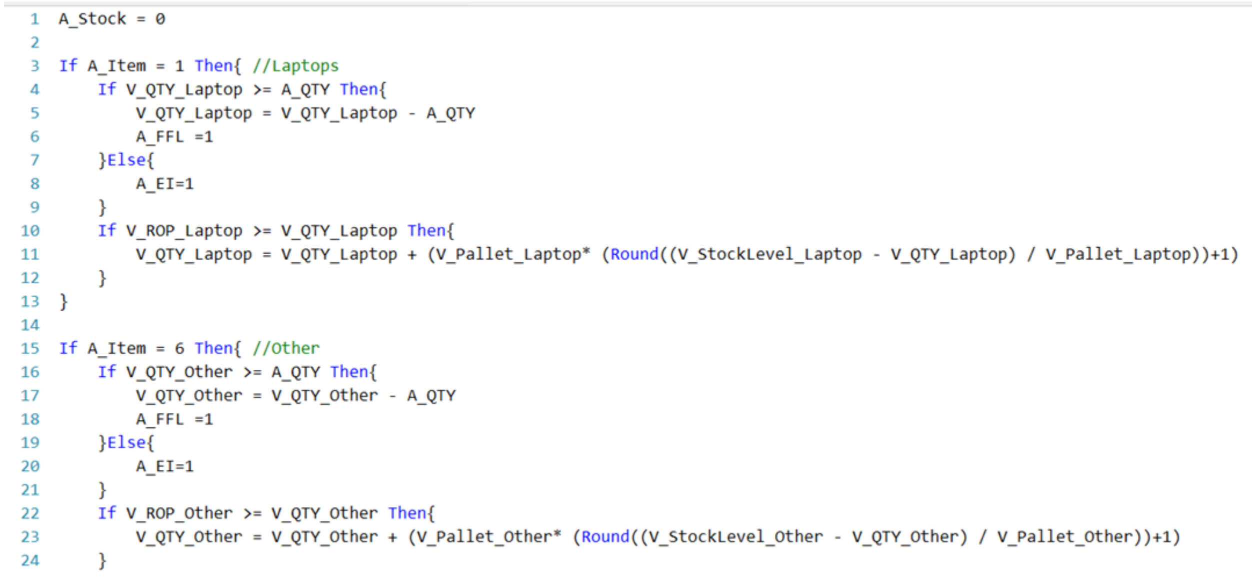
We used Attributes in our simulation to sort where orders were supposed to go as well as other things such as which TSP processed the ticket. Our Attributes were A_FFL, A_War, A_Stock, A_EI, A_Proc, A_Qty, A_Item, A_TSP, and A_Ticket. The Attribute A_FFL stood for fulfilled and was given to an order once it had been completed. An order needed this attribute to leave the system. The attributes A_War, A_stock, A_EI, and A_Proc were our routing attributes. These attributes determined whether orders should go to Stock, Export/Import or procurement. If an Order had a warranty request, then it was given the Attribute A_War and could proceed to the Warranty location. The Attribute A_Ticket corresponded to the ticket number which was in a separate file that contained the raw data.

Our arrivals file contained 600 rows of orders. This number represented a semester worth of orders. In our arrivals file we had all our attributes. Each attribute was assigned either a 1 or a 0. If the attribute had a 1 then it was sent to that location. Our attribute A_FFL had a zero all the way until the end when it was complete. One it was complete it was given a 1 and then it could leave the system. In our arrivals file each item was assigned a number 0-6. These numbers corresponded to a category of items. For example, 0 corresponded to desktops, 1 corresponded to laptops, 2 corresponded to docks and so on.

We had 38 Variables in our Simulation. All our Variables controlled the inventory levels at stock location, the waiting times for when our orders were placed and the quantity that could be ordered on a pallet. For each item in our simulation, we gave it a Variable such as V_QTY_Webcam. In this instance this Variable would control the Quantity of webcams held at the stock location. Another variable that we had was V_OT_Docks. This Variable controlled how long the order time was to order the Docks. The Variables to order more stock got a little more complicated. These variables were V_ROP_Monitor, V_Stocklevel_Monitor, and V_Pallet_Monitor. In this instance these Variables controlled the point at which new monitors would get reordered. When the V_QTY dropped below a set quantity at V_ROP, then the V_Stocklevel variable would increase by the pallet size set by V_Pallet. We built these variables for every category of item in our simulation

The Processing at the procurement location was relatively straight forward. We used a simple If statement that corresponded to an attribute. If the attribute had a 1 then it was sent to that location. The

In stock location the processing was a little more complicated. We had split everything into two parts. The first part of the processing controlled the inventory levels. When an order comes into the stock location the quantity of the item would be pulled and sent back to procurement. If the order was too big for the stock location to handle it would go to the second part of our processing. In the second part of that processing, the order would be sent to the import export location to have more quantity ordered so that the order could be met. The processing for the stock location looked like what is in the image below. We had a section built for every single category that we had in the simulation.



```
A_Stock = 0

If A_Item = 1 Then{ //Laptops
    If V_QTY_Laptop >= A_QTY Then{
        V_QTY_Laptop = V_QTY_Laptop - A_QTY
        A_FFL =1
    }Else{
        A_EI=1
    }
    If V_ROP_Laptop >= V_QTY_Laptop Then{
        V_QTY_Laptop = V_QTY_Laptop + (V_Pallet_Laptop* (Round((V_StockLevel_Laptop - V_QTY_Laptop) / V_Pallet_Laptop))+1)
    }
}

If A_Item = 6 Then{ //Other
    If V_QTY_Other >= A_QTY Then{
        V_QTY_Other = V_QTY_Other - A_QTY
        A_FFL =1
    }Else{
        A_EI=1
    }
    If V_ROP_Other >= V_QTY_Other Then{
        V_QTY_Other = V_QTY_Other + (V_Pallet_Other* (Round((V_StockLevel_Other - V_QTY_Other) / V_Pallet_Other))+1)
    }
```

Our processing at the Import/Export location was somewhat straightforward. When an order comes into that location, it is ordered and waits the order time set by the Variable V_OT. If another order arrives, it also waits there until the order time has passed. Once this is done the order is routed up and through the simulation and back to procurement. All excess inventory is then sent to the Order location.

The processing of our warranty location was the simplest of all our locations. Every warranty request was sent from TSP or Procurement and then waited the set time by the variable V_Warranty. Once the set time had been waited the item was given its fulfilled attribute and could be routed through and out of the simulation.

SIMULATION ANALYSIS

Through our simulation we were able to determine that adding a safety stock to the OIT department would be beneficial to them for a couple reasons. In our simulation we found that having a safety stock was 53% more efficient than ordering everything. It is difficult to determine how much more efficient this type of inventory management would be for their current system. Another benefit of implementing the safety stock inventory system would be improved customer service. If OIT had the safety stock, then they could constantly be handing out inventory to customers knowing that there was more on the way, thus improving their customer service and reducing the time it takes to complete a ticket.

STOCKING RECOMMENDATIONS

Based on our simulation, we recommend that OIT implement the use of safety stock. Adding safety stock to the OIT inventory management would allow them to fulfill orders more efficiently, leading to better customer service levels. An obstacle to overcome with safety stock is that you must pay a higher upfront cost because you are buying twice as much, as well as you also must find a large place to store your larger inventory. Another advantage of safety stock it that it would prevent stockouts and also provide some resilience to the changing supply chain. The benefits of having a safety stock outweigh the cons and adding it would be the best way to improve their current inventory system.