Proceedings of the 2019 Winter Simulation Conference
N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, eds.

MODELING AND SIMULATION OF THE BOTTLENECKS OF AN ONLINE RESERVATION SYSTEM

José C. Conti
Edson L. Ursini
Paulo S. Martins

School of Technology
University of Campinas (Unicamp)
R. Paschoal Marmo, 1888
Limeira, SP 13484-332, BRAZIL

ABSTRACT

This work intends to present an analytical and a simulation model for an online reservation system and examine its performance associated with its bottlenecks. The modeling and evaluation of reservation systems should be carried out through analytical models, whenever these models are available. However, it is known that in general, such models have approximations that do not consider several details of the real world. Because the multiprocessing system in consideration uses database queries (and also uses preemption resume priority) and has a certain degree of complexity, our approach uses an approximate analytical model that is validated by a discrete-event simulation model (and vice-versa). Both models are supported by data obtained from measurements of a real-world system. Once validated, the model can be analyzed for other input data and distributions through simulation.

1 INTRODUCTION

In the last few years, there has been a growing increase in querying data and request services from the most diverse types of production systems, and these demands also include cloud-based reservation systems. These systems cater to a large number of requests simultaneously and may reach millions of users scattered worldwide. Failure of such systems may pose a relatively high risk with severe financial, economic or social consequences. One way of mitigating such concerns is to devise methods for robust and accurate simulation modeling, analysis, and optimization. Such methods must be extensively validated either with analytical models or real data or both. They must identify the bottlenecks of such systems and evaluate their performance in the face of possible and unexpected overloading. However, specifically with online reservation systems, there is a lack of validated simulation models that evaluate the processing capacity of the server in relation to the volume of queries and requests that are performed.

The use of simulation models constitutes an opportunity to provide an alternative or eventually replace experiments that occur in these production systems, or that are still in the implementation phase, not economically effective, or even non-realizable. A simulation model may identify complex and important characteristics of real-world systems, thus allowing the representation of the behavior that such systems would show when they become productive. The main challenge is the synchronization of conditions and the mismatch between specific concepts in use for the analytic approach and the discrete models.

Within this context, in this work, a discrete event simulation model of a cloud-based reservation system is proposed. It is an online reservation datacenter system consisting of n-servers and n-databases, for which it is desired to determine the mean response time for a request and the mean percentage server utilization. Another goal of this work is to validate the result of the simulation model with an equivalent and simplified

analytical model, initially using its main bottlenecks. Both models adhere to the available measured data, i.e. they are as close as possible to the observed data, and the simulation model seeks to support the limitations of the analytical model. Within this context, the main contributions of this work are as follows:

- *Modeling and simulation*: The approach, method, and tool provided may allow, through a number of case studies, the reduction of response times of requesting data in processing systems as well as the better characterization of the system bottlenecks in order to improve dimensioning and thus avoid idle or lacking system capacity;
- *Validation*: We used simplified analytical models to validate the simulation model, including preemptive mechanisms, disk channel and disk files;
- *Performance analysis:* Once validated, we used other types of distribution replacing the exponential. For example, the Gamma, Erlang-k, Uniform and the Triangular were used because they have a variation coefficient less than one and are short-tailed. On the other hand, the Weibull and the Lognormal were used because they can be long-tailed and can have a variation coefficient larger than one. These characteristics are relevant since they may impact system behavior in different ways.

The remainder of this work is structured as follows: Section 2 presents the related work, Section 3 introduces the modeling approach, Section 4 shows the simplified analytical and the simulation models and three case studies containing the analysis of the bottlenecks. Finally, in Section 5 we present the conclusions and future work.

2 BACKGROUND AND RELATED WORK

The key areas of this work are related to performance evaluation through simulation and analysis, and research on reservation systems. In the following paragraphs, we briefly address each one of these topics.

The performance evaluation of systems can be obtained through a real system or through a simulation model. The increasing complexity of discrete systems makes the simulation and validation a demanding task for the design of heterogeneous systems. The global validation of these systems requires new techniques that offer high levels of abstraction and precision of simulation, from the time point of view. The main challenge is the time to synchronize and accommodate different concepts for analytic discrete models. Within this context, Bouchhima et al. (2007) proposes a co-simulation framework for analytic and discrete systems.

Sambamoorthy et al. (2011) show the vision of an intelligent network which is based on the widespread use of modern digital communication techniques for today's systems. As measures of this area, control technologies are being developed and implemented. The role of the communication network is becoming prominent. The dynamics of the energy system are influenced by the delays in communication in the network. Therefore, the extensive integration of the energy system and its communication infrastructure requires that these systems be studied as a single distributed system. The authors show how to alleviate this problem by proposing an errorless implicit synchronization mechanism for the co-simulation framework.

Online reservation systems have escalated in size and number over the last years to facilitate the purchase of goods and services. As an example, one of the contributions is related to a review on the software program Online Bus Ticket Reservation System (OBTRS) for a bus transportation system, a facility which is used to reserve seats, cancellation of reservation and different types of route enquiries used on securing quick reservations. OBTRS is built for managing and computerizing the traditional database, ticket booking and tracking bus and travel made. It maintains all customer, bus, and reservation details.

The approach reduces the time used for making a reservation at the bus terminal and also increases efficiency. The application also has the ability to automatically update records in various files, thus relieving the companys staff the stress of working from file security of data, Nwakanma et al. (2015). Their work exemplifies a body of work that look at the efficiency of reservation systems from an application high-level

perspective. Instead, our work examines performance at the bottlenecks of a reservation system at a much lower level abstraction, i.e. from the task/job level.

In their paper, Fuerst et al. (2018) motivate the need for dynamic resource reservation schemes and show how this is provided by their proposal, namely Kraken. They evaluate Kraken via extensive simulations and a preliminary Hadoop prototype. In cloud environments, the absence of rigorous network performance guarantees may lead to unpredictable task execution times. To address this issue, recently, there have been several proposals on how to provide guaranteed network performance. These proposals, however, rely on computing resource reservation schedules a priori.

Unfortunately, this is not currently practical in cloud environments, where application demands are inherently unpredictable (e.g., due to differences in the input data sets or phenomena, such as failures and stragglers). To overcome these limitations, the authors' proposed solution allows minimum guarantees (at runtime) for both network bandwidth and computing resources. Unlike previous work, their system does not require prior knowledge about the resource requirements of the applications. Instead, it allows the modification of reservations at runtime. Kraken achieves this through an online resource reservation scheme, which comes with provable optimality guarantees.

In Ursini et al. (2016), the authors present a methodology to measure the link capacity and the delay of packets in multi-service networks with Quality of Service requirements using discrete-event simulation. It was observed a significant reduction in the simulation time without significant loss of precision by exploring the reduction of the time of the sample variance due to the large time scale difference between the events occurring in the application (service layer) and the network layer. The simulation models were first validated in relation to the analytical models and then increased with small, significant changes, for example, by just changing the type of service distribution (from exponential to constant, or exponential to Weibull).

In conclusion, we are not aware of any research in the literature that approaches online reservation systems from the perspective adopted in this work, i.e. with an analytical model that is supported by measurements on a real-word reservation system, and a simulation model that is based on the analytical as well as real data, and each model validating the other.

3 MODELING APPROACH

We wish to determine the mean server response time for a given request in a cloud-based reservation system such as the one shown in Figure 1. The mean response time is the time that includes the arrival of a request and the processing of the request including all interrupts caused by such request until the request is finally completed and exit the system. Note that disk accesses such as disk reads and writes are excluded from the server processing (bottleneck 1). The disk channel is treated as bottleneck 2. The disk channel and the disk files are both treated as bottleneck 3. This analysis includes the bottlenecks and does not consider the whole system and the total delay, which is abstracted away, but may be considered in future work (e.g. inclusion of memory-access latency, etc.). Although this analysis is not complete, it is essential, since it may reveal at an early stage critical issues with the system design and dimensioning.

Possible redundancies that are related only to reliability and not to the capacity of the system are not considered. A request is submitted to the system and it is stored in the kernel memory. When a request message is completely assembled, the server is interrupted (*Message-Entry interrupt*) and the message is placed in the worklist. The worklist is organized as a First-In-First-Out (FIFO) discipline. Thus, when a request message reaches the top of the list, it obtains access to the server and begins processing. The system consists of a cloud processing infrastructure with one server that performs a number of accesses to a storage system in order to fulfill client reservation requests.

There are three performance bottlenecks in this system: The first is the interrupt handling on the server itself; the second is the access to the storage, and the third includes both the file-queues (disk contention) and the storage (Figure 1). One client reservation request requires a fixed number of disk accesses to be fulfilled. The pending-disk-requests counter keeps track of the number of remaining disk requests. The

processing of a message in the server consists of one or more disk-file requests (i.e. reads or writes) for either data or programs; once one disk access request is completed, the message is moved to a waiting buffer (area). The control of the server is given to the next message in the waiting list. If there is no further processing, the server moves to the IDLE state. The file requests are sent to the appropriate file queues. When both the access channel and the required disk file are free, a read or write command is given to retrieve the data. Thus, the status of a message request may be: 1) queued for disk access, 2) processing (on the server), and 3) on-hold in the waiting buffer.

The assumption is that the searches are randomly distributed by the file system as a whole, such that there is uniform access. When each file request is completed, the server is interrupted (*File-request-completed* interrupt) and the wait counter for the message that originated that file request is decremented. If the waiting counter is not zero, the message continues to wait. On the other hand, when the counter reaches zero, the message is put on the waiting list for even more processing. When a message (processing) is completed, a response is sent back to the client and the server is again interrupted (*Message-reply* interrupt) for a short period.

Once the server process exits for the given message request, the remnants of the message are cleared from the kernel. Incoming messages consist of five different types. The number of each per peak day is shown in Table 1, along with the number of file or message accesses required. The data is from a global company not disclosed to preserve its identity. The application was adapted to a modern cloud reservation system (Alpha 1971). Notice that peak day is a day when the intensity of traffic is maximum. These peak hours are the ones for which the system traffic calculations are performed.

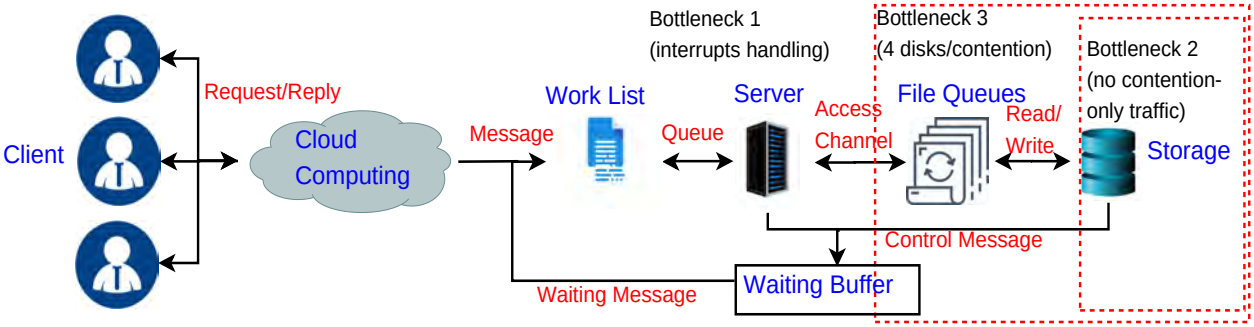

Figure 1: Architecture of the cloud-based reservation system.

The method used to evaluate the performance of the first bottleneck consists of four steps:

1. Measurement of input and system data;
2. Reduction of the input data to events per time unit;
3. Determination of the interrupt rate;
4. Determination of the server's mean response time and utilization.

These steps are addressed in the following sections. Table 2 shows the percentage for each type of message per peak day and the number of disk accesses required by each type.

Next, we determine the types of input rates for the worklist, interrupt rate, and server processing time (this part of the Interrupt handler must be treated as a single server queue with preemptive discipline with resumption from the point at which the service was suspended). Table 3 provides the types of interrupt and processing corresponding to the traffic for the server due to Message entry, Message reply, and File-request complete. The complete interrupt rates for the server are also given in Table 3.

Table 1: Input message traffic per peak hour.

Message type	# messages	# of data reads	# data writes	# program reads
1. Inquiry	25900	2	-	-
2. Sell	31700	2	1	-
3. Passanger data	47500	1	1	-
4. End transaction	31700	4	2	1
5. Cancel	7200	4	2	2
Total	144000	13	6	3

Table 2: Input traffic reduced to events per time unit (tu).

Message type	% of msgs	% of msgs $\times 4$	Data read	Data write	Program read
1. Inquiry	0.18	0.72	1.44	-	-
2. Sell	0.22	0.88	1.76	0.88	-
3. Passenger data	0.33	1.32	1.32	1.32	-
4. End transaction	0.22	0.88	3.52	1.76	0.88
5. Cancel	0.05	0.20	0.80	0.40	0.40
Total access	-	-	8.84	4.36	1.28
Total	1.00	4.00	Sum of disk accesses per tu = 14.48		

Table 3: Interrupt traffic in the server (X_1).

Interrupt type	Time in $tu \times 10^{-3}$	Interrupts/tu
1. Message entry	1	4
2. Message reply	2 ± 1	4
3. File-request complete	2 ± 1	14.48
Total	-	22.48 entries/tu

Table 4 provides the estimate of the processing sequence for each type of message. It includes both the Server IH (Interrupt Handler) processing requirements and the I/O (Input / Output) requirements. For example, in Table 4, the Inquiry message type sequentially requires S(3 u) - 3 time units in the Server, R - 0 tu in the server, since it executes in the disk-file system, S(1 u) - 1 time unit in the Server, R (same as before), S(6 u) - 6 time units in the Server, Rp - 0 tu in the server, since it executes outside the server, and O - 0 time units in the server, since it executes outside the system. On the other hand, the frequency of the entries of jobs in the worklist are obtained from Tables 1, 2 and 4. This information is summarized in Table 5. In this table, $E[X_{2S}]$ is the mean entry time per message type and $E[X_{2S}^2]$ is its second moment per message type.

4 ANALYTICAL AND SIMULATION MODEL

In this section, we first calculate the performance of the model before presenting the implementation of the same using DES simulation with Arena, Rockwell-Automation (2019). The goal is to estimate the job's mean response time and processor (server, disk channel, and disk files) utilization considering the bottlenecks. The data (numerical values) and the full analytical model is presented in Conti et al. (2019). In all case studies, we consider one server and four storage units for the sake of simplicity. However, the case may be extended to n servers and n storage units without compromising the validity of the model. Three case studies are defined and discussed as follows.

4.1 Case 1 - Server: Bottleneck 1

The mean response time for one job entry (T_{q2}) in the server (for a message to get through the system) is given by Equation (1), Jaiswal (1961):

$$T_{q2} = \frac{1}{1-\rho_1}\left[E[X_2] + \frac{\lambda_1 \times E[X_1^2] + \lambda_2 \times E[X_2^2]}{2 \times (1-\rho_1-\rho_2)}\right] \tag{1}$$

where ρ_1 is the utilization of the server due to interrupts, ρ_2 is the utilization of the server due to job processing, λ_1 is the mean interrupt rate, λ_2 is the job arrival rate, $E[X_1]$ is the mean interrupt time, $E[X_2]$ is the mean of the server entry time, $E[X_1^2]$ is the second moment of X_1, and $E[X_2^2]$ is the second moment of X_2. The mean $E[X_1]$ of interrupt time is given by Conti et al. (2019), Equation (2):

$$E[X_1] = \frac{(4\times 1)+(4\times 2)+(14.48\times 2)}{22.48} = 1.82\ tu \times 10^{-3} \tag{2}$$

The mean of the server entry time $E[X_2]$ using the fourth and fifth columns of Table 5 is given by Equation (3):

$$E[X_2] = \frac{2.16\cdot(3.33)+3.52\cdot(8.75)+3.96\cdot(3.00)+6.16\cdot(8.00)+1.40\cdot(8.00)}{17.20} = 6.42\ tu\cdot 10^{-3} \tag{3}$$

The calculation of the second moment of the interrupt time $E[X_1^2]$ and second moment of the entry time $E[X_2^2]$ can be seen in Conti et al. (2019), as well as further information about the analytical model, in which we add 8 $\frac{entries}{tu}$ with the 14.48 $\frac{entries}{tu}$ $\rightarrow \lambda_1$=22.48 $\frac{entries}{tu}$ (Table 3). The use of the server due to interrupts is given by Equation (4), i.e. the average interrupt rate times the average interrupt time ($E[X_1]$=0.00182 tu):

$$\rho_1 = 22.48\frac{entries}{tu} \times 0.00182\frac{tu}{entries} = 0.041 = 4.1\% \tag{4}$$

The server utilization due to the processing of the entries (message processing) ρ_2 of the worklist is given by the arrival rate of entries ($\lambda_2 = 17.20$ entries/tu) times the average execution time of a entry ($E[X_2] = 0.00642$ tu), as in Equation (5):

$$\rho_2 = 17.20\frac{entries}{tu} \times 0.00642\frac{tu}{entries} = 0.111 = 11.1\% \tag{5}$$

The total server utilization ρ_T is then given by $\rho_T = \rho_1 + \rho_2 = 0.152 = 15.2$. Numerically, from Equation (1), we have Equation (6) for preemptive resume discipline in a lower-priority queue:

$$T_{q2} = \frac{1}{1-0.041}\left[6.42 + \frac{22.48\times 0.00374 + 17.20\times 0.1163}{2\times(1-0.152)}\right] = 7.97\ tu\times 10^{-3} \tag{6}$$

Table 4: Job processing (service) time sequence by message type (u = unit = tu $\times$ 10^{-3}).

Message type	Jobs
1. Inquiry	S(3), R, S(1), R, S(6), Rp, O
2. Sell	S(3), R, S(1), R, S(30), Rp, W, S(1), O
3. Passenger data	S(3), R, S(5), Rp, W, S(1), O
4. End transaction	S(3), R, S(1), R, S(15), Rp, W, Pr, S(10), R, S(1), R, S (25), W, S(1), O
5. Cancelation	S(3), R,Pr, S(1), R, S(5), R, W,Pr, S(10), R, S(1), R, S(35), W, S(1), O
Legend: S- Server, R- Read, W- Writing, Rp- Reply, O- Output, Pr- Program Reading	

Table 5: Traffic to the worklist.

Msg type	Msgs per tu	Server entries per msg	Server entries per tu	Mean proc. time $E[X_{2S}]$ ($tu \times 10^{-3}$)	Second moment $E[X_{2S}^2]$ ($tu^2 \times 10^{-6}$)
1. Inquiry	0.72	3	2.16	3.33	15.3
2. Sell	0.88	4	3.52	8.75	228.0
3. Pass. data	1.32	3	3.96	3.00	11.7
4. End Trans.	0.88	7	6.16	8.00	137.4
5. Cancel.	0.20	7	1.40	8.00	194.7
Total	4.0	–	17.20 ent./tu	–	–

The discrete event simulation model for the system including bottleneck 1 is shown in Figure 2. An event generator creates entities which are specialized into messages or interrupts. Messages remain in partial processing while there are pending requests:

- *Message generator:* It determines the arrival time for the request and it is defined by the Exponential Distribution (1 / 17.20) (Table 5). Considering the five message types, we have: *Inquiry*, with probability of requests being 2.16 / 17.20; *Sell*, with probability 3.52 / 17.20 ; *Passenger data*, with probability 3.96 / 17.20; *End of transaction*, with probability 6.16 / 17.20 and *Cancellation*: with probability 1.40 / 17.20;
- *Interrupt generator:* Another generator to consider 22.48 interrupts / tu or an interrupt of the server every 1 / 22.48 tu (more interrupts in relation to the number of messages) for each type of service. Therefore, the interrupts have three types of services: Message Entry with 4/22.48, for Message Reply with 4/22.48 and File-request Complete with 14.48/22.48 (Table 3).

The interrupts are related to the processing of messages, although for the sake of the implementation they were treated separately (without sacrificing functionality or performance). The processing of the simulation model takes into account whether or not there are pending requests (i.e. partial processing) for any given message under processing. The model takes into account the preemption of jobs in the server in bottleneck 1.

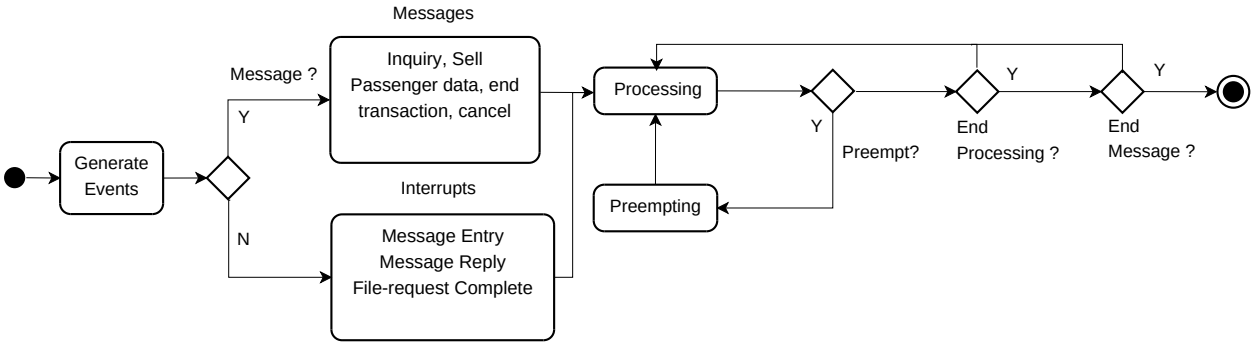

Figure 2: Simulation model - Server - bottleneck 1.

The implementation of the system (code) can be found in Conti et al. (2019). The result of this analysis can be seen in Figure 3. Eight replications were executed, each with 10000 tu, and the results of the messages, interrupts and server were observed. The *Interrupts* variable indicates the average processing time of the interrupts. The *Messages* variable indicates the average processing time of the messages. The simulators *Interrupt-handler Busy* variable indicates the average wait time of the server for both the worklist and the interrupts (the residence time is the sum of the waiting time plus the service time). For the Messages variable, the replications presented results with an average value of 7.92 $tu \times 10^{-3}$, and a

Figure 3: Server performance (% utilization) per replication: blue=messages (left bar), green = interrupts (middle bar), yellow = IH busy (right bar).

confidence interval between 7.90 $tu \times 10^{-3}$ and 7.94 $tu \times 10^{-3}$. For the Interrupts variable, the replications presented results with an average value of 1.860 $tu \times 10^{-3}$ and a confidence interval between 1.859 $tu \times 10^{-3}$ and 1.861 $tu \times 10^{-3}$. For the Interrupt handler (I. H.) Busy variable, the replications presented results with an average value of 15.143 % and a confidence interval between 15.140 % and 15.146 % of Interrupt handler usage. The results indicate that the average processing times for both the messages and the interrupts observed in the simulation model are close to the analytical model. We also observed that the percentage of Interrupt handler utilization of the simulation model is also close to the result of the analytical model.

4.2 Case 1 - Server: Bottleneck 1 - Using other probability distributions

To improve the simulation model, other probability distributions are used to try to identify details about the analytical model. However, in practice, we must make new measurements in the system to adjust the best possible distribution. The results for the new distributions can be seen in Table 6. The Weibull, Gamma, Lognormal, Uniform, and Triangular distributions were applied and compared to the Exponential distribution used in the simulation model.

Table 6: Other probability distributions ($tu \times 10^{-3}$) - Case Study 1.

Distribution	Expo	Weib	Gamm	Logn	Unif	Tria
Messages	7.92	7.85	7.93	8.53	7.28	6.82
Interrupts	1.86	1.87	1.87	1.87	1.87	1.86
Interrupt handler Busy (%)	15.1	15.6	15.3	15.1	15.0	15.1

By recalculating the values of the Exponential distribution EXPO(1/17.20) used in the simulation model, we have the following values for the distributions: *Weibull*: WEIB(0.0568, 1.05), *Gamma*: GAMM(0.0571, 1.01), *Log Normal*: LOGN(0.05814, 0.11628), *Uniform*: UNIF(0.001 , 0.11627), *Triangular*: TRIA(0.01, 0.05814, 0.10628).

We have used the same mean value from the experimental data to calculate the new probability distributions. For the messages, the Weibull, Gamma, Log-Normal and Uniform distributions present a small variation in relation to the Exponential distribution used in the simulation model. On the other hand, for the Triangular distribution, a larger variation in results was observed. For the Interrupts and interrupt handler, there are no large variations of the results between the distributions under analysis. Table 7 presents the variation of the results of the analytical model with the simulation model. For the messages, the analytical model presents 7.97 tu $\times 10^{-3}$ of processing time, the simulation model results in 7.92 tu $\times 10^{-3}$ of processing time with 0.6 % of the variation between them. For the Interrupts, the analytical model presents 1.82 tu $\times 10^{-3}$ of processing time, the simulation model equals 1.86 tu $\times 10^{-3}$ with 2.2 % of the variation. For the server, the analytical model presents 15.2 % and the simulation model yields 15.1

% with 0.6 % of the variation. The variation between the simulation and analytical models is calculated from the ratio (Analytical model - Simulation model)/Analytical model.

Table 7: Analytical Model vs. Simulation Model ($tu \times 10^{-3}$) - Case Study 1.

Model	Analytical	Simulation	% Variation	Validation
Messages (system time, Tq2)	7.97	7.92	0.6	✓
Interrupts (system time)	1.82	1.86	2.2	✓
Interrupt handler Busy (%)	15.2	15.1	0.6	✓
system time = queueing + service time				

4.3 Case 2 - Disk Channel: Bottleneck 2

The total disk turnover (given by Table 2) is λ = 14.48 access/tu (time unit). The use of the disk channel is given by: $\rho_c = \lambda T_c = 14.48 \times 0.266 = 0.385 = 38.5\%$, where 0.266 *tu* is the mean of Channel Service. Further information about the analytical model can be found in Conti et al. (2019). The analysis of the simulation model is presented in Table 8.

The simulation model for Disk Channel is shown in Figure 4. There are 14.48 accesses / tu, and the input data is proportionally distributed according to its type. The parameters for Data read, Data write and Program read can be found in Table 9. The objective of the simulation model is to evaluate the mean and the variance of the processing time, actually the traffic that is $\lambda \times$ *average holding time* $= \rho$. We inserted 30 servers to the model (it may be any large number) and applied the First-In-First-Out (FIFO) queue discipline, Statistics (Average Response Time) and (Standard Deviation of Response Time) and 4000 tu for simulation time. It is observed that the mean and variance values, as expected, are validated with the ρ of the analytical model, Conti et al. (2019).

Table 8: Disk Channel - Results of the Simulation Model (%) - Bottleneck 2.

Model	Analytical	Simulation	C.I.	% Variation	Validation
Disk Channel	38.50	38.45	38.06 ; 38.83	0.003	✓

4.4 Case 2 - Disk Files with Disk Channel: Bottlenecks 2 and 3

The equations and further information about the analytical model can be found in Conti et al. (2019). The total disk turnover given by Table 2 is λ = 14.48 entries/tu. The use of the disk channel is given by $\rho_c = \lambda T_c = 14.48 \times 0.266 = 0.385 = 38.5\%$. The use of each access mechanism is given by $\frac{\lambda T_p}{m} = \frac{14.48 \times 0.118}{4} = 0.427 = 42.7\%$, where 0.118 *tu* is the mean response time considering an adapted M/G/1 model (machine repairmen with external inputs and four parallel mechanisms), and fitting an Erlang-k distribution as in Conti et al. (2019), (Figure 4.) The simulation model for Disk Channel and Disk Files is shown in Figure 4. The assumption is that there are four disk files and thus at most four requests can be serviced simultaneously at any time. The Process (Server) serves four disk files. We insert an Arrive that has a rate of: $1/(14.48/4 \times 4)$. That is, the system is set to work with 4 messages / tu and when we evaluate the increase to 8 msg / tu, for example, that rate becomes $1/(14.48/4 \times 8)$. Since we only have 4 disk files, there is a Hold 4 block that holds the new requests until there are less than 4 accesses in the system. We suggest fitting an Erlang-k to this model from the mean squared values (Mean Response time squared) and the square of the standard deviation of response time (variance). The simulation was executed for three runs during 3000 tu each run. The result shows for the Average Response Time the value of 0.12290 tu, and for the Standard Deviation Response Time, the value of 0.3402 tu. The results obtained from the simulation model for the disk channel and disk files are presented in Table 10 and compared with the results obtained from the analytical models. For Disk Channel, we note that the analytical model

Table 9: Probabilities for disk channel utilization (from Table: 2).

Parameter	Data read	Data write	Program read
Probability	8.84 / 14.48	4.36 / 14.48	1.28 / 14.48
Process time (tu $\times$ 10^{-3})	UNIF(0.25)+2.5+2	UNIF(0.25) +27.5+2	UNIF(0.25)+25+2

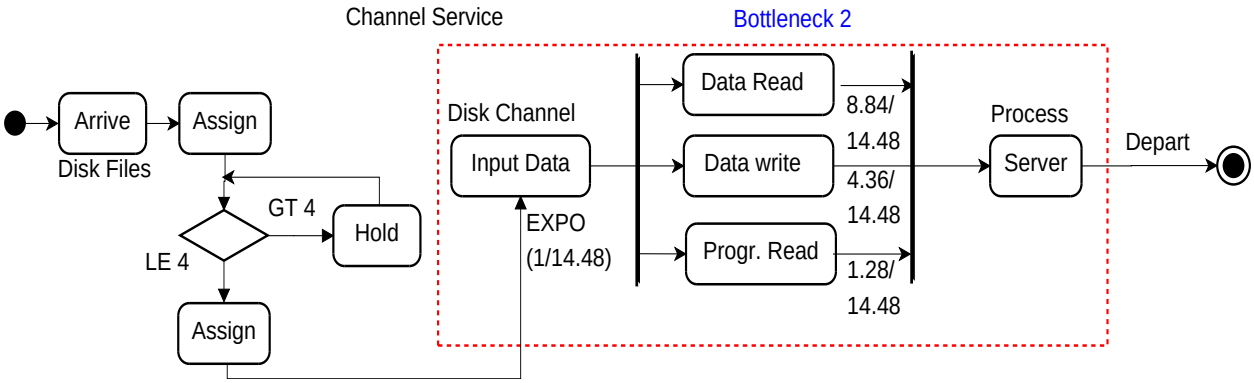

Figure 4: Simulation Model - Disk Files with Disk Channel - bottlenecks 2 and 3.

presents a server utilization of 38.50 %, and the simulation model 38.45 % with a confidence interval between 38.06 % and 38.83 %. The variation between the analytical model and the simulation model is 0.003 %. For Disk Files, we note that the analytical model results in a server utilization of 42.70 %, the simulation model yields 44.49 % with a confidence interval between 44.15 % and 44.83 %. The variation value, i.e. (Analytical model - Simulation model)/Analytical, between the analytical and the simulation model, is 0.040 %.

Table 10: Disk Files with Disk Channel - Utilization (%) - Bottlenecks 2 and 3.

Model	Analytical	Simulation	C.I.	% Variation	Validation
Disk Channel	38.50	38.45	38.06 ; 38.83	0.003	✓
Disk Files	42.70	44.49	44.15 ; 44.83	0.040	✓

4.5 Case 3 - Disk Files with Disk Channel: Bottlenecks 2 and 3 - increasing the input traffic

This case presents the validation for four msg/tu regarding mean and variance of response time. Table 11 shows satisfactory results for both the analytical and the simulation model. The analytical model was adjusted with Erl-17 phases and the simulation model was adjusted with Erl-13 phases. We present the results obtained considering the application of a number of messages per tu. We also show the validation of the analytical model by the simulation model (Table 12).

It is very important that analytical models are used, even if approximate, to compare with simulation models and vice-versa. The basic model, with 4 msg / tu was validated by the simulation model. From there, the simulation model was less pessimistic than the analytical model. The analytical model did not allow the use of a load with 8 msg / tu, whereas the simulation model allowed it. The reservation system may still be used even with 9 msg / tu. However, the simulation model allows us to estimate the server's utilization with a load of almost 87 % at 8 msg / tu. Then, two tests were performed, with 8 msg / tu (using arrivals with coefficient of variation (c.v.) lower than one, Erl-3, and with c.v. greater than one, Lognormal (with the second parameter equal to three times the mean). We can see that the disk channel utilization remains the same (items 3, 6 and 7, Table 12)), but the use of Lognormal, item 7, makes it unfeasible because the traffic ρ exceeds 1.0. In this way, having the simulation model validated, it would be worthwhile to make measurements "in the field" on the arrival rates of each type of request to the disk-files

in order to better estimate the arrivals. In fact, the most significant overhead (utilization and delays) is not the channel occupation, but the set of disk channel and disk files.

Table 11: Disk Files & Channel - Mean and variance of response time (4 msg/ tu - Bottlenecks 2 and 3).

Model	Analytical	Simulation	Validation
Mean	165 tu $\times 10^{-3}$	172 tu $\times 10^{-3}$	✓
Variance	8900 tu^2 $\times 10^{-6}$	9667 $tu^2 \times 10^{-6}$	✓

Table 12: Disk Files with Disk Channel - Analysis of the Simulation Model (%) - Bottlenecks 2 and 3.

Item	Msg/ tu	Distribution	Disk Files %	Disk Channel %	Validation
1	4	Expo	44.49	38.45	✓(Table 10)
2	7	Expo	53.64	67.47	-
3	8	Expo	61.40	76.73	-
4	9	Expo	80.97	86.57	-
5	10	Expo	188.29	96.16	-
6	8	Erlang-3	50.98	76.84	-
7	8	LogNormal	136.77	76.75	-

5 SUMMARY AND CONCLUSIONS

In this work, we proposed a discrete-event simulation model for a multi-processing reservation system. The model consisted of a server and message queries that share a file/storage system. We validated the simulation model with an analytical model considering message types and server interrupts. We have not identified in the literature on performance evaluation of online reservation systems, simulation models that have been validated for the analysis of the message traffic in a multiprocessing system. In the absence of such validated model, it is necessary to resort to data from a real system to carry out such performance analysis. Such campaigns can be costly and in many cases disrupt the operation of a real-world system.

In addition to server utilization, we evaluated the processing time for messages and interrupts. For model processing in the first bottleneck, we considered eight replications and we have not found significant variations between them. We also evaluated the utilization of the disk files and disk channel considering the increase in the number of messages. Other probability distributions were taken into account, which allowed further validation/verification of the simulation model. This model may be adapted and further extended to accommodate other (possibly more complex) configurations.

As future work, we suggest the use of other distributions, by changing the probability types of the input values of the model, as well as the increase of the number of servers used in the processing. The model and method used constitute an essential tool for tunning the behavior of existing systems as well for planning and dimensioning of the performance of future generations.

REFERENCES

Alpha, C. 1971. *Multi Processing System*. 2rd ed. New York, NY: Company Alpha.

Bouchhima, F., M. Brière, G. Nicolescu, M. Abid, and E. M. Aboulhamid. 2007. "A SystemC/Simulink Co-simulation Framework for Continuous/Discrete-events Simulation". In *Proceedings of the 2006 IEEE International Behavioral Modeling and Simulation Workshop*, 1–6. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Conti, J. C., E. L. Ursini, and P. S. Martins. 2019. "Analytical Modeling of a Cloud-Based Reservation System". Technical report. https://www.researchgate.net/publication/332380564_Analytical_Modeling_of_a_Cloud-Based_Reservation_System, accessed 8^{th} August 2019.

Fuerst, C., S. Schmid, L. Suresh, and P. Costa. 2018, February. "Kraken: Online and Elastic Resource Reservations for Cloud Datacenters". *IEEE/ACM Transactions on Networking*. 26(1):422–435.

Jaiswal, N. K. 1961. "Preemptive Resume Priority Queue". *Operations Research* 9(5):732–742.
Nwakanma, I., C. Etus, I. Ajere, and U. Agomuo. 2015. "Online Bus Ticket Reservation System". *Statistics and Computing* 1:1–17.
Rockwell-Automation 2019. "Arena Simulation Model". https://www.rockwellautomation.com/rockwellsoftware/simulation.page, accessed 29^{th} July 2019.
Sambamoorthy, S., S. Shukla, J. Thorp, and L. Mili. 2011. "Power System and Communication Network Co-simulation for Smart Grid Applications". In *In Proceedings of the Innovative Smart Grid Technologies (ISGT)*, 1–6. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
Ursini, E. L., P. S. Martins, V. S. Timoteo, and F. R. Massaro. 2016. "Modeling and Simulation Applied to Link Dimensioning of Stream IP Traffic with Incremental Validation". In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. M. Roeder, R. Szechtman, and E. Zhou, 3049–3060. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

JOSE C. CONTI is a doctoral candidate at the School of Technology, Unicamp. He holds a Master Degree in Technology from Unicamp. He works with database systems' analysis, discrete event simulation and data integration systems. He has experience with SAS, Oracle SQL. His email is j078615@dac.unicamp.br.

EDSON L. URSINI is an Associate Professor at School of Technology, Unicamp with a doctoral degree in Electrical Engineering (electronics and communications) from Campinas State University. He works with data networks, stochastic processes, queuing models, teletraffic and data traffic, discrete event simulation, continuous simulation, fuzzy logic systems, and performance analysis. His email is ursini@ft.unicamp.br.

PAULO S. MARTINS is an Associate Professor at the University of Campinas and holds a DPhil degree in Computer Science from The University of York, UK. He works with network performance analysis, real-time systems, and discrete-event simulation. He has worked in both academia (USA, Brazil) and industry (USA, UK) including automotive and military projects with the US Air Force. His email is paulo@ft.unicamp.br.