Chapter

MODELING AND SIMULATION OF INTEGRATION SOLUTIONS BASED ON QUEUEING THEORY

Arlete Kelm Wiesner, Sandro Sawicki*[*]*, Fabricia Roos-Frantz and Rafael Z. Frantz

Unijuí University, Department of Exact Sciences
and Engineering, Ijuí, RS, Brazil

PACS 05.45-a, 52.35.Mw, 96.50.Fm. **Keywords:** Enterprise Application Integration, Simulation, Queueing Theory.

Abstract

Companies often acquire or develop their own applications to support decision making and improve their business processes. These applications compose the heterogeneous software ecosystem, since they are developed without considering their possible integration, making their reuse difficult. The Enterprise Application Integration (EAI) area provides methodologies, techniques and tools for companies to develop integration solutions. In this scenario, the integration technology called Guaraná enables software engineers to design integration solutions with a high level of abstraction using a model-centric approach. The computational analysis of integration solutions is an important activity that contributes to increase the quality of the integration solutions

[*]E-mail address: sawicki@unijui.edu.br

developed as well as to predict their behavior and to find possible performance bottlenecks. The approach generally adopted by software engineers is based on the design, implementation, and testing of integration solutions. However, implementing all these steps involves costs and risks. In this context, we have introduced the areas of Enterprise Application Integration and Discrete-event Simulation (DES) in this chapter and we propose a new approach to identify potential performance bottlenecks in the design phase. The Queueing Theory was used to develop a formal simulation model from a static conceptual model. An integration solution designed in the Guaraná integration technology was used as a case study. The experimental results of the simulation model allow us to evaluate the behavior of the integration solution when exposed to different workloads in order to identify possible performance bottlenecks in its structure. The proposed simulation model was verified and validated through formal techniques widely used in the literature.

1. Introduction

The area of Enterprise Application Integration uses computational techniques and tools so companies can integrate data and functionality offered by different applications. Hohpe e Woolf [1] assign an integration solution the task of making legacy applications reusable and all of their functionality available. The goal of an integration solution is to keep a series of data and application functionalities in sync or to develop new functionalities over existing ones, so that the integration solution does not change the applications [2, 3].

The presence of heterogeneous software ecosystems [4] in companies demands integration solutions. Several messaging-based technologies and integration standards support the design and implementation of integration solutions. Camel [5], Spring Integration [6], Mule [7], and Guaraná [8] are some of the technologies that support integration project design and development. An integration solution consists of a new application that must follow a software development process, which typically includes analysis, design, implementation, testing, and evolution phases.

Usually, the approach adopted by software engineers to analyze behavior under critical operating scenarios is to collect consistent data in the construction and execution of the integration solution. With this approach, performance bottleneck problems are only detected after the implementation and testing steps, increasing the time and cost of the application. In order to support this process, the Discrete-Event Simulation (DES) is known to be a technique that uses mathematical models that allow the study and analysis of behavior and performance without the necessity of changes in the real system, and thus predict a future behavior.

In this sense, this study presents a method for the transformation of static conceptual models into formal simulation models using Queueing Theory in order to identify possible performance bottlenecks in integration solutions still in the design phase. This work utilizes a real-world integration solution designed through Guaraná integration technology. The simulations were performed using the Simulink/SimEvent tool.

This chapter is organized as follows. Section 2 presents a brief background on Guaraná integration technology, its Domain Specific Language (DSL), and concrete syntax. This section also covers simulation systems and models as well as Queueing theory. Section 3 describes related work. Section 4 presents the Case Study and formulation of the integration problem. Section 5 introduces the Proposed Formal Model. Section 6 discusses the experimental results. Finally, Section 7 presents the conclusions.

2. Background

Section 2 presents a brief background on Guaraná integration technology, its Domain Specific Language (DSL), and concrete syntax. It also covers simulation systems and models, as well as Queueing theory concepts.

2.1. Enterprise Application Integration (EAI): Guaraná Technology

An enterprise application integration solution needs to keep data and information from the company's software ecosystem in sync as well as enable new functionality to be developed from existing ones without the applications being modified [9–11].

In this context, Guaraná technology is used to design enterprise application integration solutions so that software engineers can focus on creating models to solve the problem without having to worry about the technical details of their implementation.

2.1.1. Domain Specific Language (DSL)

Conceptual models designed in Guaraná technology are platform independent, hence software engineers do not rely on specific knowledge in low-level integration technologies to build integration solutions. This feature allows engineers to centralize their efforts on designing problem-solving models, thereby reducing the costs involved, as the engineer does not need to learn to use the distinct and often complex implementation technologies [8].

The concepts needed to design models of enterprise application integration solutions in Guaraná technology are shown in Figure 1. The functionalities and structure of an integration solution are completely defined using building blocks: ports, processes, tasks, slots, and links. Tasks are provided in toolboxes and are sorted according to their semantics.

According to Frantz et al. [8], the starting point of the concepts is the integration solution that represents a set of processes that cooperate to integrate several applications. The processes can be logically divided into processes that contain the communication logic with the integrated applications and processes that contain the solution integration logic. A message is an abstraction of a part of the information that is exchanged and transformed, through the integration solution, consisting of a header and a body. The header contains predefined properties, such as message identifier, correlation identifier, and message priority. The structure of messages depends entirely on the integration solutions in which they are involved.

Tasks perform message processing and may have one or more entries through which messages are received and one or more outputs through which messages resulting from processing are dispatched. In Guaraná technology, tasks are classified according to their semantics:

- *Router*: router tasks do not change the status of the messages they process, but they route messages through a process.

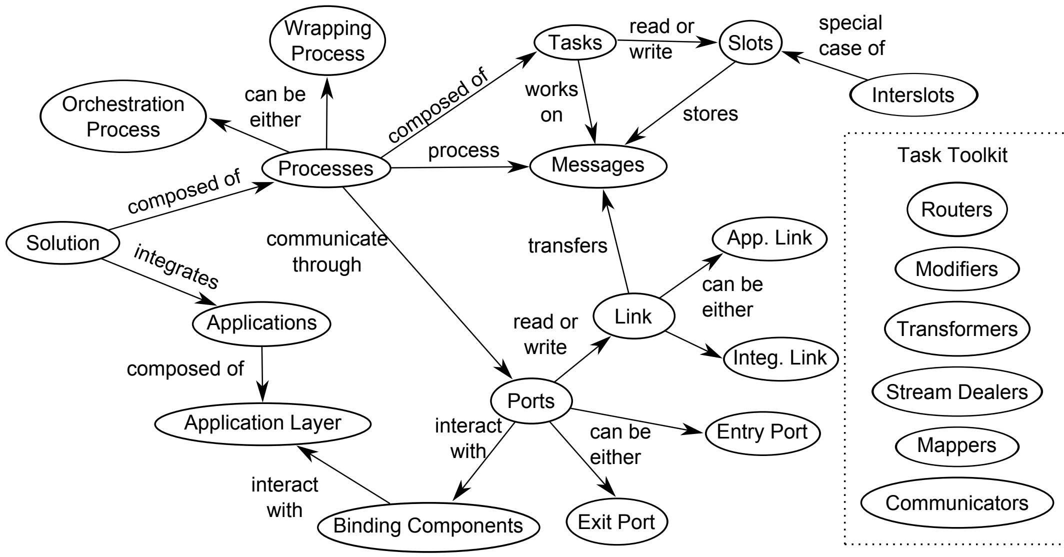


Figure 1. Conceptual Map of Guaraná Technology (from Frantz et al. [8])

- *Modifier*: modifying tasks add or remove message data so that it does not result in messages with a different schema. An example of this functionality is the task that adds data in the correlated message.
- *Transformer*: transforming tasks help translate one or more messages into a new message, with a different schema.
- *Stream Dealer*: tasks that work with a stream of bytes and help compress/decompress, encrypt/decrypt, or encode/decode messages.
- *Mapper*: mapping tasks change the format of messages that process, for example, from a stream of bytes in an XML document.
- *Communicator*: communicator-type tasks are used on ports to interact with communication components, commonly known as adapters.

2.1.2. Concrete Syntax

The concrete syntax used to represent the concepts of Guaraná technology is shown in Table 1. These graphic representations express all concepts needed to design integration solutions: Application, Process, EntryPort, ExitPort, IntegrationLink, ApplicationLink, Slot, and Task.

The graphical representation of the tasks is generic because they are provided in special toolboxes, known as toolkits. Therefore, they are not part of the language core. Beside the icon, there are protrusions that represent the entrances and exits of the task. These inputs and outputs are connected to slots, thus allowing connection and communication of tasks.

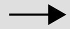
Notation	Description	Notation	Description
	*Application*		*IntegrationLink*
	*Process*		*ApplicationLink*
	*EntryPort*		*Slot*
	*ExitPort*		*Task*

Table 1. Symbology of concrete syntax (from Frantz et al. [8])

2.2. Systems and Simulation Model

A model can be defined as a representation of the real system and enables the study and analysis of a system without the need for actual implementation. The model can be sufficiently detailed or valid to allow the analyst to make the same decisions that would be made using the real system. Its main advantage lies in the possibility of making alterations in the system, for the purpose of studying and analyzing the results, without having to change the actual system [12].

The mathematical model is a representation of the real system. According to Hillier and Lieberman [13], one of the main strengths of the mathematical model is the abstraction of the essence of the problem, revealing its underlying structure and providing the cause-effect relationships contained in the system. In an analytical solution, mathematical formulas are used to represent the systems. These models have a high degree of abstraction and, therefore, of simplification in relation to the systems they represent [14].

From Law and Kelton's point of view [12], analytical solutions use traditional mathematical and statistical techniques to analyze and obtain accurate information about the system. However, many systems are so complex that the mathematical models that represent these systems are equally complex, making it impossible to use analytical solutions because the mathematical relationships are extensive and the computation complex. In this case, we try to use the simulation because it can model the complex characteristics of the system, including its stochastic (or probabilistic) and dynamic aspect, without making the model difficult to study.

The simulation can be done manually. However, running the simulation typically requires the generation and processing of a large amount of data. This is likely to be time-consuming and error-prone due to the numerous repetitions of the same mathematical operations. Thus, these simulated experiments should be inevitably carried out on a computer [13].

Kelton [12], initially classify the models as deterministic or stochastic. A model is deterministic when the variables do not have any probabilistic components. In the simulation of a deterministic model, the input data are considered constant, thus generating constant outputs. According to Sawicki et al. [15], a deterministic model for a known set of data input, produces a unique output set.

A stochastic or probabilistic model uses random variables as inputs that determine and produce random outputs. The output of a stochastic simulation must be analyzed as a statistical estimate of the real characteristics of a system. A stochastic simulation model represents a system that evolves probabilistically over time. To mimic real system performance, probability distributions are used to randomly generate the events that occur in the system [13]. Generally, stochastic models are more complex, but better represent a real system when compared to deterministic models.

2.3. Queueing Theory

Queueing theory is an analytical method that studies queueing through mathematical formulas and uses queueing models to represent the various types of queueing systems that emerge in practice. Fogliatti and Mattos [16] define Queueing theory as the analytical modeling of queueing systems that aims to determine and evaluate performance measures, which express the productivity/operability of this system. These measures include the number of customers in the queue, the waiting time for the service, and the idle time of the servers.

According to Fogliatti and Mattos [16], a queueing system is any process where clients coming from a population arrive to receive a service. When demand is greater than serviceability, in terms of flow, customers wait and, after service, leave the system. Hillier and Lieberman [13] report that the elements that make up the process are the population from which the arrivals of the clients in need service come from. Clients enter the queueing system and, due to unavailability of immediate service, form a queue. Clients are selected at certain times for a service by a rule known as queue discipline. After clients are serviced, they leave the queueing system.

Specifying a queueing model typically requires performance characteristics to be declared. These characteristics are arrival process, attendance process, number of servers, queueing capacity, population size, and queue discipline. In queueing system modeling, it is common to use a notation to summarize these six characteristics. In the following sections, we present this notation and the definitions of the six characteristics.

2.3.1. Queue Service Policies

The queue is characterized by the available space defined by the maximum number of clients it can hold. The queues are called finite or infinite according to capacity (size) limited or unlimited, respectively. Queueing models generally support unlimited capacity (infinite queue), even when limited, but this limitation is relatively large.

The queueing discipline defines the order in which customers are selected from the queue for service. The most used discipline is FIFO (First In, First Out), in which the service is performed according to the order of arrival, the first client that arrives is the first to be served. Other types of queueing discipline can also be used as: the last arriving client is the first to be served (LIFO - Last-In, First-Out), the service follows a random order (SIRO - service in random order), and selection in order of priority.

2.3.2. Formal Notation of Queue System

In order to classify a queueing system and briefly specify the queueing model, we use the notation proposed by Kendall [17], which has the expression A/B/s/K/m/Z, where:

- A represents the arrival process;
- B is the service time distribution;
- s is the number of servers;
- K the number of places in the system;
- m the calling population;
- Z the queue's discipline.

The types of distributions for times between successive arrivals and time of service are usually represented by the following nomenclature:

- M: exponencial (markovian or poisson distribution);
- Ek: an Erlang distribution with k as the shape parameter;
- Hk: hyper-exponential distribution with parameter k;
- D: deterministic;
- G: general.

According to Fogliatti and Mattos [16], some authors use a simplification of the notation, omitting the letters K, m, and Z when system capacity and population size are infinite (∞) and the queueing discipline is First In, First Out (FIFO).

For example, the notation M/M/1/∞/∞/FIFO or only M/M/1 indicates a queueing system with the following characteristics:

- M: arrivals occur according to a Poisson distribution;
- M: service time with exponential distribution;
- 1: a single server;
- ∞ : the buffer size is infinite;
- ∞ : the population size is infinite;
- $FIFO$: first-in-first-out discipline.

3. Related Work

An integration solution may be characterized as a system whose model is classified as a discrete event system. Several papers use Queueing Theory and simulation of models to evaluate the efficiency of discrete event systems. However, in the literature, there were no studies that analyzed enterprise application integration solutions through the simulation and/or Queueing Theory. Thus, this work is an important contribution to the development of research in the IEA area, as it may be considered an initial step in this approach. Considering this context, this study focused on studies that used Queueing Theory and simulation to comprehend and analyze performance measures of discrete-event simulation.

Abensur et al. [18] proposed conditions for the formulation of strategies consistent with the Brazilian self-service banking system, based on the evaluation of the performance variables obtained by the application of the queuing theory on the case studied. The authors estimated the arrival flow of users served by ATMs by using the information recorded in the bank's computer system. In order to estimate the average occupation time of self-service equipment, the authors used the individual processing time of the main transactions used by the users plus a time due to the idle permanence of the users in front of the equipment.

Changfu and Zhenyu [19] sought to establish mathematical and simulation models to analyze and evaluate the impact of the efficiency caused by the variation of requisition transactions in the queuing system of the e-Business environment. According to the authors, by means of the mathematical manipulation of the Queueing Theory formulas, the structural adjustment in the handling of the transaction request in the system of queues in the e-Business environment can exhaust the average waiting time. They also indicate that simplifying inbound and outbound documentation may decrease the average waiting time and, therefore, be expected to significantly decrease waiting time.

Kamali et al. [20] proposed, evaluated, and estimated network traffic monitoring based on the Queueing Theory. The authors state that in a heterogeneous environment, network traffic monitoring is necessary for evaluating the efficiency and reliability of constant network operations. They examine the performance and prediction of network traffic management, giving a suggestion to control traffic performance based on queuing theory, simulating the LAN network environment, which can be a model for the real world (LAN network environment). The authors used queuing theory variables and formulas to obtain parameters, such as bandwidth capacity, packet size, and distances between two computers that determine network traffic. They then performed the simulation of the heterogeneous network environment and used several tests to control parameters and values.

Camelo et al. [21] described, through the Queueing Theory and the simulation, the analysis of ship service characteristics in the shipment of iron ore and manganese at the Ponta da Madeira Maritime Terminal. The Queueing Theory was used in the study to analyze the average number of ships in queue and in the system and the average length of stay in queue and in the system in Pier I and Pier III and the simulation technique to simulate the operation of Pier IV, which was in the implantation phase. With the data obtained, the authors defined the queuing system as an $M/M/1/\infty/\infty/FIFO$ model and applied the values of the variables previously mentioned in the formulas of the variables used as performance measures of queuing theory to observe the performance of the existing system for loading iron ore and manganese.

Doy et al. [22] proposed to study, through the simulation of a queuing model, the e-mail service of the computer network of the Institute of Mathematics and Statistics of the University of São Paulo, to identify statistical characteristics and evaluate its performance. According to the authors, the results showed that the system has space to absorb growth on several fronts, showing itself to be practically insensitive to doubling the number of messages arrivals. On the other hand, the stops in the system have an important effect on the measures studied. The effect of the unfolding was also evaluated. They originate from lists of interest and have a huge potential multiplier in message traffic over the network. In the system studied, the increase applied to the size of the lists can be largely absorbed by the existing capacity in the system.

4. Case Study: Integration Problem

The integration problem analyzed as a case study consists of a real problem of a Telephone Exchange management. The proposed solution aimed to improve the functionality of the telephone exchange application and automate the billing of personal calls made by employees using the telephones of the Regional University of the State of Rio Grande do Sul (Unijuí) Telephone Exchange. The integration solution enables the calls made by the employee identified by a personal access code to be automatically deducted from the payroll.

The conceptual model designed with the Guaraná technology to solve the integration problem of the Unijuí University Telephone Exchange, proposed by Frantz [23], is shown in Figure 2.

The solution integrates five applications: Call Center System (Telephone Exchange), Human Resources, Payroll, E-mail Server, and SMS Notifier. Each application runs on a different platform; Human Resources and Payroll are legacy systems developed by the University and the other applications were acquired. In addition, the E-mail Server provides POP3 and SMTP interfaces, and the other applications are designed without regard to the possibility of integration. The Telephone Exchanges registers all the calls that each employee makes from the university telephones, and this is possible because each employee has a personal access code, which must be entered before dialing the number you wish to call. This code is used to correlate the links with the Human Resources and Payroll application information. The Human Resources application provides personal information about employees and each month the Payroll calculates the salary of all employees, including deductions. The E-mail Server and SMS Notifier, respectively, run the University e-mail and SMS services, which are used for notification.

The integration solution designed in Guaraná is composed of an orchestration process that exogenously coordinates the applications involved in the solution, as represented in Figure 2. The workflow begins at the $P0$ gateway, which periodically searches and checks the PBX to find new calls. Each call results in a message that is then entered into the orchestration process and contains the call information. Within the process, the messages are forwarded to task $T0$, via slot $S0$. Whenever a task is executing incoming messages, they wait in the slots in a queue and are subsequently selected for FIFO processing.

The task $T0$ (Filter) filters and discards the toll-free messages and only forwards those that have costs to the S1 slot to be executed by the task $T1$ (Replicator). This task creates a copy of the message, which is forwarded to slot $S2$ to be processed by the task $T2$ (Translator), which translates the message content into the Human Resources application format, and is forwarded to slot $S3$ to query the application via port $P1$.

Messages return through port $P1$ and slot $S5$ forwards it to task $T3$ (Correlator), which analyzes incoming messages and outputs a set of correlated data. Slots $S6$ and $S7$ conduct the correlated messages for the $T4$ task (Context-based Content Enricher), which adds

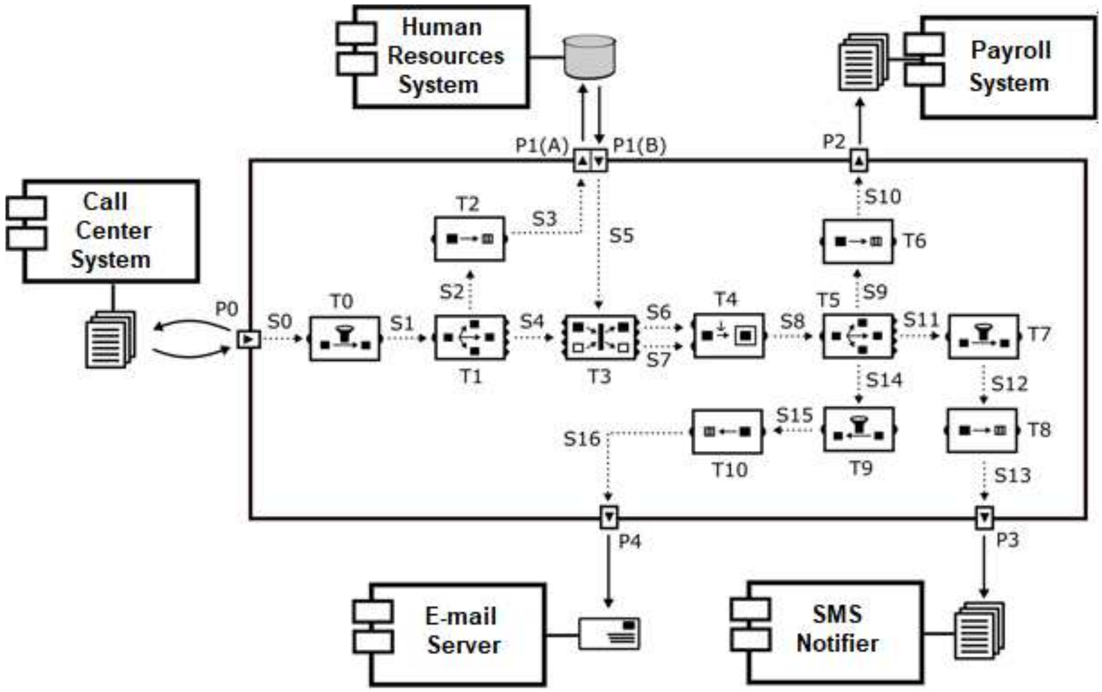


Figure 2. Conceptual model of the integration solution using Guaraná technology

information that has been returned from the Human Resources application in the other correlated copy. The $T5$ task then creates three copies of the enriched message and sends it to the Payroll, E-mail Server, and SMS Notifier applications. The outgoing port $P2$ informs the payroll debit orders, the copies sent to the SMS Notifier, and E-mail Server will be filtered if the outgoing ports $P3$ and $P4$ receive messages without enough information from the recipients, for example phone number and email address.

5. Formal Model Proposed

In Queueing Theory, models, according to their characteristics, represent systems. However, the systems have elements common to all that are part of the basic process. The elements that make up the process are customers who need service, arrive in time, enter the queuing system and, due to unavailability of immediate service, form a queue. Clients are selected at certain times for service by a rule known as queue discipline. After the servers service the client, it leaves the queue system.

A generic process of a service system that can be represented by a queue model is illustrated in Figure 4 (a). This process has a similar structure to the conceptual model of an integration solution designed in Guaraná. In this structure, clients arrive, form a queue, and wait a certain amount of time to be serviced. A portion extracted from an integration solution designed in Guaraná is illustrated in Figure 3 (b), in order to demonstrate the equivalence between the elements of the Queueing Theory and those of an integration solution. The messages represent the clients, the slots, the queues, and the services tasks.

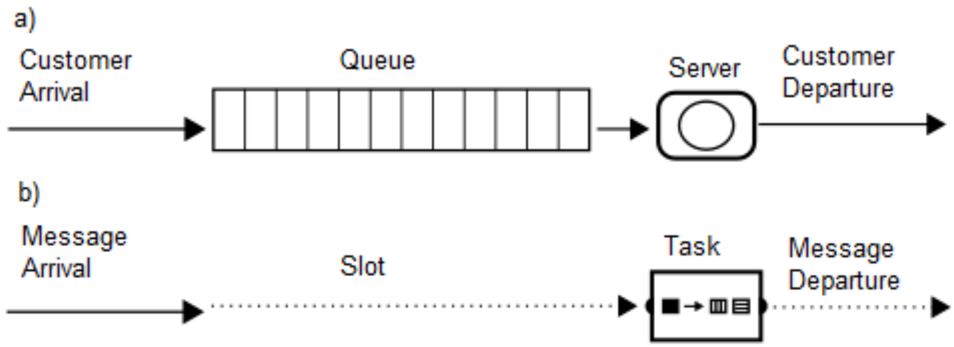


Figure 3. Model of a service system (a) Guaraná DSL (b)

Another common feature between an integration solution and a discrete event system is the relationship between its elements and the structure of operation. The relationship between the conceptual elements of a Discrete-Event Simulation (DES) is shown in Figure 4.

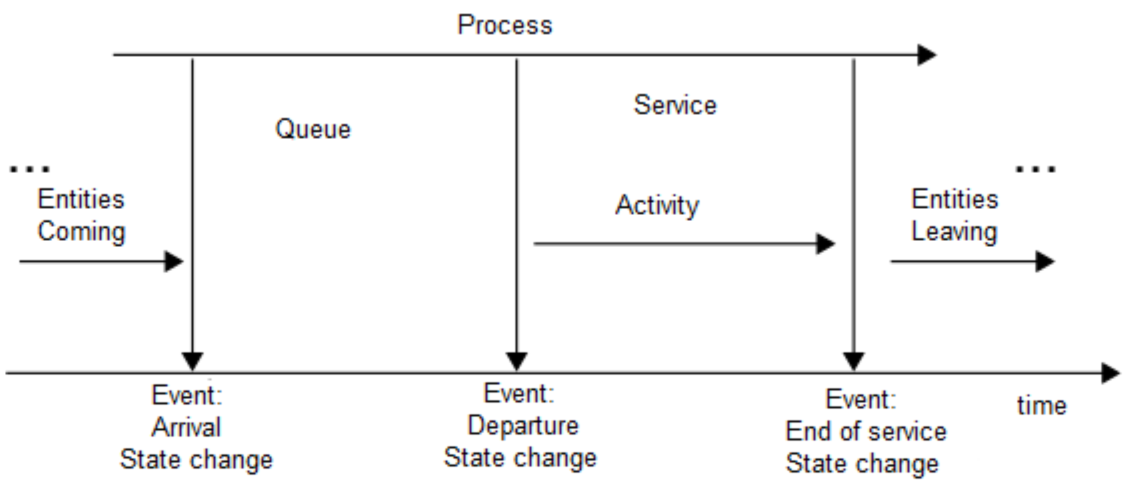


Figure 4. Relationship between elements that compose a Discrete-Event Simulation

5.1. Integration Solution Variables

Distribution of probability was used to correctly characterize the arrival process, which requires the specification of a parameter called the arrival rate to represent the average number of messages arriving in the solution per unit time. In this context, the average message arrival rate is an essential random variable. To quantify this variable, in the Queue Theory, the symbol λ and IC are used for the average interval between message arrivals. Based on the Queue Theory, the process of executing a message in the task may be quantified by a random variable and happens in a way analogous to the arrival process. In this way, a probability distribution is also used. The symbol μ indicates the average execution rate and the average execution time of a task. It is also possible to calculate the average number of messages processed in the task (NA), the utilization rate ($\boldsymbol{\rho}$) and the idle task rate ($\mathbf{1}-\boldsymbol{\rho}$). In Queue Theory, the number of servers is an important variable. In the integration solution, the number of threads (s) used to execute a task is analogous to this variable.

To identify possible performance bottlenecks, the slot variables are related to the number of messages and waiting time. The variables are, respectively, the average number of messages in the slot (NF) and the average time in the slot (TF). The integration solution seen as a system also has important variables that allow observing and studying the behavior of the solution. These variables are the average number of messages (NS) and the average time of stay of the messages in the solution (TS). The location of the variables identified in an integration solution is shown in Figure 5.

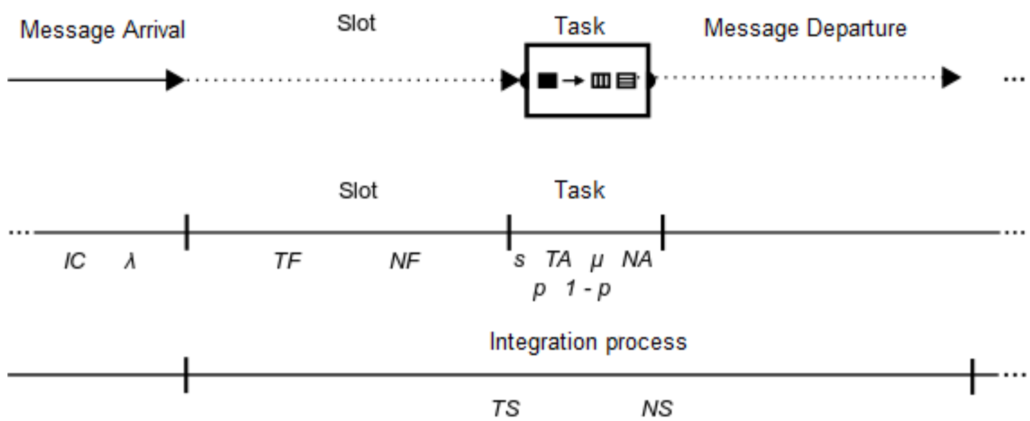


Figure 5. Location of variables in an integration solution

5.2. Mathematical Model

In the M/M/1 model, the times between successive arrivals and service times follow the exponential distribution. The average arrival rate (λ) to the queuing system and average attendance rate (μ) are constant, regardless of the state of the system.

A system enters state 0 only from state 1. Therefore, only by means of an output (termination of service) of the state $n = 1$. Thus, P_1 is the probability of steady state in state 1 and represents the proportion of time that the system remains in state 1. Given that when the system is in state 1, μ is the average service rate and μP_1 is the average rate at which the system enters state 0. On the other hand, considering the state $n = 0$ the system leaves state 0 only by means of an arrival. Given that when the system is in state 0, λ is the average rate of arrival of clients, P_0 is the proportion of time the system remains in state 0, and λP_0 is the average rate at which the system exits state 0.

The equilibrium equation for state 0 is:

$$\mu P_1 = \lambda P_0 \tag{1}$$

In order to solve the equilibrium equations, it is necessary to solve them in relation to one of the variables, being the most convenient P_0. Thus, the Equilibrium Equation (1) for state 0 is used to find P_1 in terms of P_0.

$$P_1 = \frac{\lambda}{\mu} P_0 \tag{2}$$

For all other ($n = 1,\ 2,\ ...$), there are always two transitions, one going out and another going into the state. Therefore, each side of the equilibrium equations (rate that enters = rate that exits) for these states is given by the sum of the average rates of the respective transitions involved. In state 1, λP_0 and μP_2 are occurring, respectively, in the transition from state 0 (arrival of a customer) and state 2 (end of a service). They are coming out of state 1, λP_1 e μP_1 that represent, respectively, the transition to state 2 (arrival of a client) and to state 0 (termination of a service). Thus, for state 1, the equilibrium equation is:

$$\lambda P_0 + \mu P_2 = \lambda P_1 + \mu P_1 \tag{3}$$

Isolating P_2 to the left, we have:

$$\begin{aligned} P_2 &= \frac{\lambda P_1 + \mu P_1 - \lambda P_0}{\mu} \\ &= \frac{\lambda}{\mu} P_1 + \frac{1}{\mu}(\mu P_1 - \lambda P_0) \end{aligned} \tag{4}$$

From Equation (1) it is known that $\mu P_1 = \lambda P_0$, replacing in Equation (4) are obtained:

$$\begin{aligned} P_2 &= \frac{\lambda}{\mu} P_1 + \frac{1}{\mu}(\lambda P_0 - \lambda P_0) \\ &= \frac{\lambda}{\mu} P_1 \end{aligned} \tag{5}$$

To write P_2 in terms of P_0, we replaced the Equation (2) into Equation (5).

$$\begin{aligned} P_2 &= \frac{\lambda}{\mu} \left(\frac{\lambda}{\mu} \right) P_0 \\ &= \left(\frac{\lambda}{\mu} \right)^2 P_0 \end{aligned} \tag{6}$$

The Equation of Equilibrium of state 2 is given by:

$$\lambda P_1 + \mu P_3 = \lambda P_2 + \mu P_2 \tag{7}$$

Isolating P_3, we have:

$$\begin{aligned} P_3 &= \frac{\lambda P_2 + \mu P_2 - \lambda P_1}{\mu} \\ &= \frac{\lambda}{\mu} P_2 + \frac{1}{\mu}(\mu P_2 - \lambda P_1) \end{aligned} \tag{8}$$

Replacing the Equation (5) into Equation (8) are obtained:

$$\begin{aligned} P_3 &= \frac{\lambda}{\mu}\frac{\lambda}{\mu}P_1 + \frac{1}{\mu}\left(\mu\frac{\lambda}{\mu}P_1 - \lambda P_1\right) \\ &= \left(\frac{\lambda}{\mu}\right)^2 P_1 + \frac{1}{\mu}\left(\lambda P_1 - \lambda P_1\right) \\ &= \left(\frac{\lambda}{\mu}\right)^2 P_1 \end{aligned} \tag{9}$$

To write P_3 in terms of P_0, we replaced the Equation (2) into Equation (9).

$$\begin{aligned} P_3 &= \left(\frac{\lambda}{\mu}\right)^2 \frac{\lambda}{\mu}P_0 \\ &= \left(\frac{\lambda}{\mu}\right)^3 P_0 \end{aligned} \tag{10}$$

Observing the probability that the system is in the states ($n = 0, 1, 2$), by induction, we conclude that:

$$P_n = \left(\frac{\lambda}{\mu}\right)^n P_0 \tag{11}$$

The occupancy rate of a server is given by:

$$\rho = \frac{\lambda}{\mu} \tag{12}$$

Replacing the Equation (12) into Equation (11) are obtained:

$$P_n = \rho^n P_0 \tag{13}$$

The sum of all probability must be equal to 1,

$$\sum_{n=0}^{\infty} P_n = 1 \tag{14}$$

implies that,

$$\begin{aligned} \sum_{n=0}^{\infty} \rho^n P_0 &= 1 \\ \frac{1}{P_0} &= \sum_{n=0}^{\infty} \rho^n \end{aligned} \tag{15}$$

The second term of Equation (15) is the sum of a geometric series obtained by adding the infinite terms of a geometric progression and converging if and only if, $\rho < 1$. Thus, for the existence of a steady-state (equilibrium) system, the mean attendance rate must be higher than the mean arrival rate ($\mu > \lambda$). Hillier and Lieberman [13] justify, when $\mu \leq \lambda$, the solution is overloaded, since the sum to calculate P_0 diverge. diverges. By starting the queue system operation with no clients, the clerk would be successful in supporting clients who arrive in a short period of time, but this is impossible in the long run and the queue will grow without limits. Even when $\mu = \lambda$, the number of clients in the queue system grows slowly with no limits over time. The sum of the geometric series is:

$$\sum_{n=0}^{\infty} \rho^n = \frac{1}{1-\rho} \quad \text{the series converges if } |\rho| < 1 \tag{16}$$

Thus, replacing the Equation (16) into Equation (15) can be obtained P_0.

$$\begin{aligned} \frac{1}{P_0} &= \frac{1}{1-\rho} \\ P_0 &= 1-\rho \end{aligned} \tag{17}$$

Substituting the Equation (17) into Equation (13) we obtain the equation that allows us to directly calculate the probability of n clients being in the queue system.

$$P_n = \rho^n(1-\rho) \tag{18}$$

Performance measures for the queue systems can be obtained from P_n.

$$NS = \sum_{n=0}^{\infty} nP_n \tag{19}$$

To obtain the average number of customers in the system, initially the Equation (18) is replaced in Equation (19).

$$\begin{aligned} NS &= \sum_{n=0}^{\infty} n\rho^n(1-\rho) \\ &= (1-\rho)\sum_{n=0}^{\infty} n\rho^n \end{aligned} \tag{20}$$

The sum of Equation (20) can be rewritten as:

$$\begin{aligned} \sum_{n=0}^{\infty} n\rho^n &= \rho + 2\rho^2 + 3\rho^3 + 4\rho^4 + ... \\ &= \rho(1 + 2\rho + 3\rho^2 + 4\rho^3 + ...) \\ &= \rho\sum_{n=1}^{\infty} n\rho^{n-1} \end{aligned} \tag{21}$$

Replacing the sum of Equation (21) into Equation (20), we have:

$$NS = (1-\rho)\rho \sum_{n=1}^{\infty} n\rho^{n-1} \tag{22}$$

The derivative of ρ^n in relation to ρ is given by:

$$\frac{d}{d\rho}\rho^n = n\rho^{n-1} \tag{23}$$

Thus, satisfied the conditions to invert the sum and derivative, we obtained:

$$\begin{aligned} NS &= (1-\rho)\rho \sum_{n=1}^{\infty} n\rho^{n-1} \\ &= (1-\rho)\rho \sum_{n=0}^{\infty} \frac{d}{d\rho}\rho^n \\ &= (1-\rho)\rho \frac{d}{d\rho} \sum_{n=0}^{\infty} \rho^n \\ &= (1-\rho)\rho \frac{d}{d\rho}\left(\frac{1}{1-\rho}\right) \\ &= (1-\rho)\rho \frac{1}{(1-\rho)^2} \end{aligned} \tag{24}$$

Simplifying the Equation (24) and substituting rho, we have:

$$\begin{aligned} NS &= \frac{\rho}{(1-\rho)} \\ &= \frac{\frac{\lambda}{\mu}}{1-\frac{\lambda}{\mu}} \end{aligned}$$

Thus, the average number of customers in the system is given by the formula:

$$NS = \frac{\lambda}{\mu - \lambda} \tag{25}$$

Considering the formula of Little (Equation 30) and isolating TS,

$$\begin{aligned} NS &= \lambda TS \\ TS &= \frac{NS}{\lambda} \end{aligned} \tag{26}$$

the formula of the mean time of permanence in the system can be obtained using Equation (25).

$$\begin{aligned} TS &= \frac{\frac{\lambda}{\mu-\lambda}}{\lambda} \\ &= \frac{\lambda}{\mu-\lambda}\left(\frac{1}{\lambda}\right) \\ &= \frac{1}{\mu-\lambda} \end{aligned} \tag{27}$$

Using the Equation of relation (25) and isolating NF on the left,

$$\begin{aligned} NS &= NF + \frac{\lambda}{\mu} \\ NF &= NS - \frac{\lambda}{\mu} \end{aligned} \tag{28}$$

and replacing NS for Equation (25), we get the formula for the average number of clients in the queue.

$$\begin{aligned} NF &= \frac{\lambda}{\mu-\lambda} - \frac{\lambda}{\mu} \\ &= \frac{\lambda\mu - \lambda(\mu-\lambda)}{\mu(\mu-\lambda)} \\ &= \frac{\lambda\mu - \lambda\mu + \lambda^2}{\mu(\mu-\lambda)} \\ &= \frac{\lambda^2}{\mu(\mu-\lambda)} \end{aligned} \tag{29}$$

In order to obtain the mean time of queue time, the Little Equation relation (30) is used, isolating TF,

$$\begin{aligned} NF &= \lambda TF \\ TF &= \frac{NF}{\lambda} \end{aligned} \tag{30}$$

and substituting NF for Equation (29), we have the formula:

$$\begin{aligned} TF &= \frac{\frac{\lambda^2}{\mu(\mu-\lambda)}}{\lambda} \\ &= \frac{\lambda^2}{\mu(\mu-\lambda)}\left(\frac{1}{\lambda}\right) \\ &= \frac{\lambda}{\mu(\mu-\lambda)} \end{aligned} \tag{31}$$

The summary of the M/M/1 model formulas that treat the main random variables to obtain the performance measures of the queuing system is shown in Table 2.

Name	Description	Equation
NS	Average number of customers in the system	$NS = \frac{\lambda}{\mu - \lambda}$
TS	Average time of stay in the system	$TS = \frac{1}{\mu - \lambda}$
NF	Average number of customers in the queue	$NF = \frac{\lambda^2}{\mu(\mu - \lambda)}$
TF	Average time of stay in the queue	$TF = \frac{\lambda}{\mu(\mu - \lambda)}$

Table 2. Equations of M/M/1 model

5.3. Simulation Model

The characterization of an integration solution as a discrete event system, through the Queueing Theory, was demonstrated by equivalence of the slot with a queue. A slot has infinite capacity and, in this study, is considered to organize the messages according to FIFO discipline. Thus, in the simulation model, the FIFO Queue block was used in order that the queue discipline is FIFO and the entity storage capacity can be set as infinite.

In the characterization, the equivalence of a task with a server was also demonstrated. Thus, any task may be considered equivalent to the Single Server block. Tasks execute a message over a period. Runtime is not constant, and to make it random in the simulation model, the Single Server block will receive a signal from the Event Based Random Number block.

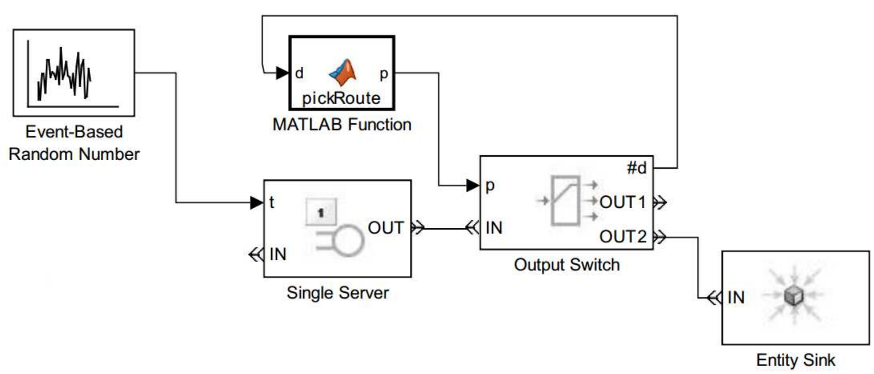


Figure 6. Tasks Filter $T0$, $T7$ and $T9$ using SimEvents

However, the Single Server block presents the role of a server in general, and each task has a specific functionality. For example, the Filter task, when performing message processing, performs the filtering of unwanted messages, and the Replicator task copies the message. In this way, it was not possible to represent most tasks using only the Single

Server and Event Based Random Number blocks. To obtain equivalence, in addition to these two blocks, it was necessary to use the Output Switch, MATLAB Function, Entity Sink, Replicate, and Entity Combiner blocks.

For the Filter task, it was necessary to use several blocks (Figure 6). The Output Switch block allows determining, by the parameter Switching criterion, by which output port the entity will leave the block. The switching criterion used was a signal from the MATLAB Function block. In this block, a function was created to determine, using percentage, the entities that will follow in the simulation flow and those that will be sent to the Entity Sink block, representing the filtered messages.

The blocks used to obtain the functions of the Replicator task are shown in 7. The Replicate block creates a copy of the entity. The number of copies of the entity is determined by setting the number of output ports.

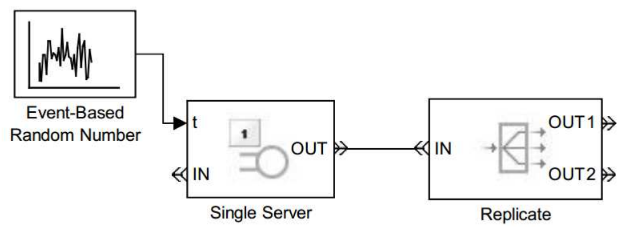


Figure 7. Tasks Replicator $T4$ and $T5$ using SimEvents

For the Translator task, no block was identified in the SimEvents library to perform its function. The absence of this functionality does not interfere in the simulation and results, because the Single Server block represents the processing time of this task (Figure 8).

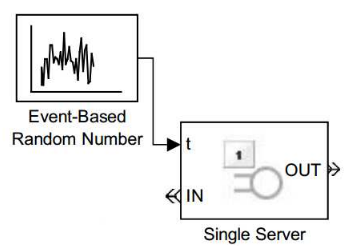


Figure 8. Tasks Translator $T2$, $T6$, $T8$ and $T10$ using SimEvents

The Entity Combiner block has equivalent functionality to the Correlator and Context-based Content Enricher tasks. The block detects when the entities are ready for the combining operation and generates a new entity for each set of incoming entities. This block accepts entities only when the input port of the next block is available. Thus, to represent the execution time of both tasks, the Single Server block will be used to delay the combination operations (Figure 9).

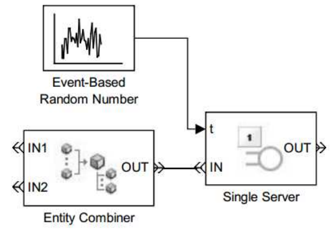


Figure 9. Tasks Correlator $T3$ e $T4$ using SimEvents

6. Experimental Results

The process of arrival and attendance based on Queueing Theory is characterized by probability distributions. In this study, the integration solution is considered equivalent to a queuing system represented by the $M/M/1$ model. In this queuing model, the times between successive arrivals and service times follow the exponential distribution, there is only one attendant and the queue discipline is FIFO. Therefore, in the simulation model, the time between arrivals and the service time were configured with exponential distribution and the Single Server and $FIFO$ Queue blocks were used. The use of a Single Server block in each task is equivalent to a thread being allocated to perform a task.

The times between arrivals were estimated to obtain different rates of arrivals of messages, to allow the analysis of the performance of the solution under different workloads. In this context, based on the proposed formal simulation model, the behavior of the integration solution was analyzed experimentally in seven different scenarios. In all scenarios, the simulation time was set at 24 hours and the response time in the presented values, varying only the time between message arrivals at 16, 8, 4, 2, 1, 0.5, and 0.25 seconds, respectively. According to the definition of Queue Theory, the average arrival rate of messages per second for the time between arrivals used in the simulation in each scenario are: (Scenario 1: 0.0625), (Scenario 2: 0.125), (Scenario 3: 0.25), and (Scenario 4: 0.5).

6.1. Discussion and Data Analysis

Our approach considered the outliers in the analysis of the results. In this context, we calculated the mean ($\overline{x}$) standard deviation (s), and coefficient of variation (CV), with and without outliers, in the seven scenarios of the results of the variables average number of messages in the slots and rate of utilization to infer if the central position measure and dispersion measures are greatly affected by the extreme values. For the idle rate, these inferences were not performed because it is expressed mathematically as $1 - \rho$. Thus, the results of this variable are not obtained directly from the simulation, but from the utilization rate.

In scenario 1, values below the lower limit in slot $S4$ and above the upper limit in $S0$, $S5$, $S12$, and $S13$ were found. By comparing the results of Table 3, it is possible to observe that these values do not significantly affect the average, as the difference is not big between the average with and without outliers. In the standard deviation, the difference is also small, which shows that there is little dispersion in the values of the set around the mean. The

Slots	With outliers			Without outliers		
	$\overline{x}$	s	CV (%)	$\overline{x}$	s	CV (%)
S0*	0,00100	0,00011	10,88408	0,00097	0,00008	7,79131
S1	0,00089	0,00011	12,43839	0,00089	0,00011	12,43839
S2	0,00089	0,00011	11,90574	0,00089	0,00011	11,90574
S3	0,00094	0,00015	15,73158	0,00094	0,00015	15,73158
S4*	0,26947	0,00418	1,55169	0,26998	0,00337	1,24969
S5*	0,00374	0,00036	9,74989	0,00370	0,00032	8,58258
S8	0,00093	0,00010	11,10110	0,00093	0,00010	11,10110
S9	0,00090	0,00011	12,30599	0,00090	0,00011	12,30599
S10	0,00089	0,00010	11,36067	0,00089	0,00010	11,36067
S11	0,00088	0,00010	11,45787	0,00088	0,00010	11,45787
S12*	0,00079	0,00010	12,84668	0,00078	0,00009	11,62736
S13*	0,00085	0,00013	15,26382	0,00084	0,00011	13,09171
S14	0,00089	0,00013	14,05391	0,00089	0,00013	14,05391
S15	0,00086	0,00010	11,68624	0,00086	0,00010	11,68624
S16	0,00092	0,00011	12,37015	0,00092	0,00011	12,37015

Table 3. NF variable statistics in scenario 1

coefficient of variation below 30% proves that the sample is homogeneous (there is little variation). This means that the average is a good measure of core trend. The sample of slot $S4$ is more homogeneous in the group because it has the lowest coefficient of variation.

Slots	With outliers			Without outliers		
	$\overline{x}$	s	CV (%)	$\overline{x}$	s	CV (%)
S0*	0,00425	0,00031	7,31142	0,00421	0,00028	6,55419
S1	0,00375	0,00019	5,03787	0,00375	0,00019	5,03787
S2*	0,00376	0,00024	6,50162	0,00380	0,00020	5,30961
S3*	0,00376	0,00032	8,39788	0,00371	0,00026	7,01539
S4*	0,62515	0,00943	1,50813	0,62402	0,00773	1,23849
S5	0,01588	0,00072	4,55512	0,01588	0,00072	4,55512
S8	0,00375	0,00019	4,97307	0,00375	0,00019	4,97307
S9	0,00381	0,00030	7,75060	0,00381	0,00030	7,75060
S10	0,00375	0,00028	7,42410	0,00375	0,00028	7,42410
S11	0,00372	0,00023	6,20130	0,00372	0,00023	6,20130
S12	0,00333	0,00017	5,05838	0,00333	0,00017	5,05838
S13	0,00341	0,00017	4,99683	0,00341	0,00017	4,99683
S14	0,00370	0,00019	5,04457	0,00370	0,00019	5,04457
S15	0,00373	0,00024	6,45425	0,00373	0,00024	6,45425
S16	0,00369	0,00025	6,80510	0,00369	0,00025	6,80510

Table 4. NF variable statistics in scenario 2

Scenario 2 presented values below the lower limit in slot $S2$ and above the upper limit in $S0$, $S3$, and $S4$. By analyzing the results expressed in Table 4, one notices that there is not much difference between the statistics with and without outliers. However, the elimination of extreme values results in a sample with more homogeneous data. The slot $S4$ in this scenario also has the most homogeneous sample.

The data obtained from the simulation of scenario 3 presented values below the lower limit in slot $S5$ and above the upper limit in $S1$, $S13$, and $S15$. In this scenario, the mean is also not sensitive to extreme values, as it is perceived that there is no significant variation with the mean without outliers. The coefficient of variation of the data is low, showing that the results are homogeneous. Slot $S4$ presents the lowest coefficient of variation of the data series (Table 5).

Slots	With outliers			Without outliers		
	$\overline{x}$	s	CV (%)	$\overline{x}$	s	CV (%)
S0	0,01781	0,00075	4,21905	0,01781	0,00075	4,21905
S1*	0,01608	0,00071	4,41246	0,01601	0,00063	3,90970
S2	0,01583	0,00069	4,37151	0,01583	0,00069	4,37151
S3	0,01601	0,00054	3,36893	0,01601	0,00054	3,36893
S4	1,93626	0,04948	2,55567	1,93626	0,04948	2,55567
S5*	0,07439	0,00228	3,06727	0,07481	0,00182	2,43669
S8	0,01602	0,00059	3,68765	0,01602	0,00059	3,68765
S9	0,01612	0,00051	3,14089	0,01612	0,00051	3,14089
S10	0,01585	0,00051	3,18916	0,01585	0,00051	3,18916
S11	0,01580	0,00054	3,43452	0,01580	0,00054	3,43452
S12	0,01437	0,00050	3,45448	0,01437	0,00050	3,45448
S13*	0,01433	0,00060	4,18448	0,01426	0,00050	3,52198
S14	0,01608	0,00036	2,22699	0,01608	0,00036	2,22699
S15*	0,01585	0,00056	3,50409	0,01574	0,00043	2,75294
S16	0,01564	0,00063	4,03079	0,01564	0,00063	4,03079

Table 5. NF variable statistics in scenario 3

Scenario 4 presented values below the lower limit in slots $S5$ and $S13$ and above the upper limit in slots $S5$, $S9$, $S10$, $S13$, and $S14$. The data are homogeneous, since they have a low coefficient of variation (Table 6. Extreme values do not affect the mean because there is no significant difference compared to the mean without outliers.

Considering scenarios 1, 2, and 3, it is not possible to say that $S4$ is a bottleneck, although it is possible to see this tendency, since in all three scenarios it presents the highest value, followed by $S5$. In scenario 4, the facility that represents the human resource application in the simulation model is no longer in steady state and it is perceived that this directly influences $S4$, as the value increases from approximately 2 (scenario 3) to approximately $3,277$. This significant increase justifies why the messages in slot $S4$ await the correlation that was sent by the solution to fetch information in the Human Resources application.

7. Conclusion

This chapter proposes the development of a formal simulation model based on the mathematical formalism of Queueing Theory as the objective of analyzing the behavior and

Slots	With outliers			Without outliers		
	$\overline{x}$	s	CV (%)	$\overline{x}$	s	CV (%)
S0	0,08320	0,00196	2,35129	0,08320	0,00196	2,35129
S1	0,07433	0,00150	2,01860	0,07433	0,00150	2,01860
S2	0,07397	0,00187	2,52907	0,07397	0,00187	2,52907
S3	0,07413	0,00157	2,11905	0,07413	0,00157	2,11905
S4	3276,87273	158,04968	4,82319	3276,87273	158,04968	4,82319
S5*	0,26864	0,00918	3,41868	0,26729	0,00484	1,81052
S8	0,05038	0,00140	2,77255	0,05038	0,00140	2,77255
S9*	0,05025	0,00131	2,59838	0,05011	0,00114	2,27083
S10*	0,04979	0,00153	3,07869	0,04945	0,00101	2,04342
S11	0,04981	0,00112	2,25842	0,04981	0,00112	2,25842
S12	0,04488	0,00132	2,95050	0,04488	0,00132	2,95050
S13*	0,04477	0,00144	3,21065	0,04482	0,00081	1,79903
S14*	0,04995	0,00159	3,18708	0,04979	0,00140	2,80628
S15	0,04887	0,00141	2,88965	0,04887	0,00141	2,88965
S16	0,04871	0,00107	2,18697	0,04871	0,00107	2,18697

Table 6. NF variable statistics in scenario 4

identifying possible performance bottlenecks of integration solutions, projected in Guaraná technology, based on its conceptual models. For the simulation of the conceptual model, which is the result of a real problem, the SimEvents tool was used, as it provides mechanisms for simulating discrete events and is a component of the Simulink library in Matlab. The simulation approach presented for behavior analysis and identification of possible performance bottlenecks for integration solutions can help in the quality of integration solutions developed using Guaraná technology. It was noticed that the EAI area using the simulation method is still the subject of little research. Therefore, this study aims to increase research in the scope of simulation of enterprise application integration solutions. The statistical outputs of the blocks of the simulation model allowed us to monitor and analyze the performance measures of the solution; average number of messages in the slots, average time of messages in the slots, utilization rate, and idle rate of the tasks. The analysis of these measures allowed the identification of the behavior of the solution in different work rates and the occurrence of possible performance bottlenecks in this integration solution.

References

[1] Gregor Hohpe and Bobby Woolf. *Enterprise integration patterns: Designing, building, and deploying messaging solutions*. Addison-Wesley Professional, 2004.

[2] Daniela L. Freire, Rafael Z. Frantz, Fabricia Roos-Frantz, and Sandro Sawicki. A methodology to rank enterprise application integration platforms from a performance perspective: an analytic hierarchy process-based approach. *Enterprise Information Systems*, (1):1–31, 2019.

[3] Daniela L. Freire, Rafael Z. Frantz, Fabricia Roos-Frantz, and Sandro Sawicki. Survey

on the run-time systems of enterprise application integration platforms focusing on performance. *Software: Practice and Experience*, 49(3):341–360, 2019.

[4] David G Messerschmitt and Clemens Szyperski. Software ecosystem: Understanding an indispensable technology and industry, 2003.

[5] Claus Ibsen and Jonathan Anstey. *Camel in action*. Manning Publications Co., 2010.

[6] Mark Fisher, Jonas Partner, Marius Bogoevici, and Iwein Fuld. *Spring Integration in action*. Manning Publications Co., 2012.

[7] David Dossot, John D'Emic, and Victor Romero. *Mule in action*. Manning, 2014.

[8] Rafael Z Frantz, Antonia M Reina Quintero, and Rafael Corchuelo. A domain-specific language to design enterprise application integration solutions. *International Journal of Cooperative Information Systems*, 20(02):143–176, 2011.

[9] Daniela L. Freire, Rafael Z. Frantz, and Fabricia Roos-Frantz. Ranking enterprise application integration platforms from a performance perspective: An experience report. *Software: Practice and Experience*, 49(5):921–941, 2019.

[10] Daniela L. Freire, Rafael Z. Frantz, and Fabricia Roos-Frantz. Towards optimal thread pool configuration for run-time systems of integration platforms. *International Journal of Computer Applications in Technology*, (in-press):1–18, 2019.

[11] Daniela L. Freire, Rafael Z. Frantz, Fabricia Roos-Frantz, and Sandro Sawicki. Optimization of the size of thread pool in runtime systems to enterprise application integration: a mathematical modelling approach. *Trends in Applied and Computational Mathematics*, (20):169–188, 2019.

[12] Averill M Law and W David Kelton. *Simulation modeling and analysis*. McGraw Hill Boston, 2000.

[13] Frederick S Hillier and Gerald J Lieberman. *Introduction to Operational Research*. McGraw Hill, 2010.

[14] Paulo José de Freitas Filho. *Introduction to modeling and simulation of systems: with applications in Arena tool*. Visual Books, 2001.

[15] Sandro Sawicki, Rafael Z Frantz, Vitor Manuel Basto Fernandes, Fabricia Roos-Frantz, Iryna Yevseyeva, and Rafael Corchuelo. Characterising enterprise application integration solutions as discreteevent system. In *Handbook of Research on Computational Simulation and Modeling in Engineering*, pages 255–282. IGI Global, 2015.

[16] Maria Cristina Fogliatti and Neli Maria Costa Mattos. Queueing theory. *Rio de Janeiro: Interscience*, pages 1–20, 2007.

[17] David G Kendall. Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded markov chain. *The Annals of Mathematical Statistics*, pages 338–354, 1953.

[18] Eder Oliveira Abensur, Israel Brunstein, Adalberto Fishmann, and Linda Lee Ho. Trends for brazilian banking self-service: a strategic approach based on queuing theory. *Mackenzie Administration Journal*, 4(2), 2008.

[19] Liu Changfu and Liu Zhenyu. Research of transaction request handling queueing system in the e-business environment based on queuing theory. In *Information Processing, 2009. APCIP 2009. Asia-Pacific Conference on*, volume 2, pages 589–592. IEEE, 2009.

[20] Seyed Hossein Kamali, Maysam Hedayati, Abdol Said Izadi, and Hamid Reza Hoseiny. The monitoring of the network traffic based on queuing theory and simulation in heterogeneous network environment. In *Computer Technology and Development, 2009. ICCTD'09. International Conference on*, volume 1, pages 322–326. IEEE, 2009.

[21] Gustavo Rossa Camelo, Antônio Sérgio Coelho, Renata Massoli Borges, and Rosimeri Maria de Souza. Queueing theory and simulation applied to the shipment of iron ore and manganese in the maritime terminal of the ponta da madeira. *Books of the IME-Statistical Series*, 29(2):1, 2010.

[22] Fábio E Doy, Graça Bressan, Gustavo H de A Pereira, and Marcos N Magalhães. Simulation of the e-mail service through a queueing model. *Operational Research*, 26(2):241–253, 2006.

[23] Rafael Z Frantz. *Enterprise application integration: an easy-to-maintain model-driven engineering approach*. PhD thesis, Universidad de Sevilla, 2012.