CSI 4124 / SYS 5110

Foundation of Modelling and Simulation

Final Report

Group Name:

SM OffiCe Repair

Team members and Roles

Project Manager
Elisha Pruner - 3461729

Modelling Team
Aren Lavoie - 7262357
Joseph LeFebvre - 7374469

Simulation Team
Dorukhan Basoglu- 8942175
Hashem Hashem - 8489664
Amit Bajpai - 7550759

Wednesday December 20, 2017

Problem Description

Problem Statement

SM Office Repair is a service designed by an existing office supply store to provide the very best customer support services for customers who have purchased their office equipment. Although, to date, the operation has been successful, the task of determining the staffing requirements and method of dispatching bas become a major corporate headache.

In this analysis, SM Office Repair would like us to determine the optimal staffing levels throughout the work day that maximizes employee productivity and the number of contracts meeting time restrictions. In this analysis, customer satisfaction is the percentage of service calls that are completed within the contract timeline. Currently, customers are deemed satisfied when the percentage of service calls completed within the contract is at least 85%. SM Office Repair would also like to investigate the strategies to increase the number of service calls completed within the contract timeline to 95%.

In summary, the key problems of this simulation are: **1)** hire an optimal amount of employees while still meeting the prescribed satisfaction level,and **2)** Explore staffing options to be able to respond to the desired 95% of calls within the contract timeline. Additionally, cost in this scenario can be simply viewed as the number of employees, i.e. by minimizing the number of employees who work during regular working hours, cost can be minimized . Hours worked outside of regular working hours are paid at an overtime rate of 175%, which could have a significant effect on our results.

Additionally, the management team would like to extend the analysis to additional locations throughout the service area (i.e. with different locations and other parameters) without need to create a new model & simulation for each location. This problem will not be specifically addressed during the design of the model and simulation, however if the model is sufficiently generalized it should be applicable to other locations by simply changing parameters.

SUI Details

Office Repair Organization

Equipment Types

The customer can purchase equipment type 1000, 2000, 3000, or 4000. Each equipment type has a different service contract associated with it, and the customer can buy either basic or Premium level service contracts for each equipment type.

Service Contracts

basic: The basic contract provides a 24-hour response time. This 24 hour response time is carried out during working hours, and service is guaranteed within the next working day, excluding weekends and holidays. If a customer calls in at 4 PM on Friday, the company has until 4 PM Monday to respond.

Premium: Premium customers are provided a 3-hour response time. Similar to the basic service, this 3-hour response time is guaranteed during working hours, which exclude weekends and holidays. If the customer calls in at 4 PM on Friday, the company has until 10 AM on Monday to respond.

Employee Types

EMP_T12: This type of employee is qualified to service only the type 1000 and type 2000 equipment. EMP_T12 has a salary of $26 per hour. All work done outside regular hours (8:00 am to 5:00 pm) is paid at overtime of 1.75 times regular rate. For EMP_T12, this rate is $45.5.

EMP_ALL: This type of employee is qualified to service all equipment types: type 1000, type 2000, type 3000, and type 4000 models. Since they have a greater level of expertise, they are also paid a higher salary, earning $41 per hour. All work done outside regular hours (8:00 am to 5:00 pm) is paid at overtime of 1.75 times regular rate. For EMP_ALL, this rate is $71.75.

Dispatch center

All service request are sent to a central dispatching center for processing. Consequently, once an appropriate employee is available, the employee is contacted via cell-phone technology. After the repair service is complete, the employee then informs the dispatch center that the job has been completed and that they are available for another job.

Repair service

Working Hours: SM Office Repair provides equipment repair from 8 AM to 5 PM, Monday through Friday.

Closed for service: SM Office Repair does not provide service on bank holidays or on weekends.

Dispatching a Call

Dispatcher Hours: Dispatchers are available from 8 AM to 5 PM. The dispatcher never dispatches employee after 4:30 PM. After 4:30 PM the work is assigned on the next working day.

Call Received at the Dispatching Centre : All requests for service are made to a central dispatching center. Dispatchers are the main point of contact for the customer.

Dispatching the service: If a qualified service person is currently available, the individual is called using cell-phone technology and is assigned the service call. The dispatcher never sends anyone after 4:30 to any call.

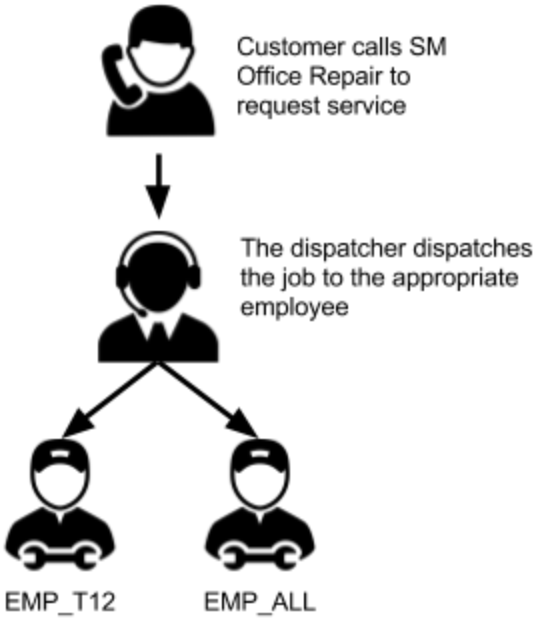


Figure 1: Dispatcher Behaviour

Servicing a Call

service time: After the dispatcher provides an employee their job, the employee travels to the customer's location and immediately starts the service.

Job Complete: When the job is complete, the employee informs the dispatcher. The dispatcher can then send the employee to the next location.

Return Visits: Very few service calls require a return visit. This only occurs 1.5 % of the time. Return visits must be carried out by the same service personnel who originally began the work, and must be scheduled on the next workday.

Servicing After Hours: If any basic services are in progress after 5 PM, the service personnel are allowed to work on those calls until they are completed or until 5:30 PM, whichever comes first. If a service call is interrupted, the service person will return at 8 AM the next workday to complete the service. Premium service calls that are in progress after 5 PM are always completed. service personnel working after 5 PM are paid overtime, at an additional 75% cost over their salary.

Lunch and Replenishing Repair Items: All employees are allocated a one-hour lunch break. service personnel never interrupt a service call for lunch, and they take their one-hour break between service calls. service personnel try to schedule their lunch between 11:30 AM and 1:30 PM. Replenishing repair items takes 10% of their idle time, and is needed to replenish their stock of repair items and other inventory that they stock.

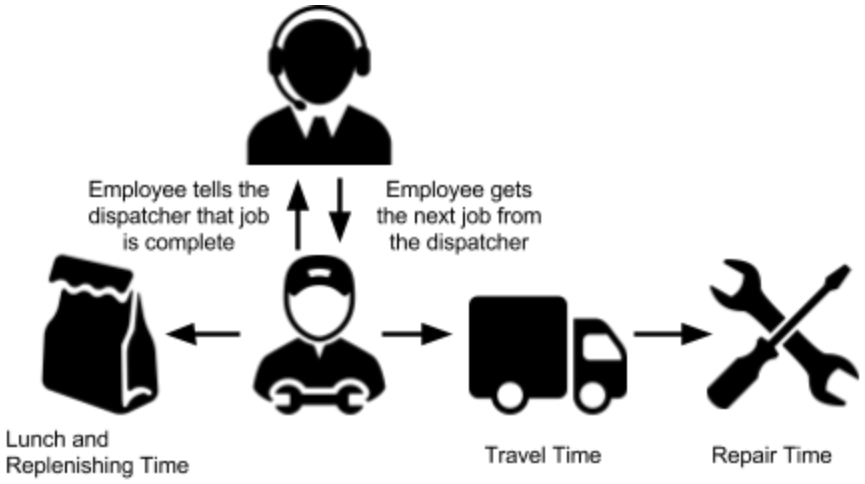


Figure 2: Employee Behaviour

Team Roles

Project Manager
Elisha Pruner

Modelling Team
Aren Lavoie
Joseph LeFebvre

Simulation Team
Dorukhan Basoglu
Hashem Hashem
Amit Bajpai

Experiment and Output Analysis
Amit Bajpai
Joseph Lefebvre
Elisha Pruner

Project Goals

the key problems of this simulation are: **1)** hire an optimal amount of employees constrained to the specified satisfaction level i.e. minimum number of employees that still meet the prescribed satisfaction level (of 85%) , **2)** Determine the desired staffing to be able to respond to the desired 95% of calls within the contract timeline.

Parameters

The parameters in our system are the two types of employees in the resource (R) **EMP_T12** and **EMP_ALL**.

- **R.Employee[EMP_T12][numEmployeesT12]**
    - Vary the number of *EMP_T12* in this resource between different experimental runs.
- **R.employee.[EMP_ALL][numEmployeesALL]**
    - Vary the number of *EMP_ALL* in this resource between different experiment runs.

Experimentation

Study:
The study will be a steady state study. Most calls will travel through the system transiently between daily hours of operations (8am to 5pm); however, some will be carried over to the next business day if the call service was not completed by 5:30.

Observation Interval:
- The time units will be minutes
- The right hand bound must be determined with experimentation.

Experiment Procedure:
Initial number of employees
- Take average service time of each type of contract (calculated from data provided)
- Multiply by the number of calls per day per type of contract
- Divide this value by 9 work day hours
- This gave us an estimate of 11 **EMP_T12** and 10 **EMP_ALL**, however to account for randomness of the system we start at 8 employees for each type.

Varying parameters
- Divide the desired satisfaction rate by the actual satisfaction rate, the satisfaction rate is calculated by dividing the number of calls which were serviced in their contract time frame by the total number of calls.
    - Denoted as *DCS / ACS - desired customer satisfaction / actual customer satisfaction*
    - This gives a ratio which we can use to increase the current number of employees by multiplying the number of employees by DCS/ACS
- First increase number of **EMP_ALL** employees until the desired CS rate for TYPE3000 and TYPE4000 calls is achieved.
    - Start with EMP_T12 = 8
    - If TYPE3000 and TYPE4000 calls are not achieving the desired customer satisfaction levels after the simulation, increase the number of **EMP_ALL** by 1. Continue increasing EMP_ALL until the satisfaction level is achieved
- Secondly, repeat the process with **EMP_T12** employees in order to obtain the optimal amount of each type of employee.
    - Start at EMP_ALL = 8
    - If TYPE1000 and TYPE2000 calls are not achieving the desired customer satisfaction levels after the simulation, increase the number of **EMP_T12** by 1. Continue increasing until the satisfaction level is achieved
- Experiments are repeated until ACS is greater than the DCS of 85% for the base case.

- Experiments are repeated again until ACS is greater than the DCS of 95% for the optimal case.
- We are tracking the fixed costs per day which is determined by the number of employees of each type. We are also tracking overtime hours by keeping tracking of the amount worked after 5:30 and accumulating the additional pay in a simple scalar output variable.

Output

satisfactionLevelT12

- Derived by dividing the number of completed contracts of Type 1000 and 2000 by the total number of contracts (i.e. iC.Call) of Type 1000 and 2000 received.

satisfactionLevelT34

- Derived by dividing the number of completed contracts of Type 3000 and 4000 by the total number of contracts (i.e. iC.Call) of Type 3000 and 4000 received.

satisfactionLevelAll

- Derived by dividing the total number of completed contract for all types of equipment by the total number contracts (i.e. iC.Call) of all types received.

averageDailyCost

- Derived by dividing the total cost of Employee hours worked and their overtime pay by the total number of days in the experiment (this is the RH bound of the system to be determined during experimentation).

ABCmod Conceptual Model

High Level Conceptual Model

Assumptions

- Assume that all premium contracts being worked on at 5:30 are completed before the start of the next day.
- Assume the number of calls per hour does not change between days, it is the same as described in *Table 1* of the *Annex.*
- The four job types are used in our model are: JOB_1000_2000_B, JOB_1000_2000_P, JOB_3000_4000_B, JOB_3000_4000_P. Jobs are put in the queue as defined in the dispatcher rules (see next point) after the customer calls in for a repair. JOB_1000_2000_B and JOB_1000_2000_P are the jobs for the type 1000 and 2000 equipment along with basic and premium versions of the contract. These two job queues can be serviced by EMP_T12 and EMP_ALL employees. JOB_3000_4000_B and JOB_3000_4000_P are the jobs for type 3000 and 4000 equipment along with basic and premium versions of the contracts. These two job queues can be serviced by EMP_ALL employees.
- The **dispatcher center rules** is the hierarchy by which calls are dispatched based on their equipment type and service type. The rules are:
    1. If Jobs.[Job_3000_4000_P] is not empty and if EMP_ALL is available, Send EMP_ALL to Job. Otherwise move on to 2.
    2. If Jobs.[Job_1000_2000_P] is not empty, and if EMP_T12 is available send EMP_T12 to job. Otherwise if EMP_ALL is available send EMP_ALL to job. Otherwise got to 3.
    3. If Jobs.[Job_3000_4000_B] is not empty, and if EMP_ALL is available then Send EMP_ALL to Job. Otherwise go to 4.
    4. If Jobs.[Job_1000_2000_B] is not empty, and if Employee EMP_T12 is available, Send EMP_T12 to Job. Otherwise if EMP_ALL is available, Send EMP_ALL to job.
- We are not concerned with optimizing dispatching problem (which can introduce travelling salesman fracets to our model). All the dispatching is done with above-mentioned rules.

Simplifications

- When Time reaches 5:30 all non-premium (basic) calls are interrupted and completed the next day at 8 am. It is assumed that they leave early enough to get there at exactly 8 am, i.e. do not need to model a second travel time.
- The characteristic that replenishing supplies consumes 10% of idle time will not be modelled because idle time does not have any effect on the behavior or the output of the model.
- Return visits are defined as calls that were completed but the client has a problem and a return visit is required. These occur very infrequently (1.5% of calls) and we will not include them in the model as they should not impact the optimal outputs values of experimentation.

Structural View

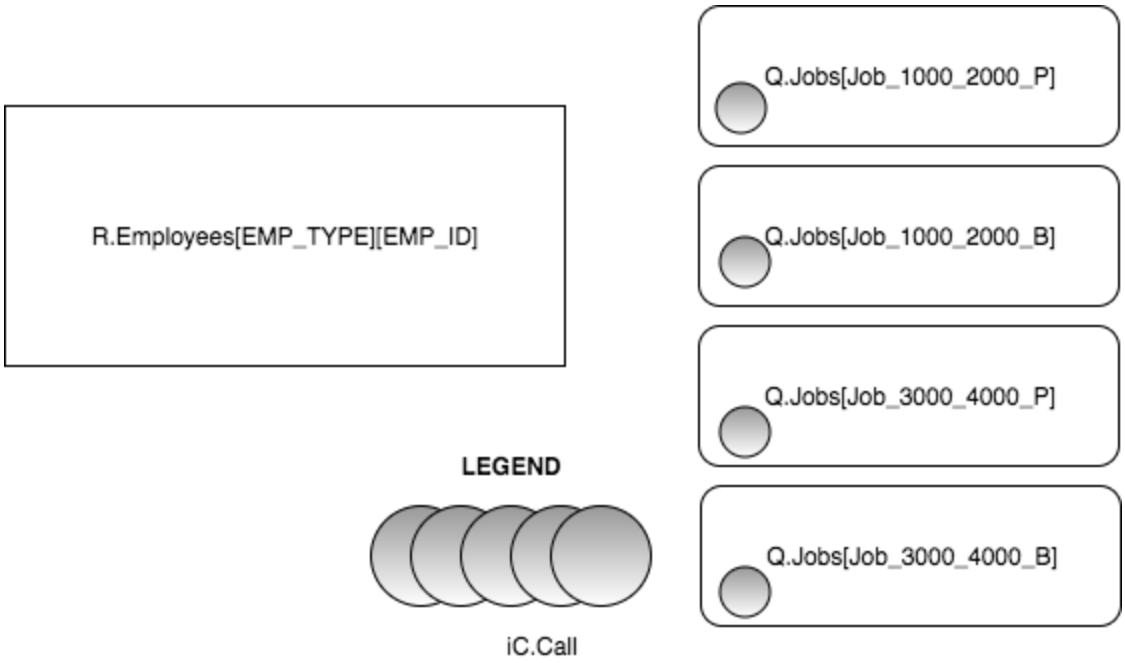


Figure 3: Structural View

Entity Structures

iC.Call

- Represents a request from a client for a service call

Resource Employees with scope Set

- The resource Employees is a set with 2D structure and two identifiers : the first to identify the type of employee, and the second that identifies the specific employee of that type.

Queue Jobs with scope Set

- The set contains 4 different queues to hold instances of iC.Call which are ready to be served by an employee
- The four queues are
    - Q.Jobs[Job_1000_2000_P]
    - Q.Jobs[Job_1000_2000_B]
    - Q.Jobs[Job_3000_4000_B]
    - Q.Jobs[Job_3000_4000_P]
- Based on the decision made by the Dispatcher rules (see *Assumptions)*, an incoming instance of call will be placed in queue based on iC.Call.EquipmentType and iC.Call.serviceType.

Behavioural View

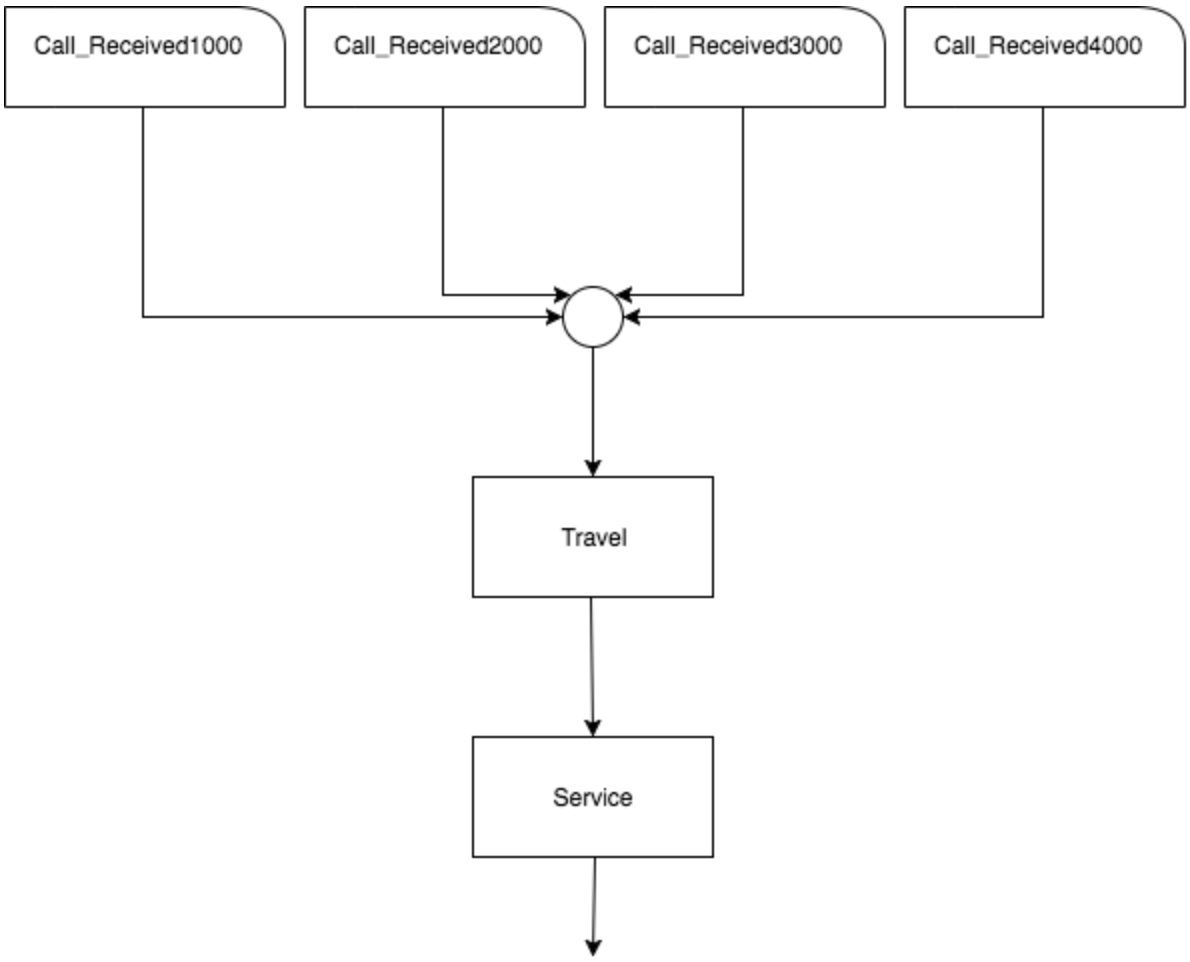


Figure 4: Life-cycle diagram of a call

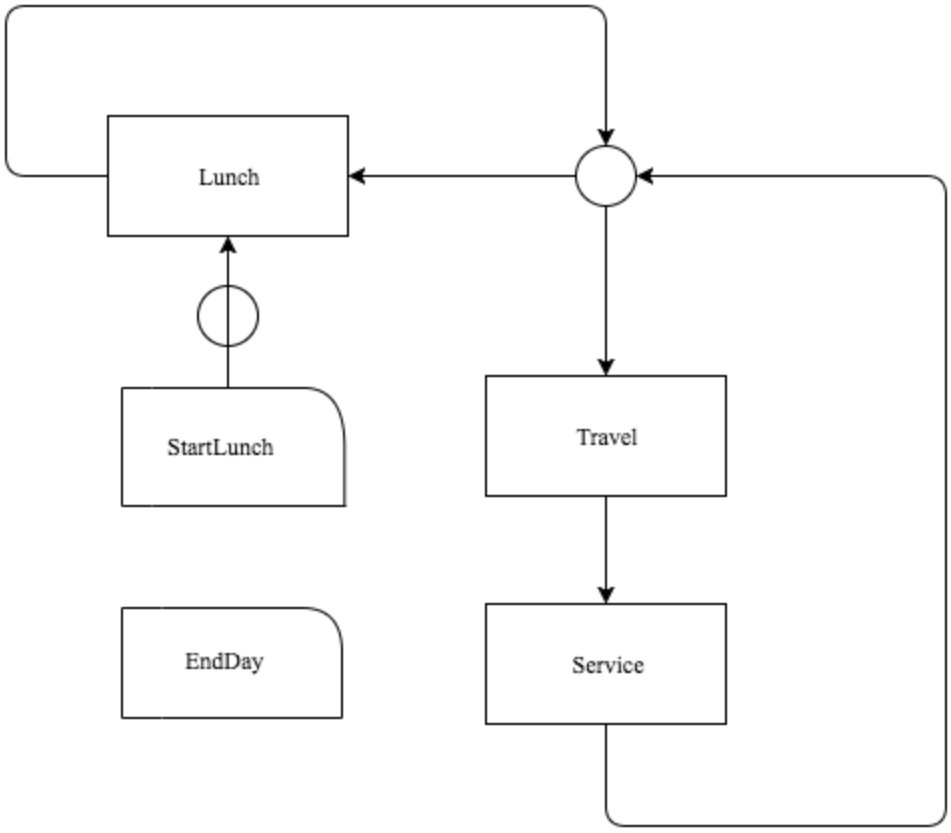


Figure 5: Life Cycle of Employee

Scheduled Action constructs

Call_Received1000: A service call for equipment type 1000 has arrived in the system and is instantly placed into a Job queue by the dispatcher rules.

Call_Received2000: A service call for equipment type 2000 has arrived in the system and is instantly placed into a Job queue by the dispatcher rules.

Call_Received3000: A service call for equipment type 3000 has arrived in the system and is instantly placed into a Job queue by the dispatcher rules.

Call_Received4000: A service call for equipment type 4000 has arrived in the system and is instantly placed into a Job queue by the dispatcher rules.

StartLunch: Executed at the 11:30AM every day. This action signals when employees can start going to lunch.

EndDay: Executed at the end of every day at midnight. At EndDay, reset the employee.lunchTaken to false, and calculate the fixed total cost and the average daily cost. Based on the data files, the latest time for a premium service along with its travel time does not go past 8pm at night. So executing EndDay every midnight will not cause any conflicts.

Sequel Activity Constructs

Service: Employee repairing the equipment on location, the duration of this event is modeled by the data in Annex.

Conditional Activity constructs

Travel: This conditional activity begins when an available Employee is assigned to a service call. The activity determines the travel time to get to the customer. When the travel time is complete, it executes the sequel active service.

Lunch: This conditional activity relieves an Employee for lunch when they become idle for the first time after 11:30AM each day.

Input

Exogenous Input (Entity Stream)			
Variable Name	**Description**	**Domain Sequence**	**Range Sequence**
uCallArrival1000	Input entity stream for equipment 1000 call arrival time	RVP.DuCallArrival1000, arrival of calls modelled using the exponential distribution	N/A, only one call is answered at a time
uCallArrival2000	Input entity stream for equipment 2000 call arrival time	RVP.DuCallArrival12000, arrival of calls modelled using the exponential distribution	N/A, only one call is answered at a time
uCallArrival3000	Input entity stream for equipment 3000 call arrival time	RVP.DuCallArrival3000, arrival of calls modelled using the exponential distribution	N/A, only one call is answered at a time
uCallArrival4000	Input entity stream for equipment 4000 call arrival time	RVP.DuCallArrival4000, arrival of calls modelled using the exponential distribution	N/A, only one call is answered at a time
Endogenous Input (Semi-Independent)			
Variable Name	**Description**	**Value(s)**	**Range Sequence**
uTravelTime	Time the employee takes to reach the client	RVP.uTravelTime(), models the travel time	
userviceType	Contract type purchased by the customer	RVP.userviceType(equipType) returns the contract type <BASIC, PREMIUM>	

userviceTime	Time it takes to service the equipment	RVP.userviceTime1000(equipType) returns the service time in minutes for equipment type <TYPE1000, TYPE2000, TYPE3000, TYPE4000>
uEndDay	When the current day is scheduled to end	DVP.EndDay() returns the time scheduled time for ending the day.

Detailed Conceptual Model

Structural Components

Constants		
Name	**Description**	**Value**
EMP_T12_HOURLY_WAGE	Hourly wage paid to EMP_T12 employees	$26
EMP_ALL_HOURLY_WAGE	Hourly wage paid to EMP_ALL employees	$41
EMP_T12_OVERTIME_WAGE	Hourly overtime wage for EMP_T2	$45.5
EMP_ALL_OVERTIME_WAGE	Hourly overtime wage to EMP_ALL	$71.75
EMPLOYEE_T12	Identifier for resource set to indicate EMP_T12 type employees	0
EMPLOYEE_ALL	Identifier for resource set to indicate EMP_ALL type employees	1
Job_1000_2000_P	Identifier for resource set to indicate Job_1000_2000_P type jobs	0
Job_1000_2000_B	Identifier for resource set to indicate Job_1000_2000_B type jobs	1
Job_3000_4000_P	Identifier for resource set to indicate Job_3000_4000_P type jobs	2
Job_3000_4000_B	Identifier for resource set to indicate Job_3000_4000_B type jobs	3
LUNCH_DURATION	The duration of lunch given to employees	60 minutes
Parameters		
Name	**Description**	**Values**
numEmployeesT12	Number of employees that can repair equipment type 1000 and 2000	Calculated initial value to number req. for valid results

numEmployeesALL	Number of employees that can repair ALL types of equipment	Calculated initial value to number req. for valid results
satisfactionLevel	The satisfaction level defined by the experiment	The first experiment runs with a satisfaction level of 85%. The second experiment runs with a satisfaction level of 95%

Resource Set[2][maxEmployees]: *Employees*	
This Resource Set represents all the employees in our system. The set has two identifies , *EMP_TYPE* and *EMP_ID*. The EMP_TYPE identifier has values <EMP_T12, EMP_All> and determines what kind of equipment the employees can respond to. The EMP_ID identifier distinguishing between individual employee. **maxEmployees is the max of numEmployeesT12 and numEmployeeALL	
Attributes	Description
status	Represents the current status of the employee. It is an enum with options {READY_FOR_CALL, TAKING_LUNCH, SERVICING_CALL}
lunchTaken	Whether or not the employee has taken lunch at the time

Queue/Set[4] : *Jobs*	
Each job queue stores a particular call received depending on the type of equipment and contract type.. Set identifier determines EquipmentType and serviceType of the Job queue. Identifiers allowed are <Job_1000_2000_P, Job_1000_2000_B, Job_3000_4000_P, Job_3000_4000_B>	
Attributes	Description
list	List of jobs of a specifiC contract type and equipment type (determined by set identifier) that are waiting to be responded to by employee
n	The number of jobs in each list

Consumer Class : *Call*	
A call is a surrogate for the customer. This is the fundamental and the only consumer class in the model.	
Attributes	Description
equipmentType	Denotes the type of Equipment that must be repaired. The values of the attribute

	can be <*TYPE1000, TYPE2000, TYPE3000, TYPE4000*>.
serviceType	Denotes the quality of service provided to the customer. Directly affects the response time. The attribute can have values of <BASIC, PREMIUM>.
timeIn	Time that the call was entered into the system.

Behavioural Components

Time units: Minutes

Observation interval: Steady state study - starts at 0 and the right hand end of the observation interval is to be determined during experimentation.

Action: *Initialise*	
TimeSequence	*<0>*
Event SCS	R.Employees[EMP_T12][EMP_ID].status = READY_FOR_CALL R.Employees[EMP_ALL][EMP_ID].status = READY_FOR_CALL Q.Jobs[Job_1000_2000_P] ← 0 Q.Jobs[Job_1000_2000_B] ← 0 Q.Jobs[Job_3000_4000_P] ← 0 Q.Jobs[Job_3000_4000_P] ← 0 SSOV.contractsT12satisfied ← 0 SSOV.totalNumberT12Contracts ← 0 SSOV.contractsT34satisfied ← 0 SSOV.totalNumberT34Contracts ← 0 SSOV.fixedTotalCost ← 0 SSOV.overtimeCost ← 0 SSOV.averageDailyCost ← 0

Output

Output	
Simple Scalar Output Variables (SSOV's)	
Name	**Description**
contractsT12satisfied	Number of contracts of type 1000 and type 2000 that have been completed on time
totalNumberT12Contracts	Total number of contracts of equipment type 1000 and 2000 that have entered the system.
contractsT34satisfied	Number of contracts of type 3000 and type 4000 that have been completed on time
totalNumberT34Contracts	Total number of contracts of equipment type 3000 and 4000 that have entered the system.
overtimeCost	Keeps track of the total overtime cost. It is indifferent of when it occurred or which employee received the pay. Simply increased at the end of the servicing activity if conditions are met.
getAverageDailyCost()	The average cost to run the office repair service each days. Returns : (fixedTotalCost + overtimeCost)/totalNumberDays fixedTotalCost = ((8hrs * numEmployessT12 * EMP_T12_HOURLY_WAGE) +(8hrs * numEmployeeAll * EMP_ALL_HOURLY_WAGE)) *totalNumberDays totalNumberDays = Current_time div (Integer division) 1440
getSatisfactionLevelT12	contractsT12satisfied / totalNumberT12Contracts
getSatisfactionLevelT34	contractsT34satisfied / totalNumberT34Contracts
getSatisfactionLevelALL	Total contracts completed of all types / total calls received of all types: satisfactionLevelAll = (contractsT12satisfied + contractsT34satisfied) / (totalNumberT12Contracts + totalNUmberT12Contracts)
getOvertimeCost()	Over time cost incurred till time t (overtimeCost)

User Defined Procedures

User-Defined Procedures	
Name	**Description**
GetEmployeeForLunch()	Returns the two identifiers <EMP_TYPE, EMP_ID> from the Employees resource set for an employee who is eligible to take lunch.
CanStartLunch()	Returns True as long as long as an employee can start lunch.
ReadyToTakeCall()	Returns true if there is an employee available and a compatible call in the a queue that the employee can service. In other words , if there is a non-empty job queue (i.e. has a call in it) and an employee who can service this call, return true. Otherwise return false.

	**Note: the UDP should also return false if the time is greater than 4:30, because no employees are dispatched after 4:30.
GetEmployeeForCall()	Returns the two identifiers <EMP_TYPE,EMP_ID> for Employees and the identifier for Jobs. This gives an employee and the queue which contains the next call to be serviced. This procedure implements the dispatcher rules, sending employees to jobs in the specified priority. *SEE HLCM - Simplications and Assumptions for priority.* .
UpdateContractSSOVs(iC.Call)	Updates all the SSOV output variables that describe number of contracts and number of contracts satisfied. Specifically, it updates SSOV.contractsT12satisfied, SSOV.totalNumberT12Contracts, SSOV.contractsT34satisfied, SSOV.totalNumberT34Contracts.
UpdateOvertimeSSOVs(etypeId)	Updates SSOV.overtimeCost based on the current time and the wage of the type of employee that was servicing the call. Since this UDP is called from terminating event of the service activity, the current time will represent the time at which the call finishes. **In the case that calls are completed the next day, i.e. they were not be finished before 5:30, 30 minutes of overtime of the appropriate wage is added.
ComputeServiceDuration(icCall, etypeId)	This UDP returns the service time of a call given by RVP.uSericeTime(equipmentType) UNLESS the call is of type BASIC and will not be completed before 5:30, then the service time duration is extended to include an overnight idle time. This UDP also adds 30 minutes of overtime of the appropriate wages to SSOV.overtimeCost, because 30 minutes of overtime will be incurred from 5:00 to 5:30.
StartIdleEmpLunch()	UDP starts a new lunch activity for every employee with status of READY_FOR_CALL (i.e. idle)

Deterministic Value Procedures

Deterministic Value Procedures	
Name	**Description**
DVP.EndDay()	Gives the time of the EndDay action which occurs every night at midnight
DVP.StartLunc()	GIves the time of the next StartLunch action which occurs every day at 11:30

Input Constructs

Random Variate Procedures		
Name	**Description**	**Data Model**
RVP.uServiceType (equipType)	Returns the service type <BASIC, PREMIUM> the call based on the equipment type parameter has values <TYPE1000, TYPE2000, TYPE3000, TYPE4000>	FIXED PROBABILITY Type / Basic / Premium TYPE1000 35% 65% TYPE2000 35% 65% TYPE3000 25% 75% TYPE4000 15% 85%

RVP.uServiceTime (equipType)	Gives a random service time for the time to repair the equipment for each call	Gaussian distribution from the input Type1000.dat, Type2000.dat, Type3000.dat, Type4000.dat, where the mean and standard deviation is:
		
		meanSrvTm1000 = 14.79;
		meanSrvTm2000 = 44.36;
		meanSrvTm3000 = 60.27;
		meanSrvTm4000 = 130.17;
		
		stdDevSrvTm1000 = 5.85;
		stdDevSrvTm2000 = 15.06;
		stdDevSrvTm3000 = 27.40;
		stdDevSrvTm4000 = 29.84;

RVP.uTravelTime()	Gives the travel time to a service call	Triangular distribution where
		minTravelTm = 10;
		maxTravelTm = 45;
		meanTravelTm = 30;

RVP.DuCallArrival1000( )	Gives the arrival time for service calls.	Exponential distribution, Exponential(MEAN) where MEAN is based on the time of day: 8AM - 9AM: MEAN = (60 min/h) / (7 calls/h) = 8.57 for $0 \leq (t \% 1440) < 60$ 9AM - 10AM: MEAN = (60 min/h) / (12 calls/h = 5.0 for $60 \leq (t \% 1440) < 120$ 10AM - 11AM: MEAN = (60min/h) / (10 calls/h) = 6.0 for $120 \leq (t \% 1440) < 180$ 11AM - 12PM: MEAN = (60 min/h) / (7 calls/h) = 8.57 for $180 \leq (t \% 1440) < 240$ 12 PM - 1PM:

		MEAN = (60 min/h) / (5 calls/h) = 12.0 for 240 ≤ (t % 1440) < 300 <u>1PM - 2PM:</u> MEAN = (60 min/h) / (4 calls/h) = 15.0 for 300 ≤ (t % 1440) < 360 <u>2PM - 3M:</u> MEAN = (60 min/h) / (5 calls/h = 12.0 for 360 ≤ (t % 1440) < 420 <u>3PM - 4PM:</u> MEAN = (60 min/h) / (4 calls/h) = 15.0 for 420 ≤ (t % 1440) < 480 <u>4PM - 5PM:</u> MEAN = (60 min/h) / (3 calls/h) = 20.0 for 480 ≤ (t % 1440) < 540
RVP.DuCallArrival2000( )	Gives the arrival time for service calls.	Exponential distribution, Exponential(MEAN) where MEAN is based on the time of day: <u>8AM - 9AM:</u> MEAN = (60 min/h) / (8 calls/h) = 7.5 for 0 ≤ (t % 1440) < 60 <u>9AM - 10AM:</u> MEAN = (60 min/h) / (11 calls/h = 5.45 for 60 ≤ (t % 1440) < 120 <u>10AM - 11AM:</u> MEAN = (60min/h) / (8 calls/h) = 7.5 for 120 ≤ (t % 1440) < 180 <u>11AM - 12PM:</u> MEAN = (60 min/h) / (9 calls/h) = 6.66 for 180 ≤ (t % 1440) < 240 <u>12 PM - 1PM:</u> MEAN = (60 min/h) / (6 calls/h) = 10.0 for 240 ≤ (t % 1440) < 300 <u>1PM - 2PM:</u> MEAN = (60 min/h) / (4 calls/h) = 15.0 for 300 ≤ (t % 1440) < 360 <u>2PM - 3M:</u> MEAN = (60 min/h) / (3 calls/h = 20.0 for 360 ≤ (t % 1440) < 420 <u>3PM - 4PM:</u> MEAN = (60 min/h) / (3 calls/h) = 20.0 for 420 ≤ (t % 1440) < 480 <u>4PM - 5PM:</u> MEAN = (60 min/h) / (2 calls/h) = 30.0 for 480 ≤ (t % 1440) < 540
RVP.DuCallArrival3000( )	Gives the arrival time for service calls.	Exponential distribution, Exponential(MEAN) where MEAN is based on the time of day: <u>8AM - 9AM:</u> MEAN = (60 min/h) / (5 calls/h) = 12.0 for 0 ≤ (t % 1440) < 60 <u>9AM - 10AM:</u> MEAN = (60 min/h) / (6 calls/h = 10.0 for 60 ≤ (t % 1440) < 120 <u>10AM - 11AM:</u> MEAN = (60min/h) / (5 calls/h) = 12.0

		for 120 ≤ (t % 1440) < 180 11AM - 12PM: MEAN = (60 min/h) / (4 calls/h) = 15.0 for 180 ≤ (t % 1440) < 240 12 PM - 1PM: MEAN = (60 min/h) / (3 calls/h) = 20.0 for 240 ≤ (t % 1440) < 300 1PM - 2PM: MEAN = (60 min/h) / (3 calls/h) = 20.0 for 300 ≤ (t % 1440) < 360 2PM - 3M: MEAN = (60 min/h) / (2 calls/h = 30.0 for 360 ≤ (t % 1440) < 420 3PM - 4PM: MEAN = (60 min/h) / (2 calls/h) = 30.0 for 420 ≤ (t % 1440) < 480 4PM - 5PM: MEAN = (60 min/h) / (1 calls/h) = 60.0 for 480 ≤ (t % 1440) < 540
RVP.DuCallArrival4000( )	Gives the arrival time for service calls.	Exponential distribution, Exponential(MEAN) where MEAN is based on the time of day: 8AM - 9AM: MEAN = (60 min/h) / (2 calls/h) = 30.0 for 0 ≤ (t % 1440) < 60 9AM - 10AM: MEAN = (60 min/h) / (3 calls/h = 20.0 for 60 ≤ (t % 1440) < 120 10AM - 11AM: MEAN = (60min/h) / (4 calls/h) = 15.0 for 120 ≤ (t % 1440) < 180 11AM - 12PM: MEAN = (60 min/h) / (3 calls/h) = 20.0 for 180 ≤ (t % 1440) < 240 12 PM - 1PM: MEAN = (60 min/h) / (2 calls/h) = 30.0 for 240 ≤ (t % 1440) < 300 1PM - 2PM: MEAN = (60 min/h) / (1 calls/h) = 60.0 for 300 ≤ (t % 1440) < 360 2PM - 3M: MEAN = (60 min/h) / (1 calls/h = 60.0 for 360 ≤ (t % 1440) < 420 3PM - 4PM: MEAN = (60 min/h) / (1 calls/h) = 60.0 for 420 ≤ (t % 1440) < 480 4PM - 5PM: MEAN = (60 min/h) / (1 calls/h) = 60.0 for 480 ≤ (t % 1440) < 540

Action: Call_Received1000

A call from a customer is received that is of equipment type 1000 and service type basic or Premium	
TimeSequence	RVP.DuCallArrival1000()
Event SCS	iC.Call ← SP.Derive(Call) iC.Call.equipmentType ← TYPE1000 iC.Call.serviceType ← RVP.userviceType(TYPE1000) iC.Call.timeIn ← t IF iC.Call.serviceType == basic Then SP.insertQue(Job_1000_2000_B, iC.Call) Else SP.insertQue(Job_1000_2000_P, iC.Call)

Action: Call_Received2000	
A call from a customer is received that is of equipment type 2000 and service type basic or Premium	
TimeSequence	RVP.DuCallArrival2000()
Event SCS	iC.Call ← SP.Derive(Call) iC.Call.equipmentType ← TYPE2000 iC.Call.serviceType ← RVP.userviceType(TYPE2000) iC.Call.timein ← t IF iC.Call.serviceType == basic Then SP.insertQue(Job_1000_2000_B, iC.Call) Else SP.insertQue(Job_1000_2000_P, iC.Call)

Action: Call_Received3000	
A call from a customer is received that is of equipment type 3000 and service type basic or Premium	
TimeSequence	RVP.DuCallArrival3000()
Event SCS	iC.Call ← SP.Derive(Call) iC.Call.equipmentType ← TYPE3000 iC.Call.serviceType ← RVP.userviceType(TYPE3000) iC.Call.timein ← t IF iC.Call.serviceType == basic Then SP.insertQue(Job_3000_4000_B, iC.Call) Else SP.insertQue(Job_3000_4000_P, iC.Call)

Action: Call_Received4000
A call from a customer is received that is of equipment type 4000 and service type basic or Premium

TimeSequence	RVP.DuCallArrival4000()
Event SCS	iC.Call ← SP.Derive(Call) iC.Call.equipmentType ← TYPE3000 iC.Call.serviceType ← RVP.userviceType(TYPE4000) iC.Call.timein ← t IF iC.Call.serviceType == basic Then SP.insertQue(Job_3000_4000_B, iC.Call) Else SP.insertQue(Job_3000_4000_P, iC.Call)

Action: *EndDay*	
This scheduled action construct progresses the simulation to the next work day. Gives a value to the semi-independent variable *uEndDay*.	
TimeSequence	*DVP.EndDay()*
Event SCS	For each R.Employees[EMP_T12][EMP_ID] **then** R.Employees[EMP_T12][EMP_ID].lunchTaken ← false For each R.Emplyees[EMP_ALL][EMP_ID] **then** R.Employees[EMP_ALL][EMP_ID].lunchTaken ← false

Action: *StartLunch*	
This scheduled action construct sends all currently Idle employees on their lunch break. This action is triggered at 11:30 AM every day.	
TimeSequence	*DVP.StartLunch()*
Event SCS	UDP.SendIdleEmpLunch() *(Start a new Lunch activity for every Idle Employee)*

Behavioural Constructs

Activity: *Lunch*	
The employee takes their lunch break	
Precondition	UDP.CanStartLunch() (*return true as long as an employee can start lunch)*

Event SCS	<etypeid, empid> ← UDP.GetEmployeeForLunch() R.Employees[etypeId][empId].status ← TAKING_LUNCH
Duration	*LUNCH_DURATION*
Event SCS	R.Employees[etypeId][empId].status = READY_FOR_CALL R.Employee[etypeId][empId].lunchTaken = true

Activity: *Travel*	
The employee travels to the next customer	
Precondition	UDP.ReadyToTakeCall() (returns True if there is an appropriate employee available to respond to a call in any of the Job queues)
Event SCS	<etypeId, empId, jobQueueIdent> ← UDP.GetEmployeeForCall() R.Employees[etypeId][empId].status ← SERVICING_CALL iC.Call ← SP.Remove(Jobs[JobQueueIdent]) (*remove Call from Job Queue and store locally, in Travel so that it can be passed to the sequel activity SERVICE)*
Duration	RVP.uTravelTime()
Event SCS	SP.Start(Service, etypeId, empId, iC.Call) *(Start sequal activtiy service)*

Activity: *Service*	
This activity is initiated after an employee has completed their travel time and has arrived at the client's location. The employee will now service the equipment	
Causal	(etypeId, empId, iC.Call)
Event SCS	

Duration	UDP.ComputeServiceDuration(icCall, etypeId) (*The time it takes the employee to actually perform the repair*)
Event SCS	*(etypeId, empId, iC.Call are received from Travel)* UDP.UpdateContractSSOVs(iC.Call) UDP.UpdateOvertimeSSOV(eTypeId) (*Free the employee)* R.Employees[etypeId][empId].status ← READY_FOR_CALL

Design of Validation Experimentation

Given the simpliCity of the model, it is possible to validate the model using a trace log as described below. The model shall be validated for both the base case and the alternate case.

Trace Logging

The state of the simulation model is monitored by tracking the status of satisfactionLevelT12, satisfactionLevelT34, and satisfactionLevelAll. These values are presented on a single line as follows:

```
Clock: XXXXXX
Number of T12 Employees: XX, Number of ALL Employees: XX
T12Satisfied: XXXX, totalT12: XXXX, T34Satisfifed XXXX, totalT34: XXXX
SatisfactionLevelT12: XX, SatisfactionLevelT34: XX, SatisfactionLevelAll: XX
```

where the xx's are replaced with current values of the clock and attributes accordingly.

Simulation Model

Design of Simulation Model and Program

The simulation model is implemented in the class `OfficeRepair` (an extension of the ABSmod/J class `AOSimulation` model) and a number of other classes used to implement the various constructs from the ABCmod conceptual model. All Java classes that make up the Java Office Repair simulation model are placed in the Java package `simModel`.

The following table shows how the various ABCmod entity structures are mapped to Java classes and how objects instantiated from these classes are reference by the `OfficeRepair` class.

Entity Structures		
ABCmod Construct	**Java Class**	**Object References**
iC.Call	`Call` The enumerated data type equipmentType defines the TYPE1000, TYPE2000, TYPE3000, TYPE4000 equipment. The enumerated data type serviceType defines the service as either basic or PREMIUM	Typically by the reference variable iC.Call in the various methods that manipulate Call objects
R.Employee	`Employee` Implemented as a 2D array of objects Employee[empType, empID]	simModel.rEmployee
Q.Jobs	`ArrayList` (Standard Java Class) Notes: - The various methods available in the ArrayList class provide the implementation of the various ABCmod procedures, such as SP.InsertQue (qCustLine.add) SP.RemveQue (qCustLine.remove).	simModel.qJobs

	- The attribute n is maintained within the ArrayList object (adjusted automatically when ArrayList methods are called). The method qJobs.size() provides the value of the Q.Jobs.n attribute.	

The following table provides mapping between the conceptual model Action/Activities to Java classes.

Actions / Activities	
ABCmod Constructs	**Java Classes**
Call_Recieved1000	Call_Recieved1000
Call_Recieved2000	Call_Recieved2000
Call_Recieved3000	Call_Recieved3000
Call_Recieved4000	Call_Recieved4000
EndDay	EndDay
Initialise	Initialise
Lunch	Lunch
Service	Service
StartLunch	StartLunch
Travel	Travel

Results of the Validation Experimentation

The java class Experiment1.java contains the code to generate output logs of both the SBL trace overtime as well as the outputs of 10 different runs. To demonstrate validation two cases of Experiment1 were run: **1)** 6 EMP_T12 employees and **2)** 6 EMP_ALL employees. The following partial log files demonstrate the proper expected behaviour of the model. However the

entire log files can be found in the attached SBLTrace_Experiment1_XXXX.txt and Outputs_Experiment1_XXXX.txt files.

The beginning of the log file shows that all output variables are initially zero, as well as the first calls of different types beginning to arrive. The two scheduled actions of the model, StartLunch, EndDay have also been scheduled to their proper first occurence, 11:30 and Midnight, respectively. A stop condition is also scheduled at one week for validation purposes.

---------- Clock: 0.0 Number of T12 Employees: 6, Number of ALL Employees: 6 T12Satisfied: 0, totalT12: 0, T34Satisfifed 0, totalT34: 0 SatisfactionLevelT12: 0%, SatisfactionLevelT34: 0%, SatisfactionLevelAll: 0% ----------SBL---------- TimeStamp:0.2514024604655989 Activity/Action: simModel.Call_Received3000 TimeStamp:2.0220795029543703 Activity/Action: simModel.Call_Received1000 TimeStamp:4.530657231181888 Activity/Action: simModel.Call_Received2000 TimeStamp:16.638200040035432 Activity/Action: simModel.Call_Received4000 TimeStamp:210.0 Activity/Action: simModel.StartLunch TimeStamp:960.0 Activity/Action: simModel.EndDay TimeStamp:10080.0 Stop Notification ---------- ----------

At the beginning of the day, most of the employees are going to be free and therefore after 2 minutes the first travel activities begin to appear as employees leave to respond to service cals. Note the end of the travel activity now appears on the SBL trace.

Clock: 2.0220795029543703
Number of T12 Employees: 6, Number of ALL Employees: 6
T12Satisfied: 0, totalT12: 0, T34Satisfifed 0, totalT34: 0
SatisfactionLevelT12: 0%, SatisfactionLevelT34: 0%, SatisfactionLevelAll: 0%
----------SBL----------
TimeStamp:4.530657231181888 Activity/Action: simModel.Call_Received2000
TimeStamp:5.600690490384437 Activity/Action: simModel.Call_Received1000
TimeStamp:16.638200040035432 Activity/Action: simModel.Call_Received4000
TimeStamp:24.892166451827585 Activity/Action: simModel.Call_Received3000
TimeStamp:27.705707571745897 Activity/Action: simModel.Travel
TimeStamp:28.74087376043046 Activity/Action: simModel.Travel
TimeStamp:210.0 Activity/Action: simModel.StartLunch
TimeStamp:960.0 Activity/Action: simModel.EndDay
TimeStamp:10080.0 Stop Notification

In the next block, we can see that multiple travel activities have initiated and the First travel event at 27.705 is about to terminate (i.e. the clock is very close). Yet there are still no service activities initiated.

Clock: 26.016124480507298
Number of T12 Employees: 6, Number of ALL Employees: 6
T12Satisfied: 0, totalT12: 0, T34Satisfifed 0, totalT34: 0
SatisfactionLevelT12: 0%, SatisfactionLevelT34: 0%, SatisfactionLevelAll: 0%
----------SBL----------
TimeStamp:27.705707571745897 Activity/Action: simModel.Travel
TimeStamp:28.74087376043046 Activity/Action: simModel.Travel
TimeStamp:30.810454135759183 Activity/Action: simModel.Travel
TimeStamp:32.61531572542714 Activity/Action: simModel.Call_Received2000
TimeStamp:34.60595875345674 Activity/Action: simModel.Travel
TimeStamp:35.890488293078946 Activity/Action: simModel.Travel
TimeStamp:41.22411988506467 Activity/Action: simModel.Call_Received3000
TimeStamp:42.39347047749756 Activity/Action: simModel.Travel
TimeStamp:42.43778790636902 Activity/Action: simModel.Travel
TimeStamp:43.80097373741944 Activity/Action: simModel.Travel
TimeStamp:44.09076545121491 Activity/Action: simModel.Travel
TimeStamp:44.21867371902474 Activity/Action: simModel.Call_Received1000
TimeStamp:49.26631206679564 Activity/Action: simModel.Travel
TimeStamp:49.79492125658679 Activity/Action: simModel.Travel
TimeStamp:53.58771396769863 Activity/Action: simModel.Travel
TimeStamp:69.24027334335973 Activity/Action: simModel.Call_Received4000
TimeStamp:210.0 Activity/Action: simModel.StartLunch
TimeStamp:960.0 Activity/Action: simModel.EndDay
TimeStamp:10080.0 Stop Notification

In the next SBL output, the clock is now at the terminating time of the travel highlighted in the previous section. However, now the first service activity has been scheduled with an end time of 41.761.

Clock: 27.705707571745897
Number of T12 Employees: 6, Number of ALL Employees: 6
T12Satisfied: 0, totalT12: 0, T34Satisfifed 0, totalT34: 0
SatisfactionLevelT12: 0%, SatisfactionLevelT34: 0%, SatisfactionLevelAll: 0%
----------SBL----------
TimeStamp:28.74087376043046 Activity/Action: simModel.Travel
TimeStamp:30.810454135759183 Activity/Action: simModel.Travel
TimeStamp:32.61531572542714 Activity/Action: simModel.Call_Received2000
TimeStamp:34.60595875345674 Activity/Action: simModel.Travel
TimeStamp:35.890488293078946 Activity/Action: simModel.Travel
TimeStamp:41.22411988506467 Activity/Action: simModel.Call_Received3000

TimeStamp:41.76162726521227 Activity/Action: simModel.Service
TimeStamp:42.39347047749756 Activity/Action: simModel.Travel
TimeStamp:42.43778790636902 Activity/Action: simModel.Travel
TimeStamp:43.80097373741944 Activity/Action: simModel.Travel
TimeStamp:44.09076545121491 Activity/Action: simModel.Travel
TimeStamp:44.21867371902474 Activity/Action: simModel.Call_Received1000
TimeStamp:49.26631206679564 Activity/Action: simModel.Travel
TimeStamp:49.79492125658679 Activity/Action: simModel.Travel
TimeStamp:53.58771396769863 Activity/Action: simModel.Travel
TimeStamp:69.24027334335973 Activity/Action: simModel.Call_Received4000
TimeStamp:210.0 Activity/Action: simModel.StartLunch
TimeStamp:960.0 Activity/Action: simModel.EndDay
TimeStamp:10080.0 Stop Notification

After 11:30 , as employees finish their service calls they will start to go on their lunch. This is seen below as the clock is past 210 (which is 11:30), as lunch activities are now being scheduled.

Clock: 227.4881014351003
Number of T12 Employees: 8, Number of ALL Employees: 8
TimeStamp:227.54406559963252 Activity/Action: simModel.Call_Received2000
TimeStamp:230.18945432475232 Activity/Action: simModel.Travel
TimeStamp:231.70993429251243 Activity/Action: simModel.Call_Received4000
TimeStamp:233.13319365314442 Activity/Action: simModel.Call_Received1000
TimeStamp:236.60540438597533 Activity/Action: simModel.Service
TimeStamp:237.84136112226867 Activity/Action: simModel.Travel
TimeStamp:241.1353878074152 Activity/Action: simModel.Call_Received3000
TimeStamp:243.05345283590552 Activity/Action: simModel.Service
TimeStamp:247.3627866398657 Activity/Action: simModel.Service
TimeStamp:256.6553408107854 Activity/Action: simModel.Service
TimeStamp:259.69995793723945 Activity/Action: simModel.Service
TimeStamp:261.6168569327582 Activity/Action: simModel.Service
TimeStamp:276.41470961767715 Activity/Action: simModel.Lunch

At midnight the EndDay action executes and the next one is scheduled for 2400 (which is simple the next day at midnight). In this log, you can also see that the next StartLunch action is scheduled as well.

Clock: 960.0
Number of T12 Employees: 8, Number of ALL Employees: 8
T12Satisfied: 34, totalT12: 65, T34Satisfifed 22, totalT34: 30
SatisfactionLevelT12: 52%, SatisfactionLevelT34: 73%, SatisfactionLevelAll: 59%
----------SBL----------
TimeStamp:1020.0 Activity/Action: simModel.Lunch
TimeStamp:1440.0 Activity/Action: simModel.Call_Received2000

TimeStamp:1440.0 Activity/Action: simModel.Call_Received1000
TimeStamp:1440.0 Activity/Action: simModel.Call_Received3000
TimeStamp:1440.0 Activity/Action: simModel.Call_Received4000
TimeStamp:1650.0 Activity/Action: simModel.StartLunch
TimeStamp:2400.0 Activity/Action: simModel.EndDay
TimeStamp:10080.0 Stop Notification

The next log demonstrates the behaviour that any of type Basic that will not be finished before 5:30, it is scheduled to be finished the next day. The following log show that a basic call came into the system and the service activity was scheduled for the next day.

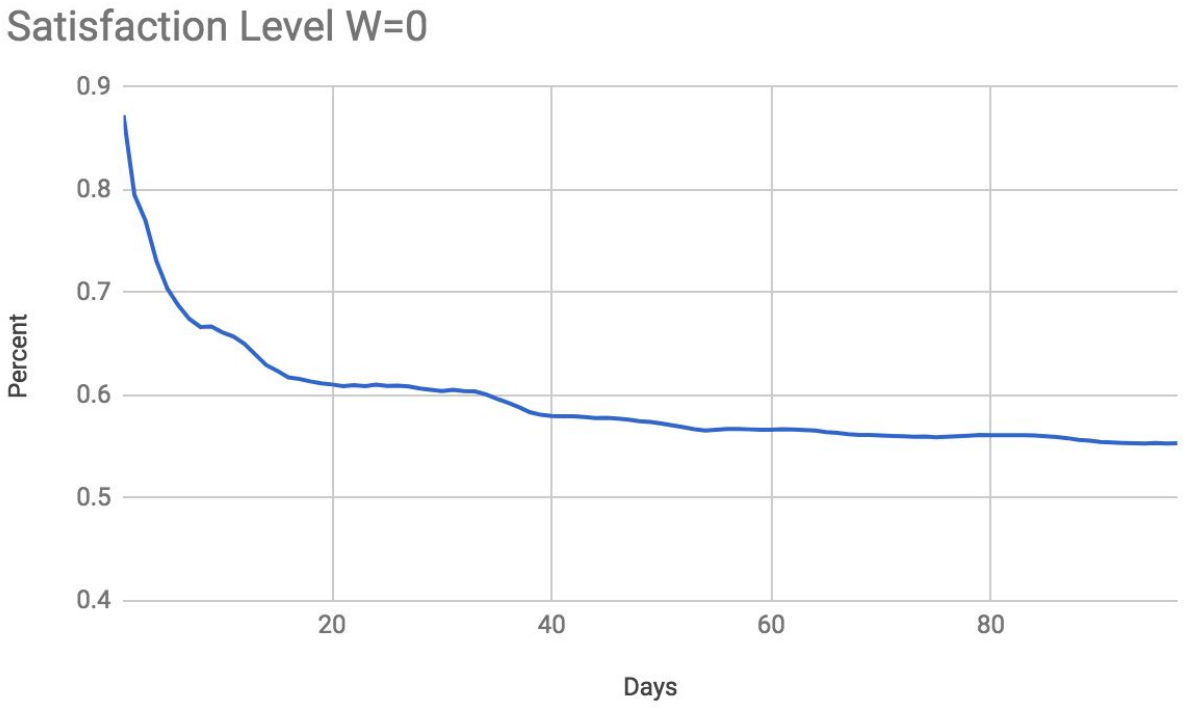
****DURATION******
CALL_is_BASIC
Clock: 521
Service time is : 63.16400454209476
****
Clock: 521.8425783012303
Number of T12 Employees: 8, Number of ALL Employees: 8
T12Satisfied: 57, totalT12: 63, T34Satisfifed 11, totalT34: 20
SatisfactionLevelT12: 90%, SatisfactionLevelT34: 55%, SatisfactionLevelAll: 82%
----------SBL----------
TimeStamp:522.8367536265141 Activity/Action: simModel.Call_Received2000
TimeStamp:525.6629382579437 Activity/Action: simModel.Service
TimeStamp:526.443665617458 Activity/Action: simModel.Service
TimeStamp:527.0572587743264 Activity/Action: simModel.Travel
TimeStamp:528.3883575110833 Activity/Action: simModel.Service
TimeStamp:529.3389228643684 Activity/Action: simModel.Service
TimeStamp:531.5391436167941 Activity/Action: simModel.Service
TimeStamp:531.616922483061 Activity/Action: simModel.Call_Received3000
TimeStamp:531.7691740310917 Activity/Action: simModel.Call_Received1000
TimeStamp:532.3558359380479 Activity/Action: simModel.Service
TimeStamp:534.0218506647783 Activity/Action: simModel.Service
TimeStamp:537.0831703294921 Activity/Action: simModel.Travel
TimeStamp:539.2935466103037 Activity/Action: simModel.Service
TimeStamp:539.3229045099107 Activity/Action: simModel.Travel
TimeStamp:541.2916238927273 Activity/Action: simModel.Service
TimeStamp:555.6291831903555 Activity/Action: simModel.Service
TimeStamp:586.8906884779744 Activity/Action: simModel.Call_Received4000
TimeStamp:604.4012370350979 Activity/Action: simModel.Service
TimeStamp:960.0 Activity/Action: simModel.EndDay
TimeStamp:1455.006582843325 Activity/Action: simModel.Service

TimeStamp:1650.0 Activity/Action: simModel.StartLunch
TimeStamp:10080.0 Stop Notification

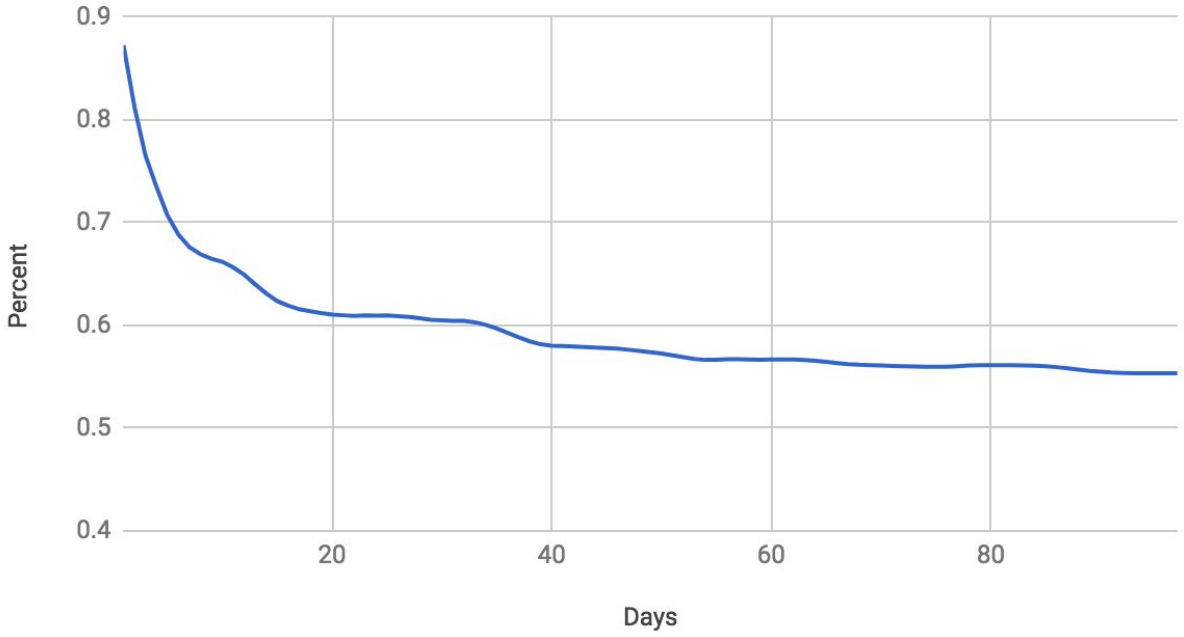
Experimentation and Analysis

Steady State Observation Interval

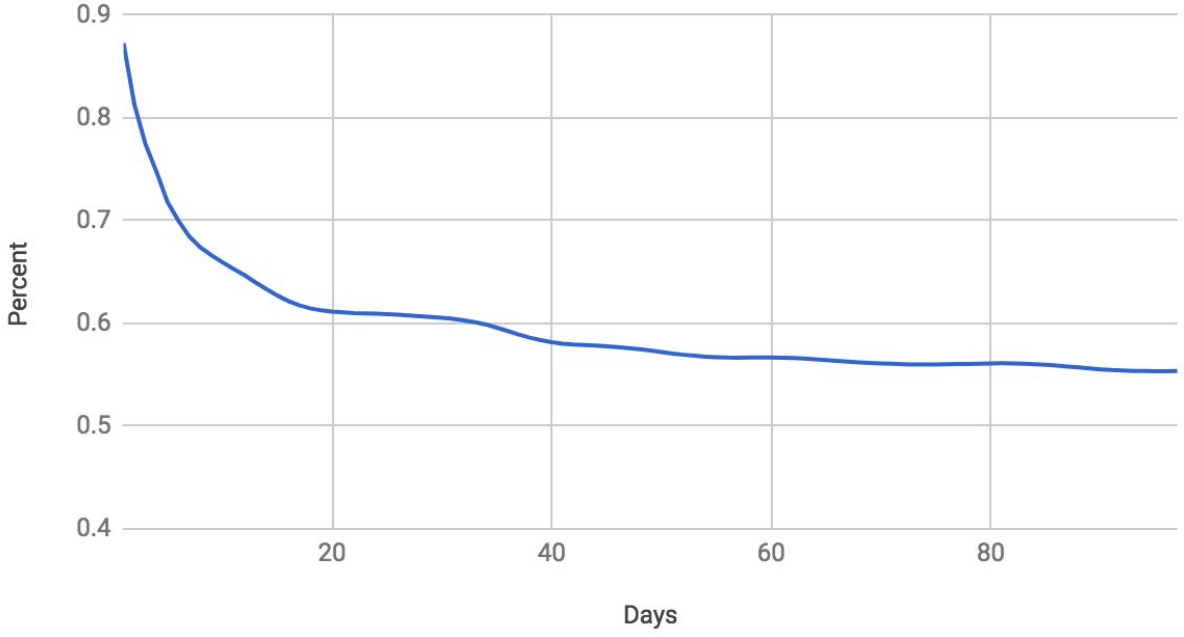
This study is a steady state study and requires a warm-up time and endtime. The warm-up accounts for the fact that all employees are initially idle and they can respond to calls instantly as they come in. Accordingly, this warm-up time allows for a more accurate confidence interval of the true average of the output. To compute the warm-up time, we applied Welch's averaging method for both satisfactionLevelAll and averageDailyCost with W = 0, W = 1, W =3 . The java class WarmUp.java implements the Welch's average with 10 simulation runs, an interval length of 1 day and 100 intervals. The data generated is found in the attached WarmUp.xlsx. The graphs are for each W value are shown below.

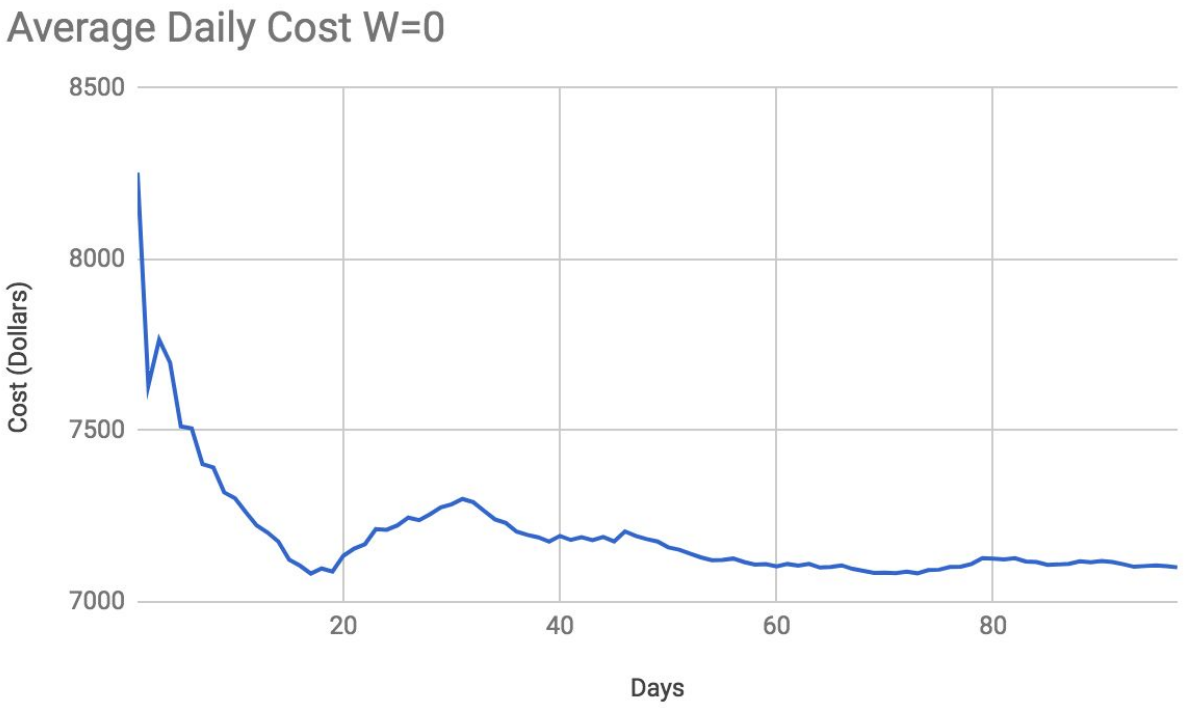


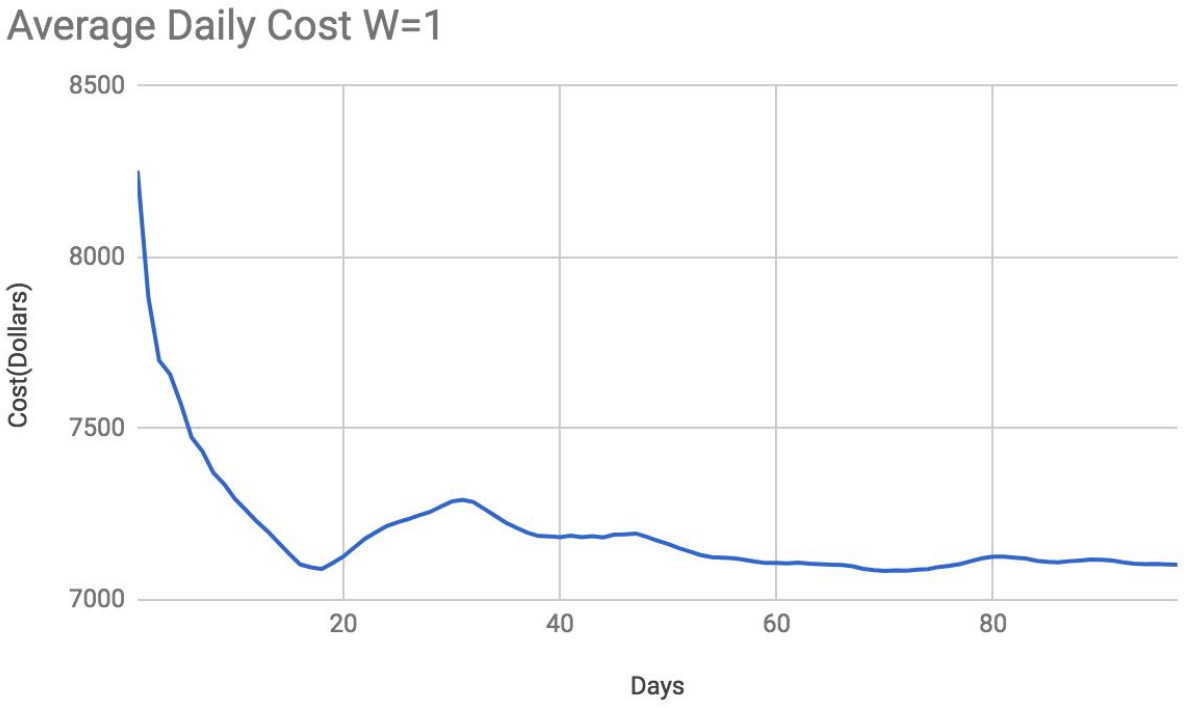
Satisfaction Level W=1

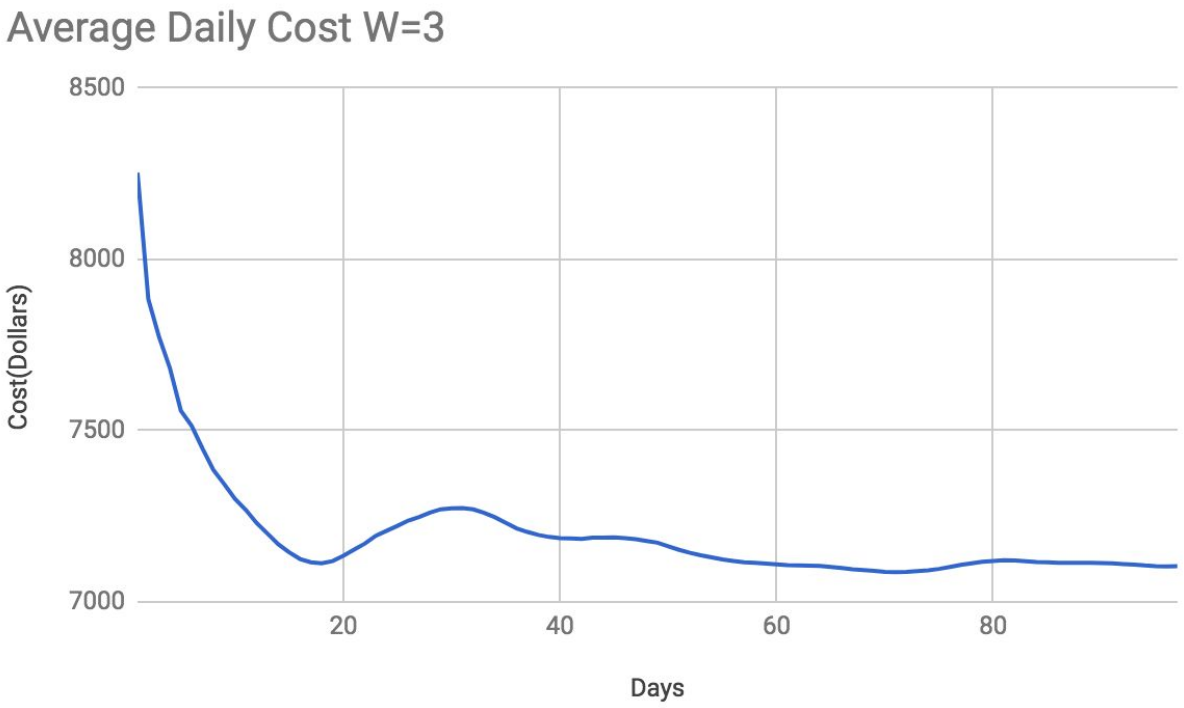


Satisfaction Level W=3




Average Daily Cost W=0
8500
8000
7500
7000
Cost (Dollars)
20
40
60
80
Days


Average Daily Cost W=1
8500
8000
7500
7000
Cost(Dollars)
20
40
60
80
Days



For the satisfactionLevelAll , when W = 0 , W = 1 , W = 3 , the value begins to stabilize at around 70 weeks. For average daily cost the values again stabilize near 80 weeks. This makes sense because the average daily cost is the overtime cost will stabilize over time as the rate at which employees receive and can respond to calls reaches an equilibrium. In conclusion we will use a warm-up time of 80 days or 11.5 weeks for experimentation.

To conclude our observation interval we must perform experimentation to determine a suitable right hand bound. The class Experiment2 was designed to examine the effect using a 15 week, 25 week, and 35 week observation interval with the number of simulation runs ranging from 20 to 1000. The results of the experiment are shown below.

```
Number of Employees of type T12 = 13
Number of Employees of type ALL = 13
End Time = 15 weeks (151200.0 hours), TimeStamp: 20:09:53
End Time = 25 weeks (252000.0 hours), TimeStamp: 20:10:31
End Time = 35 weeks (352800.0 hours), TimeStamp: 20:11:20
```

Number of employees 13

tf:	15 weeks				25 weeks				35 weeks			
n	yb(n)	s(n)	z(n)	z(n)/yb(n)	yb(n)	s(n)	z(n)	z(n)/yb(n)	yb(n)	s(n)	z(n)	z(n)/yb(n)
Satisfaction Level T12												
20	0.718	0.055	0.021	0.0298	0.686	0.038	0.015	0.0214	0.671	0.028	0.011	0.0163
30	0.714	0.052	0.016	0.0226	0.682	0.035	0.011	0.0159	0.667	0.027	0.008	0.0124
40	0.712	0.052	0.014	0.0194	0.681	0.035	0.009	0.0136	0.668	0.028	0.008	0.0113
60	0.706	0.050	0.011	0.0153	0.677	0.033	0.007	0.0106	0.665	0.027	0.006	0.0086
80	0.712	0.054	0.010	0.0142	0.683	0.038	0.007	0.0104	0.669	0.030	0.006	0.0083
100	0.705	0.055	0.009	0.0129	0.679	0.038	0.006	0.0093	0.667	0.030	0.005	0.0074
1000	0.698	0.052	0.003	0.0039	0.673	0.038	0.002	0.0029	0.662	0.030	0.002	0.0023
Satisfaction Level T34												
20	0.882	0.022	0.008	0.0095	0.882	0.015	0.006	0.0064	0.879	0.012	0.005	0.0055
30	0.883	0.022	0.007	0.0076	0.883	0.015	0.005	0.0051	0.883	0.013	0.004	0.0046
40	0.884	0.021	0.005	0.0062	0.884	0.016	0.004	0.0049	0.883	0.014	0.004	0.0043
60	0.884	0.018	0.004	0.0045	0.883	0.015	0.003	0.0038	0.882	0.014	0.003	0.0034
80	0.885	0.019	0.003	0.0039	0.884	0.015	0.003	0.0032	0.883	0.013	0.002	0.0028
100	0.885	0.019	0.003	0.0035	0.884	0.016	0.003	0.0029	0.883	0.014	0.002	0.0026
1000	0.883	0.019	0.001	0.0011	0.882	0.015	0.001	0.0009	0.882	0.012	0.001	0.0007
Satisfaction Level All												
20	0.771	0.041	0.016	0.0207	0.749	0.028	0.011	0.0142	0.738	0.020	0.008	0.0105
30	0.768	0.037	0.011	0.0149	0.747	0.025	0.008	0.0102	0.736	0.018	0.006	0.0076
40	0.767	0.037	0.010	0.0128	0.746	0.024	0.007	0.0087	0.737	0.020	0.005	0.0072
60	0.763	0.036	0.008	0.0101	0.743	0.023	0.005	0.0067	0.735	0.019	0.004	0.0055
80	0.768	0.039	0.007	0.0094	0.748	0.027	0.005	0.0067	0.738	0.021	0.004	0.0053
100	0.763	0.039	0.006	0.0085	0.745	0.027	0.005	0.0061	0.736	0.021	0.004	0.0048
1000	0.757	0.038	0.002	0.0026	0.741	0.028	0.001	0.0019	0.732	0.021	0.001	0.0015
Average Daily Cost												
20	12690.929	403.263	155.926	0.0123	12720.767	317.405	122.728	0.0096	12748.604	284.948	110.178	0.0086
30	12657.238	390.147	121.031	0.0096	12696.287	306.644	95.127	0.0075	12725.138	256.932	79.705	0.0063
40	12636.703	361.462	96.290	0.0076	12716.821	281.219	74.914	0.0059	12749.761	242.548	64.612	0.0051
60	12613.793	342.951	73.986	0.0059	12685.833	284.388	61.352	0.0048	12740.818	245.692	53.004	0.0042
80	12629.986	328.601	61.146	0.0048	12697.494	268.853	50.028	0.0039	12748.599	246.944	45.951	0.0036
100	12648.280	328.558	54.553	0.0043	12699.879	258.503	42.921	0.0034	12750.815	233.548	38.778	0.0030
1000	12725.695	322.570	16.794	0.0013	12766.253	244.485	12.728	0.0010	12790.918	207.811	10.819	0.0008

For our output, we are looking to achieve a confidence level of over 95%. Hence the z(n)/yb(n) values need to be less than 5%. By examining the output of the Experiment2, all four outputs: Satisfaction Level T12, Satisfaction Level T34, Satisfaction Level ALL, and Average Daily Cost, the z(n)/yb(n) are less than 5% at 15 weeks. Thus a 15 week observation interval with 20 simulation runs is chosen for experimentation to solve the modelling and simulation problem.

Experimentation

These experimentation programs were created for the project and review the following Java classes:

1) WarmUp.java
2) Experiment1.java
3) Experiment2.java
4) Experiment3.java

We used Experiment3.java to get feasibility set for our solution. The parameter vector (EMP_T12,EMP_ALL)= (8,8) gives overall satisfaction of 2% with 95% CI, while the vector(14,14) gives overall satisfaction or 96% with 95% CI. More experiments with intermediate values are performed to obtain the optimal staffing.
Here we need to clarify the term “optimal” as stated in project goals. As T12 employees are cheaper to hire , an optimal in terms of total fixed cost is different than an optimal in terms of minimum number of total employees (both meeting the satisfaction level requirements). For example the vectors (16,11) and (14,13) both give about same satisfaction level and are both optimized in terms of minimum number of employees (16+11=14+13) , (16,11) incurs least fixed cost.
After setting up a feasibility set for the values of the parameters, our main experiments are done using Experiment2.java to get statistically significant values. The methodology adopted is :

1) Start with (EMP_T12,EMP_ALL)=(8,8) as the base case
2) Increase EMP_T12 till satisfactioT12>0.85
3) Increase EMP_ALL till satisfactionT34>0.85
4) Futhers investigate the neighbouring points to obtain minimum number of employess while maintaining satisfactionALL>0.85
5) Repeat the steps for exploratory satisfaction level of 95%.

Output Analysis

For our output, we began with the number of employees of type EMP_T12 equal to 8, and we increased the number of employees of type EMP_ALL until the satisfaction level was above 85%. In our experiments, the satisfaction level of employee type EMP_ALL finally surpassed 85% when EMP_ALL = 15. See the output files ET12_8_AND_EALL_9_10_11_12.txt and ET12_8_AND_EALL_13_14_15.txt to see all of the outputs from the number of EMP_ALL = 9 until the number of EMP_ALL = 15 for the base case of EMP_T12 = 8.

		Number of employeesTALL 14		
		Number of employeesT12 8		
tf:	15 weeks			
n	yb(n)	s(n)	z(n)	z(n)/yb(n)
		Satisfaction Level T12		
20	0.646	0.024	0.009	0.0141
30	0.645	0.020	0.006	0.0097
40	0.645	0.020	0.005	0.0081
60	0.642	0.020	0.004	0.0068
80	0.642	0.019	0.004	0.0055
100	0.642	0.020	0.003	0.0051
1000	0.640	0.021	0.001	0.0017
		Satisfaction Level T34		
20	0.823	0.022	0.008	0.0102
30	0.826	0.023	0.007	0.0085
40	0.827	0.023	0.006	0.0075
60	0.825	0.022	0.005	0.0056
80	0.827	0.021	0.004	0.0048
100	0.826	0.022	0.004	0.0044
1000	0.824	0.022	0.001	0.0014

		Number of employeesTALL 15		
		Number of employeesT12 8		
tf:	15 weeks			
n	yb(n)	s(n)	z(n)	z(n)/yb(n)
		Satisfaction Level T12		
20	0.661	0.016	0.006	0.0094
30	0.660	0.014	0.004	0.0065
40	0.660	0.014	0.004	0.0055
60	0.660	0.013	0.003	0.0042
80	0.659	0.012	0.002	0.0034
100	0.659	0.012	0.002	0.0031
1000	0.658	0.013	0.001	0.0010
		Satisfaction Level T34		
20	0.912	0.019	0.007	0.0080
30	0.913	0.017	0.005	0.0058
40	0.914	0.016	0.004	0.0046
60	0.913	0.015	0.003	0.0035
80	0.914	0.015	0.003	0.0030
100	0.913	0.016	0.003	0.0028
1000	0.911	0.016	0.001	0.0009

For the second part of the experiment, we set with the number of employees of type EMP_ALL equal to 8, and we increased the number of employees of type EMP_T12 until the satisfaction level was above 85%. In our experiments, the satisfaction level of employee type EMP_T12

finally surpassed 85% when EMP_T12 = 15. See the output files EALL_8_AND_ET12_9_10_11_12.txt and EALL_8_AND_ET12_13_14_15.txt to see all of the outputs from the number of EMP_T12 = 9 until the number of EMP_T12 = 15 for the base case of EMP_ALL = 8.

Number of employeesT12 14

Number of employeesTALL 8

tf: 15 weeks

n	yb(n)	s(n)	z(n)	z(n)/yb(n)
Satisfaction Level T12				
20	0.671	0.021	0.008	0.0120
30	0.669	0.020	0.006	0.0091
40	0.668	0.018	0.005	0.0073
60	0.670	0.021	0.004	0.0067
80	0.669	0.020	0.004	0.0055
100	0.668	0.019	0.003	0.0046
1000	0.667	0.017	0.001	0.0013
Satisfaction Level T34				
20	0.019	0.006	0.002	0.1191
30	0.021	0.007	0.002	0.1075
40	0.021	0.007	0.002	0.0918
60	0.020	0.008	0.002	0.0864
80	0.021	0.009	0.002	0.0794
100	0.021	0.009	0.001	0.0678
1000	0.020	0.008	0.000	0.0210

Number of employeesT12 15

Number of employeesTALL 8

tf: 15 weeks

n	yb(n)	s(n)	z(n)	z(n)/yb(n)
Satisfaction Level T12				
20	0.928	0.038	0.015	0.0156
30	0.922	0.040	0.012	0.0134
40	0.923	0.037	0.010	0.0107
60	0.922	0.038	0.008	0.0090
80	0.923	0.038	0.007	0.0077
100	0.921	0.038	0.006	0.0069
1000	0.921	0.041	0.002	0.0023
Satisfaction Level ALL				
20	0.019	0.006	0.002	0.1167
30	0.021	0.007	0.002	0.1075
40	0.021	0.007	0.002	0.0935
60	0.020	0.008	0.002	0.0847
80	0.021	0.009	0.002	0.0803
100	0.021	0.009	0.001	0.0682
1000	0.020	0.008	0.000	0.0206

Finally, in Experiment 3 we examined the difference between EMP_T12 = 15 and EMP_ALL = 15, and EMP_T12 = 14 and EMP_ALL = 14. Although the satisfaction level for both cases was greater than 85%, when analyzing the average daily cost, using EMP_T12 = 15 and EMP_ALL = 15 was the more optimal solution. This answer makes sense, since the (15, 15) solution decreases the number of overtime hours needed to complete the work, hence decreases the total cost.

	Satisfaction Level ALL			Average Daily Cost		
Run	14,14	15,15	Difference	14,14	15,15	Difference
1	0.968	0.973	0.004	11780.517	11016.950	-763.567
2	0.968	0.973	0.005	11643.500	10925.433	-718.067
3	0.969	0.972	0.003	11845.500	11140.750	-704.750
4	0.969	0.973	0.004	11338.900	10969.267	-369.633
5	0.971	0.974	0.003	11319.183	10843.400	-475.783
6	0.974	0.974	0.001	11099.650	10907.850	-191.800
7	0.973	0.974	0.001	11303.400	10767.017	-536.383
8	0.964	0.973	0.009	11538.667	10861.450	-677.217
9	0.967	0.973	0.006	11166.417	11487.633	321.217
10	0.973	0.974	0.001	11183.767	10705.733	-478.033
11	0.970	0.974	0.004	11255.850	10933.383	-322.467
12	0.967	0.972	0.005	11850.867	10935.133	-915.733
13	0.964	0.975	0.011	11982.417	10888.583	-1093.833
14	0.971	0.973	0.002	11247.383	10949.200	-298.183
15	0.970	0.974	0.004	12014.467	11046.483	-967.983
16	0.972	0.973	0.001	11144.967	10801.567	-343.400
17	0.967	0.973	0.005	12245.517	11248.350	-997.167
18	0.950	0.973	0.024	11361.967	10982.400	-379.567
19	0.972	0.973	0.001	11275.317	10741.183	-534.133
20	0.947	0.974	0.027	11687.283	11110.383	-576.900
ybar(n)			0.006			-551.169
s(n)			0.007			327.411
zeta(n)			0.003			153.229
CI Min(n)			0.003			-704.399
CI Max(n)			0.009			-397.940

These numbers work for both the 85% and 95% cases, since there was a big jump in satisfaction from 14 to 15 employees, both when EMP_T12 was fixed at 8 and EMP_ALL increased, and when EMP_ALL was fixed at 8 and EMP_T12 increased.

We also checked the case when EMP_T12 = 16 and EMP_ALL = 16 to see if the average daily cost decreased. This turned out to not be true, and the (16, 16) result did not achieve better costs than the (15, 15) case.

Hence the optimal number of employees for SM Office repair is EMP_T12 = 15 and EMP_ALL = 15.

Conclusions

The purpose of this experiment was to determine the optimal staffing levels throughout the work day, to meet the repair contracts within the time constraints. From our experiments, we found that the optimal staffing level to achieve 85% customer satisfaction was EMP_T12 = 15 and EMP_ALL = 15.

This model can be easily adopted to include more than one office centers in different regions. A central dispatcher would be able to dispatch and allocate the employees in all the regions with minimal changes to the simulation model. We can also recommend another study which takes into account the dispatching problem (minimizing travel time) to be incorporated in dispatching algorithm.

Annex A - Data Modelling

Input Data

- SM OffiCe Repair acquired four data files, that represents the service time by equipment type.
    - The data files are: TYPE1000.dat , TYPE2000.dat , TYPE3000.dat , TYPE4000.dat

Table 1: Calls per hour for each equipment type

Time Period	Type 1000	Type 2000	Type 3000	Type 4000
8 AM - 9 AM	7	8	5	2
9 AM - 10 AM	12	11	6	3
10 AM - 11 AM	10	8	5	4
11 AM - Noon	7	9	4	3
Noon - 1 PM	5	6	3	2
1 PM - 2 PM	4	4	3	1
2 PM - 3 PM	5	3	2	1
3 PM - 4 PM	4	3	2	1
4 PM - 5 PM	3	2	1	1

Table 2: Percentage of basic and premium contracts per equipment type

Time Period	Type 1000	Type 2000	Type 3000	Type 4000
basic	35	35	25	15
Premium	65	65	75	85

Generated Data

service Time

- The service time for type 1000 machines is determined using a Gaussian distribution the mean and standard deviation of TYPE1000.dat. These values are meanSrvTm1000 = 14.79; stdDevSrvTm1000 = 5.85;
- The service time for type 2000 machines is determined using a Gaussian distribution the mean and standard deviation of TYPE2000.dat. These values are meanSrvTm2000 = 44.36; stdDevSrvTm2000 = 15.06;
- The service time for type 3000 machines is determined using a Gaussian distribution the mean and standard deviation of TYPE3000.dat. These values are meanSrvTm3000 = 60.27; stdDevSrvTm3000 = 27.40;
- The service time for type 4000 machines is determined using a Gaussian distribution the mean and standard deviation of TYPE4000.dat. These values are meanSrvTm4000 = 130.17; stdDevSrvTm4000 = 29.84;

Arrival Time

The arrival time is generated using the inter-arrival mean. This mean is calculated by dividing 60 minutes/hour by the rates found in Table 1. This mean is found for every hour of the work day. The results of these means for each equipment type are listed below:

Equipment Type 1000:
8AM - 9AM:
MEAN = (60 min/h) / (7 calls/h) = 8.57
9AM - 10AM:
MEAN = (60 min/h) / (12 calls/h = 5.0
10AM - 11AM:
MEAN = (60min/h) / (10 calls/h) = 6.0
11AM - 12PM:
MEAN = (60 min/h) / (7 calls/h) = 8.57
12 PM - 1PM:
MEAN = (60 min/h) / (5 calls/h) = 12.0
1PM - 2PM:
MEAN = (60 min/h) / (4 calls/h) = 15.0
2PM - 3M:
MEAN = (60 min/h) / (5 calls/h = 12.0
3PM - 4PM:
MEAN = (60 min/h) / (4 calls/h) = 15.0
4PM - 5PM:
MEAN = (60 min/h) / (3 calls/h) = 20.0

Equipment Type 2000:
8AM - 9AM:
MEAN = (60 min/h) / (8 calls/h) = 7.5

9AM - 10AM:
MEAN = (60 min/h) / (11 calls/h = 5.45
10AM - 11AM:
MEAN = (60min/h) / (8 calls/h) = 7.5
11AM - 12PM:
MEAN = (60 min/h) / (9 calls/h) = 6.66
12 PM - 1PM:
MEAN = (60 min/h) / (6 calls/h) = 10.0
1PM - 2PM:
MEAN = (60 min/h) / (4 calls/h) = 15.0
2PM - 3M:
MEAN = (60 min/h) / (3 calls/h = 20.0
3PM - 4PM:
MEAN = (60 min/h) / (3 calls/h) = 20.0
4PM - 5PM:
MEAN = (60 min/h) / (2 calls/h) = 30.0

Equipment Type 3000

8AM - 9AM:
MEAN = (60 min/h) / (5 calls/h) = 12.0
9AM - 10AM:
MEAN = (60 min/h) / (6 calls/h = 10.0
10AM - 11AM:
MEAN = (60min/h) / (5 calls/h) = 12.0
11AM - 12PM:
MEAN = (60 min/h) / (4 calls/h) = 15.0
12 PM - 1PM:
MEAN = (60 min/h) / (3 calls/h) = 20.0
1PM - 2PM:
MEAN = (60 min/h) / (3 calls/h) = 20.0
2PM - 3M:
MEAN = (60 min/h) / (2 calls/h = 30.0
3PM - 4PM:
MEAN = (60 min/h) / (2 calls/h) = 30.0
4PM - 5PM:
MEAN = (60 min/h) / (1 calls/h) = 60.0

Equipment Type 4000

8AM - 9AM:
MEAN = (60 min/h) / (2 calls/h) = 30.0
9AM - 10AM:
MEAN = (60 min/h) / (3 calls/h = 20.0
10AM - 11AM:
MEAN = (60min/h) / (4 calls/h) = 15.0
11AM - 12PM:
MEAN = (60 min/h) / (3 calls/h) = 20.0
12 PM - 1PM:
MEAN = (60 min/h) / (2 calls/h) = 30.0
1PM - 2PM:
MEAN = (60 min/h) / (1 calls/h) = 60.0
2PM - 3M:
MEAN = (60 min/h) / (1 calls/h = 60.0
3PM - 4PM:

MEAN = (60 min/h) / (1 calls/h) = 60.0
4PM - 5PM:
MEAN = (60 min/h) / (1 calls/h) = 60.0

Travel Time

- Travel time is estimated to have a triangular distribution with a mean time of 30 minutes, a minimum of 10 minutes to reach a location, and a maximum of 45 minutes.