University of Toronto
Faculty of Applied Science and Engineering, Mechanical & Industrial Engineering
MIE490 Capstone Design Course
Final Design Specification(FDS)

Executive Summary

Ceridian is a global human capital management (HCM) software company that helps its clients manage all aspects of their employee's employment lifecycle. Ceridian's flagship product Dayforce is a cloud platform that offers workforce management (WFM) solutions which improve operational efficiency for clients from a wide range of industries, especially from hospitality. One component of Dayforce is the scheduling module which enables Dayforce users to roster their employees, deciding each employee's roles and responsibilities as well as shift schedule. The current module can only provide basic schedule information but fail to cover advanced performance metrics such as utilization rates and customer average waiting time. Ceridian continuously tries to serve their clients' business needs better. Therefore, Ceridian cooperated with the capstone team to develop a simulation module to help its clients gain further insights into their schedules. The team is tasked to build a proof-of-concept simulation framework and an illustration of how this framework can be integrated into existing software. This would provide the development team a foundation to build upon and would accelerate the development of the simulation module.

The final design specification document starts with the project overview which consists of the background, the current methodology employed, and the stakeholders of the project being identified and explained. State-of-the-art methodologies in discrete event simulation and its application in retail and hospitality industries are also discussed. Followed by the project requirements, including functions, objectives, constraints and service environment of the final simulation application are explicitly identified. The Exploratory Data Analysis analyzes the current weekly shift schedules of a restaurant client of Ceiridan. Through analyzing 13 weeks of customer arrival patterns and the employee shift schedule, the team approximated the customer arrival distributions for different periods in a week, as well as the staffing level and service distributions for each employee. The distributions and parameters are used for a case study on the restaurant's current schedule.

The following section describes the implementation of the final design in detail. The deliverable files, input file format, core simulation algorithms and the Java application structure are explained clearly. In the next section, the simulation results of the restaurant scenario are demonstrated. The current restaurant daily server utilization rate ranges from 66% to 78%, depending on the day of the week. These statistics could serve as evidence if the restaurant decides to redesign their employee shift schedule to reduce cost of labor while maintaining current average customer wait time. The team also attempted to perform sensitivity analysis on the staffing level and interpret the impact on the system performance metrics.

Finally, the final design is validated against and has met all the project requirements. The capstone team is currently integrating all findings for the final presentation. After the demonstration, the team will further assist the development team at Ceridian to utilize the application, and probably provide directions for future updates and integration into Dayforce.

Executive Summary **1**

1.0 Project Landscape **3**
1.1 Problem Statement 3
1.1.1 Background 3
1.1.2 Problem Definition 3
1.2 State-of-Art Review 5

2.0 Project Requirements **5**
2.1 Functions 5
2.2 Objectives 6
2.3 Constraints 6
2.4 Stakeholders 7
2.5 Service Environment 8
2.5.1 Virtual Environment 8
2.6 Design for Excellence Principle 8
2.6.1 Design for Flexibility 8
2.6.2 Design for Reliability 8
2.6.3 Design for Reusability 9

3.0 Exploratory Data Analysis (EDA) **9**
3.1 Arrival Distribution 9
3.2 Staffing Level 9
3.3 Service Time Distribution 10

4.0 Detailed Design & Implementation **11**
4.1 Design Overview 11
4.1.1 Deliverable Outline 11
4.2 Input Parameters 12
4.3 Core Idea of the Application Development 13
4.3.1 Event-driven Simulation Framework 13
4.3.2 Queuing Network 14
4.4 Application Structure 15

5.0 Testing Results Analysis **18**
5.0.1 Performance Metrics Description 18
5.1 Result Summary 18
5.2 Sensitivity Analysis 20
5.2.1 Change on Staffing Level 20
5.3 Validation Against Project Requirements 21

6.0 Future Work Recommendations **23**

7.0 Conclusion **23**

Reference **24**

Appendix **26**
Appendix A - Estimated Arrival Rate 26
Appendix B - Estimated Staffing Level 28
Appendix C 30
Appendix C.1 - The format of the Input file “input.txt” for Monday 30
Appendix C.2 - Output for One Week 31
Appendix C.3 - Output for the segments on Monday 33
Appendix C.4 - Input with Staffing Level updated for Sensitivity analysis 34

1.0 Project Landscape

1.1 Problem Statement

1.1.1 Background

Ceridian, founded in 1992, is an American-based human capital management (HCM) software company that provides solutions and services for organizations of all sizes, from small businesses to global organizations [1]. Ceridian's flagship product Dayforce is a cloud platform that combines Human Resources (HR), payroll, benefits, workforce management, and talent management in a single application.

Dayforce offers workforce management (WFM) solutions to improve operational efficiency for clients from a wide range of industries. One component of the WFM solution is the scheduling module. Dayforce users take advantage of the scheduling module to roster their employees, deciding when each employee should work and what they should do [2]. The current module provides basic information such as service coverage rate, scheduled employee work hours, and utilization rate. However, it does not provide performance statistics such as projected wait times, queue lengths, throughput rate, etc. which may be of greater use to clients.

Therefore, in response to Dayforce users' interest in the projected performance metrics of schedules that they create, Dayforce has proposed that developing a simulation module may help clients gain insights into their schedules to maximize efficiencies and improve strategic benefits. As this is a new area to the Ceridian software development team, a proof-of-concept framework and an illustration of how this framework can be integrated into existing software would be useful. This would provide the development team a foundation to build upon and would accelerate the development of the simulation module.

1.1.2 Problem Definition

The problem Ceridian is facing right now is that Dayforce does not schedule analysis. The current scheduling module in the Dayforce platform cannot monitor the system performance metrics, thus Dayforce's users cannot evaluate and improve on their current scheduling. Therefore, the development team at Ceridian is motivated to investigate whether simulation could be integrated into the Dayforce scheduling module to provide schedule analysis.
The capstone team is tasked to assess the viability of such a simulation module and aims to implement a proof-of-concept discrete event simulation software application that allows users to simulate their business schedules as a queuing system.

1.1.3 Background Information on Queuing Systems

A queueing system can be described as a system having a service facility at which units of some kind, usually called "customers", arrive for service. Whenever there are more units in the system than the service facility can handle simultaneously, a queue develops. The waiting units take their turn for service according to a predefined rule, and after service they leave the system. Thus, the input to the system consists of the customers demanding service, and the output is the serviced customers. Based on the paper *"Birth-and-Death Queueing Systems:*

Exponential Models"[3], a queueing system is usually characterized by the following components:

" The first component is the input process. Let the customers arrive at the instants $t_0, t_1, t_2 ...$*; then the interarrival times are* $u_r = t_r - t_{r-1}$*, where* $r = 1, 2, 3....$
The random variables u_r *are assumed to be statistically independent, and their probability distribution* $A(u)$ *is called the interarrival time distribution or, simply, the arrival distribution......*

The second component is the queue discipline. This can be described as the rule determining the formation of a queue or queues and the manner in which a customer or customers are selected for service from those waiting. The most common queue discipline is "first come, first served," according to which the units enter service in order of their arrival......

The third component is the service mechanism. The time that elapses while a unit is being served is called its service time. The service times $v_1, v_2, v_3 ...$ *of the successive units are assumed to be independent of one another and of the arrival distribution, and their probability distribution* $B(v)$ *is called the service-time distribution....."*

In this project, the capstone team strived to develop a software application which allows users to build their own queuing systems with user-defined input process and service mechanism. We assumed that the queue discipline must be "first in, first out". Users will need to provide input-parameters including the customer arrival distributions, the number of service stations in the system, the number of servers in each station, the service time distributions for each server in each station, and the start and end time in each server's schedule.

For example, in Section 3.0 below, the team will introduce a real-world case study on one of Ceridian's clients: Whitbread PLC, the largest hospitality company in the UK [4]. Ceridian provided us data for a restaurant branch of Whitbread PLC, including the customer average inter-arrival times in each one-hour period during business hours, from Monday to Sunday, as well as the employee schedule. From the employee schedule, the team deduced that there are two service stations in the system: "taking order" and "cooking food". The team also knew the number of waiters taking orders from customers and the number of cooks preparing their food, as well as when the waiters and cooks start and end their shifts. The team assumed that the inter-arrival times are distributed exponentially and change in every one-hour period in a day. However, information about waiters' and cooks' service time distributions was missing, therefore the team had to conduct extensive research to estimate them.

1.2 State-of-Art Review

The team conducted extensive research on the applications of discrete-event simulation in retail, hospitality, and other types of industries which help companies improve their operational efficiencies [5]. For example, the Nebraska Medical Centre had used discrete event simulation models to simulate the current medical system in a virtual environment, then developed insights from simulation results to identify and remove bottlenecks, reduce patient and surgeon travel distance, ensure the use of operating rooms during staff peak hours, and reduce patient turnover rate [6]. Another example of simulation's application in hospitality is that a group of students developed a simulation model to reduce the average waiting time in the queues for the BGSU Students Union(BTSU) Cafeteria [7].

Researchers had dedicated a lot of efforts into investigating the application of simulation in the retail and hospitality industries. The simulation had been a widely accepted tool for analyzing performance in the service sector. Restaurants, in particular, have been the topic of several studies in the last decade. In 1996, researchers Farahmand and Martinez developed one general model of a fast-food restaurant lobby and drive-through to optimize scheduling and maximize operational efficiency [8]. Later on, researchers Field, Mcknew, and Kiessler applied Monte Carlo modeling to simulate two types of buffet restaurants in 1997 [9]. On the top of that, Jaynes, S. L., and Hoffman, J. O. conducted a case study on leveraging the discrete event simulation for quick-service restaurant traffic analysis [10]. The research explained some useful simulation techniques and methods, as well as key performance indicators that our team could incorporate to build a simulation-based application which could project performance metrics. The solution that the capstone team is developing will address efficiencies in the retail and hospitality scheduling.

2.0 Project Requirements

2.1 Functions

The design solution must meet all functional requirements, as listed below, for a *reliable*, *satisfactory*, and *adequate* user experience.

Primary Functions:

F1. Read input parameters (i.e., derived distributions) to prepare the construction of a discrete event simulation model.

F2. Generate arrival and service events based on derived arrival and service time distributions and parameters.

F3. Implement a discrete event simulation algorithm to project performance metrics, such as average service time and time in system, etc.

F4. Project performance metrics in a meaningful way to the user

Secondary Functions:

F5. Derive arrival and service time distribution, and staffing level from historical data.

F6. Include options to visualize the final results to improve readability (i.e. text, chart).

2.2 Objectives

The following objectives are listed in descending order of importance based on the client's requirements. The detailed description and measurable metrics are in the table 1 below.

Table 1 : Objectives, Goals and Metrics of the Capstone Project

Objective	Objective goals	Metric
O1. Validity	Using the Monte Carlo method with ten iterations to avoid the long-running time, the average simulation results should not deviate from Simio's results by more than 25% numerically [11].	The validity deviation is measured by comparing the difference between the results of widely used simulation software Simio, and the simulation results by using the Monte Carlo Simulation method [11].
O2. Scalability	The design should be able to simulate a small to medium scale real-world system, with 1-5 possible service stations and 1-10 servers at each station. The time complexity of the design should not be greater than the O(n^2), measured by the complexity of the simulation algorithm.	• The number of service stations and the number of servers at each station in the system. • We defined scalability in terms of time complexity, using Big O. The execution time should grow at most quadratically with respect to the product of the number of service stations and the number of servers in each service station [12].
O3. A diverse set of performance metrics	The design should be able to calculate at least 2 different performance metrics [13].	The number of performance metrics in the output.
O4. Easy integration into Dayforce	The discrete event simulation algorithm should be easily converted to the language Ceridian's system uses.	The number of incompatibilities after integrating into the Dayforce platform.

2.3 Constraints

The following constraints are identified based on the client's requirements, which limit the designs in environmental and technical aspects. To be qualified as a solution, the deliverable must satisfy these constraints:

Environmental Constraints [14]:

C1. The discrete event simulation algorithm must be coded in C# or Java
C2. The executable prototype should not depend on any commercial software package

Technical Constraints [14]:

C3. The final design must implement a fully developed discrete event simulation algorithm
C4. The simulation output must be visible and manipulable
C5. The simulation input should be a .csv or .txt file

2.4 Stakeholders

The stakeholders of the design consist of the Software Development Team from Ceridian, the Capstone Design Team, and the supervising professor from the University of Toronto. The respective interests and effects on project functions, objectives, and constraints (FOCs) are listed in the table 2 below.

Table 2: Stakeholders of the Capstone Project

Stakeholders	Interests	Effects on FOCs
	<u>Internal Stakeholders</u>	
Software Development Team (Ceridian)	● The primary user or tester of the final deliverable ● Investigate whether simulation could be used for Dayforce's workforce management module to provide meaningful insights about workforce schedules to its enterprise users ● Learn the feasibility and benefit of the final prototype ● Full implementation and maintenance of the design in the future	● Ensure that the deliverable executes accurately based on reading in basic data needed for discrete event simulation (F1, F2, F3, F5, O1, O2, O3, C1, C2, C3) ● Ensure that the tool reliably and efficiently meets the team's requirements (F1, F2, F3, F4, F5, O1, O2, O3, O4, C1, C2, C3) ● Provides readable and meaningful output to end-user (F4, F6, O3, C4)
	<u>External Stakeholders</u>	
Capstone Design Team (University of Toronto)	● The main driver of the tool development and testing ● Apply state-of-art queuing theory and discrete event simulation techniques to design the simulation module ● Generate and present the performance metrics in a meaningful way to the end-user	● Research and implement technologies and algorithms required to develop the simulation module (F1, F2, F3, F4, F5, O1, O2, O3, C1, C2, C3) ● Analyze the data necessary for input parameters (F1, F2, F5, C5)
Capstone supervisor (University of Toronto)	● Helps and supervises the Capstone Design Team to achieve their goals ● Recommend state-of-art queuing theory and discrete event simulation techniques to design the simulation module	● Inspire and validate the work done by the Capstone Design Team ● Ensure that the simulation algorithm is valid (F1, F2, F3, F4, F5, O1, O2, C3, C1, C2, C3)

As the final solution has the most direct impact on its further development of the simulation module, the software development team has the highest interest. The Capstone supervisor has the second greatest impact as the supervisor ensures that the final solution is validated.

2.5 Service Environment

The service environment represents the operating locations of the final design. All conditions related to FOCs that may cause the potential failure of the tool have to be considered. Since the primary function is to implement a simulation module, the final solution will be deployed in a virtual environment. The intangible virtual infrastructures within the service environment are the most significant factors to be considered.

2.5.1 Virtual Environment

- The simulation module should be implemented using Java or C# and developed in the Eclipse IDE (F3, O1, O2, O4, C1, C2, C3)
- The application should run in a Windows environment (F3, F4, F6, O1, O3, C1, C2, C3)
- The discrete event simulation model is constructed using JAPROSIM library in Java, or SharpSim library in C# [15] (F2, F3, O1, O2, O3, C1, C2, C3)

2.6 Design for Excellence Principle

Design for X principles aim to optimize specific characteristics of the design, where X is the aspect being evaluated. The project particularly focuses on flexibility, reliability and reusability aspects. The corresponding approaches would be beneficial to assess, control and enhance different design features [16]. The following characteristics are considered in decreasing priority according to their adverse impact on the objectives of the project if they are violated. These principles will be assessed separately for design specifications.

2.6.1 Design for Flexibility

Design for flexibility means designing a product that is scalable in capacity and easily expandable from a software's perspective. The team makes sure the application is easy to modify or add new functionalities. For example, the application can be extended to have more probability distributions to represent customer inter-arrivals and service times. In addition, it is not difficult to modify the main simulation algorithm to keep track of more performance metrics, such as throughput rate or individual server utilization rates.

2.6.2 Design for Reliability

The design should consider software fault tolerance. Software fault tolerance is the ability for software which is running to provide service in accordance with the specification to detect and recover from a fault that is happening, or has already happened in the software [17]. Therefore, all simulation output data should be stored in a separate file during run time to prevent data loss. Users will still be able to access output data even if they accidentally close the application, or the application terminates due to unforeseeable reasons.

2.6.3 Design for Reusability

The application should implement an object-oriented algorithm which is designed for future reusability. Object-oriented programming is able to increase flexibility and maintainability since modular coding can be selectively modified without considering the whole [18].

In addition, the application should be compatible with future updates. Potential updates are extending the “Distribution” interface to have more different types of distributions, or diversifying the types of servers by implementing Java inheritance from the “Server” class. Moreover, the application could be integrated with updated tools in the future with minimal impact if Ceridian eventually decides to integrate our design into Dayforce as a new simulation module.

3.0 Exploratory Data Analysis (EDA)

To better understand the restaurant model and illustrate the feasibility of constructing such a model in Java, Cerdian provides the team a restaurant chain use case. For the scenario of this use case, it consists of two service stations connected in sequence: taking order (Front of house) and cooking food (Back of house); customers arrive at the restaurant, the server who is idle serves the customer to make an order, then the cook prepares food to complete the order.

Upon Ceridian’s requirements, the team first retrieved data from the database for the restaurant chain client Whitbred PLC from Ceridian's HCM platform, then analyzed the data using Excel to determine the required input parameters including arrival distributions and staffing level. While the service time distributions were justified based on assumptions and research papers. The analysis for each of the input parameters is discussed in the sections below.

3.1 Arrival Distribution

Considering the fact that the customer arrival may present different distributions in different periods in one day, the arrival distributions are determined by the average number of arrivals in each one-hour time slot, from the time of restaurant opening to closing, from Monday to Sunday.

As required by the Ceridian, the retrieved data was from June 03, 2019 to August 25, 2019, containing the arrival rate from the time of restaurant opening to closing. To determine the arrival distribution, the team split the data into seven categories based on each single day from Monday to Sunday. For each category, arrival rates were filtered by each hour time slot. After calculating the average number of arrivals in each hour, the inter-arrival time was assumed to follow an exponential distribution whose average arrival time varied on an hourly basis [Appendix A].

3.2 Staffing Level

The staffing level is summarized based on the current schedule, and it’s basically information about the number of employees and scheduled employee work hours. The restaurant opens and closes at fixed times from Monday to Sunday, so the weekly schedule is fixed.

As required by the Ceridian, the retrieved data was from June 03, 2019 to August 25, 2019. The data included the start and end working times of each employee at the Front and Back of the House. The team first split the data according to the day of the week. Next, the team calculated the number of employees at the front and back of the house and respective scheduled work hours after analyzing and collating. Finally, the team summarized the information into the schedules for each day of one week [Appendix B].

3.3 Service Time Distribution

The team analyzed and integrated the customer arrival distribution and staffing level based on the retrieved data from the client. However, there is no related data (service time for taking order and cooking food) in the dataset to find the best-fit distribution for the service time. Therefore, the team first developed some assumptions listed below.

Assumptions:

- Servers follow the "First Come, First Serve" (FCFS) service discipline.
- The time from leaving the queuing to sitting down is negligible.
- Customers receive a menu once seated.
- Service time for service station 1 (taking order): The time customers spend ordering drinks and staples.
- Service time for service station 2 (cooking food): The time between the customer places the order and receives of the meal.
- In upscale casual restaurants, the average time customers spend in the dining room for a party of two with two courses and either drinks or coffee service is 1 hour 15 minutes, and up to 1 hour 30 minutes with coffee or dessert service [19].
- A party of four will take approximately 1:30 on average to dine [19].
- For parties of 10 or more, the average time in the dining room will be around 2 hours [19].
- The average waiting time for orders is 11 - 25 minutes [19].
- The industry standard of kitchen-to-table service at a sit-down establishment is 15 - 20 minutes [20].

Those research papers [21][22] informed our work in the following ways. In cases where data is unavailable, it is common practice to consult experts or use a Normal or Triangular distribution [21]. The experts are familiar with the model process, knowledgeable, and experienced to estimate the center and extremes values. The experts of this project are our professional supervisor and client. Our experts concurred with this practice during the meeting. However, our team still lacked the statistical data to use in those distributions, so we used experimental research closely resembling our model process to estimate the service time distribution [22].

The team then found secondary data sources such as research papers, websites to estimate the service time distribution. The table 3 below shows the distributions for each service station that can be utilized as the input parameters for testing. However, for this use case, the team would use Normal distribution for the input testing as recommended by the supervisor.

Table 3. Service Time Distribution

Service stations	**Service time distribution**
Service Station 1: Taking order	Normal distribution(mean = 3.21, std = 1.2) [22]
	Triangular distribution(min = 1.15 , mean = 3.21, max = 5.83) [22]
Service Station 2: Cooking food	Normal distribution(mean = 17.5, std = 2) [20]
	Triangular distribution(min = 15, mean = 17.5, max = 20) [20]

4.0 Detailed Design & Implementation

4.1 Design Overview

The final design is delivered as a discrete event simulation model of the queuing network implemented in Java. To initialize the simulation, the application asks the user to input parameters to construct the simulation model. The application will project multiple key performance indicators and a statistical graph, providing information, helping the user to better design the staffing level of each service center and customer queuing for improved performance and customer service.

4.1.1 Deliverable Outline

Table 4 gives a list of files/codes provided to the Ceridian development team as the final deliverables. It includes the main deliverables which are the source codes for the application development. Among these files, the “Test.java” is the program that enables execution. The FDS reference for the java files is in Section 4.4 that illustrates more details about the application development. There is also a html file for creating the final statistical graphs based on the output data. Additionally, the final deliverable also contains a list of excel files, which includes the data retrieved from the database of Dayforce and the corresponding data analysis which has been discussed in section 3.0.

Table 4. List of Deliverable

	Name	**Description**
Main (src file)	*input.txt*	Input text file
	Distribution (folder)	Define interface distribution; Contains 4 types of distribution: *Exponential, Uniform, Normal, Triangular.*
	Customer.java	Customer class
	DateUtils.java	DateUtils class about time
	EventList.java	Event list class and Event class; record event and the occurrence time of the event
	QueuingSystem.java	Queuing system class, contains the discrete event simulation process of the queuing network

	ServiceNode.java	Service station class, every service station contains one queue and ≥1 server(s)
	Server.java	Server class
	NodeRecord.java	Record the change on number of servers, the change on number of customers, customer waiting time and time in station
	InputProcess.java	To read the input file
	Test.java	The main program for execution
Creating graph	*out.json*	Contains the output generated from program running
	result.html	Code for creating html graph, read data from *out.json*
	Other files	Automatically generated file by the compiler: *sc.iml* Package for creating graph: *Echarts.min.js* and *jquery-3.1.1.min.js*
Data analysis	*data_source.xlsx*	Contains 8 weeks data from 06/03/2019 to 08/25/2019, source from the Dayforce user, Whitbread PLC
	Arrival_distribution .xlsx	Data analysis for average customer arrival distributions of each day in one week period
	Staffinglevel.xlsx	Analysis for staffing levels of each day in one week period

4.2 Input Parameters

The input file "input.txt" contains the required input parameters for simulation. The actual format of the input file is also shown in the appendix C.1 [Appendix C]. To fully test the restaurant chain use case, the team provides one-week parameters that can be used as the input. For better illustration, Table 5 lists the value of input parameters only on Monday to do an one-day real-life example simulation.

Table 5. Table of Input Parameter Value on Monday

Input parameters	Parameter values on Monday	
Customer arrival distribution	9:00-10:00	Exponential (mean = 15.64)
	10:00-11:00	Exponential (mean = 9.33)
	11:00-12:00	Exponential (mean = 7.40)
	12:00-13:00	Exponential (mean = 25.36)
	13:00-14:00	Exponential (mean = 20)
	14:00-15:00	Exponential (mean = 17)
	15:00-16:00	Exponential (mean = 7.3)
	16:00-17:00	Exponential (mean = 11.82)
	17:00-18:00	Exponential (mean = 21)
	18:00-19:00	Exponential (mean = 35.10)
	19:00-20:00	Exponential (mean = 32.82)
	20:00-21:00	Exponential (mean = 17.36)
	21:00-22:00	Exponential (mean = 7.11)

Number of service stations	2 (service station #1: taking order; service station #2: cooking food)	
Number of servers in each station	2 servers in the station #1; 4 servers in station #2	
Staffing level	Service station #1	Server 1: 9:00 - 15:00 Server 2: 17:00 - 22:00 Server 3: 12:00 - 21:00
	Service station #2	Server 4: 6:00 - 15:00 Server 5: 11:00 - 22:30 Server 6: 15:00 - 20:00 Server 7: 17:00 - 22:30
Service time distribution	Service station #1	Server 1: Normal(mean = 3.21, std =1.2) Server 2: Normal(mean = 3.21, std =1.2) Server 3: Normal(mean = 3.21, std =1.2)
	Service station #2	Server 4: Normal(mean = 17.5, std =2) Server 5: Normal(mean = 17.5, std =2) Server 6: Normal(mean = 17.5, std =2) Server 7: Normal(mean = 17.5, std =2)

The input indicates that Dayforce users can customize those parameter values to initialize the simulation. The input parameters enable the user to consider various conditions in the context of the real-life scenario including:

- The customer arrival rate changes over time
- The number of service stages that the customer needs to go through in sequence
- The number of employees in each service stage
- The employee schedule with their expected start and end shift times
- The time that each server needs to spend on each customer.

With all of the inputs, the team is able to model a more realistic scenario for simulation.

4.3 Core Idea of the Application Development

The java application development implements the core idea of discrete event simulation based on the queuing network through the main programs from the java files such as “EventList.java” and “QueuingSystem.java”. The following sections 4.3.1 and 4.3.2 demonstrate event-driven simulation and queuing network in detail.

4.3.1 Event-driven Simulation Framework

Event-driven simulation models the operation of a system as a sequence of events in time. Each event occurs at a particular instant in time and marks a change of state in the system. Between consecutive events, no change in the system is assumed to occur; thus the simulation time can directly jump to the time of the next event [23].
Figure 1 below shows the framework of the event-driven simulation with steps labelled. There are generally 4 repeating steps[24]:

1. Select then delete the first entry (e_1, t_1) from the event list
2. Update the simulation time by advancing it to the new event time t_1

3. Update the state according to the state transition function $x' = f(x, e_1)$
4. The new feasible events triggered by e_1 will be added into the event list. The scheduled event time for event e_i is given by $t_1 + v_i$, where v_i is a lifetime obtained from the random variate generator.

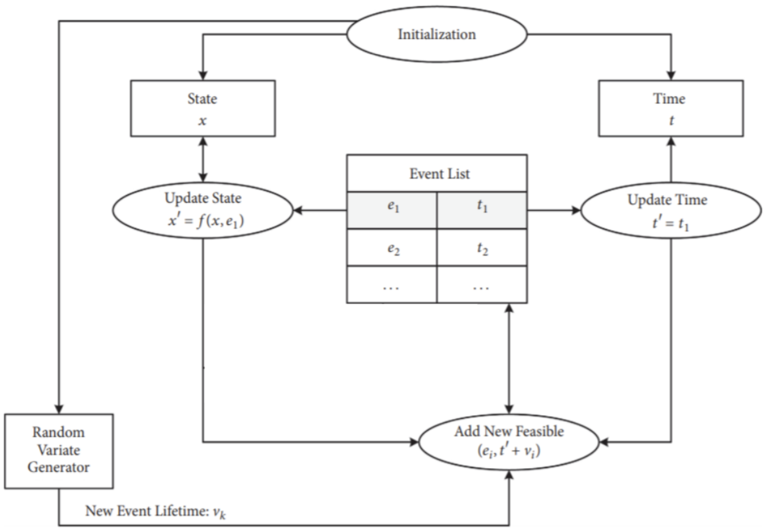


Figure 1.Event-driven simulation framework

Based on Figure 1, another detailed procedures of an event-driven simulation are described below [24][25]:

```
Simulation_time = 0
WHILE EventList is not empty
     Select then delete the first entry (e1,t1) from the EventList
     #e1 is the ID of current event selected, t1 is the duration of event e1
     Compute Simulation_time as Simulation_time + t1 #update simulation time
     x'= f(x,e1) #update the system state with the state transition function
     New_events = GenerateEvent(x',e1) #new events may be triggered by e1
     FOR every event $e_i$ in New_events
        Vi = EstimateDuration($e_i$)
         #estimate duration Vi based on arrival/service rate distribution
         scheduled_time_i = Simulation_time + Vi #schedule when ei will happen
         Insert ($e_i$,Vi) to EventList based on the scheduled time of $e_i$
    ENDFOR
ENDWHILE
```

4.3.2 Queuing Network

To implement the queuing network in which multiple service stations are connected in sequence under the framework of event-driven simulation, there are three types of events defined: i) arrival, ii) service, and iii) departure. An arrival event falls into one of the two categories: arrival from outside(external), and arrival from the prior service station.

Figure 2 demonstrates the general logic to simulate the queuing network based on the sequence of events, where the blue rectangles represent the defined types of events. An "Arrival from outside" event is created when an entity enters the system. If the queue to the first service station that the entity enters and there is at least one free server, then the "Serving" event is triggered. Otherwise, the event will wait to be updated to the next event. Once "Serving" finishes, the "Departure" event is generated. If the current service station is not the last one, the "Arrival from prior station" event is triggered. If the queue that the entity enters is empty and there is at least one free server, then another "Serving" event is generated. The program will loop over the events until the simulation time ends.

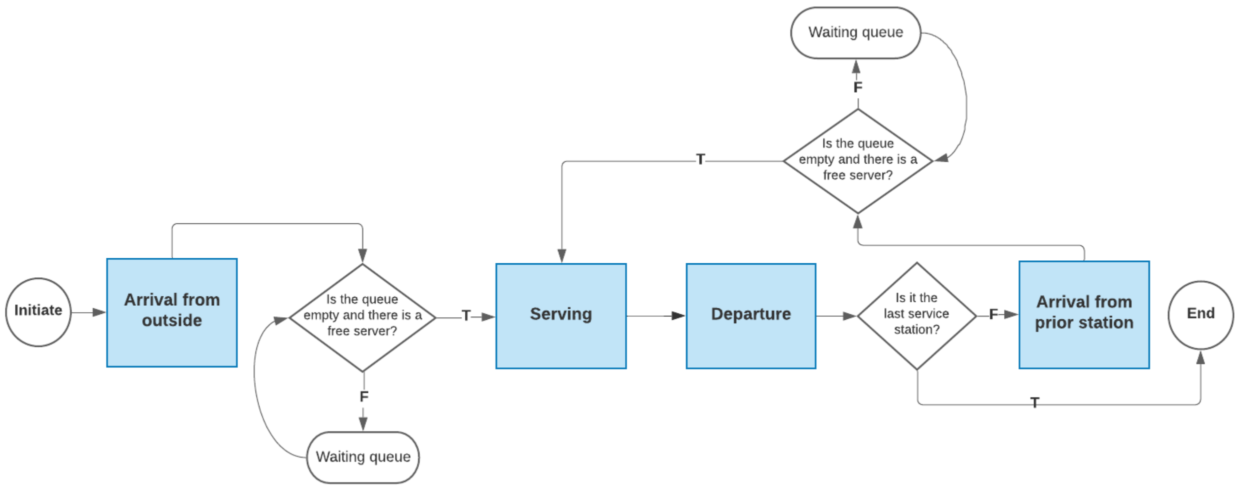

Initiate
Arrival from outside
Is the queue empty and there is a free server?
T
F
Waiting queue
Serving
Departure
Is it the last service station?
F
T
Arrival from prior station
End
Is the queue empty and there is a free server?
T
F
Waiting queue

Figure 2. Event graph modelling

4.4 Application Structure

Figure 3 is the Unified Modeling Language (UML) representation of the application which describes each Java class and how all the Java classes interact to simulate a queuing system. In order to use the application, the first step is to prepare the input data to be imported into the application. The input data text file must be formatted in the following fashion:

1. The first row should be a single number S representing the number of service stations in the system.
2. The second row should be a single number P_1 representing the number of servers in the first service station.
3. The following P_1 rows are supposed to represent the servers in the first service station. Each row contains the start and end times of the server, as well as the service distribution type and parameters.
4. Repeat step 2 and 3 until the S^{th} service station and the servers working at the S^{th} service stations are visited.
5. Starting at 9:00 AM and ending at 10:00 PM, each row represents a one-hour period with specified customer arrival distribution.

Each customer is an instance of the Customer class and has a unique ID. The application keeps track of each customer's arrival and departure moments and the customer's duration in the system as well as the waiting times at each service station.

Each server is an instance of the Server class and has a unique ID. The application keeps track of the start and end time of the server's shift. A server must be in one of the two states: busy or not busy. If the server is busy, he/she must be serving a customer at that moment. In addition, each server has his/her own service time distribution and shift schedule.

Each instance of the Event class represents a real-world event and contains information about the time of occurence and the related customer and server, as well as the type of the Event. In the scope of our project, the team defines an event as the interaction between a customer, a server and a service station that directly impacts the state of the system. The different types of events are: "A new customer arrives at the system", "A customer leaves the previous service station and arrives at the current service station", "A server starts providing service to a customer", "A customer is waiting for service" and "A customer leaves the system".

Each instance of the EventList class is a list of chronological events that drives the simulation algorithm. Therefore, an event list contains various different types of "events" that will update the simulation system state. Events in an event list are processed in a "first in, first out" order which is implemented by sorting the event list from most earliest events to future events. The system state is updated every time the application starts to process a new event.

Each instance of the QueuingSystem class is a complete queuing system which contains information about the service stations, servers and the varying customer arrival distributions for different periods in a day. The system monitors the simulation time and the current events being processed, as well as the throughput and average customer waiting times at each service station and the average time in the system.

The InputProcess class handles reading in the input.txt file. Appendix C.1 shows input data that represents the operation of the restaurant on October 1st, 2020 from 9:00 AM to 11:00 PM. InputProcess sends read-in parameters to the Test.class which initiates the core discrete event simulation algorithm. A high-level explanation of the simulation algorithm is in section 4.3. By calling the simulation function in QueueSystem, servers and customers are initialized, then the function calls itself recursively to update the customer's state in different periods of the simulation. The number of all arrivals and departures from the system are recorded, as well as the durations in the system for every customer served. The "Save()" function stores and outputs the simulation results to a .json file, which can be visualized using the result.html file.

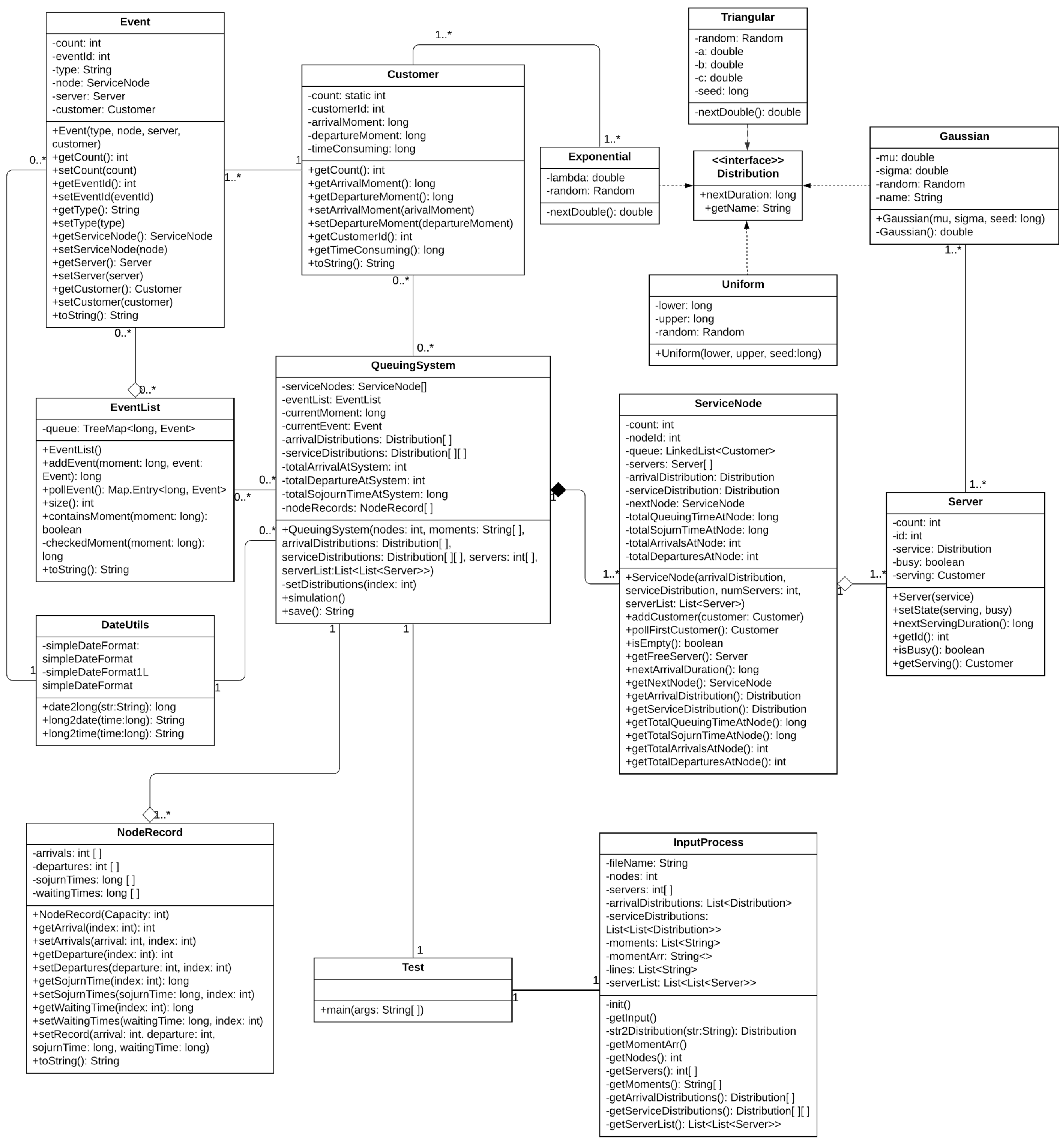

Event
-count: int
-eventId: int
-type: String
-node: ServiceNode
-server: Server
-customer: Customer
+Event(type, node, server, customer)
+getCount(): int
+setCount(count)
+getEventId(): int
+setEventId(eventId)
+getType(): String
+setType(type)
+getServiceNode(): ServiceNode
+setServiceNode(node)
+getServer(): Server
+setServer(server)
+getCustomer(): Customer
+setCustomer(customer)
+toString(): String
Customer
-count: static int
-customerId: int
-arrivalMoment: long
-departureMoment: long
-timeConsuming: long
+getCount(): int
+getArrivalMoment(): long
+getDepartureMoment(): long
+setArrivalMoment(arivalMoment)
+setDepartureMoment(departureMoment)
+getCustomerId(): int
+getTimeConsuming(): long
+toString(): String
Triangular
-random: Random
-a: double
-b: double
-c: double
-seed: long
-nextDouble(): double
Exponential
-lambda: double
-random: Random
-nextDouble(): double
<<interface>>
Distribution
+nextDuration: long
+getName: String
Gaussian
-mu: double
-sigma: double
-random: Random
-name: String
+Gaussian(mu, sigma, seed: long)
-Gaussian(): double
Uniform
-lower: long
-upper: long
-random: Random
+Uniform(lower, upper, seed:long)
EventList
-queue: TreeMap<long, Event>
+EventList()
+addEvent(moment: long, event: Event): long
+pollEvent(): Map.Entry<long, Event>
+size(): int
+containsMoment(moment: long): boolean
-checkedMoment(moment: long): long
+toString(): String
QueuingSystem
-serviceNodes: ServiceNode[]
-eventList: EventList
-currentMoment: long
-currentEvent: Event
-arrivalDistributions: Distribution[ ]
-serviceDistributions: Distribution[ ][ ]
-totalArrivalAtSystem: int
-totalDepartureAtSystem: int
-totalSojournTimeAtSystem: long
-nodeRecords: NodeRecord[ ]
+QueuingSystem(nodes: int, moments: String[ ], arrivalDistributions: Distribution[ ], serviceDistributions: Distribution[ ][ ], servers: int[ ], serverList:List<List<Server>>)
-setDistributions(index: int)
+simulation()
+save(): String
ServiceNode
-count: int
-nodeId: int
-queue: LinkedList<Customer>
-servers: Server[ ]
-arrivalDistribution: Distribution
-serviceDistribution: Distribution
-nextNode: ServiceNode
-totalQueuingTimeAtNode: long
-totalSojurnTimeAtNode: long
-totalArrivalsAtNode: int
-totalDeparturesAtNode: int
+ServiceNode(arrivalDistribution, serviceDistribution, numServers: int, serverList: List<Server>)
+addCustomer(customer: Customer)
+pollFirstCustomer(): Customer
+isEmpty(): boolean
+getFreeServer(): Server
+nextArrivalDuration(): long
+getNextNode(): ServiceNode
+getArrivalDistribution(): Distribution
+getServiceDistribution(): Distribution
+getTotalQueuingTimeAtNode(): long
+getTotalSojurnTimeAtNode(): long
+getTotalArrivalsAtNode(): int
+getTotalDeparturesAtNode(): int
Server
-count: int
-id: int
-service: Distribution
-busy: boolean
-serving: Customer
+Server(service)
+setState(serving, busy)
+nextServingDuration(): long
+getId(): int
+isBusy(): boolean
+getServing(): Customer
DateUtils
-simpleDateFormat: simpleDateFormat
-simpleDateFormat1L simpleDateFormat
+date2long(str:String): long
+long2date(time:long): String
+long2time(time:long): String
NodeRecord
-arrivals: int [ ]
-departures: int [ ]
-sojurnTimes: long [ ]
-waitingTimes: long [ ]
+NodeRecord(Capacity: int)
+getArrival(index: int): int
+setArrivals(arrival: int, index: int)
+getDeparture(index: int): int
+setDepartures(departure: int, index: int)
+getSojurnTime(index: int): long
+setSojurnTimes(sojurnTime: long, index: int)
+getWaitingTime(index: int): long
+setWaitingTimes(waitingTime: long, index: int)
+setRecord(arrival: int. departure: int, sojurnTime: long, waitingTime: long)
+toString(): String
Test
+main(args: String[ ])
InputProcess
-fileName: String
-nodes: int
-servers: int[ ]
-arrivalDistributions: List<Distribution>
-serviceDistributions: List<List<Distribution>>
-moments: List<String>
-momentArr: String<>
-lines: List<String>
-serverList: List<List<Server>>
-init()
-getInput()
-str2Distribution(str:String): Distribution
-getMomentArr()
-getNodes(): int
-getServers(): int[ ]
-getMoments(): String[ ]
-getArrivalDistributions(): Distribution[ ]
-getServiceDistributions(): Distribution[ ][ ]
-getServerList(): List<List<Server>>
1..*
1..*
0..*
1..*
1
0..*
0..*
0..*
0..*
0..*
0..*
0..*
1
1
1
1
1
1..*
1
1..*
1..*
1..*
1
1..*
1
1
1

Figure 3. UML Diagram

5.0 Testing Results Analysis

5.0.1 Performance Metrics Description

The simulation outputs are the projected performance metrics generated based on the input parameters. The capstone team identified five key performance metrics shown in Table 6 below.

Table 6. The List of Performance Metrics

Performance metrics
Avg.Time in system (min:sec)
Avg.Time per station (min:sec)
Avg.wait time per station (min:sec)
Throughput rate (per min)
Server utilization rate

Among all of the five performance metrics, the three performance metrics including average time in system, the average time in each station, and the average wait time in each station, are the key indicators that generally reflect the average time that a customer spends or waits during the system. If there is a large wait time, it means a long wait for the customers, which could result in an operation problem because it could possibly defer customers who were considering dining and cause poor customer experience. Next, throughput rate refers to the average rate of departure per hour. A high throughput rate indicates a quick speed at which something is processed. For the server utilization rate, it is an important performance indicator in business that essentially measures how much of this available time is actually used in service or other production work [26].

5.1 Result Summary

There are 7 input files for each single day from Monday to Sunday. Table 7 below shows the performance metrics corresponding to each of the seven days generated through the discrete event simulation. The generated measurement of performance demonstrates the current state of the restaurant scenario.

Table 7. The Test Outputs for One Week [Appendix C.2]

Performance metrics	**Mon**	**Tue**	**Wed**	**Thurs**	**Fri**	**Sat**	**Sun**
Avg.Time in system (min:sec)	20:40	20:34	20:42	20:20	19:59	19:48	19:32
Avg.Time in station 1: Taking order (min:sec)	03:30	03:30	03:29	03:24	03:22	03:15	03:27
Avg.Time in station 2: Cooking food (min:sec)	17:39	18:01	18:38	18:25	18:02	20:07	18:34
Avg.wait time in station 1:	00:28	00:37	00:48	00:43	00:57	01:02	01:30

Taking order (min:sec)							
Avg.wait time in station 2: Cooking food (min:sec)	01:54	02:12	02:55	03:17	03:06	06:50	06:14
Throughput rate (per min)	0.06	0.06	0.06	0.05	0.04	0.03	0.03
Server utilization rate	0.78	0.75	0.75	0.75	0.73	0.67	0.66

Based on the testing results, the performance metrics are relatively even among the seven days. Specifically, the average time spent in a system is around 20 minutes for almost each day, where the average time spent in service station of taking order is around 3 mins, and the average time spent in service station of cooking food is around 18 mins. Also, the average wait time in the service station of taking orders is no longer than 2 minutes; the average wait time in the service station of cooking food is no longer than 7 minutes. The server utilization rates of the seven days are also within a small range of around 75%.

As there is no significant difference on the metrics from day to day, the team just picked Monday as an example to observe more details on a single day. The generated statistical graph shown below visualizes the outputs and presents how some of the performance metrics such as average time in system and average wait time change over each hour time slot. From this graph, the user can intuitively perceive which time period that customer spends more time in a specific service station, and which time period that the customer spends lower time in a specific service station.

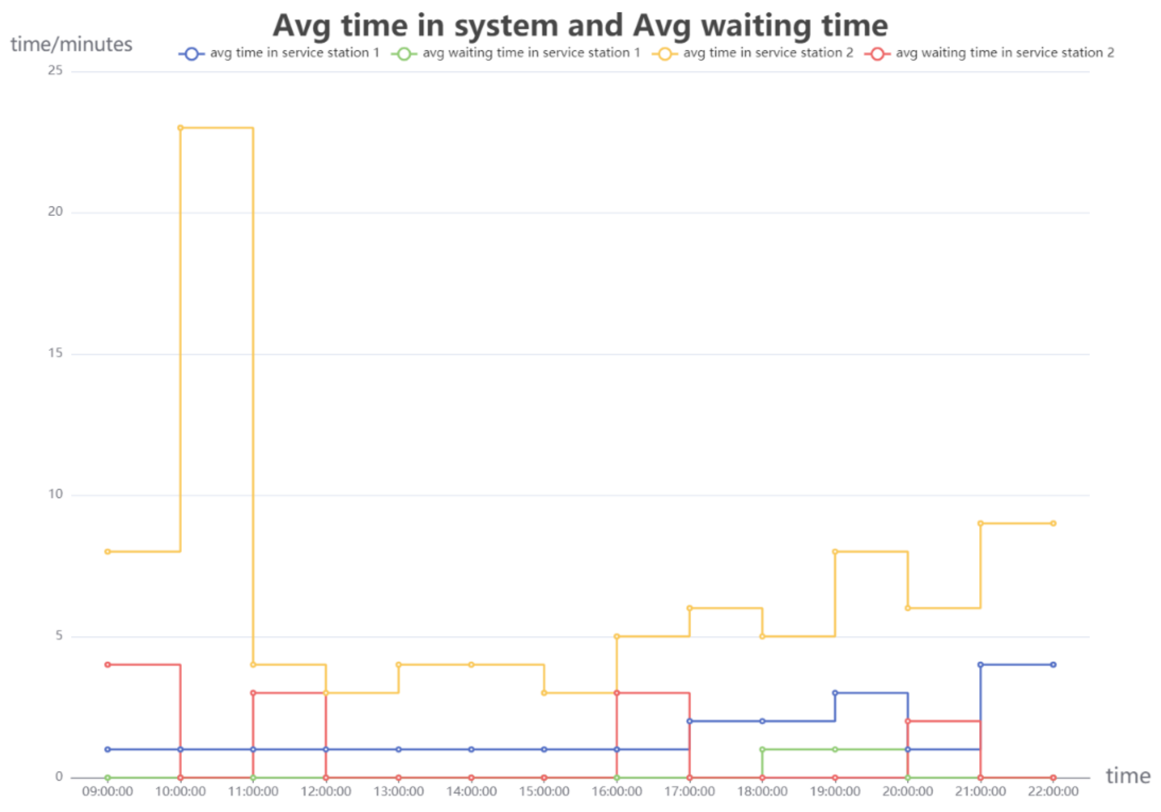


Figure 4. The statistical graph for Monday

In addition, to explore the general daily operations of the restaurant, the team also split the whole day into four time segments: Morning time 9:00-12:00, Lunch time 12:00 - 15:00, Afternoon 15:00 - 18:00, Dinner time 18:00 - 22:00; with the assumption that each segment has a new initiation to start simulation. Table 8 shows the performance metrics corresponding to each of the four segments through the discrete event simulation.

Table 8. The Test Outputs on Monday [Appendix C.3]

Performance metrics	9:00-12:00	12:00 - 15:00	15:00 - 18:00	18:00 - 22:00
Avg.Time in system (min:sec)	19:24	19:24	19:04	19:42
Avg.Time in station 1 (min:sec)	02:57	03:02	02:39	03:18
Avg.Time in station 2 (min:sec)	17:12	16:36	16:53	17:23
Avg.wait time in station 1 (min:sec)	00:16	00:22	00:19	01:03
Avg.wait time in station 2 (min:sec)	02:33	02:16	01:40	02:29
Throughput rate (per min)	0.07	0.04	0.07	0.03
Server utilization rate	0.75	0.78	0.79	0.72

Based on the test result of the four time segments on Monday, the performance metrics are also relatively even among different time periods. However, it clearly indicates that the Dinner time 18:00 - 22:00 is usually busier than other time because of the highest average time spent in the system and the highest waiting time in both two service stations. In contrast, the Afternoon 15:00 - 18:00 is the least busy time during the whole day.

To summarize, the output of the final design can provide multiple key performance indicators to the enterprise users to assess their current business, and visualizes some statistics to let the users understand more intuitively. Overall, the testing results for the use case can validate the feasibility of the design.

5.2 Sensitivity Analysis

Sensitivity analysis was also conducted based on the operations of the restaurant chain use case. The sensitivity analysis in the following section is to assess what potential benefits that the projected performance metrics can bring to through comparing the generated outputs of current staffing solutions and the alternative staffing solutions.

5.2.1 Change on Staffing Level

To evaluate the current staffing solution, the team made changes on the staffing level by reducing the number of servers on each service station. For Monday of the one week, the number of servers in the 1st station was reduced from 3 to 2, the number of servers in the 2nd station was reduced from 4 to 3. The specific change on staffing level is shown below:

Table 9. The Alternative Staffing Level

Staffing level	Service station #1 Taking order	Server 1: 9:00 - 17:00 Server 2: 17:00 - 22:00
	Service station #2 Cooking food	Server 3: 6:00 - 15:00 Server 4: 15:00 - 20:00 Server 5: 17:00 - 22:30

Table 10 below shows the generated performance metrics generated by the input parameters with the staffing level updates [Appendix C.4].

Table 10. The Updated Test Outputs on Monday

Performance metrics	9:00 - 12:00	12:00 - 15:00	15:00 - 18:00	18:00 - 22:00
Avg.Time in system (min:sec)	20:04	19:24	19:57	19:42
Avg.Time in station 1 (min:sec)	02:57	03:02	02:39	03:18
Avg.Time in station 2 (min:sec)	17:32	16:36	17:51	17:23
Avg.wait time in station 1 (min:sec)	00:16	00:22	00:19	01:03
Avg.wait time in station 2 (min:sec)	03:14	02:16	02:40	02:29
Throughput rate (per min)	0.06	0.04	0.07	0.03
Server utilization rate	0.72	0.78	0.76	0.72

Compared with the output from the original input in section 5.1 and the output from the updated input, the disparity indicates that the change in performance metrics depends on the number of servers available in that service station in that time segment. The team found that there was a slight increase in average time that customers spent in the system for the segment 9:00-12:00 and 15:00 - 18:00 due to the increase in waiting time in the second service station.

Therefore, the team can preliminarily conclude that a small reduction in the number of servers or shift hours might bring labor cost benefit to the restaurant client with only a small change on other metrics. While on the other hand, if the client is more focused on customer experience, they may avoid reducing the number of servers or work hours. Overall, through changing the input parameters, the performance indicators can present useful information to the enterprise users to help them assess their employee schedule and make further decisions based on their own business objective.

5.3 Validation Against Project Requirements

The section validates the final design testing results against the project requirements including functions, objectives and constraints. Based on the assessment, all the requirements are successfully met by this design. The project requirements and assessments are listed in the tables below. For some of the requirements, a short description for the assessment is also provided to each of them.

Table 11. Validate Final Design Against Project Requirements

Functions	Assessment
F1. Read in input parameters (i.e., derived distributions) to prepare the construction of a discrete event simulation model	✓ **Meet**
F2. Generate arrival and service events based on derived arrival and service time distributions and parameters	✓ **Meet** Arrival Event: Time and number of customers entered the system Service Event: Taking order and Cooking food
F3. Implement a discrete event simulation algorithm to project performance metrics, such as average service time and time in system, etc	✓ **Meet**
F4. Project performance metrics in a meaningful way to the user	✓ **Meet**
F5. Derive arrival and service time distribution, and staffing level from historical data	✓ **Meet** Detrive arrival distribution and staffing level based on the historical data from Ceridian's database. The service time distribution is generated by research papers and secondary sources.
F6. Include options to visualize the final results to improve readability (i.e. text, chart)	✓ **Meet** The default format for final results is text output. Visualize the final results in line charts to improve readability.

Objectives	Assessment
O1. Validity	✓ **Meet**
O2. Scalability	✓ **Meet** The designed prototype is able to simulate a real-world restaurant system in a reasonably long execution time, with 2 service stations and 2 to 9 servers at each station.
O3. A diverse set of performance metrics	✓ **Meet** There are 5 types of performance metrics that can be generated from the design.
O4. Easy integration into Dayforce	✓ **Meet** The discrete event simulation algorithm is coded in Java, therefore, there are no any

	incompatibilities with DayForce's Platform.

Constraints	Assessment
C1. The discrete event simulation algorithm must be coded in C# or Java	✓ **Meet** The discrete event simulation application is programmed in Java.
C2. The executable prototype should not depend on any commercial software package	✓ **Meet** No commercial software packages imported.
C3. The final design must implement a fully developed discrete event simulation algorithm	✓ **Meet**
C4. The simulation output must be visible and manipulable	✓ **Meet** There are two formats of the simulation output, text and chart.
C5. The simulation input should be a .csv or .txt file	✓ **Meet** The format of simulation input is text file.

6.0 Future Work Recommendations

For the current design, users can customize multiple input parameters according to their current employee's shift schedule and arrival rate data to simulate and visualize the performance metrics. For the next step, the design can consider more possible conditions that could happen during the system such as customers exiting a queue halfway, service time distribution changing over time, etc. Therefore, the whole design will develop a more complex queuing network to apply to more real-life scenarios.

In addition, the current design only focuses on backend development. As a future step, the design can add a graphical user interface to give easier access to the users. An easy-to-use interface will improve the efficiency of the user input process, enabling users to learn the system more quickly and have a better user experience.

7.0 Conclusion

The capstone team successfully developed the discrete event simulation application for future users to project performance metrics. From the restaurant case study and the following sensitivity analysis, the team has demonstrated how the application can be utilized to provide recommendations to improve performance in the context of the current scheduling module. The generated key performance metrics allow Ceridian’s HCM platform users to gain more data insights for strategic decision making. The performance metrics can clearly demonstrate the current performance of a specific process, and present how effectively the business is achieving key business objectives. It brings value to business enterprises on staffing schedules and customer experience, helps them to improve or optimize their current schedules and thereby benefit the whole business.

Reference

[1] Ceridian.com. 2020. Global HCM Software Company | Culture-Driven Innovation | Ceridian. [online] Available at: https://www.ceridian.com/company [Accessed: 26-Mar-2021].

[2] Ceridian.com. 2020. Employee Scheduling Software - Dayforce | Ceridian. [online] Available: https://www.ceridian.com/ca/products/dayforce/workforce-management/scheduling. [Accessed: 26-Mar-2021].

[3] “Queueing System,” *Queueing System - an overview | ScienceDirect Topics*. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/queueing-system#:~:text=A%20queueing%20system%20can%20be,(or%20waiting%20line)%20develops. [Accessed: 26-Mar-2021].

[4] "Whitbread", Whitbread.co.uk, 2021. [Online]. Available: https://www.whitbread.co.uk/. [Accessed: 26-Mar-2021].

[5] Vesna Bosilj Vukšić Faculty of Economics and Business, “Utilization of Discrete Event Simulation in Business Processes Management Projects: a Literature Review,” Utilization of Discrete Event Simulation in Business Processes Management Projects: a Literature Review | Journal of Information and Organizational Sciences. [Online]. Available: https://jios.foi.hr/index.php/jios/article/view/1104. [Accessed: 26-Mar-2021].

[6] "4 Definitive Discrete Event Simulation Examples | MOSIMTEC", *MOSIMTEC*, 2021. [Online]. Available: https://mosimtec.com/discrete-event-simulation-examples/. [Accessed: 26-Mar-2021].

[7] H. Rajaei and R. Khakzad, “[PDF]A Real-World Example for Student Learning: BTSU Cafeteria Simulation: Semantic Scholar,” SEMANTIC SCHOLAR, 01-Jan-1970. [Online]. Available: https://www.semanticscholar.org/paper/A-Real-World-Example-for-Student-Learning%3A-BTSU-Rajaei-Khakzad/04a1388b30c8485c89103fbb480cbfc0df2f7f71?p2df. [Accessed: 26-Mar-2021].

[8] K. Farahmand and A. F. G. Martinez, “[PDF] Simulation and animation of the operation of a fast food restaurant: Semantic Scholar,” Semantic Scholar, 01-Jan-1996. [Online]. Available: https://www.semanticscholar.org/paper/Simulation-and-animation-of-the-operation-of-a-fast-Farahmand-Martinez/5a911950674fd19c021c8d830ce67a2764eaf01d. [Accessed: 26-Mar-2021].

[9] M. McKnew, A. Field, and P. Kiessler, “A Simulation Comparison of Buffet Restaurants: Applying Monte Carlo Modeling,” Academia. [Online]. Available: https://www.academia.edu/29275744/A_Simulation_Comparison_of_Buffet_Restaurants_Applying_Monte_Carlo_Modeling. [Accessed: 26-Mar-2021].

[10] S. L. Jaynes and J. O. Hoffman, “Discrete event simulation for quick service restaurant traffic analysis,” IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/717489. [Accessed: 26-Mar-2021].

[11] Q. Chen, W. Zhang, Z. Zhang and Z. Zhai, "Evaluation of Various Turbulence Models in Predicting Airflow and Turbulence in Enclosed Environments by CFD: Part 2—Comparison with Experimental Data from Literature", ResearchGate, 2021. [Online]. Available: https://www.researchgate.net/publication/238046939_Evaluation_of_Various_Turbulence_Models_in_Predicting_Airflow_and_Turbulence_in_Enclosed_Environments_by_CFD_Part_2-Comparison_with_Experimental_Data_from_Literature. [Accessed: 26- Mar- 2021].

[12] M. lingam, “Big O is really Big! (PART 1),” Medium, 23-Apr-2019. [Online]. Available: https://blog.bitsrc.io/big-o-is-really-big-part-1-3e837ca64b1c#:~:text=Whenever we define scalability in,on, instead of using performance. [Accessed: 27-Mar-2021].

[13] "9 Queuing Metrics You Should Have at Your Fingertips", Lavi Industries, 2021. [Online]. Available: https://www.lavi.com/en/resources-detail/9-queuing-metrics.[Accessed: 26- Mar- 2021].

[14] Overview and Project Introduction. (2020)

[15] G. Dagkakis and C. Heavey, "A review of open source discrete event simulation software for operations research", 2020. [Online] Available at: https://link.springer.com/article/10.1057/jos.2015.9/tables/1[Accessed: 26- Mar- 2021].

[16] “Guidance Document on Emission Scenario Documents,” *Series on Emission Scenario Documents*, pp. 1–3, 2000. [Online]. Available: https://tl9000.org/handbooks/documents/guidance_document_design_for_XDFx_v1.3.pdf.

[17] "Software Fault Tolerance", Users.ece.cmu.edu, 2021. [Online]. Available: https://users.ece.cmu.edu/~koopman/des_s99/sw_fault_tolerance/#:~:text=Software%20fault%20tolerance%20is%20the,in%20accordance%20with%20the%20specification. [Accessed: 26- Mar- 2021].

[18] Advantages and Disadvantages of Object-Oriented Programming (OOP). The Saylor Foundation, 2021, p. 2. [Online]. Available: https://resources.saylor.org/wwwresources/archived/site/wp-content/uploads/2013/02/CS101-2.1.2-AdvantagesDisadvantagesOfOOP-FINAL.pdf.

[19] "What is the average time from start to finish the meals when we dine in a restaurant?", Quora, 2021. [Online]. Available: https://www.quora.com/What-is-the-average-time-from-start-to-finish-the-meals-when-we-dine-in-a-restaurant. [Accessed: 26- Mar- 2021].

[20] "Restaurant cook times and putting pork on your menu," *Ontario Pork News*, 04-Apr-2017. [Online]. Available: https://www.ontariopork.on.ca/retail/News/restaurant-cook-times-and-putting-pork-on-your-menu-1. [Accessed: 26-Mar-2021].

[21] Biller, B. and Nelson, B. L. 2002. Answers to the top ten input modeling questions. In Proceedings of the 2002 Winter Simulation Conference, eds. E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, 35–40. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers. Available via <https://www.informs-sim.org/wsc02papers/005.pdf> [accessed September 1, 2004].

[22] S. Curin, J. Vosko, E. Chan and O. Tsimhoni, REDUCING SERVICE TIME AT A BUSY FAST FOOD RESTAURANT ON CAMPUS. Proceedings of the 2005 Winter Simulation Conference M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds., 2021, pp. 2628-2635. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers. Available via <https://informs-sim.org/wsc05papers/331.pdf> [accessed September 1, 2005].

[23] "Discrete-event simulation", En.wikipedia.org, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Discrete-event_simulation. [Accessed: 18- Mar- 2021].

[24] A Simulation-Based Approach of QoS-Aware Service Selection in Mobile Edge Computing
J. Huang, Y. Lan, and M. Xu, "A Simulation-Based Approach of QoS-Aware Service Selection in Mobile Edge Computing," Wireless Communications and Mobile Computing, 01-Nov-2018. [Online]. Available: https://www.hindawi.com/journals/wcmc/2018/5485461/. [Accessed: 18- Mar- 2021].

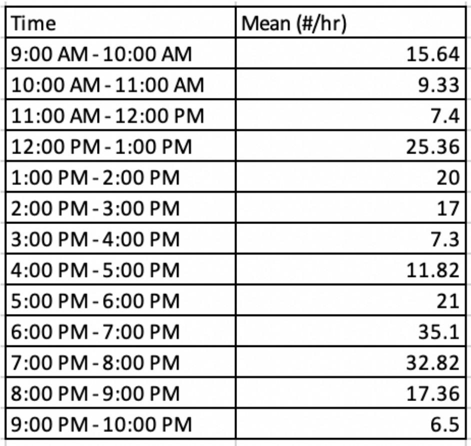
[25] "WSZiB : Lectures of Prof. dr Peter Sloot", Artemis.wszib.edu.pl, 2021. [Online]. Available: https://artemis.wszib.edu.pl/~sloot/1_4.html. [Accessed: 18- Mar- 2021].

[26] H. Cohen, "How Utilization Rate Affects Your Profitability (& How to Increase It)", Workamajig.com, 2021. [Online]. Available: https://www.workamajig.com/blog/utilization-rate#:~:text=Utilization%20rate%20is%20essentially%20a,a%20utilization%20rate%20of%2083.34%25. [Accessed: 26- Mar- 2021].

Appendix

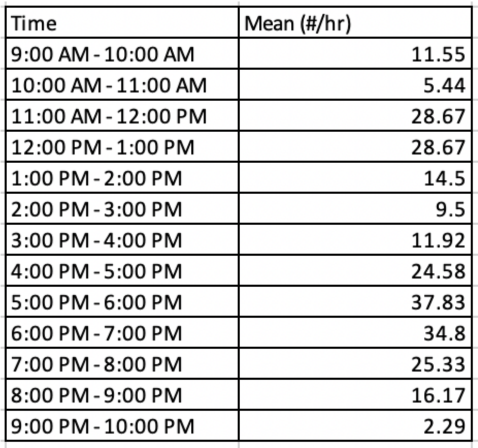
Appendix A - Estimated Arrival Rate

Table 12. Average Arrival Rate for Monday



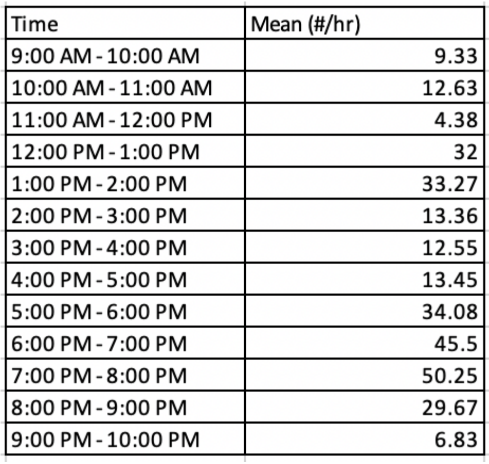
Time	Mean (#/hr)
9:00 AM - 10:00 AM	15.64
10:00 AM - 11:00 AM	9.33
11:00 AM - 12:00 PM	7.4
12:00 PM - 1:00 PM	25.36
1:00 PM - 2:00 PM	20
2:00 PM - 3:00 PM	17
3:00 PM - 4:00 PM	7.3
4:00 PM - 5:00 PM	11.82
5:00 PM - 6:00 PM	21
6:00 PM - 7:00 PM	35.1
7:00 PM - 8:00 PM	32.82
8:00 PM - 9:00 PM	17.36
9:00 PM - 10:00 PM	6.5

Table 13. Average Arrival Rate for Tuesday



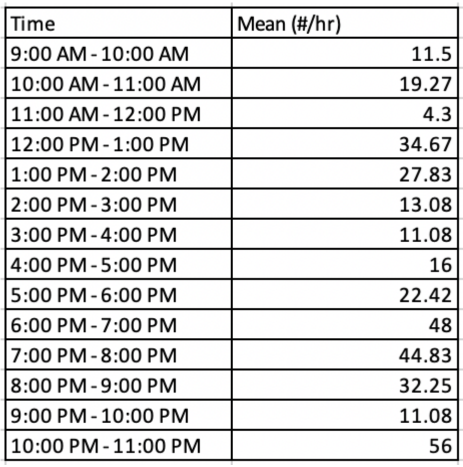
Time	Mean (#/hr)
9:00 AM - 10:00 AM	11.55
10:00 AM - 11:00 AM	5.44
11:00 AM - 12:00 PM	28.67
12:00 PM - 1:00 PM	28.67
1:00 PM - 2:00 PM	14.5
2:00 PM - 3:00 PM	9.5
3:00 PM - 4:00 PM	11.92
4:00 PM - 5:00 PM	24.58
5:00 PM - 6:00 PM	37.83
6:00 PM - 7:00 PM	34.8
7:00 PM - 8:00 PM	25.33
8:00 PM - 9:00 PM	16.17
9:00 PM - 10:00 PM	2.29

Table 14. Average Arrival Rate for Wednesday



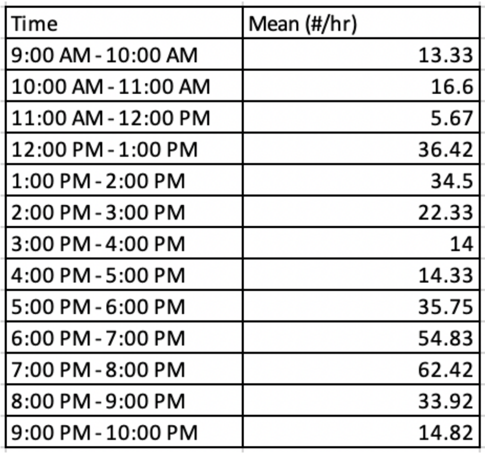
Time	Mean (#/hr)
9:00 AM - 10:00 AM	9.33
10:00 AM - 11:00 AM	12.63
11:00 AM - 12:00 PM	4.38
12:00 PM - 1:00 PM	32
1:00 PM - 2:00 PM	33.27
2:00 PM - 3:00 PM	13.36
3:00 PM - 4:00 PM	12.55
4:00 PM - 5:00 PM	13.45
5:00 PM - 6:00 PM	34.08
6:00 PM - 7:00 PM	45.5
7:00 PM - 8:00 PM	50.25
8:00 PM - 9:00 PM	29.67
9:00 PM - 10:00 PM	6.83

Table 15. Average Arrival Rate for Thursday



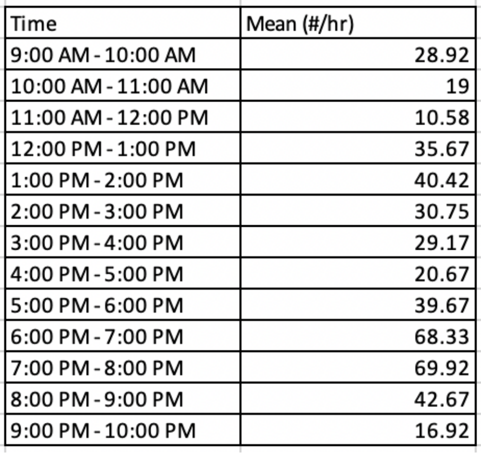
Time	Mean (#/hr)
9:00 AM - 10:00 AM	11.5
10:00 AM - 11:00 AM	19.27
11:00 AM - 12:00 PM	4.3
12:00 PM - 1:00 PM	34.67
1:00 PM - 2:00 PM	27.83
2:00 PM - 3:00 PM	13.08
3:00 PM - 4:00 PM	11.08
4:00 PM - 5:00 PM	16
5:00 PM - 6:00 PM	22.42
6:00 PM - 7:00 PM	48
7:00 PM - 8:00 PM	44.83
8:00 PM - 9:00 PM	32.25
9:00 PM - 10:00 PM	11.08
10:00 PM - 11:00 PM	56

Table 16. Average Arrival Rate for Friday



Time	Mean (#/hr)
9:00 AM - 10:00 AM	13.33
10:00 AM - 11:00 AM	16.6
11:00 AM - 12:00 PM	5.67
12:00 PM - 1:00 PM	36.42
1:00 PM - 2:00 PM	34.5
2:00 PM - 3:00 PM	22.33
3:00 PM - 4:00 PM	14
4:00 PM - 5:00 PM	14.33
5:00 PM - 6:00 PM	35.75
6:00 PM - 7:00 PM	54.83
7:00 PM - 8:00 PM	62.42
8:00 PM - 9:00 PM	33.92
9:00 PM - 10:00 PM	14.82

Table 17. Average Arrival Rate for Saturday



Time	Mean (#/hr)
9:00 AM - 10:00 AM	28.92
10:00 AM - 11:00 AM	19
11:00 AM - 12:00 PM	10.58
12:00 PM - 1:00 PM	35.67
1:00 PM - 2:00 PM	40.42
2:00 PM - 3:00 PM	30.75
3:00 PM - 4:00 PM	29.17
4:00 PM - 5:00 PM	20.67
5:00 PM - 6:00 PM	39.67
6:00 PM - 7:00 PM	68.33
7:00 PM - 8:00 PM	69.92
8:00 PM - 9:00 PM	42.67
9:00 PM - 10:00 PM	16.92

Table 18. Average Arrival Rate for Sunday

Time	Mean (#/hr)
9:00 AM - 10:00 AM	27.75
10:00 AM - 11:00 AM	35.33
11:00 AM - 12:00 PM	16.64
12:00 PM - 1:00 PM	31.5
1:00 PM - 2:00 PM	57.58
2:00 PM - 3:00 PM	52.67
3:00 PM - 4:00 PM	45.42
4:00 PM - 5:00 PM	44.92
5:00 PM - 6:00 PM	44.33
6:00 PM - 7:00 PM	54
7:00 PM - 8:00 PM	38.82
8:00 PM - 9:00 PM	20
9:00 PM - 10:00 PM	6.37

Appendix B - Estimated Staffing Level

Table 19. Estimated Staffing Level for Monday

Monday		
Server Station 1 (Take order)		
		Time Span
Morning		
Server 1	9:00 - 15:00	5.5 hours
Server 2	12:00 - 21:00	9 hours
Afternoon		
Server 3	17:00 - 22:00	5 hours
Server Station 2 (Cook)		
Morning		
Server 3	6:00 - 15:00	9 hours
Server 4	11:00 - 22:30	10.5 hours
Afternoon		
Server 5	15:00 - 20:00	6.5 hours
Server 6	17:00 - 22:30	5 hours

Table 20. Estimated Staffing Level for Tuesday

Tuesday		
Server Station 1 (Take order)		
		Time Span
Morning		
Server 1	9:00 - 14:45	5.25 hours
Server 2	12:00 - 21:00	8 hours
Afternoon		
Server 3	17:00 - 22:00	5 hours
Server Station 2 (Cook)		
Morning		
Server 4	6:00 - 14:00	7.5 hours
Server 5	9:00 - 11:00	2 hours
Server 6	11:00 - 22:30	10.5 hours
Afternoon		
Server 7	14:00 - 20:30	7.5 hours
Server 8	17:00 - 22:45	5.25 hours

Table 21. Estimated Staffing Level for Wednesday

Table 22. Estimated Staffing Level for Thursday

Wednesday		
Server Station 1 (Take order)		
		Time Span
Morning		
Server 1	9:00 - 17:00	7 hours
Afternoon		
Server 2	16:30 - 21:00	4.5 hours
Server 3	17:30 - 22:00	4.5 hours
Server Station 2 (Cook)		
Morning		
Server 4	9:00 - 15:00	6 hours
Server 5	11:30 - 22:00	10 hours
Afternoon		
Server 6	15:00 - 23:00	8 hours
Server 7	16:30 - 23:30	7 hours

Thursday		
Server Station 1 (Take order)		
		Time Span
Morning		
Server 1	9:00 - 15:00	5.5 hours
Afternoon		
Server 2	16:30 - 21:00	4.5 hours
Server 3	17:30 - 22:00	4.5 hours
Server Station 2 (Cook)		
Morning		
Server 4	9:00 - 15:00	6 hours
Server 5	11:30 - 22:00	9.5 hours
Afternoon		
Server 6	15:00 - 23:00	7.5 hours
Server 7	16:30 - 23:00	6 hours

Table 23. Estimated Staffing Level for Friday

Friday		
Server Station 1 (Take order)		
		Time Span
Morning		
Server 1	9:00 - 16:00	6 hours
Afternoon		
Server 2	16:00 - 21:00	5 hours
Server 3	17:30 - 22:00	4.5 hours
Server Station 2 (Cook)		
Morning		
Server 4	6:00 - 14:30	8 hours
Server 5	11:00 - 20:30	7.5 hours
Afternoon		
Server 6	15:00 - 22:00	6.5 hours
Server 7	15:00 - 23:30	8 hours
Server 8	16:30 - 23:30	6.5 hours

Table 24. Estimated Staffing Level for Saturday

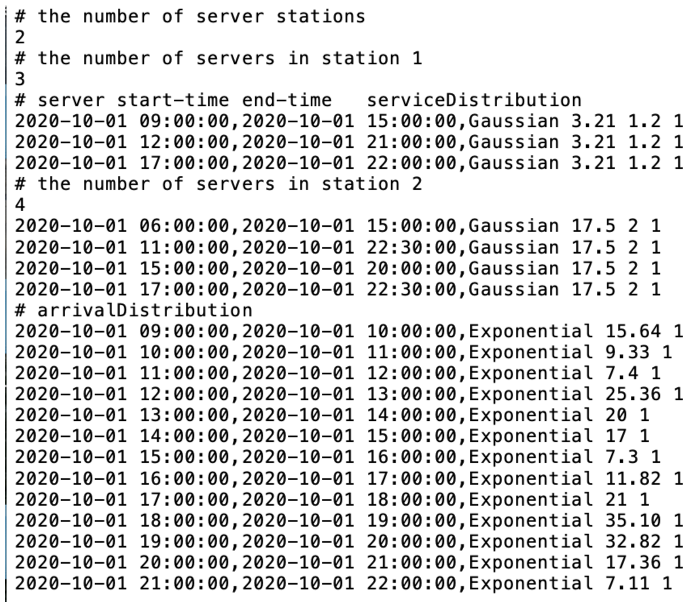
Saturday		
Server Station 1 (Take order)		
		Time Span
Morning		
Server 1	9:00 - 20:00	10 hours
Afternoon		
Server 2	12:00 - 21:45	8.25 hours
Server 3	17:00 - 23:00	6 hours
Server Station 2 (Cook)		
Morning		
Server 4	6:30 - 15:00	8 hours
Server 5	9:00 - 20:30	9.5 hours
Server 6	11:00 - 22:00	9 hours
Afternoon		
Server 7	15:00 - 23:30	8 hours
Server 8	15:00 - 23:45	8.25 hours

Table 25. Estimated Staffing Level for Sunday

Sunday		
Server Station 1 (Take order)		
		Time Span
Morning		
Server 1	8:30 - 15:00	6 hours
Server 2	10:00 - 18:00	7.5 hours
Afternoon		
Server 3	12:00 - 20:00	8 hours
Server 4	17:00 - 22:00	5 hours
Server Station 2 (Cook)		
Morning		
Server 5	6:30 - 11:00	4.5 hours
Server 6	10:30 - 20:00	8.5 hours
Server 7	10:30 - 20:30	9 hours
Afternoon		
Server 8	12:00 - 22:30	9.5 hours
Server 9	14:00 - 22:30	8 hours

Appendix C

Appendix C.1 - Input file “input.txt”



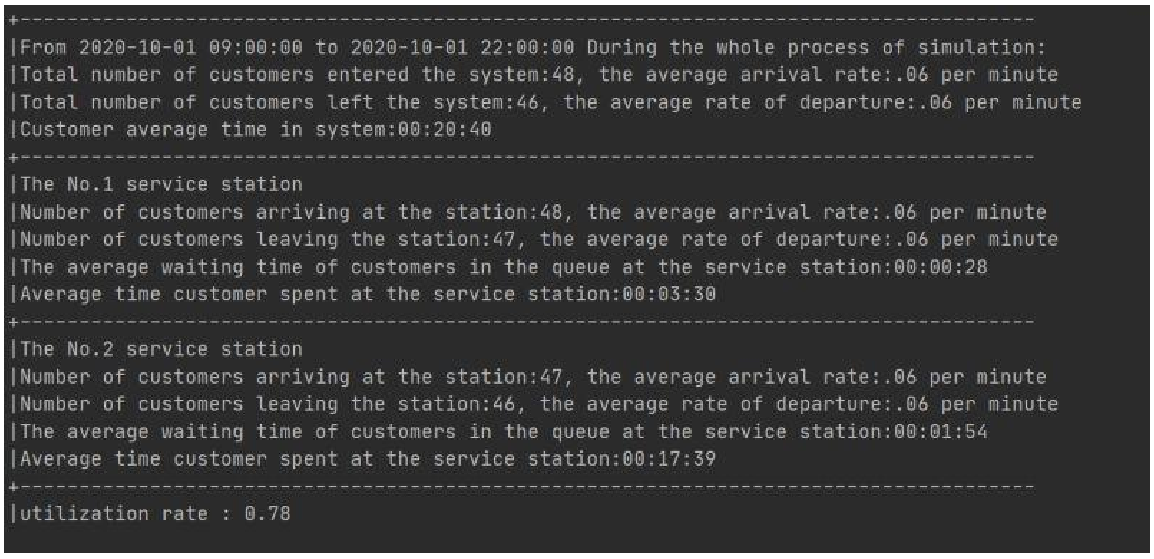
```
# the number of server stations
2
# the number of servers in station 1
3
# server start-time end-time   serviceDistribution
2020-10-01 09:00:00,2020-10-01 15:00:00,Gaussian 3.21 1.2 1
2020-10-01 12:00:00,2020-10-01 21:00:00,Gaussian 3.21 1.2 1
2020-10-01 17:00:00,2020-10-01 22:00:00,Gaussian 3.21 1.2 1
# the number of servers in station 2
4
2020-10-01 06:00:00,2020-10-01 15:00:00,Gaussian 17.5 2 1
2020-10-01 11:00:00,2020-10-01 22:30:00,Gaussian 17.5 2 1
2020-10-01 15:00:00,2020-10-01 20:00:00,Gaussian 17.5 2 1
2020-10-01 17:00:00,2020-10-01 22:30:00,Gaussian 17.5 2 1
# arrivalDistribution
2020-10-01 09:00:00,2020-10-01 10:00:00,Exponential 15.64 1
2020-10-01 10:00:00,2020-10-01 11:00:00,Exponential 9.33 1
2020-10-01 11:00:00,2020-10-01 12:00:00,Exponential 7.4 1
2020-10-01 12:00:00,2020-10-01 13:00:00,Exponential 25.36 1
2020-10-01 13:00:00,2020-10-01 14:00:00,Exponential 20 1
2020-10-01 14:00:00,2020-10-01 15:00:00,Exponential 17 1
2020-10-01 15:00:00,2020-10-01 16:00:00,Exponential 7.3 1
2020-10-01 16:00:00,2020-10-01 17:00:00,Exponential 11.82 1
2020-10-01 17:00:00,2020-10-01 18:00:00,Exponential 21 1
2020-10-01 18:00:00,2020-10-01 19:00:00,Exponential 35.10 1
2020-10-01 19:00:00,2020-10-01 20:00:00,Exponential 32.82 1
2020-10-01 20:00:00,2020-10-01 21:00:00,Exponential 17.36 1
2020-10-01 21:00:00,2020-10-01 22:00:00,Exponential 7.11 1
```

Figure 5. A Screenshot Example of the input file, for Monday

The format of the Input file “input.txt” for Monday

```
# the number of server stations
2
# the number of servers in station 1
3
# server start-time end-time   serviceDistribution
2020-10-01 09:00:00,2020-10-01 15:00:00,Gaussian 3.21 1.2
2020-10-01 12:00:00,2020-10-01 21:00:00,Gaussian 3.21 1.2 1
2020-10-01 17:00:00,2020-10-01 22:00:00,Gaussian 3.21 1.2
# the number of servers in station 2
4
2020-10-01 06:00:00,2020-10-01 15:00:00,Gaussian 17.5 2
2020-10-01 11:00:00,2020-10-01 21:30:00,Gaussian 17.5 2
2020-10-01 15:00:00,2020-10-01 20:00:00,Gaussian 17.5 2
2020-10-01 17:00:00,2020-10-01 22:30:00,Gaussian 17.5 2
# arrivalDistribution
2020-10-01 09:00:00,2020-10-01 10:00:00,Exponential 15.64 1
2020-10-01 10:00:00,2020-10-01 11:00:00,Exponential 9.33 1
2020-10-01 11:00:00,2020-10-01 12:00:00,Exponential 7.4 1
2020-10-01 12:00:00,2020-10-01 13:00:00,Exponential 25.36 1
2020-10-01 13:00:00,2020-10-01 14:00:00,Exponential 20 1
2020-10-01 14:00:00,2020-10-01 15:00:00,Exponential 17 1
2020-10-01 15:00:00,2020-10-01 16:00:00,Exponential 7.3 1
2020-10-01 16:00:00,2020-10-01 17:00:00,Exponential 11.82 1
2020-10-01 17:00:00,2020-10-01 18:00:00,Exponential 21 1
2020-10-01 18:00:00,2020-10-01 19:00:00,Exponential 35.10 1
2020-10-01 19:00:00,2020-10-01 20:00:00,Exponential 32.82 1
2020-10-01 20:00:00,2020-10-01 21:00:00,Exponential 17.36 1
2020-10-01 21:00:00,2020-10-01 22:00:00,Exponential 7.11 1
2020-10-01 22:00:00,2020-10-01 23:00:00,Exponential 1 1
```

Appendix C.2 - Output for One Week



```
+----------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:48, the average arrival rate:.06 per minute
|Total number of customers left the system:46, the average rate of departure:.06 per minute
|Customer average time in system:00:20:40
+----------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:48, the average arrival rate:.06 per minute
|Number of customers leaving the station:47, the average rate of departure:.06 per minute
|The average waiting time of customers in the queue at the service station:00:00:28
|Average time customer spent at the service station:00:03:30
+----------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:47, the average arrival rate:.06 per minute
|Number of customers leaving the station:46, the average rate of departure:.06 per minute
|The average waiting time of customers in the queue at the service station:00:01:54
|Average time customer spent at the service station:00:17:39
+----------------------------------------------------------------------------------
|utilization rate : 0.78
```

Figure 6. A Screenshot Example of the console output, for Monday

The outputs for the input files from Monday to Sunday

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:48, the average arrival rate:.06 per
minute
|Total number of customers left the system:46, the average rate of departure:.06 per
minute
|Customer average time in system:00:20:40
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:48, the average arrival rate:.06 per minute
|Number of customers leaving the station:47, the average rate of departure:.06 per
minute
|The average waiting time of customers in the queue at the service station:00:00:28
|Average time customer spent at the service station:00:03:30
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:47, the average arrival rate:.06 per minute
|Number of customers leaving the station:46, the average rate of departure:.06 per
minute
|The average waiting time of customers in the queue at the service station:00:01:54
|Average time customer spent at the service station:00:17:39
+-------------------------------------------------------------------------------------
|utilization rate : 0.78
```

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:52, the average arrival rate:.07 per
minute
|Total number of customers left the system:45, the average rate of departure:.06 per
minute
|Customer average time in system:00:20:34
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:52, the average arrival rate:.07 per minute
|Number of customers leaving the station:52, the average rate of departure:.07 per
minute
|The average waiting time of customers in the queue at the service station:00:00:37
|Average time customer spent at the service station:00:03:30
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:52, the average arrival rate:.07 per minute
|Number of customers leaving the station:45, the average rate of departure:.06 per
minute
|The average waiting time of customers in the queue at the service station:00:02:12
```

```
|Average time customer spent at the service station:00:18:01
+--------------------------------------------------------------------------------------
|utilization rate : 0.75
```

```
+--------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:45, the average arrival rate:.06 per
minute
|Total number of customers left the system:43, the average rate of departure:.06 per
minute
|Customer average time in system:00:20:42
+--------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:45, the average arrival rate:.06 per minute
|Number of customers leaving the station:45, the average rate of departure:.06 per
minute
|The average waiting time of customers in the queue at the service station:00:00:48
|Average time customer spent at the service station:00:03:29
+--------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:45, the average arrival rate:.06 per minute
|Number of customers leaving the station:43, the average rate of departure:.06 per
minute
|The average waiting time of customers in the queue at the service station:00:02:55
|Average time customer spent at the service station:00:18:38
+--------------------------------------------------------------------------------------
|utilization rate : 0.75
```

```
+--------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:41, the average arrival rate:.05 per
minute
|Total number of customers left the system:41, the average rate of departure:.05 per
minute
|Customer average time in system:00:20:20
+--------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:41, the average arrival rate:.05 per minute
|Number of customers leaving the station:41, the average rate of departure:.05 per
minute
|The average waiting time of customers in the queue at the service station:00:00:43
|Average time customer spent at the service station:00:03:24
+--------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:41, the average arrival rate:.05 per minute
|Number of customers leaving the station:41, the average rate of departure:.05 per
minute
|The average waiting time of customers in the queue at the service station:00:03:17
|Average time customer spent at the service station:00:18:25
+--------------------------------------------------------------------------------------
|utilization rate : 0.75
```

```
+--------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:35, the average arrival rate:.04 per
minute
|Total number of customers left the system:32, the average rate of departure:.04 per
minute
|Customer average time in system:00:19:59
+--------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:35, the average arrival rate:.04 per minute
|Number of customers leaving the station:35, the average rate of departure:.04 per
minute
|The average waiting time of customers in the queue at the service station:00:00:57
|Average time customer spent at the service station:00:03:22
+--------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:35, the average arrival rate:.04 per minute
|Number of customers leaving the station:32, the average rate of departure:.04 per
minute
|The average waiting time of customers in the queue at the service station:00:03:06
|Average time customer spent at the service station:00:18:02
+--------------------------------------------------------------------------------------
|utilization rate : 0.73
```

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:28, the average arrival rate:.04 per
minute
|Total number of customers left the system:27, the average rate of departure:.03 per
minute
|Customer average time in system:00:19:48
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:28, the average arrival rate:.04 per minute
|Number of customers leaving the station:28, the average rate of departure:.04 per
minute
|The average waiting time of customers in the queue at the service station:00:01:02
|Average time customer spent at the service station:00:03:15
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:28, the average arrival rate:.04 per minute
|Number of customers leaving the station:27, the average rate of departure:.03 per
minute
|The average waiting time of customers in the queue at the service station:00:06:50
|Average time customer spent at the service station:00:20:07
+-------------------------------------------------------------------------------------
|utilization rate : 0.67
```

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:22, the average arrival rate:.03 per
minute
|Total number of customers left the system:22, the average rate of departure:.03 per
minute
|Customer average time in system:00:19:32
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:22, the average arrival rate:.03 per minute
|Number of customers leaving the station:22, the average rate of departure:.03 per
minute
|The average waiting time of customers in the queue at the service station:00:01:30
|Average time customer spent at the service station:00:03:27
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:22, the average arrival rate:.03 per minute
|Number of customers leaving the station:22, the average rate of departure:.03 per
minute
|The average waiting time of customers in the queue at the service station:00:06:14
|Average time customer spent at the service station:00:18:34
+-------------------------------------------------------------------------------------
|utilization rate : 0.66
```

Appendix C.3 - Output for the segments on Monday

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 09:00:00 to 2020-10-01 12:00:00 During the whole process of simulation:
|Total number of customers entered the system:14, the average arrival rate:.08 per
minute
|Total number of customers left the system:12, the average rate of departure:.07 per
minute
|Customer average time in system:00:19:24
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:14, the average arrival rate:.08 per minute
|Number of customers leaving the station:14, the average rate of departure:.08 per
minute
|The average waiting time of customers in the queue at the service station:00:00:16
|Average time customer spent at the service station:00:02:57
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:14, the average arrival rate:.08 per minute
|Number of customers leaving the station:12, the average rate of departure:.07 per
minute
|The average waiting time of customers in the queue at the service station:00:02:33
|Average time customer spent at the service station:00:17:12
+-------------------------------------------------------------------------------------
|utilization rate : 0.75
```

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 12:00:00 to 2020-10-01 15:00:00 During the whole process of simulation:
|Total number of customers entered the system:8, the average arrival rate:.04 per minute
|Total number of customers left the system:8, the average rate of departure:.04 per
minute
|Customer average time in system:00:19:24
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:8, the average arrival rate:.04 per minute
|Number of customers leaving the station:8, the average rate of departure:.04 per minute
|The average waiting time of customers in the queue at the service station:00:00:22
|Average time customer spent at the service station:00:03:02
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:8, the average arrival rate:.04 per minute
|Number of customers leaving the station:8, the average rate of departure:.04 per minute
|The average waiting time of customers in the queue at the service station:00:02:16
|Average time customer spent at the service station:00:16:36
+-------------------------------------------------------------------------------------
|utilization rate : 0.78
```

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 15:00:00 to 2020-10-01 18:00:00 During the whole process of simulation:
|Total number of customers entered the system:14, the average arrival rate:.08 per
minute
|Total number of customers left the system:12, the average rate of departure:.07 per
minute
|Customer average time in system:00:19:04
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:14, the average arrival rate:.08 per minute
|Number of customers leaving the station:14, the average rate of departure:.08 per
minute
|The average waiting time of customers in the queue at the service station:00:00:19
|Average time customer spent at the service station:00:02:39
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:14, the average arrival rate:.08 per minute
|Number of customers leaving the station:12, the average rate of departure:.07 per
minute
|The average waiting time of customers in the queue at the service station:00:01:40
|Average time customer spent at the service station:00:16:53
+-------------------------------------------------------------------------------------
|utilization rate : 0.79
```

```
+-------------------------------------------------------------------------------------
|From 2020-10-01 18:00:00 to 2020-10-01 22:00:00 During the whole process of simulation:
|Total number of customers entered the system:10, the average arrival rate:.04 per
minute
|Total number of customers left the system:8, the average rate of departure:.03 per
minute
|Customer average time in system:00:19:42
+-------------------------------------------------------------------------------------
|The No.1 service station
|Number of customers arriving at the station:10, the average arrival rate:.04 per minute
|Number of customers leaving the station:10, the average rate of departure:.04 per
minute
|The average waiting time of customers in the queue at the service station:00:01:03
|Average time customer spent at the service station:00:03:18
+-------------------------------------------------------------------------------------
|The No.2 service station
|Number of customers arriving at the station:10, the average arrival rate:.04 per minute
|Number of customers leaving the station:8, the average rate of departure:.03 per minute
|The average waiting time of customers in the queue at the service station:00:02:29
|Average time customer spent at the service station:00:17:23
+-------------------------------------------------------------------------------------
|utilization rate : 0.72
```

Appendix C.4 - Input with Staffing Level updated for Sensitivity analysis

```
# the number of server stations
2
```

```
# the number of servers in station 1
2
# server start-time end-time   serviceDistribution
2020-10-01 09:00:00,2020-10-01 17:00:00,Gaussian 3.21 1.2 1
2020-10-01 17:00:00,2020-10-01 22:00:00,Gaussian 3.21 1.2 1
# the number of servers in station 2
3
2020-10-01 06:00:00,2020-10-01 15:00:00,Gaussian 17.5 2 1
2020-10-01 15:00:00,2020-10-01 20:00:00,Gaussian 17.5 2 1
2020-10-01 17:00:00,2020-10-01 22:30:00,Gaussian 17.5 2 1
# arrivalDistribution
```