Redesigning the Repair Network of Toner It Down! Inc.

ORIE 4580 Final Project

Anthony Lea ajl339

Borja Ocejo Elizondo bo97

Leeds Rising lar294

Christina Zhou cyz9

December 6, 2019

Executive Summary

The leading document management system manufacturing company, Toner It Down! Inc. (TIDInc.), is currently operating in the Syracuse area and has in place a repair division that features a network of ten business centers and a dispatch center. The network is maintained by mechanics as well as van operators and their vans. The goal of TIDInc. is to become more competitive with other providers in the region by improving the time it takes to satisfy customer requests. In order to achieve this, our team has conducted a preliminary study that focuses on one model of copiers currently in use, and this has allowed us to determine the number of mechanics and vans that will meet the service goals set forth by the business. The results of this report can help TIDInc. better utilize their workers, improve the time it takes to serve their clients, and optimize their current system overall. Our recommendation is that TIDInc. hires 8 mechanics and 3 vans to operate on a daily basis.

At this moment, based on feedback from managers, on-site diagnosis and on-site repair times of copiers cannot be further reduced at this time. Thus, we turned our focus to worker assignment, response, and delivery times. There are two main areas where TIDInc. is currently lagging behind in terms of service and can improve it's initial response time and delivery time for replaced customers: initial response time (the time interval between when a customer places the call and when a mechanic reaches the customer location) and delivery time (the time interval between when a customer places the call and when a functional copier is delivered and installed for the customer, which is only is recorded when the copier cannot be fixed on-site). Keeping these metrics in mind, we found the lowest number of mechanics and vans necessary to keep initial response time under 1 hour - with a more precise average of .96 hours - and our delivery time for replaced copiers under 3 hours - with a more precise average of 2.60 hours - to be 8 and 3 respectively.

To complete this study we created a simulation model that replicates a day of TIDInc. operations. Using real data collected over 60 days of TIDInc. business, our team was able to create probability distributions that allow us to reproduce the events that occur over the course of a day. We are able to generate a realistic distribution of customer requests at service times as well as a realistic rate of copiers that require replacement. The collected data also allowed us to

approximate diagnosis and service times across the study, with randomness included to represent the variance in a day's work. We also incorporated the physical limitations of the system presented, such as travel time between the centers and the way different entities are able to travel. By simulating a day of work thousands of times using these random distributions we were able to discern these distributions with confidence.

To further support our assertion of 8 mechanics and 3 vans, we analyzed the utilization rates of mechanics and vans. Utilization rate shows the percent of time the mechanic or van is in use. It is important to establish the balance of not having too many idle workers (too low of a utilization rate) and not having any free workers when a new request enters the system (too high of a utilization rate). Our mechanic utilization is on average 62% and our van utilization is on average 16%. Our mechanic utilization rate varies from the vans because of the shorter time period for one job assignment, the larger number of mechanics versus vans, and the fact that on average only 17% of calls require a van delivery component.

At $140,000 per mechanic and $100,000 per van, our assignment of 8 mechanics and 3 vans will cost the company $1,420,000 for a year of service. This investment will allow TIDInc. to compete in the industry in the Syracuse area. Our simulation model and analysis provides us with the confidence that the 8 and 3 assignments will solve the goal. Further in this report, we provide further analysis of test runs beyond the key metrics in this summary, explain any assumptions we have made or taken in its creation, and verify our results by proving model correctness. We also completed a sensitivity analysis, which shows our model will not change with slight variation in customer request timings.

Problem Definition

Currently, Toner It Down! Inc. has a complex repair network and process for their products in use. With a set number of mechanics and vans, once a customer request is made and received at the Dispatch Center, the central home base of all mechanics and vans, a mechanic is sent out to the customer location. Once the mechanic reaches the customer location, he/she is able to identify whether the mechanic can fix the copier, or whether a van needs to be dispatched to replace the faulty copier. If the copier can be fixed on site, the mechanic does so and leaves. Otherwise, the mechanic leaves and a van is sent to the customer location to replace the broken copier and take it back to the Dispatch Center to be restored. Some important details we were given are that each customer/business center has the same type of copier, each van can only carry up to one copier at a time, the time it takes to swap out copiers at the dispatch center and the customer centers have expected times and distributions, and the speed of the vans and cars are approximately averaged at a constant rate (60 km/h).

By their request, we are focusing primarily on initial response time and delivery time.. As mentioned before, our goal is to aim to find certain metrics where the initial response time is less than an hour, and the delivery time of a copier is less than 3 hours respectively. Thus, by developing a model based off of their current systems, we can advise TIDInc. on the number of mechanics (including cars) and the number of vans needed to keep their company fully functional and to meet important performance metrics in order to stay competitive.

Modeling Approach

Before we analyze how the simulation model was approached at a high level, it is important to explicitly state and explain the assumptions that we made that factored into the construction of our model.The first major assumption that we made was that mechanics - upon leaving the dispatch center for the first time - will stay out in the field until it sees another call to service. This is important for initial response time, delivery time, and mechanic utilization - as mechanics are making a journey of different lengths from their previously serviced location to an incoming call than they might have if they left from the dispatch center (another viable representation might have sent mechanics back to the dispatch center upon being idle for some time). For example, if there is a call to BC1, there is a different response time if the mechanic comes from the dispatch center versus if the mechanic comes from its previous location, BC2 (0.75 hours versus 0.33 hours, respectively). Likewise, delivery time is simply the summation of time waiting for a mechanic to start working on the call, the time to travel to the location of the call, the time to address the call, and then the time for a van to travel to the site of the call with the new copier and install said copier. While the travel time for the mechanic to the site of the call is somewhat negligible in this case relative to the other values being summed (the 95% confidence interval for delivery time was [2.5639, 2.6447]) it still nonetheless serves as an important clarification to where our calculations came from. The last metric affected by this assumption was mechanic utilization, which is calculated as the amount of time every mechanic spends traveling to the sites of calls and addressing them divided by the total amount of time in the simulation period. This assumption will thus impact mechanic utilization, as the amount of time spent traveling to sites of calls is of the same magnitude as the amount of time spent servicing said calls (which is on the order of a fraction of an hour).

The second major assumption that our model ran on was in the way that we calculated our model time frame. We analyzed over a single day, but instead of waiting for a downtime with no more mechanics or vans working (or some other time period relevant to the events occurring), we simply ended the timeframe of our model once our "current time" indicator was past the 24 hour mark (more on the analysis of our time frame can be found in section 1 of the appendix).

The final assumption that we made was in the way that we handled assigning mechanics to calls when a call came in and all mechanics were busy. In the ideal case, we would assign the mechanic based off of finish time and proximity to the caller's location. To build this into our model, however, proved to be quite complicated, and a realistic simplification that we chose to make was to have these calls be assigned to the *next free mechanic* instead. This has the potential to change the mechanic travel times associated with handling our backlog of client calls, as there exist cases where the earliest finished mechanic might be far from the location of the call but the mechanic finishing only a few minutes later was just 10 or so minutes away. In such a case, we would be adding nearly an hour onto our response time. We argue, however, that such an assumption is realistic, as it would likely be very hard to keep this level of communication and optimality in regards to response time upheld.

Bearing these assumptions in mind, we can now move into a high level explanation of our model. The first piece of our model is that we set it to increment for each occurrence of an event, whether that event is a new customer call or the completion of a task (either a van or mechanic servicing a customer). As a result of this, we ensure that whenever a new call comes in, we address that call and incorporate into our actions at the exact time of day it happens, and whenever a mechanic or a van finishes servicing a customer, we incorporate the ramifications of either the mechanic or van finishing (thus being able to service another waiting customer). We will continue to run our model as long as we have not exceeded a one day period (touched upon in assumptions above) and will continue to maintain this list of impending major events as time goes on. It is important to note that jumping between these three categories of events will cover the three major points of change that need to be addressed in our model, and we will now dive into each and how they changed the current state of events.

The first type of key event that we have to handle is a service call. In the case that a customer has a call come in, we first calculate the time that the next call will come, and add this to the list of important events that we maintain and will continue to check from. Doing this right as we process an event (meaning "current time") ensures that the next event will happen after a proper interarrival time. More on this logic is covered in appendix section 3.

Since we have a call, we now can look for a mechanic to service it. We do this by creating a list of free mechanics, and then looking for the one with shortest travel time from the mechanic's current location to the location of the call. Once we have found that mechanic, we set their status to busy and the time the mechanic finishes to be the time it takes the mechanic to travel there plus the time of the repair. If there are no currently free mechanics, however, we add this call to a "backlog" for mechanics to address at a later time - this backlog will be addressed in order of time of call (early calls will be prioritized first) and every time a mechanic becomes free, it will check to see whether or not there is a call in the backlog - this ensures that we never waste time having a mechanic be free and an unserviced call existing. Furthermore, at the time of mechanic servicing a request, we need to take into account whether or not this call will require a van - if it does, we should call a van as soon as the mechanic is finished with its diagnosis, and in the case that it is not, we should add the appropriate onsite repair time to the amount of time that this mechanic will have to stay busy servicing this call (this is covered in more detail in appendix section 2).

The next key event that can occur is a mechanic finishing with its service time. The major check that we have to make is whether or not a van is needed after this call - as mentioned before, we are maintaining this data through the list of events. In the case that it is needed, we would check for a free van, and - if one is found - assign it immediately. Since all vans start from the dispatch center, we can trivially take the first free van we find and assign it to be the one servicing this call for a van. We set this van as busy so that it cannot be assigned to another call, and makes its finish time set to be the summation of the time it takes to swap at the dispatch center, swap at the business center, and travel to and from the location of the call (this is elaborated upon in appendix 4). With this information, we can add the time this van finishes servicing the client as a key event to be checked on in the list we maintain.

In the case that no free van exists, then we would want to add this call to the van backlog, which should be addressed as soon as a van becomes free. We maintain this event as the time the mechanic finished and the location of the call so that the next free van can service this event.

Finally, we need to check if any events exist in our mechanic backlog for calls. If a call is pending a mechanic, we should set this mechanic to immediately start servicing that call, making the mechanic busy once again (more on this logic is elaborated upon in appendix 5). In the process, we would also take the pending call out of the backlog and create another mechanic event, as this mechanic has just become busy once again and now has a new finish time.

The final type of key event that we need to address is a van finishing its service and arriving back at the dispatch center. In the case there are events in the van backlog, we would assign this van to the earliest call in the backlog (elaborated upon in appendix 6), and then update this van to be busy for set time (again elaborated upon in appendix 4). Otherwise, we simply set this van to no longer be busy so that it can be assigned when needed later on.

With these three events covered, we will have successfully handled each major event that occurs throughout our simulation window and made the appropriate state changes (in regards to which mechanics or vans are busy, what events exist on our key events list, and what calls have been serviced). After our simulation of a day has terminated, we will calculate the appropriate metrics that occurred over the course of that day (response time, delivery time, utilizations, etc.) record them, which - as we repeat this 24 hour simulation over many iterations - will ultimately form the histograms and confidence intervals that are analyzed later in this report.

Data Analysis

In order to accurately drive our model and create a simulation which efficiently depicts TIDInc.'s current repair process, we conducted extensive data analysis on the company data. There were several important numbers and distributions that we had to derive in order to correctly simulate the model:

1. *Proportion of Repair Copiers:* Once a mechanic arrives at the location and diagnoses the copier, we need to determine the proportion of copiers that need replacement. Given the number of copiers that needed replacement over a 60 day span, we were able to develop a 95% confidence interval of roughly [17%, 19%]. Thus, we can now randomly generate the number of repairs needed by estimating that approximately 18% of the total customer calls would need replacement.
2. *On-site Repair Time Distribution:* In order to determine the amount of time it would take a mechanic to conduct an on-site repair, we graphed all the given times in minutes that it took to repair the copiers. Given the histogram of on-site repair times, we were able to overlay approximations of different distributions of the data in to determine which would best fit our model. We came to the conclusion that both the Rayleigh and Beta distribution fit our on-site repair times, and ultimately chose Beta along with its respective distribution parameters to generate our on-site repair times within the model.
3. *Initial Diagnosis Times:* Once a mechanic arrives at the customer location, the mechanic must diagnose the current situation. We evaluated the time of diagnosis by calculating both the mean and standard deviation for each of the ten business centers (customers). From the analysis, we were able to see how the Business Centers 1, 4, 5, 6, 7, 8, and 10 each had similar mean times of about 16 minutes and a standard deviation of roughly 3

minutes. On the other hand, the Business Centers 2, 3, and 9 have mean diagnosis times of about 22 minutes and a standard deviation of around 5 minutes. From this realization, we decided to place the Business Centers into two separate groups and graphed histograms of their respective times. From our graphs, we assumed that the initial diagnosis times are approximately normal, and we could model the initial diagnosis time based on the normal distribution along with the respective mean and standard deviation parameters we had calculated earlier.

4. *Fraction of calls from each business center:* While generating interarrival times of each call, we needed to find the probability that the customer call would originate from any of the ten given business centers. Thus, we calculated the mean and 95% confidence intervals for each of the customers. From the analysis, 3.95% for business center 1, 8.15% for business center 2, 10.79% for business center 3, 13.5% for business center 4, 11.84% for business center 5, 5.5% for business center 6, 12.39% for business center 7, 5.79% for business center 8, 13.75% for business center 9, and 14.24% for business center 10 originate from the respective customer centers.
5. *Function for Customer arrival rate:* Given that we know the times in which each customer request arrives, we needed to create a rate function as a function of time. By fitting a poisson process, we were able to determine that the rate function can be represented as $\lambda(t) = -0.067t^2 + 1.581t - 1.289$ for the times between hours 3 and 22. Otherwise, for the rest of the day, we determine that the rate is a fixed constant at 1.137.
6. *Distribution of Copier Swap Times:* From the given description, we can determine that the distribution of the copier swap time at the customer location can be modeled as a Triangular distribution with minimum 20 minutes, maximum 60 minutes, and most likely 30 minutes. Conversely, the distribution of the copier swap time at the dispatch center can be modeled as a Triangular distribution with minimum 10 minutes, maximum 25 minutes, and most likely 15 minutes,

Thus by using the numbers and distributions we determined above, we were able to accurately model the TIDInc. repair process.

Verification of Model

In order to conduct model verification, we must simplify our model so that we can easily test the theoretical times we should be achieving. To simplify our model, we first set the number of business centers to one, so that the mechanics will only travel to one location. Second, we set all of our distributions to be the exponential distribution in order to maintain standard queueing model practices. For example, if a mechanic conducts a repair, the distribution of beta we changed to be exponential. Third, when the mechanic has the initial diagnosis time, we limit the number of groups from two to one to simplify the service time. Next, we decided to limit the number of mechanics and vans to one and one respectively in order to keep the model simple. Finally, we conducted two tests where mechanic only had to do repairs, or where there were only replacements needed for the copiers. Thus by watering down our model, we can easily see how certain times stay the same.

In addition, we also verified the model by changing the number of vans and mechanics intuitively. For example, if we increase the number of mechanics, then we expect our initial response time confidence interval to decrease, and our mechanic utilization to also decrease. Conversely, if we increase the number of vans, then we would expect that the delivery times to replace copiers to decrease and the van utilization to also decrease. By checking our metrics, we can thus verify our model intuitively.

Model Analysis and Findings

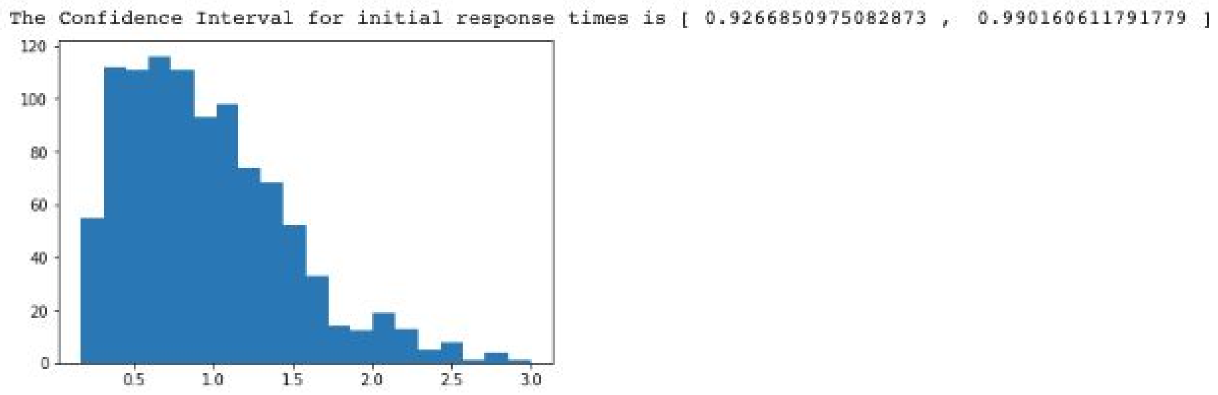
As stated at the beginning of our report, TIDInc. is looking to find the minimum number of vans and mechanics the company needs to match their competitors on two metrics - the initial response time and the delivery time for replaced copiers. TIDInc. believes their competitors have an average initial response time of less than one hour and an average delivery time for replaced copiers of less than three hours.

In total, we computed four main metrics including the primary two described above. Our metrics are computed on the scale of a day. This means that for each day in the simulation, we collect an average for each measurement and store it in a global list. At the end of the 1000 iteration simulation, we are then able to calculate a total average over all days, and a corresponding 95% confidence interval, that shows the variance and exactness of our results.

The first metric we computed is average initial response time. In our model, this metric was computed across two separate cases. In the event a mechanic was free, the initial response time was simply the travel time from the last job (or dispatch center). In the event all mechanics were busy, we added the waiting time for a mechanic to free up as well as the travel time. At the end of each day in the simulation we took the mean of all response times, and at the end of all the iterations we took the total mean and confidence interval. Our findings below support our assignment of 8 mechanics. Our histogram shows that the vast majority of our initial response times are very low, and ensure that customers are responded to timely, ensuring the customer service desired. When ran with fewer than 8 mechanics, the mean and upper bound of the confidence interval exceeded 1 hour, leaving us with our given solution.

Simulation Mean: .9584 hours

95% Confidence Interval: [.92668, .99016]



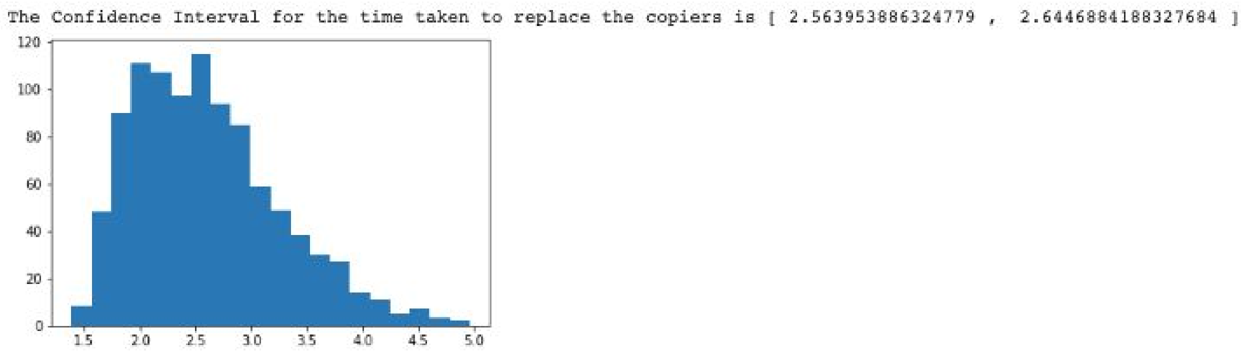
```
The Confidence Interval for initial response times is [ 0.9266850975082873 ,  0.990160611791779 ]
```

Figure: This image is a histogram displaying the distribution of all the daily averages of initial response time. The vast majority of days fall in the category of exceptional service.

For delivery time for replaced copiers it was more complex, as this metric is only collected only for the requests that require replacing a broken copier, and it features more terms to sum. It is the time interval between when a customer places the call and when a functional copier is delivered and installed for the customer, so it is the sum of response time, diagnosis time, possible van waiting time, van travel time, and installation time. To calculate this in our model, we summed the total of the five terms that make up the metric and divided it by the number of times a van had to be called. We did this by keeping a sum of those specific times, by incrementing during their calculations, and a total number of van visits counter that incremented after it was determined a van was to be called. At the end of each day we had a mean for the day and at the end of all the iterations we took the total mean and confidence interval. Our findings below support our assignment of 3 vans. By analyzing the model, we saw in scenarios with 1 or 2 vans, the delivery time shot up above 3 hours. 3 vans was the best solution to reach the goal.

Simulation Mean: 2.6043

95% Confidence Interval: [2.5639, 2.6446]



```
The Confidence Interval for the time taken to replace the copiers is [ 2.563953886324779 ,  2.6446884188327684 ]
```

Figure: This image is a histogram displaying the distribution of all the daily averages of copier installation time. It shows a majority of days fall in the category of exceptional service, and a large drop off when it came to days with high averages.

Our last two metrics were mechanic and van utilization rates. This gives us an idea of waiting times and availability of each resource within the network. These were each calculated by keeping track of the total time "worked" for each resource. Each mechanic and van had a utilization parameter that we would update for each task. In the context of mechanics, travel time, diagnosis time, and on-site repair time was counted under this metric. For vans, all travel time, installation and swapping at the business center, and swapping in a new copier at the dispatch center was included. The numbers were normalized by dividing by 24 hours for each task to get the percentage of the day worked for each task. Then, for each day we took the average across all the mechanics and all the vans respectively. And at the end of the simulation we used this list of all the averages per day to get a histogram and total average for each metric. While there was a large range of rates for mechanics, our overall mean around 62% shows that we kept a good balance of having mechanics occupied, while still keeping the arrival time low by having free mechanics at a variety of locations. Our rates for vans were much lower due to the low proportion of requests needing copier replacement. The longer overall task of the job however led us to still need 3 vans.

Simulation Mean Mechanic Utilization: 0.6215

95% Confidence Interval: [.61500, .62791]

```
The Confidence Interval for mech utilization is [ 0.615007920844697 ,  0.6279074832261002 ]
```

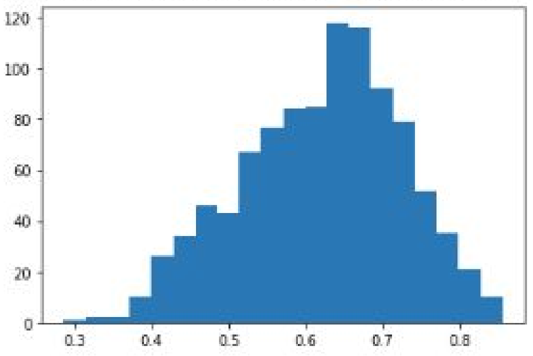


Figure: This is a histogram of average utilization rate across all mechanics on each day.

Simulation Mean Van Utilization: 0.1561

95% Confidence Interval: [.15078, .16155]

```
The Confidence Interval for van utilization is [ 0.15078043490546983 ,  0.16155822108249512 ]
```

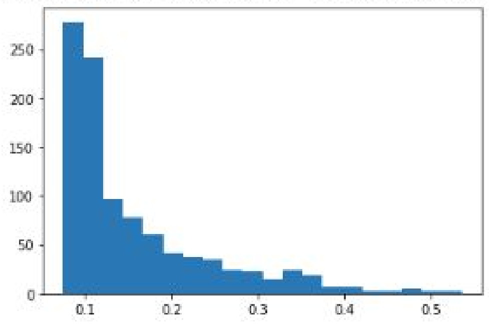


Figure: This is a histogram of average utilization rate across all vans on each day.

To confirm that our selection of 3 vans and 8 mechanics is correct, we performed a sensitivity analysis on the three distributions that we assumed - the mechanic diagnosis time distribution for business centers in group 1, the mechanic diagnosis time distribution for business centers in group 2, and the time it takes a mechanic to repair a copier if it doesn't need replacement. The tables below summarize the initial response time and delivery time confidence intervals, and the rows and columns represent changes of 10% and 25% to the parameters that the respective distributions require. For the normal distributions in Table I and Table II, the columns represent the percent change from our assumed mean and the rows represent the percent change from our assumed standard deviation. For the beta distribution in Table III, the columns represent percent change from our assumed alpha parameter (shape) and the rows represent percent change from our assumed beta parameter. The entries in the table represent the 95% confidence intervals for the initial response time and delivery time.

Original Confidence Intervals:

- CI for Initial Response Time: `[.9267, .9902]`
- CI for Delivery Time: `[2.5639, 2.6446]`

(Table I) Normal Distribution for Mechanic Diagnosis Time for Group 1 Business Centers

Std dev\Mean	-25%	-10%	0%	+10%	+25%
-25%	[0.8186, 0.8765] [2.4210, 2.4970]	[0.8594, 0.9025] [2.4721, 2.5271]	[0.8901, 0.9264] [2.5068, 2.5524]	[0.9212, 0.9532] [2.5431, 2.5828]	[0.9527, 0.9820] [2.5726, 2.6083]
-10%	[0.9337, 0.9602] [2.5559, 2.5886]	[0.9286, 0.9529] [2.5496, 2.5797]	[0.9360, 0.9589] [2.5573, 2.5855]	[0.9455, 0.9672] [2.5694, 2.5961]	[0.96081, 0.9817] [2.5839, 2.6094]
0%	[0.9510, 0.9708] [2.5739, 2.5982]	[0.9460, 0.9649] [2.5684, 2.5917]	[0.9475, 0.9657] [2.5706, 2.5929]	[0.9519, 0.9695] [2.5746, 2.5962]	[0.9615, 0.9786] [2.5849, 2.6058]
+10%	[0.9542, 0.9707] [2.5773, 2.5975]	[0.9528, 0.9688] [2.5747, 2.5942]	[0.9546, 0.9701] [2.5760, 2.5950]	[0.9568, 0.9719] [2.5785, 2.5970]	[0.9637, 0.9784] [2.5859, 2.6038]

+25%	[0.9580, 0.9724] [2.5792, 2.5967]	[0.9568, 0.9708] [2.5786, 2.5957]	[0.9575, 0.9712] [2.5804, 2.5971]	[0.9612, 0.9746] [2.5844, 2.6008]	[0.9663, 0.9796] [2.5896, 2.6057]

(Table II) Normal Distribution for Mechanic Diagnosis Time for Group 1 Business Centers

Std dev\Mean	-25%	-10%	0%	+10%	+25%
-25%	[0.7813, 0.8407] [2.3748, 2.4552]	[0.8424, 0.8863] [2.4491, 2.5058]	[0.8777, 0.9137] [2.4908, 2.5367]	[0.9300, 0.9632] [2.5449, 2.5857]	[0.9701, 1.0008] [2.5904, 2.6275]
-10%	[0.9395, 0.9670] [2.5596, 2.5932]	[0.9314, 0.9562] [2.5537, 2.5846]	[0.9364, 0.9599] [2.5583, 2.5872]	[0.9499, 0.9722] [2.5738, 2.6011]	[0.9697, 0.9911] [2.5951, 2.6212]
0%	[0.9564, 0.9766] [2.5818, 2.6065]	[0.9513, 0.9706] [2.5764, 2.6001]	[0.9538, 0.9724] [2.5796, 2.6022]	[0.9596, 0.9776] [2.5865, 2.6084]	[0.9728, 0.9904] [2.6007, 2.6219]
+10%	[0.9625, 0.9794] [2.5894, 2.6099]	[0.9586, 0.9749] [2.5863, 2.6062]	[0.9589, 0.9748] [2.5860, 2.6053]	[0.9626, 0.9781] [2.5892, 2.6080]	[0.9720, 0.9871] [2.5979, 2.6163]
+25%	[0.9648, 0.9795] [2.5908, 2.6086]	[0.9609, 0.9753] [2.5861, 2.6035]	[0.9606, 0.9746] [2.5853, 2.6023]	[0.9642, 0.9779] [2.5884, 2.6051]	[0.9720, 0.98565] [2.5963, 2.6127]

(Table III) Beta Distribution for Mechanic Repair Time

scale\shape	-25%	-10%	0%	+10%	+25%
-25%	[0.9475, 1.0145] [2.5779, 2.6586]	[1.0139, 1.0628] [2.6192, 2.6769]	[1.0589, 1.1000] [2.6575, 2.7050]	[1.1063, 1.1431] [2.6994, 2.7415]	[1.1571, 1.1911] [2.7439, 2.7823]
-10%	[1.1085, 1.1392] [2.7068, 2.7419]	[1.0860, 1.1141] [2.6875, 2.7199]	[1.0779, 1.1039] [2.6828, 2.7128]	[1.0820, 1.1065] [2.6855, 2.7137]	[1.0930, 1.1165] [2.6938, 2.7207]
0%	[1.0670, 1.0893]	[1.0544,	[0.9310,	[1.0466,	[1.0519, 1.0706]

	[2.6738, 2.6995]	1.0755] [2.6636, 2.6880]	0.9941] [2.6592, 2.6826]	1.0659] [2.6569, 2.6794]	[2.6617, 2.6835]
+10%	[1.0341, 1.0520] [2.6479, 2.6689]	[1.0237, 1.0411] [2.6400, 2.6603]	[1.0170, 1.0337] [2.6357, 2.6554]	[1.0130, 1.0293] [2.6314, 2.6505]	[1.0155, 1.0313] [2.6333, 2.6519]
+25%	[1.0022, 1.0176] [2.6224, 2.6405]	[0.9929, 1.0079] [2.6142, 2.6319]	[0.9855, 1.0001] [2.6069, 2.6242]	[0.9811, 0.9954] [2.6040, 2.6209]	[0.9806, 0.9944] [2.6026, 2.6192]

We picked 25% for the maximum percent change to a parameter since the change is big enough to cover all potential variation for the parameters. The expected value of these parameters should not vary more than 25%.

As can be seen above, when we vary the mean and standard deviation for the normal distribution representing mechanic diagnosis times, the confidence intervals for initial response time and delivery time are still within our maximum of 1 hour and 3 hours, respectively.

When we vary the shape and scale parameters in the beta distribution representing mechanic repair times, we see that our confidence intervals for delivery time are still within the maximum of 3 hours. While many of our confidence intervals for initial response time now exceed an hour, we know these percent changes for the parameters of the beta distribution are not realistic, and the response time values are only slightly above our maximum. The highest upper bound on the initial response confidence interval that exceeds an hour is around 1.19 hours. This accounts for a little less than 12 minutes over the maximum of one hour.

Given that the findings for the normal distribution are within our maximum times, the fact that the percent changes for the beta parameters are extreme, and the most extreme value is only 12 minutes above and occurs at an edge case, we conclude that our findings and recommendation of having 3 vans and 8 mechanics are valid and accurate.

Conclusion

Based on our research and in-depth analysis above, we have determined that 8 mechanics and 3 vans were the optimal and minimal number of workers needed in order to maintain competitive in our performance times as well as to minimize cost. With a yearly cost of $1,420,000, we are confident that TIDInc can maintain as a leading document management system within the Syracuse area, and can service its customers in a timely and speedy manner. We properly analyzed real test data to construct the model, we properly verified the correctness of the model, and we conducted sensitivity analysis on our results to ensure that this recommendation is accurate and correct.

Appendices:

Appendix 1: Explanation of Iterating Over Event Times

One of the key components of our model is the way that we iterate over various event times. We chose to do this by iterating over every time either a new call is made, a mechanic finishes servicing a customer, or a van finishes servicing a customer. By iterating in this way, we are ensuring that:

1. Every incoming call is immediately checked for whether or not it can be serviced. In the case that it can be serviced, we find and send the closest *currently available* mechanic out to service it. In the case that it can not be currently serviced, we add this call to the "MechBackLog" - this allows us to keep track of calls that still need a mechanic and, again, could not be initially serviced with one before proceeding.
2. Every time a mechanic finishes (each time the "CurrentEvent" is a "MechEvent") we have the mechanic that just finished check to see whether or not the MechBackLog has unserviced calls in it. In the case that it has unserviced calls in it, we have this now-free mech be immediately assigned to the first event off of the backlog - this ensures that there

is no downtime from a mech finishing to a mech servicing a call for a mech, in the case that there is a currently pending call for a mechanic.

3. Analogously to mechanics, every time a van finishes (each time the “CurrentEvent” is a “VanEvent”) we have the van that just finished check to see whether or not the VanBackLog has unserviced calls (this time, from a mechanic for a van) in it. In the case that there are unserviced calls in the van backlog queue, we have this now-free van be immediately assigned to the first event off of the backlog - this ensures that there is no downtime from a van finishing to a van servicing a call for a van, in the case that there is a currently pending call for a van.

As a result of iterating in this way, we thus ensure that there are no downtimes between major events happening throughout a given simulation period and the corresponding actions that need to take place for that event (mechanics being assigned as quickly to mechanic calls as they come in, and vans being assigned as quickly as possible to van calls when they come in.)

Appendix 2: Calculating Appropriate Mechanic Service Times

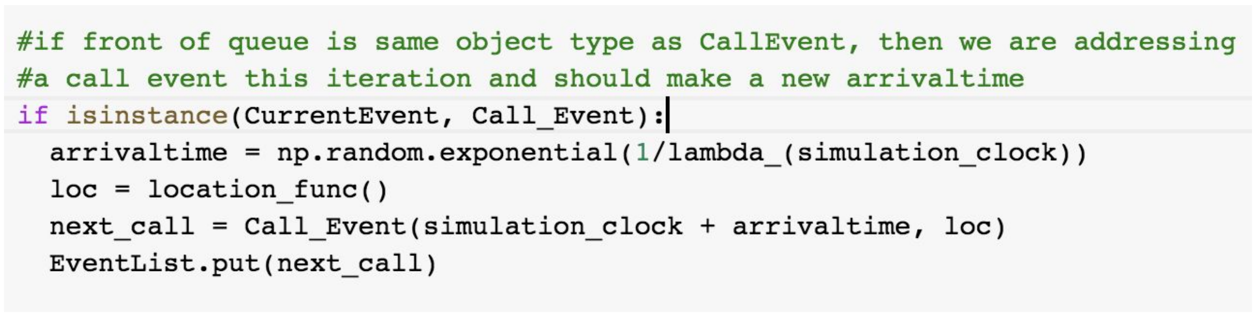
Part of our model needs to take into account whether or not a van is needed for this particular customer call. In the case that it is, would not have any onsite repair time, and in the case that it is not, we would add additional onsite repair time for the mechanic to do. The below accomplishes this logic, where we maintain for each call (referred to here as a CurrentEvent) the p value, which we calculate as soon as the call is made prior.

```
#if no van needed and mechanic does repair
if CurrentEvent.p > .1795:
  time = np.random.beta(2.617986716241127, 7.461801823981)

#if van is needed then mechanic does not do repair
else:
  time = 0
```

Appendix 3: Creating accurate interarrival times between customer calls

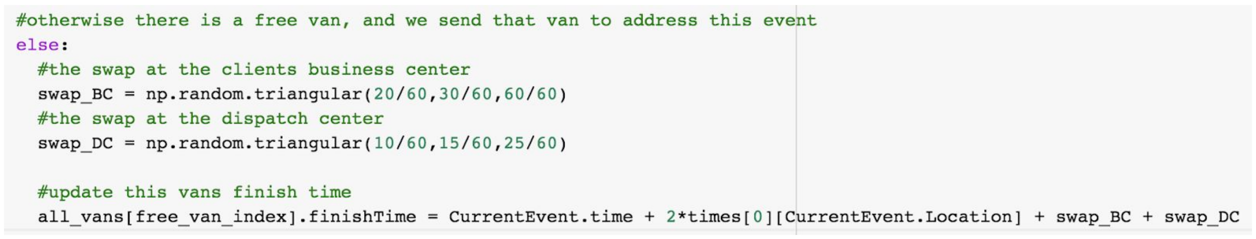
When processing a customer call, we are currently at the exact time that the customer call is happening. As such, we should schedule the next customer call to occur after the appropriate interarrival time, which is done through the calculation of interarrival time below. Then, the next customer call event will occur at simulation_clock (the current time) plus arrival time (the interarrival time between calls, dependent on the lambda function), and so we add this to the key events to be addressed in our list.



```
#if front of queue is same object type as CallEvent, then we are addressing
#a call event this iteration and should make a new arrivaltime
if isinstance(CurrentEvent, Call_Event):
  arrivaltime = np.random.exponential(1/lambda_(simulation_clock))
  loc = location_func()
  next_call = Call_Event(simulation_clock + arrivaltime, loc)
  EventList.put(next_call)
```

We then add this event of a customer arriving to our EventList as next call, meaning that when we process next_call (arrive at the time next_call happens) we update our state depending on if a mechanic can be assigned to this new call.

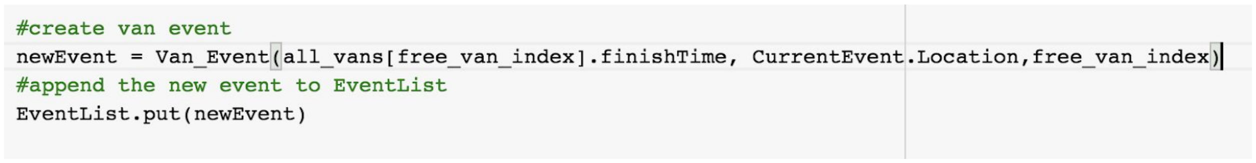
Appendix 4: Calculation of the time a van will be busy



```
#otherwise there is a free van, and we send that van to address this event
else:
  #the swap at the clients business center
  swap_BC = np.random.triangular(20/60,30/60,60/60)
  #the swap at the dispatch center
  swap_DC = np.random.triangular(10/60,15/60,25/60)

  #update this vans finish time
  all_vans[free_van_index].finishTime = CurrentEvent.time + 2*times[0][CurrentEvent.Location] + swap_BC + swap_DC
```

As mentioned, we wish for this van (referred to by maintaining the index of the free van and accessing it from the complete list of vans) to be busy until it has traveled to and from the location, completed the swap at the business center, and completed the swap at the dispatch center. Such a calculation is assigned to all_vans[free_van_index].finish, and then the time when this van will be free is added as a key event to our maintained event list with the code below.

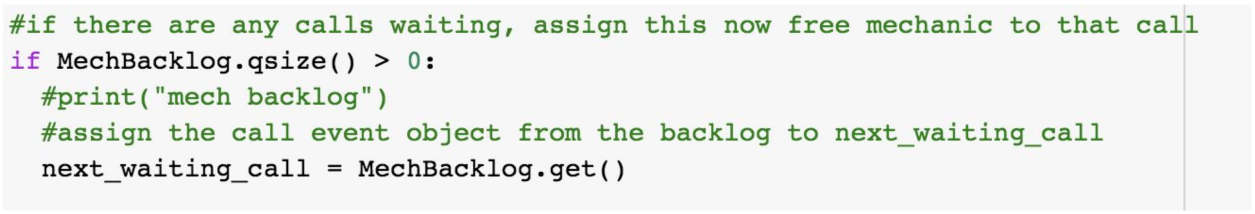


```
#create van event
newEvent = Van_Event(all_vans[free_van_index].finishTime, CurrentEvent.Location,free_van_index)
#append the new event to EventList
EventList.put(newEvent)
```

Such logic allows us to ensure that the van will not be assigned to complete a different van call during the time that it is busy (up until its finish time), and that we will reassess if this van can be assigned elsewhere once this particular call for the van is completed (as we will update our state upon arriving at this NewEvent being encountered when going over the EventList).

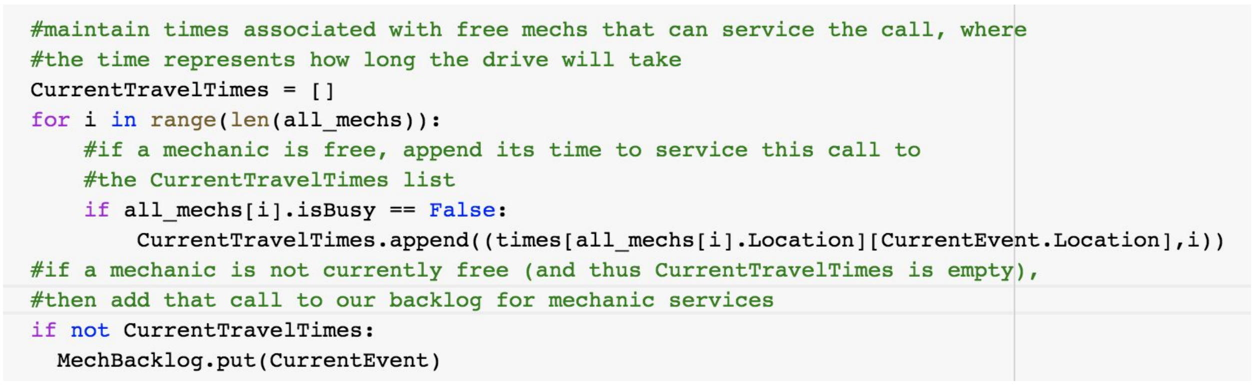
Appendix 5: Maintaining a Mechanic Backlog

When maintaining a mechanic backlog, we need to maintain a priority queue so that the earliest call added to the mechanic backlog is addressed before ones added later, and done by a mechanic as early as possible. Because MechBacklog is made as a priority queue, where calls are compared by the time they occurred, this becomes trivial, as the .get() function will allow us to pull the call that arrived the earliest easily. Furthermore, by checking right as a mech becomes free (and after we see whether or not that event they just finished servicing needs a fan) whether or not the mech backlog has any entries, we ensure that we do not waste any time assigning a mech to a service call.



```
#if there are any calls waiting, assign this now free mechanic to that call
if MechBacklog.qsize() > 0:
  #print("mech backlog")
  #assign the call event object from the backlog to next_waiting_call
  next_waiting_call = MechBacklog.get()
```

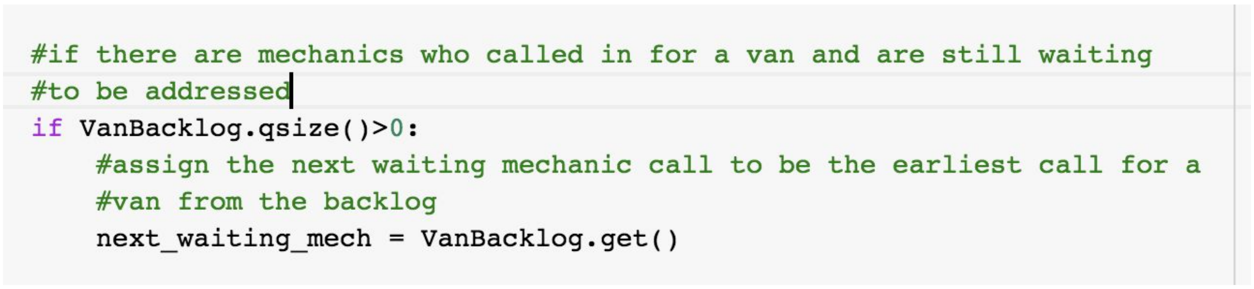
For adding the call to the MechBacklog, we check this when processing a call. If no mechs are currently available (meaning that the CurrentTravelTimes list is empty), then we add the call event to the mech backlog. This allows the call to appear in the backlog the moment that a call fails to be addressed when it comes in.



```
#maintain times associated with free mechs that can service the call, where
#the time represents how long the drive will take
CurrentTravelTimes = []
for i in range(len(all_mechs)):
    #if a mechanic is free, append its time to service this call to
    #the CurrentTravelTimes list
    if all_mechs[i].isBusy == False:
        CurrentTravelTimes.append((times[all_mechs[i].Location][CurrentEvent.Location],i))
#if a mechanic is not currently free (and thus CurrentTravelTimes is empty),
#then add that call to our backlog for mechanic services
if not CurrentTravelTimes:
  MechBacklog.put(CurrentEvent)
```

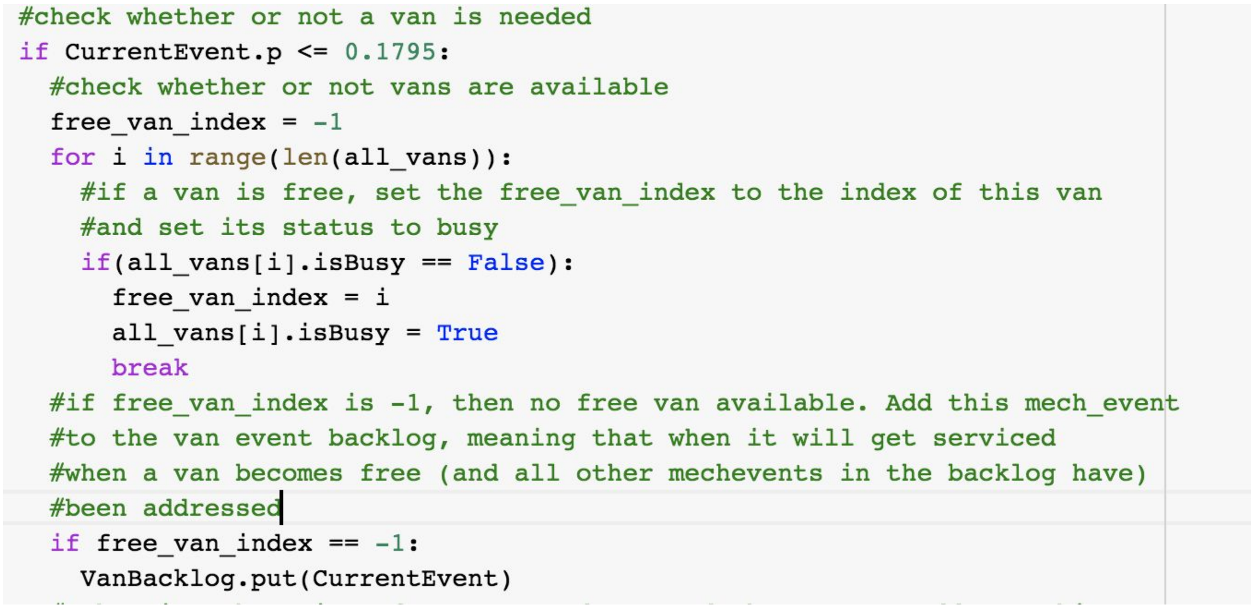
Appendix 6: Maintaining a Van Backlog

When maintaining a van backlog, we need to maintain a priority queue so that the earliest call from a mechanic added to the van backlog is addressed before ones added later, and done by a van as early as possible. Because VanBacklog is made as a priority queue, where calls are compared by the time they occurred, this becomes trivial, as the .get() function will allow us to pull the call that arrived the earliest easily. Furthermore, by checking right as a van becomes free whether or not the van backlog has any entries, we ensure that we do not waste any time assigning a van to a waiting call from a mech.



```
#if there are mechanics who called in for a van and are still waiting
#to be addressed
if VanBacklog.qsize()>0:
    #assign the next waiting mechanic call to be the earliest call for a
    #van from the backlog
    next_waiting_mech = VanBacklog.get()
```

For adding the call to the VanBacklog, we check this when processing a mechanic event (meaning a mechanic has just finished). If no vans are currently available (meaning that the free_van_index was never set to a van index, and is thus still -1), then we add the call event to the van backlog. This allows the van event to appear in the backlog the moment that a van call fails to be addressed when it comes in.



```
#check whether or not a van is needed
if CurrentEvent.p <= 0.1795:
  #check whether or not vans are available
  free_van_index = -1
  for i in range(len(all_vans)):
    #if a van is free, set the free_van_index to the index of this van
    #and set its status to busy
    if(all_vans[i].isBusy == False):
      free_van_index = i
      all_vans[i].isBusy = True
      break
  #if free_van_index is -1, then no free van available. Add this mech_event
  #to the van event backlog, meaning that when it will get serviced
  #when a van becomes free (and all other mechevents in the backlog have)
  #been addressed
  if free_van_index == -1:
    VanBacklog.put(CurrentEvent)
```