



Produto & Produção, vol. 17, n. 1, p. 1-12, mar. 2016

RECEBIDO EM 27/06/2014. ACEITO EM 01/12/2015.

João José de Assis Rangel

Universidade Candido Mendes - UCAM-Campos
joao@ucam-campos.br

João Victor da Silveira Costa

Universidade Candido Mendes - UCAM-Campos
joaovdscostal@gmail.com

Quézia Manuela Gonçalves Laurindo

Universidade Candido Mendes - UCAM-Campos
manulaurindo92@hotmail.com

Túlio Almeida Peixoto

Universidade Candido Mendes - UCAM-Campos
tulioap@gmail.com

Ítalo de Oliveira Matias

Universidade Candido Mendes - UCAM-Campos
italo@ucam-campos.br

Análise do fluxo de operações em um servidor de *e-mail* através de simulação a eventos discretos com o *software* livre Ururau

RESUMO

Este trabalho apresenta um modelo de simulação a eventos discretos para análise do processamento de um servidor de *e-mail*. São analisados o envio e o recebimento de mensagens, levando em consideração o período do *lead time* dos mesmos. O modelo de simulação considerou todo processamento do servidor, desde a chegada de *e-mails* até o envio para os destinatários. Assim, o modelo proporcionou a visualização dos tempos gastos em cada processo, a utilização dos recursos e, em especial, o principal problema enfrentado pela empresa, que é o alto *lead time* das mensagens no servidor. As simulações foram realizadas utilizando o *software* livre de simulação Ururau. Os resultados mostraram a possibilidade de formação de filas devido ao aumento da quantidade de *e-mails* e usuários e, desta forma, auxiliar os gestores a avaliar a performance do sistema e decidir a melhor maneira de alocar recursos de *hardware* e *software*.

Palavras-chave: Simulação, Servidor, *Lead time*.

ABSTRACT

This paper presents a discrete event simulation model for analysis of processing of a mail server administered by ESSys Solutions LTD (ESSys soluções LTDA). Sending and receiving messages are checked taking into account the period of their lead time. The simulation model considered all server processing from arrival to sending e-mails to recipients. The model provided a visualization of time spent on each process, the use of resources and, in particular, the main problem faced by the company, which is the highest lead time of messages on the server. The simulations were carried out using the free simulation software Ururau. The results showed the possibility of queues due to the increase of the amount of e-mails and users and, this way, help the managers evaluate the performance of the system and decide the best way to allocate hardware and software resources.

Keywords: *Simulation, Server, Lead time.*

1. Introdução

O objetivo deste trabalho foi simular o tráfego de envio e recebimento de mensagens do servidor de *e-mail* administrado pela ESSys soluções LTDA. A análise foi realizada a fim de auxiliar a empresa na avaliação da utilização dos seus recursos computacionais. Segundo Miranda *et al.* (2012), a simulação a eventos discretos tem sido utilizada de forma crescente para auxílio à tomada de decisões devido, principalmente, à sua versatilidade, flexibilidade e poder de análise. No setor de serviços, como a área de Internet, por exemplo, embora sua aplicação ainda não seja muito difundida nos ambientes de manufatura e logística, esta tendência começa a ser percebida (LAW, 2007 e THOMAS, *et. al.*, 2015).

A ESSys soluções LTDA é uma empresa que trabalha com prestação de serviços e distribui, no Brasil, diversas soluções para pontos de venda ou comércio eletrônico. As simulações realizadas neste trabalho permitiram aos seus gestores verificarem a performance do sistema e analisarem a possibilidade de investimento em *hardware* e/ou *software* para atender à demanda requisitada por seus usuários e possíveis expansões futuras.

As simulações foram realizadas com o *software* Ururau, apresentado em detalhes em Peixoto *et al.* (2013). Este *software* é de código aberto e livre de custos (FOSS em Inglês: *free and open source software*). O Ururau é a continuação do projeto da biblioteca de *software* Java Simulation Library (JSL), proposto por Rosseti (2008). No trabalho, Rosseti apresenta a estrutura interna do JSL e indica, como sugestão de trabalho futuro, a construção, dentre outros itens, de uma interface gráfica para dar suporte à construção de modelos de simulação por analistas não especialistas em linguagem Java. Foi justamente desta indicação que surgiu a proposta que deu origem ao projeto do *software* Ururau.

Para a construção de um modelo de simulação com o Ururau, o modelador precisa apenas arrastar e soltar com o mouse os elementos de modelagem para a área de trabalho, de acordo com a lógica do sistema em análise. Recentemente, alguns trabalhos foram realizados empregando o Ururau na análise de sistemas a eventos discretos. Dentre estes, podem-se destacar Silva *et al.* (2014), Rangel e Cordeiro (2015) e Rangel *et al.* (2013). Nestes trabalhos, foram realizadas análises dos sistemas sob investigação com bons resultados que permitiram, assim, auxiliar na tomada de decisão dos problemas em questão.

Apesar da existência de mais de cinquenta *softwares* que podem ser empregados na construção de modelos de simulação a eventos discretos, como citado em Swain (2007), ainda são poucos os *softwares* do tipo livre. Recentemente, King and Harrison (2013) apresentaram o também FOSS denominado de JaamSim, e destacaram como uma das suas principais características a de permitir que os usuários criem seus próprios objetos de alto nível para uma determinada classe de modelos. Assim, uma vez definida uma classe de modelos como, por exemplo, modelos de simulação de sistemas de processamento de um servidor de *e-mail*, podem-se criar os próprios objetos para futuras modelagens de um sistema com características semelhantes.

2. Descrição do sistema

O sistema sob análise refere-se à operação de um servidor de *e-mail* que processa mensagens durante 24 horas por dia, 7 dias por semana e 30 dias por mês. Normalmente, o servidor precisa estar disponível durante todo este período para que os seus usuários possam utilizar o serviço na forma requerida. Essa operação ininterrupta requer grande quantidade de processamento e armazenamento de dados. A Figura 1 descreve a estrutura do serviço de *e-mail* realizado. Ou seja, o servidor recebe mensagens tanto dos clientes que possuem conta no servidor quanto de servidores externos, processa as mesmas e as envia a outros usuários da mesma rede. Quando o *e-mail* é destinado a usuários de servidores externos (gmail.com, outlook.com, etc.), ele redireciona as mensagens eletrônicas para os mesmos.

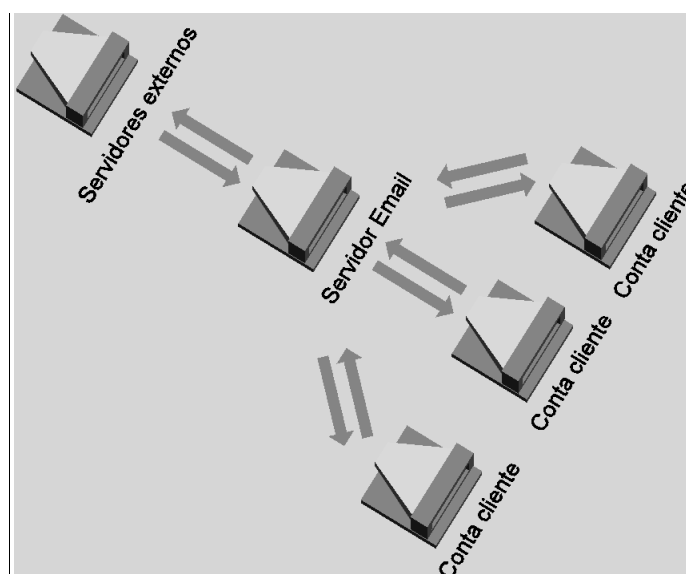


Figura 1 – Funcionamento do sistema de envio e recebimento de *e-mails* no servidor

Fonte: Próprio autor

Existem, aproximadamente, 55 contas de *e-mails* criadas no servidor. Porém, o fluxo de informações trocadas entre elas é muito elevado, devido à grande necessidade de se manter uma troca de conteúdo, bem como o fluxo de informação, por se tratar de um servidor que automatiza vários processos internos como o de notificação de processos e acontecimentos de uma empresa. O servidor utilizado para o processamento das mensagens emprega a tecnologia de “processamento em nuvem”. A arquitetura da computação em nuvem também reduz de maneira inteligente o consumo de energia. É utilizado um servidor com 2 processadores contendo 4 núcleos cada, 16Gb de memória RAM e 1Tb de capacidade de armazenamento. O sistema operacional utilizado é o *Red Hat Enterprise Linux Server*. Em termos de *software*, ele é equipado com o Zimbra, que é responsável por fazer o gerenciamento do envio de *e-mail*, e também com um *software* cuja função é fazer a análise de spam e de vírus nos *e-mails* processados pelo mesmo.

O tráfego no servidor é intenso. O histórico de todo processamento é armazenado em um arquivo de Log para que possa ser analisado, posteriormente, caso exista algum problema no processamento das informações.

Uma pequena amostra do arquivo de Log foi coletada por alguns instantes para ilustrar o tamanho de fluxo da informação nos servidores. Este arquivo está apresentado no Anexo A. Por questões de confidencialidade e espaço no texto, não é apresentado o referido arquivo completo. Vale ressaltar ainda que, no arquivo de Log, foram removidas diversas informações importantes para a empresa e que não puderam ser ditas devido a normas internas da mesma. Foi utilizado, como referência, o dia 27 de setembro de 2013, por ser um dia típico de operação do sistema.

No total desse dia, colheu-se uma amostra de 651.052 *e-mails*, tanto enviados quanto recebidos, totalizando o geral. Nesse total, 576.158, ou 88,5 %, chegaram ao seu destinatário ou podem ter sido considerados como um *e-mail* autêntico, pois não eram spam nem considerados vírus. Já, 62.859, ou 9,65%, foram detectados pelo *software* como spam, e os outros 12.035, ou 1,85%, foram constatados pelo *software* antivírus como um vírus. Esses dados são apresentados pela Tabela 1.

Tabela 1 – Quantidades e percentual de *e-mails* processados.

Tipo de <i>e-mail</i>	Quantidade	Percentual
Autênticos	576.158	88,50%
Spam	62.859	9,65%
Vírus	12.035	1,85%
Totais	651.052	100%

Fonte: (ESsys soluções LTDA.).

A Figura 2 ilustra, com mais detalhes, a lógica do funcionamento do fluxo de mensagens no servidor. Quando uma mensagem é processada pelo servidor, ela passa por uma fila de processamento, pois todas as mensagens são analisadas uma a uma, tanto pelo *software* anti-spam quanto pelo antivírus. Quando essa fila ultrapassa o limite, as mensagens são gravadas em memória auxiliar. Para manter um desempenho regular e sem perdas, deve-se ter até mil mensagens na fila de processamento de forma que os usuários não percebam lentidão no sistema. Após a mensagem ser processada e identificada como vírus ou spam, ela sai do sistema, neste caso, já que o sistema avalia apenas o servidor.

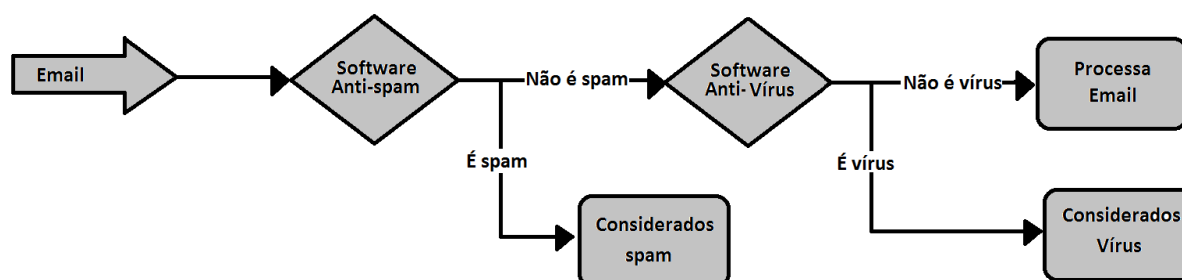


Figura 2 – Lógica do sistema do servidor de *e-mail*.

Fonte: Próprio autor.

Um problema encontrado é o período de *lead time* do envio e dos recebimentos das mensagens no servidor. Considera-se um índice aceitável de 1 a 3 minutos o tempo entre um envio e o recebimento desse *e-mail* por parte do usuário (destinatário). Esse tempo foi referenciado nesta pesquisa como *lead time* (tempo desde o envio até o recebimento de uma mensagem eletrônica). Esse número foi estipulado por meio de uma pesquisa com consulta online com 30 usuários do servidor de *e-mail* da empresa objeto deste estudo. O número de usuários consultado foi estipulado de forma preliminar para esta análise, já que não são conhecidas indicações para servidores de *e-mails* corporativos semelhantes. No entanto, este ponto requer maior refinamento estatístico, mas não está no escopo deste trabalho.

3. Modelo de simulação

O modelo de simulação foi construído utilizando a metodologia proposta por Banks (2010), com os seguintes passos específicos para este trabalho: Formulação e análise do problema; planejamento do projeto; formulação do modelo conceitual; coleta de macroinformações e dados; tradução do modelo; verificação e validação; projeto experimental; experimentação; interpretação e

análise estatística dos resultados; e documentação e apresentação dos resultados. O modelo conceitual foi elaborado utilizando a linguagem IDEF-SIM, proposta por Montevechi *et al.* (2010).

Esta linguagem conceitual utiliza e adapta elementos lógicos das técnicas de modelagem de processos de negócios IDEF0 e IDEF3, permitindo a elaboração de modelos conceituais com informações úteis ao modelo computacional. Assim, a linguagem permite uma melhor documentação de modelos computacionais, facilitando o entendimento da lógica e regras operacionais do sistema projetado (PEREIRA *et al.*, 2014).

As simulações apenas foram iniciadas após o modelo ter sido verificado e validado por completo, seguindo, adicionalmente, a metodologia proposta por Sargent (2013). O modelo conceitual foi elaborado com apoio do *software* DIA. Este *software* tem por finalidade auxiliar a implementação de modelos conceituais em IDEF-SIM. Tanto a linguagem IDEF-SIM como o *software* DIA foram desenvolvidos pela Universidade Federal de Itajubá (UNIFEI). O *software* DIA pode ser baixado livremente em: (<http://www.unifei.edu.br/projeto-dia>).

O modelo de simulação foi baseado em dados históricos do processo de informações do servidor de *e-mail*. Cada rodada de simulação foi realizada com 120 segundos. Foram executadas rodadas com 150 e 200 repetições. Uma vez apresentados os mesmos resultados, sem diferença estatística significativa entre eles, a escolha foi para fazer experimentos com 150 replicações cada. O Anexo A apresenta uma pequena parte do arquivo utilizado para a amostra e foi escolhido a partir de um dia típico de operações.

A Tabela 2 apresenta os parâmetros de rodada do modelo de simulação para o envio e recebimento de *e-mail*.

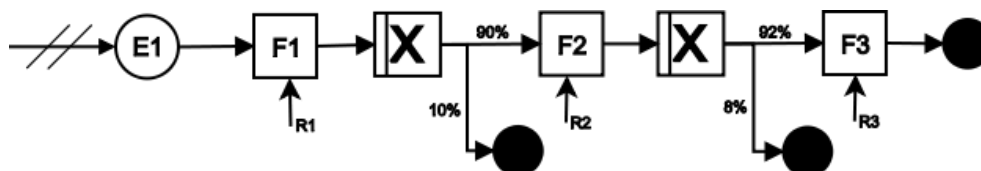
Tabela 2 – Parâmetro geral de rodada do modelo de simulação de envio e recebimento de *e-mail*.

Parâmetros do sistema	
Tempo de replicação	120 segundos
Número de replicações	150 vezes
Tempo de aquecimento	0 minuto

Fonte: Próprio autor

O modelo de simulação utilizou funções do tipo LogNormal para a chegada de *e-mails*, para o processamento do *software* anti-spam e, também, para o processamento do *software* antivírus. Para o processo de entrega de *e-mail* para os usuários locais ou então para os usuários externos, utilizou-se a função Normal. Foram utilizadas distribuições estatísticas no modelo obtidas por meio da análise dos Logs do servidor de *e-mail*, servindo como dados de entrada para a obtenção das mesmas. O *software* utilizado para gerar todas as funções de distribuição dos processos do sistema foi o *Input Analyzer*.

A Figura 3 apresenta o modelo conceitual do sistema com as informações referentes aos dados do sistema.



Legenda

	Descrição	Parâmetros
E1	Chegada de <i>e-mails</i>	Função: LOGN(1.92, 2.06) seg; 1 por vez; máximo infinito; primeira chegada 0 seg
F1	Processo de verificação de spam	Função: LOGN(0.575, 0.361) seg; quantidade: 1
F2	Processo de verificação de vírus	Função: LOGN(0.275, 0.081) seg; quantidade: 1
F3	Processo de entrega de <i>e-mail</i>	Função: NORM(3.39, 1) seg; quantidade: 1
R1	Recurso <i>software</i> Anti-Spam	Quantidade: 1
R2	Recurso <i>software</i> Antivírus	Quantidade: 1
R3	Recurso entrega de <i>e-mail</i>	Quantidade: 1

Figura 3 - Modelo conceitual do sistema do servidor de *e-mail*.

Fonte: Próprio autor.

4. Experimentos e análise dos resultados

As Figuras 4 e 5 apresentam os modelos de simulação desenvolvidos nos *softwares* Ururau e Arena, respectivamente. Observe a semelhança entre os modelos de simulação. Note ainda a semelhança destes modelos de simulação e o modelo conceitual em IDEF-SIM mostrado na Figura 3. Esta semelhança pode ser utilizada para acelerar a curva de aprendizado e migração de um ambiente para outro.

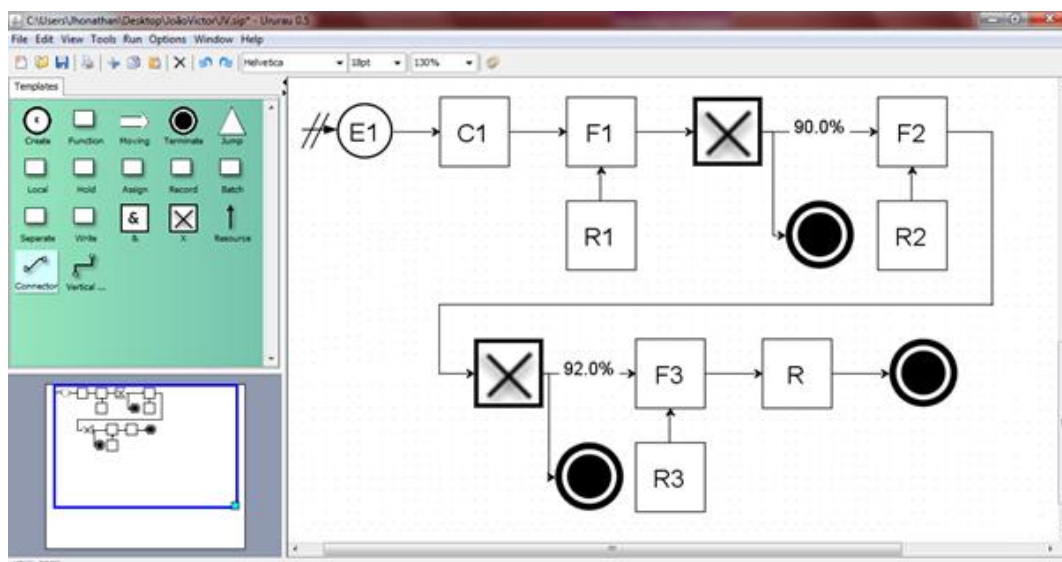


Figura 4 - Modelo construído no ambiente de simulação Ururau.

Fonte: Próprio autor.

O modelo de simulação funciona da seguinte forma: No momento em que a mensagem chega ao servidor de *e-mail*, ela é encaminhada para um processo de análise e verificação do conteúdo para validar se é spam ou não. Com as mensagens que não são validadas como spam, é feita uma nova verificação das mesmas para checar a existência ou não de vírus. Caso não haja vírus nas mensagens, elas são disparadas para a caixa de *e-mail* do usuário.

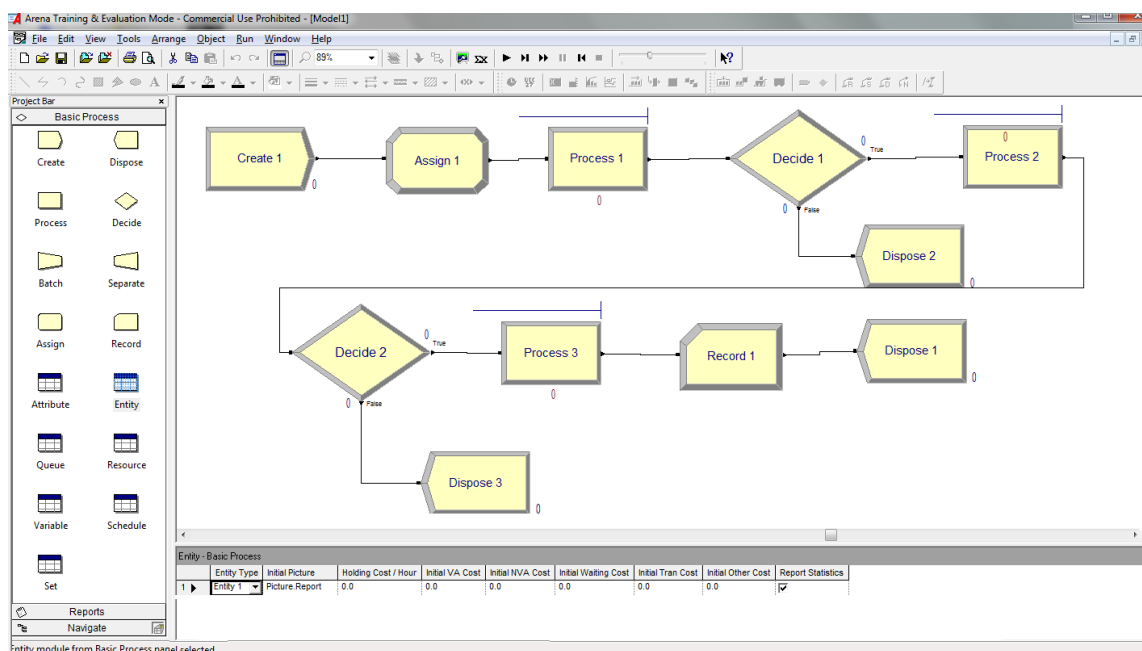


Figura 5 - Modelo construído no *software* Arena.

Fonte: Próprio autor.

Os experimentos foram realizados através de um projeto fatorial 2^k , que determina o efeito de K fatores, cada um dos quais com apenas duas possibilidades ou níveis (MONTGOMERY, 2009).

Por ser um dos principais objetivos do trabalho analisar a adequação do servidor para a demanda atual de *e-mail* e, principalmente, um provável constante crescimento de envio e recebimento de *e-mails* futuramente, o primeiro fator escolhido para verificação foi o tempo entre a chegada dos *e-mails*. Será observado o que ocorre quando se diminuem 50% do tempo de chegada, aumentando, dessa forma, a quantidade de *e-mails* que são processados pelo servidor, fazendo, assim, um teste massivo na sua capacidade de processar e encaminhar as mensagens aos respectivos destinatários.

No final de 2012, o Instituto Sophos revelou que o Brasil é o 5º no ranking global de envio de spam e parece que a situação tende a agravar-se. Isso pode acarretar vários problemas para uma instituição, principalmente vazamento de informações e ataque de hackers. Considera-se, então, um fator importante para ser analisado no modelo proposto. A quantidade inicial será multiplicada por 2, sobrecarregando o processamento do servidor e gerando uma grande fila neste processo de verificação. A Tabela 3 mostra como ficam os fatores e seus níveis.

Tabela 3 – Fatores escolhidos para o experimento.

Fator	Descrição	Nível atual (1)	Nível proposto (2)
A	Tempo de chegada dos <i>e-mails</i>	LOGN(1.92, 2.06);	LOGN(1.92, 2.06)*0,50;
B	Mensagens consideradas spam	LOGN(0.575, 0.361);	LOGN(0.575, 0.361) * 2;

Fonte: Próprio autor.

Devido ao acompanhamento dos experimentos, tem-se a possibilidade de analisar o tamanho das filas dos processos de verificação de spam e de *e-mail* e os que consomem mais processamento no servidor. Outro fator que pode ser analisado ainda é a utilização dos recursos do *software* responsável pelo anti-spam, antivírus e entrega dos *e-mails*. Um dos fatores mais importantes para análise é o *lead time*. Os cenários de simulação estão descritos no Quadro 1.

Quadro 1 – Cenários dos experimentos simulados.

Cenários	Fatores	
	Tempo entre a chegada dos e-mails	Mensagens consideradas spam
1	Nível 1	Nível 1
2	Nível 1	Nível 2
3	Nível 2	Nível 1
4	Nível 2	Nível 2

Fonte: Próprio autor.

A Tabela 4 apresenta resultados dos experimentos simulados no *software* Ururau. O cenário 1 apresenta os fatores que representam a situação atual do servidor. Apesar do serviço de entrega estar sendo consumido em 97,16%, a fila para acessar o mesmo serviço está com aproximadamente 9 *e-mails*. Isto demonstra que não há sobrecarga no servidor.

Tabela 4 – Resultados dos experimentos simulados.

Cenário	Variáveis						
	Número na fila da verificação de spam (un)	Utilização do recurso anti-spam (%)	Número na fila da verificação de vírus (un)	Utilização do recurso antivírus (%)	Número na fila de entrega (un)	Utilização do recurso entrega (%)	Lead time (seg)
1	0,05	30,51	0,00	13,10	9,45	97,16	24,62
2	0,47	60,40	0,00	12,93	8,73	96,40	24,27
3	0,52	59,67	0,00	25,73	33,46	98,99	41,41
4	12,06	97,20	0,00	21,01	23,94	98,30	42,47

Fonte: Próprio autor.

No cenário 2, cuja ideia foi dobrar a quantidade de spam a ser processada, percebe-se que todos os recursos estão em um nível bom de aceitação. Aumentou-se também o processamento do recurso do *software* anti-spam, e o *lead time* melhorou um pouco mais, permanecendo os dados estáveis.

Já nos cenários 3 e 4, pode-se perceber uma grande alteração na massa dos dados. Nos dois casos, dobra-se a quantidade de mensagens que chegam ao servidor. Note o elevado número de operações de utilização do recurso de entrega de *e-mail* para 98,99%, no cenário 3, e 98,30%, no cenário 4. Ocorre também uma elevada alteração no tamanho da fila do processo de entrega, que ficou com aproximadamente 33, no cenário 3, e 24, no cenário 4. E, por fim, percebe-se um *lead time* maior, de 41,41 segundos, no cenário 3, e de 42,47 segundos, no cenário 4.

A literatura disponível apresenta diversos trabalhos com aplicações de simuladores para avaliar desempenho de sistemas com dados de servidores. Com estes simuladores, é possível modelar e simular uma variedade de modelos de sistemas de dados, por meio de vários serviços oferecidos e também suas interações. Em recente trabalho, Dogan *et. al.*, 2014 apresentam uma aplicação semelhante ao trabalho aqui tratado.

Por fim, com intenção de mostrar a potencialidade do *software* Ururau, as Figuras 6 e 7 apresentam, de forma comparativa, os resultados obtidos nos experimentos simulados em relação ao Arena. Observe a equivalência entre os valores. Note que não há diferença estatística significativa ($p > 0,05$) entre eles. É importante ressaltar que o *software* Ururau ainda está em fase embrionária enquanto o Arena já é utilizado há décadas. No entanto, para pequenos sistemas, o *software* livre já pode ser utilizado com segurança.

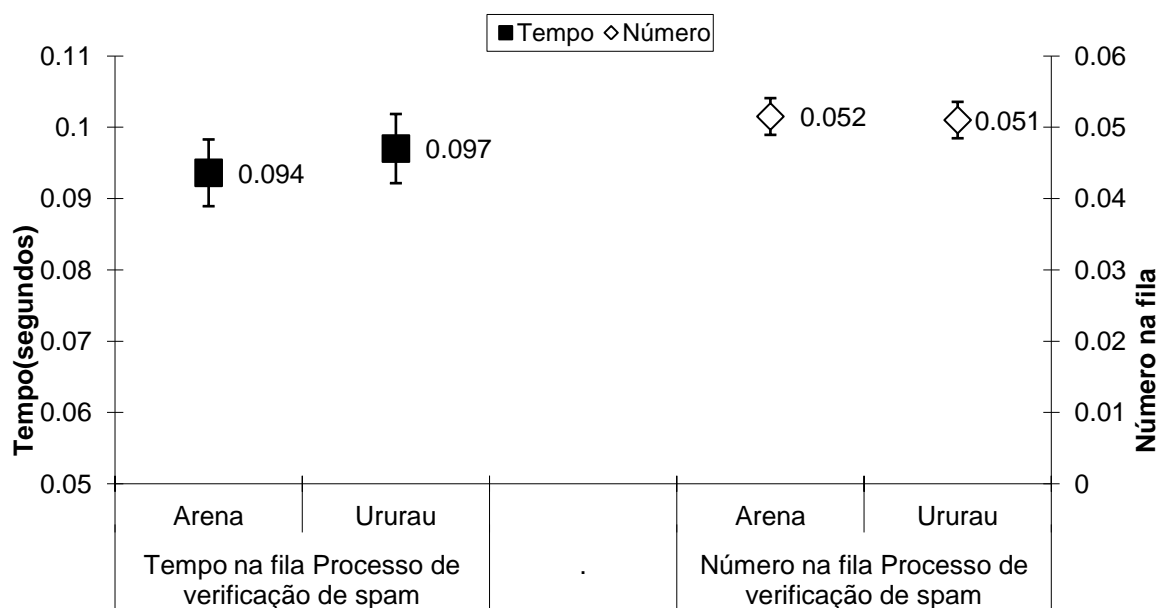


Figura 6 – Comparação dos dados em Ururau e Arena referentes à fila do processo de verificação de spam.

Fonte: Próprio autor

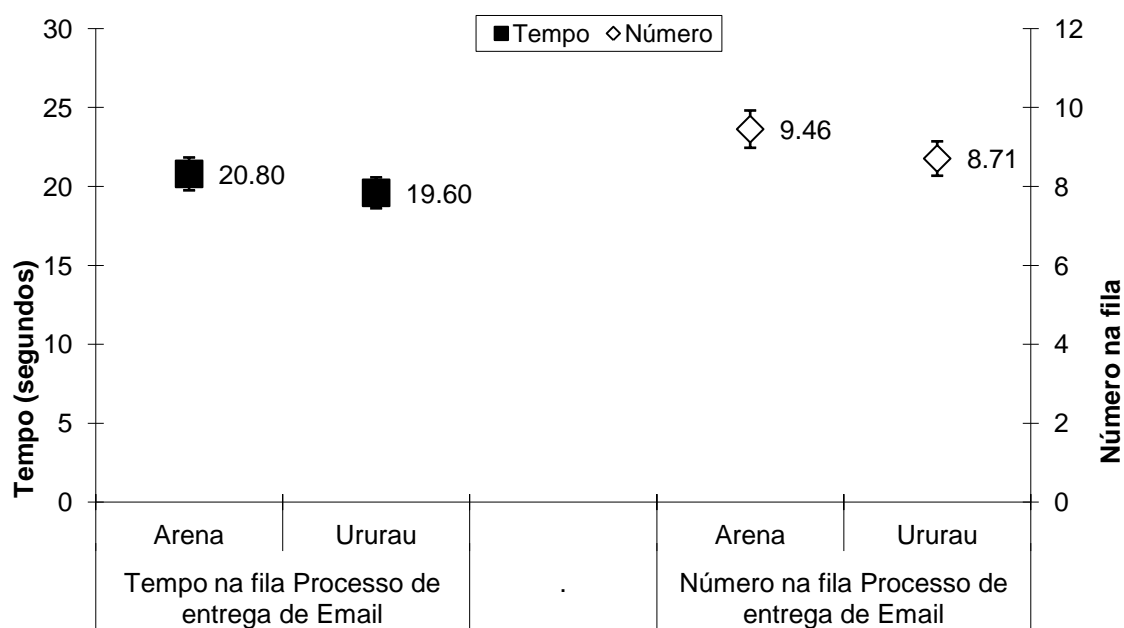


Figura 7 – Comparação dos dados em Ururau e Arena referentes à fila do processo de entrega de e-mail.

Fonte: Próprio autor.

5. Conclusões

O modelo de simulação a eventos discretos do servidor de e-mail possibilitou analisar o processamento de envio e recebimento de e-mail, levando em consideração o período do *lead time* dos mesmos. A análise considerou todo o processamento do servidor, desde a chegada de e-mails até o

envio para os destinatários finais. Com estas informações, os gestores da empresa podem avaliar a performance do sistema e decidir a melhor forma de alocação dos recursos de *hardware* e *software*.

Os resultados demonstram que o servidor atual está atendendo à demanda de *e-mails* existentes. No entanto, com o constante aumento da quantidade de *e-mails* e, ainda, a chegada de novos usuários, essa alta demanda pode exigir um processamento muito elevado. Como resultado, pode haver a formação de filas maiores e atraso na entrega de *e-mails*. Para os gestores da empresa, é importante diagnosticar previamente, pois pode haver a necessidade de um aumento de capacidade de *hardware/software* no servidor.

Vale destacar ainda que as simulações foram realizadas utilizando o *software* livre de simulação Ururau. Os resultados mostraram que o *software* obteve resultados equivalentes aos de outro *software* comercial tradicionalmente utilizado, sem diferença estatística significativa entre eles. Uma questão que se abre aqui, então, é a possibilidade de realização deste tipo de análise com *softwares* nacionais e livres de custos. Por se tratar de um *software* de código aberto e livre de custos, existe a opção para que pessoas interessadas possam compreender melhor a aplicação prática e a concepção interna da estrutura computacional do código fonte de um *software* para desenvolvimento de modelos de simulação a eventos discretos. Assim, pode-se expandir a área de simulação para além das restrições dos *softwares* comerciais e códigos proprietários.

Agradecimentos

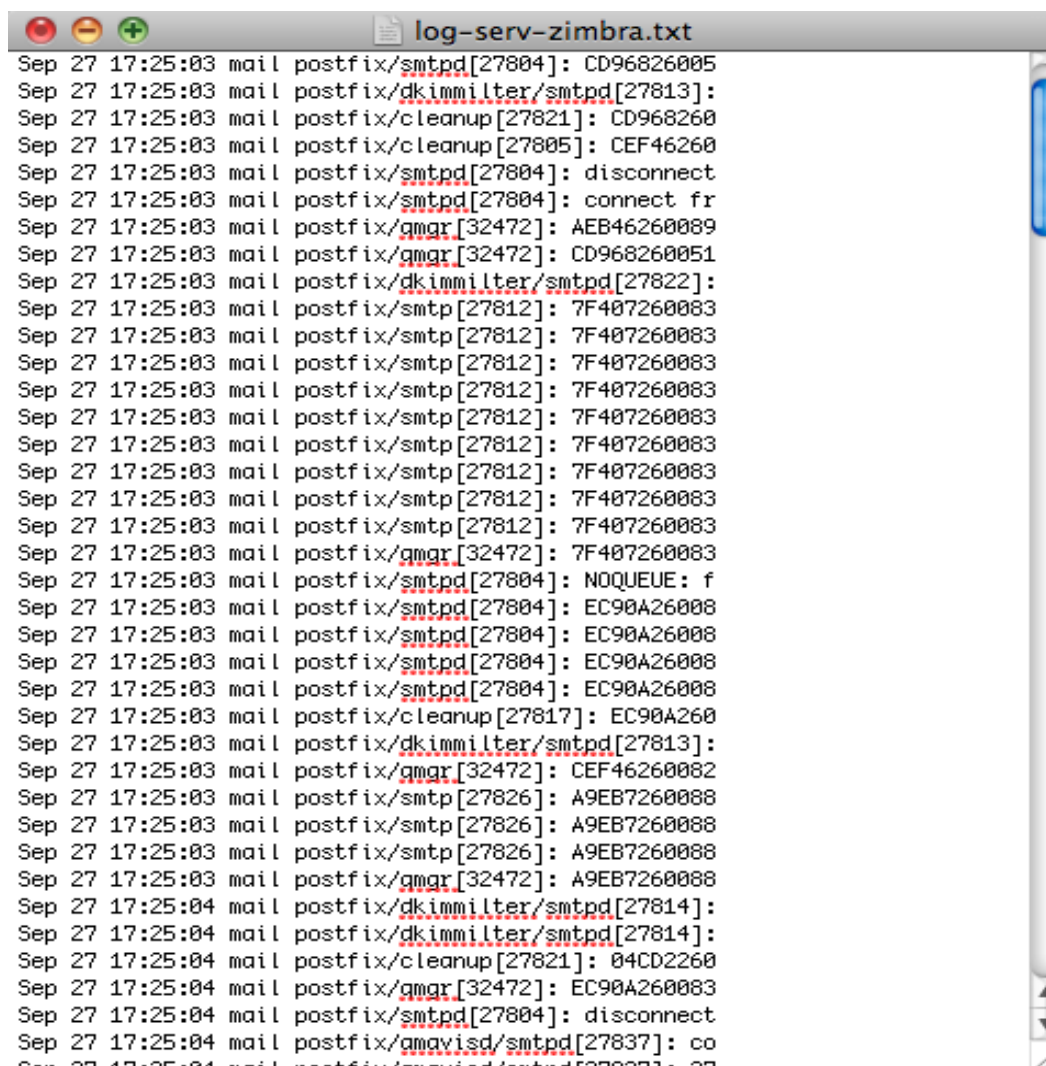
Os autores gostariam de agradecer Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) pelo suporte financeiro para esta pesquisa.

Referências

- BANKS, J.; CARSON, J. S.; NELSON, B. L.; NICOL, D. M. *Discrete-event system simulation*. 5nd ed. New Jersey: Prentice Hall, 2010.
- DOĞAN, A.; ATANAK, M. M.; TANDOĞAN, S.; ALTUĞ, R. O.; ŞENEL, H. G. *DGridSim: a multi-model discrete-event simulator for real-time data grid systems*. *Simulation*, v. 90 (11), p. 1209-1230, 2014.
- KING, D. H. AND HARRISON, H. S. *Open-Source Simulation Software "JAAMSIM"*. In: Proceedings of the WINTER SIMULATION CONFERENCE, 2013.
- LAW AM. *Simulation modeling and analysis*. 4th ed. New York: McGraw-Hill, 2007.
- MIRANDA, R.C., MONTEVECHI, J.A.B, PENTEADO, K. M., AGUIAR, M.S., PINHO, A. F. **Análise de uma unidade de processamento de roupas de um hospital através da simulação a eventos discretos**. *Produto & Produção*, vol. 13, n. 3, p. 07-24, out. 2012.
- MONTEVECHI, J.A.B.; LEAL, F.; DE PINHO, A.F.; DA SILVA COSTA, R.F.; DE OLIVEIRA, M.L.M.; DA SILVA, A.L.F. (2010) *Conceptual modeling in simulation projects by mean adapted IDEF: An application in a Brazilian tech company*. In: Winter Simulation Conference, 2010, Baltimore. Proceedings of the 2010 WINTER SIMULATION CONFERENCE, 2010. 1624-1635.
- MONTGOMERY, D.C. *Design and analysis of experiments*, 7th Edition, John Wiley and Sons, 2009.
- PEIXOTO, T. A.; RANGEL, J. J. A.; MATIAS, I. O.; MONTEVECHI, J. A. B.; MIRANDA, R. C. Ururau - **Um Ambiente para Desenvolvimento de Modelos de Simulação a Eventos Discretos**. *Pesquisa Operacional para o Desenvolvimento*, v. 5, p. 373-405, 2013.
- PEREIRA, T. F.; MONTEVECHI, J. A. B.; MIRANDA, R. C.; FRIEND, J. D. *Integrating soft systems methodology to aid simulation conceptual modeling*. *International Transactions in Operational Research*, v. 1, p. 1-20, 2014.

- RANGEL, J. J. A. E CORDEIRO, A. C. A. *Free and Open-Source Software for Sustainable Analysis in the Logistics Systems Design*. Journal of Simulation, v. 9, p.27-42, 2015.
- RANGEL J. J. A., OLIVEIRA G. L, PEIXOTO T. A, CARDOSO L. D., MATIAS I.O. and SHIMODA E.. *Simulation Model in a Free and Open-Source Software for Carbon Monoxide Emissions Analysis*. In: Proceedings of the WINTER SIMULATION CONFERENCE R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, 2013.
- ROSSETI, M.D. *Java Simulation Library (JSL): an open-source object-oriented library for discrete-event simulation in Java*. In: Int. J. Simulation and Process Modelling, Vol. 4, No. 1, 69-87, 2008.
- SARGENT, R. G. *Verifications and Validation of Simulation Models*. Journal of Simulation, vol. 7, no. 1, p. 12–24, 2013.
- SILVA, M. G. D.; RANGEL, J. J. A.; SILVA, D. V. C.; PEIXOTO, T. A.; MATIAS, I. O.. **Decisão com Redes Neurais Artificiais em Modelos de Simulação a Eventos Discretos**. Pesquisa Operacional para o Desenvolvimento, v.6, n.2, p. 299-317, 2014.
- SWAIN, J. J.. *Discrete Event Simulation Software: New Frontiers in Simulation*, OR/MS Today - INFORMS, v. 34, n. 5, p.32-43, 2007.
- THOMAS, A. W.; MICHAEL, O. R.; RICHARD, F.; MICHAEL, P. H.. *A federated simulation method for multi-modal transportation systems: combining a discrete event-based logistics simulator and a discrete time-step-based traffic microsimulator*. Simulation, v. 91 (2), p. 148-163, 2015.

Anexo A: Demonstração do arquivo de Log. (Fonte: ESSys soluções LTDA).



```
log-serv-zimbra.txt
Sep 27 17:25:03 mail postfix/smtpd[27804]: CD96826005
Sep 27 17:25:03 mail postfix/dkimilter/smtpd[27813]:
Sep 27 17:25:03 mail postfix/cleanup[27821]: CD968260
Sep 27 17:25:03 mail postfix/cleanup[27805]: CEF46260
Sep 27 17:25:03 mail postfix/smtpd[27804]: disconnect
Sep 27 17:25:03 mail postfix/smtpd[27804]: connect fr
Sep 27 17:25:03 mail postfix/qmgr[32472]: AEB46260089
Sep 27 17:25:03 mail postfix/qmgr[32472]: CD968260051
Sep 27 17:25:03 mail postfix/dkimilter/smtpd[27822]:
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/smtp[27812]: 7F407260083
Sep 27 17:25:03 mail postfix/qmgr[32472]: 7F407260083
Sep 27 17:25:03 mail postfix/smtpd[27804]: NOQUEUE: f
Sep 27 17:25:03 mail postfix/smtpd[27804]: EC90A26008
Sep 27 17:25:03 mail postfix/smtpd[27804]: EC90A26008
Sep 27 17:25:03 mail postfix/smtpd[27804]: EC90A26008
Sep 27 17:25:03 mail postfix/smtpd[27804]: EC90A26008
Sep 27 17:25:03 mail postfix/cleanup[27817]: EC90A260
Sep 27 17:25:03 mail postfix/dkimilter/smtpd[27813]:
Sep 27 17:25:03 mail postfix/qmgr[32472]: CEF46260082
Sep 27 17:25:03 mail postfix/smtp[27826]: A9EB7260088
Sep 27 17:25:03 mail postfix/smtp[27826]: A9EB7260088
Sep 27 17:25:03 mail postfix/smtp[27826]: A9EB7260088
Sep 27 17:25:03 mail postfix/qmgr[32472]: A9EB7260088
Sep 27 17:25:04 mail postfix/dkimilter/smtpd[27814]:
Sep 27 17:25:04 mail postfix/dkimilter/smtpd[27814]:
Sep 27 17:25:04 mail postfix/cleanup[27821]: 04CD2260
Sep 27 17:25:04 mail postfix/qmgr[32472]: EC90A260083
Sep 27 17:25:04 mail postfix/smtpd[27804]: disconnect
Sep 27 17:25:04 mail postfix/amavisd/smtpd[27837]: co
Sep 27 17:25:04 mail postfix/amavisd/smtpd[27837]: 27
```