Evento: XXII Jornada de Pesquisa

MODELAGEM MATEMÁTICA E SIMULAÇÃO NO CONTEXTO DOS MOTORES DE EXECUÇÃO DAS PLATAFORMAS DE INTEGRAÇÃO: UM ESTUDO EXPLORATÓRIO.[1]

MATHEMATICAL MODELING AULAS QUILO ND SIMULATION IN THE CONTEXT OF RUNTIME ENGINES OF INTEGRATION PLATFORMS : AN EXPLORATORY STUDY

Igor Gamste Haugg[2], Rafael Z. Frantz[3], Fabricia Roos-Frantz[4], Sandro Sawicki[5]

[1] Pesquisa desenvolvida no Mestrado de Modelagem Matemática, pertencente ao Grupo de Pesquisa em Computação Aplicada (GCA)
[2] Bolsista PROSUP/CAPES, Aluno do Curso de Mestrado em Modelagem Matemática da Unijuí
[3] Professor Doutor do Departamento de Ciências Exatas e Engenharias
[4] Professora Doutora do Departamento de Ciências Exatas e Engenharias
[5] Professor Doutor do Departamento de Ciências Exatas e Engenharias

Resumo

Atualmente a utilização de aplicações é imprescindível para dar suporte aos processos de negócio das empresas. Normalmente essas aplicações foram desenvolvidas por diferentes empresas, seguindo distintas tecnologias e não foram projetadas para trabalhar de forma colaborativa. Integrar aplicações é fazer com que diferentes aplicações que não foram desenvolvidas com o objetivo de integração, possam colaborar e dar suporte a novos processos de negócio. Soluções de integração podem ser desenvolvidas através de diferentes plataformas, que fornecem componentes para modelar e implementar as soluções, por exemplo: linguagem de domínio específico (DSL), kit de ferramentas de desenvolvimento, ferramenta de monitoramento e motor de execução. Dentre as características disponibilizadas pelas plataformas, neste trabalho será analisado o motor de execução, como a arquitetura destes motores de execução são organizados em uma fila, os mesmos podem ser representados com teoria das filas. Para a realização de modelagem de sistemas com teoria das filas, existem duas abordagens, matemática e simulação. A abordagem matemática é realizada através da construção de modelos matemáticos, onde busca-se uma fórmula matemática para representar o funcionamento dos motores de execução. A simulação permite representar através de modelos de simulação o funcionamento dos motores de execução. Neste trabalho será apresentado um estudo sobre teoria das filas, com o objetivo de encontrar qual abordagem melhor se aproxima ao funcionamento do motor de execução das plataformas de integração de aplicações empresariais. A partir dos estudos realizados, foi possível identificar que a abordagem matemática não pode ser aplicada para os motores de execução, pois a mesma exige que as entradas do sistema sejam constantes. Portanto, a simulação é a abordagem mais adequada para modelar o motor de execução.

Palavras-chave: Integração de Aplicações Empresariais, Teoria das Filas, Motor de Execução.

Evento: XXII Jornada de Pesquisa

Abstract

Nowadays the use of applications is indispensable to give support to business processes of companies. Normally these applications were developed by different companies, following distinct technologies and not designed to work in a collaborative way. In most cases, the business processes involve data e functionalities present in distinct applications, which requires the collaboration of these applications. Integrate applications is to make different applications that were not developed with the purpose of integration, may collaborate and give support to new business processes. Solutions of integration can be developed through different platforms, which provides components to model and implement the solutions, for example: domain specific language (DSL), development toolkit, monitoring tool and runtime engine. Among the characteristics offered by the platforms, in this work the runtime engine will be analyzed, as the architecture of these execution engines are organized in a queue, they can be represented with queue theory. For the modeling of systems with queuing theory, there are two approaches, mathematics and simulation. The mathematical approach is performed through the construction of mathematical models, where a mathematical formula is used to represent the operation of the runtime engines. The simulation allows to represent through simulation models the operation of the runtime engines. This paper will present a study on queuing theory, with the objective of finding which approach best approaches the running of the runtime engine of enterprise application integration platforms. Following the results, was possible to identify that the mathematical approach cannot be applied to the execution engines, since it requires that the system inputs be constant. Therefore, simulation is the most appropriate approach to modeling the runtime engine.

Keywords: Enterprise Application Integration, Queue Theory, Runtime Engine.

INTRODUÇÃO

A utilização de aplicações é imprescindível para dar suporte aos processos de negócio das empresas. Normalmente, as empresas possuem em seu ecossistema de software (MESSERSCHMITT, 2005) diversas aplicações desenvolvidas internamente na empresa ou adqueridas de terceiros. Desta forma, o ecossistema de software torna-se heterogêneo, com aplicações que possivelmente foram desenvolvidas utilizando diferentes linguagens de programação, diferentes sistemas operacionais, diferentes modelos de dados, e, geralmente, não foram projetadas para trabalhar de forma colaborativa (FRANTZ; CORCHUELO; ROOS-FRANTZ, 2016). Na sua maioria, os processos de negócio envolvem dados e funcionalidades presentes em aplicações distintas, o que por sua vez requer a colaboração destas aplicações.

A área de integração de aplicações empresariais (EAI) busca oferecer metodologias, técnicas e ferramentas para fazer com que diferentes aplicações, que não foram desenvolvidas com o propósito de trabalhar em conjunto, possam colaborar (HOHPE; WOOLF, 2004). EAI é uma combinação de tecnologias com a função de dar suporte aos processos de negócio, por meio do desenvolvimento de soluções de integração. Uma solução de integração orquestra de forma exógena um conjunto de aplicações, geralmente sem que essas percebam que estão sendo integradas, e sem introduzir dependência das aplicações com a solução, permitindo o reuso de

dados e funcionalidades presentes em cada uma (LINTHICUM, 2000).

Existem diversas plataformas de integração disponíveis para o desenvolvimento de soluções de integração. Dentre as plataformas open-source que representam o estado da arte, estão Camel (IBSEN, 2010), Spring *Integration* (FISHER, 2012), Mule (DOSSOT; D'EMIC; ROMERO, 2012), WSO2 ESB (INDRASIRI, 2016), ServiceMix (KONSEK, 2013) e Guaraná (FRANTZ; CORCHUELO; ROOS-FRANTZ, 2016). Essas plataformas suportam os padrões de integração documentados por HOHPE e WOOLF (2004), que se tornaram uma referência na comunidade EAI e vêm inspirando diversas plataformas de integração.

Usualmente, essas plataformas de integração fornecem uma linguagem de domínio específico, um kit de ferramentas de desenvolvimento, um ambiente para realização de testes, uma ferramenta de monitoramento e um motor de execução. A linguagem de domínio específico está voltada para a elaboração de modelos conceituais para a solução de integração, com um nível de abstração próximo ao domínio do problema. O kit de desenvolvimento é um conjunto de ferramentas de software que permite a implementação da solução, ou seja, a transformação do modelo conceitual à código executável. O ambiente para realização de testes permite provar partes individuais ou toda a solução de integração, com o objetivo de identificar e eliminar possíveis defeitos na implementação da mesma. A ferramenta de monitoramento é utilizada para acompanhar, em tempo de execução, o funcionamento da solução de integração e detectar erros que possam ocorrer durante o processamento das mensagens. O motor de execução proporciona todo o suporte necessário à execução dessas soluções de integração. Por isso, o desempenho dos motores de execução está diretamente relacionado com o desempenho das soluções de integração, ou seja, quanto melhor for o desempenho do motor de execução, melhor será o desempenho da solução de integração.

A arquitetura dos motores de execução das plataformas open-source identificadas estão organizados em torno de uma fila FIFO, nessa fila são anotadas as tarefas da solução de integração que estão aptas a serem executadas, e a execução das mesmas é realizada por um conjunto de *threads* previamente configuradas no motor de execução. Os motores de execução destas plataformas estão implementados com a tecnologia Java (GOSLING, 2000), a qual fornece uma API (Application Programming Interface) denominada *Executor* que possibilita a criação e alocação das *threads* em um *pool* de *threads*. Porém, o dimensionamento deste *pool* de *threads* é de responsabilidade dos engenheiros de software.

Teoria das Filas (ADAN; RESING, 2002)é uma área do conhecimento que estuda o melhor dimensionamento dos recursos de sistemas baseados em filas e permite criar modelos de simulação ou modelos matemáticos para estudar o comportamento desses sistemas. Esse artigo apresenta um estudo que busca encontrar qual é a melhor abordagem para modelar o funcionamento dos motores de execução das plataformas de integração. Isso é necessário, pois a partir da escolha da melhor abordagem para o desenvolvimento dos modelos será possível construir modelos que permitam compreender o funcionamento de diferentes soluções de integração.

O restante deste artigo tem sua estrutura organizada como segue. A Seção 2 apresenta a

metodologia utilizada para o desenvolvimento deste trabalho. A Seção 3 apresenta um breve referencial teórico sobre teoria das filas. A seção 3 exibe-se o modelo conceitual dos motores de execução e em seguida as conclusões deste trabalho.

METODOLOGIA

A realização desta pesquisa está dividida em quatro etapas. A primeira etapa consiste em uma revisão da literatura sobre teoria das filas, a partir de bases bibliográficas de artigos científicos e livros especializados. A segunda etapa consiste no estudo do funcionamento dos motores de execução das plataformas de integração de aplicações. Por fim, a terceira etapa centra-se na elaboração de um modelo conceitual, para representar o funcionamento do motor de execução e descobrir qual a abordagem de teoria das filas pode representar da melhor maneira o motor de execução.

REFERENCIAL TEÓRICO

No estudo de modelagem de sistemas, é comum a existência de problemas de dimensionamento onde a solução é complexa. O cenário pode ser uma fábrica, o trânsito de uma cidade, uma aplicação sendo utilizada em um computador, etc. Normalmente o objetivo é dimensionar de forma precisa, a quantidade correta de equipamentos ou recursos, e também identificar qual é o melhor fluxo dentro do sistema, ou seja, busca-se a otimização do sistema estudado (PRADO, 2014).

Em sistemas, quando há um grande acúmulo de processos esperando para serem executadas na fila, podem surgir gargalos de desempenho. Nestes casos, o ideal seria dimensionar os sistemas para que não ocorram filas. Porém, como nem sempre é possível trabalhar com a inexistência de filas, é utilizado a teoria das filas, para que, através de fórmulas matemáticas ou simulação computacional, minimizar as implicações causadas pelas filas, e assim tornar o sistema mais eficiente (PRADO, 2014).

Para melhor entendimento do funcionamento de um sistema de filas é necessário conhecer as suas principais características: clientes, processo de chegada, processo de atendimento, número de atendentes, fila e disciplina da fila (HILLIER; LIEBERMAN, 2013). Os clientes são provenientes de uma população, quando a população é muito grande, ela é definida como infinita (PRADO, 2014). O processo de chegada é definido pelo comportamento das chegadas de clientes no sistema, podendo ser: determinístico ou estocástico. Quando o número de chegadas e o momento onde os clientes chegam ao sistema é conhecido, o comportamento é determinístico. Quando as chegadas apresentam um comportamento aleatório, o comportamento é estocástico. Este comportamento pode ser representado através de uma distribuição de probabilidade (FOGLIATTI; MATTOS, 2007). O processo de atendimento segue o mesmo conceito do processo de chegada, pode ser determinístico ou estocástico. O número de servidores representa a quantidade de servidores disponíveis para realizar atendimento aos clientes que estão aguardando na fila. A fila é onde os clientes aguardam até que algum atendente esteja disponível para realizar o atendimento (PRADO, 2014). A disciplina da fila define a ordem pela qual os clientes são selecionados da fila para o atendimento. Esta disciplina normalmente segue o modelo *First In First Out* (FIFO), ou

seja, o atendimento é realizado de acordo com a ordem de chegada, porém existem outras disciplinas, como por exemplo, *Last In First Out* (LIFO), onde o atendimento é realizado de acordo com o último cliente a chegar, também pode ser através de uma disciplina aleatória, ou então através de uma disciplina com prioridades, a qual necessita de uma política de prioridades previamente especificada (PRADO, 2014).

A abordagem matemática de sistemas de filas exige estabilidade no fluxo de chegada e no processo de atendimento. Para isto, os valores da taxa média de chegada e taxa média de atendimento devem se manter constantes ao longo do tempo (PRADO, 2014). Além disso, é necessário que a taxa média de atendimento seja maior que a taxa média de chegada. Isso significa que a taxa de utilização dos atendentes deve ser menor que 1, ou seja, os atendentes devem ser capazes de atender o fluxo de chegada de clientes. Um exemplo de um sistema que não é estável, é de um funcionamento de um banco, onde poderíamos verificar que o fluxo de chegada de clientes possui variação durante o dia da seguinte forma: no período entre as 10:00 até 12:00 a chegada de clientes possui um fluxo médio, durante o período de 12:00 até 14:00 a chegada de clientes possui um fluxo alto e no período de 14:00 até 16:00 horas, o fluxo se mantém médio. Desta forma, é possível identificar que não existe estabilidade para o ritmo de chegada durante todo o período de atendimento, portanto não podemos analisar seu funcionamento pela teoria das filas, a única forma de utilizar a abordagem matemática para este tipo de sistema, seria dividir o período global em períodos parciais. Porém, isso torna ainda mais complexa a abordagem pela teoria das filas (PRADO, 2014). Desta forma, quando um sistema de filas não é estável, ele deve ser estudado por meio da simulação (PRADO, 2014).

A simulação pode ser definida como uma técnica de solução de um problema pela análise de um modelo que descreve o comportamento do sistema. Atualmente, é possível com o auxílio de ferramentas a construção de modelos de simulação capazes de representar todo o funcionamento de um sistema, por exemplo, um sistema de banco, de uma fábrica (PRADO, 2014). Um modelo de simulação permite a representação de diversas interações entre as partes que compõem um sistema. Portanto, pode ser definido como uma abstração da realidade, que visa se aproximar do verdadeiro comportamento do sistema. Ao modelar um sistema, o objetivo é capturar o que realmente é importante no sistema, sendo assim, o modelo deve ter mais simples que o sistema real (CHWIF; MEDINA,2006).

Os sistemas reais, por possuir uma natureza dinâmica (que muda de estado ao longo do tempo) e aleatória, geralmente possuem uma maior complexidade. Desta forma, um modelo de simulação consegue capturar as características deste sistema com maior fidelidade, buscando repetir o comportamento do sistema real em um computador. As principais razões para a utilização de simulação são: quando o sistema real ainda não existe ou quando há inviabilidade de interferência no sistema real. No primeiro caso, a simulação é utilizada para planejar o futuro sistema. Por exemplo, um novo hospital, uma nova fábrica ou um novo ambiente de suporte a negócios na Internet. No segundo caso, ocorre quando a alteração direta no sistema real não é possível, por motivo de inviabilidade financeira, onde alterar o sistema real sem saber as consequências desta alteração, pode trazer prejuízos para a empresa (PRADO, 2014).

RESULTADOS

O motor de execução é o responsável pela execução de uma solução de integração. Este motor de execução possui um elemento central chamado *Scheduler*, o qual é o núcleo do motor, sendo considerado a parte mais importante. O *Scheduler* é responsável pela coordenação de todas as atividades presentes em uma instância executada. Além disso, também é o responsável por executar as tarefas. Em execução, o *Scheduler* possui uma fila de tarefas, um conjunto de threads e três monitores (FRANTZ; REINA QUINTERO; CORCHUELO, 2011).

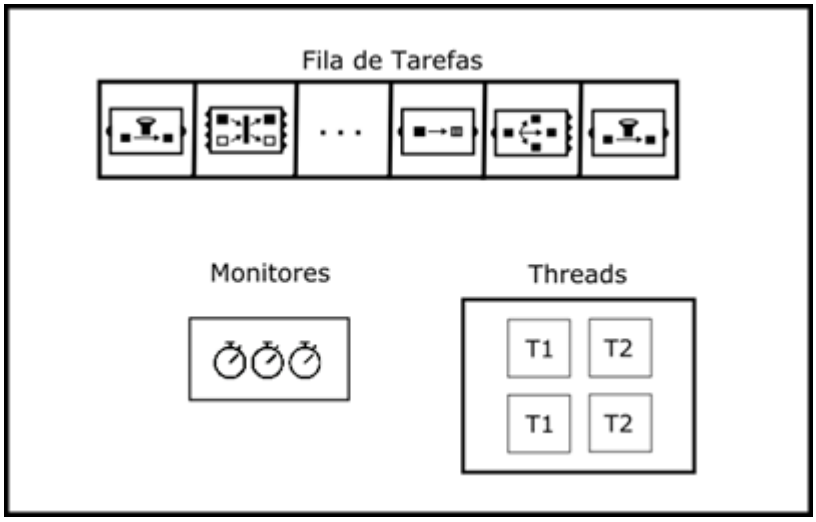

Figura 1. Elementos principais do *Scheduler*.

A fila de tarefas tem como função armazenar as unidades de trabalho a serem processadas. As unidades de trabalho possuem uma referência a uma tarefa e também uma anotação de quando esta tarefa pode ser executada. Normalmente, a tarefa possui um tempo de execução agendado para a hora atual, ou seja, a tarefa pode ser executada o mais rápido possível. Porém, é possível que seja definido um tempo específico para executar esta tarefa, desta forma, a tarefa correspondente será atrasada até que esse tempo tenha decorrido. Definir um tempo específico para executar uma tarefa é útil para implementar tarefas que precisam ser executadas periodicamente. Essa fila segue a disciplina onde o primeiro a entrar é o primeiro a ser executado (FIFO) (FRANTZ; REINA QUINTERO; CORCHUELO, 2011).

Os monitores fornecem informações estatísticas sobre o uso de memória, CPU e sobre a fila de tarefas. O monitor de memória registra informações sobre o uso da memória do sistema. O monitor das *threads* registra o tempo de execução do sistema e o tempo utilizado pelas *threads* na execução das fila de tarefas. O monitor da fila registra o tamanho da fila e o número total de

tarefas que foram processadas. Os monitores são executados por *threads* específicas e independentes, que tornam-se ativas em intervalos regulares em períodos pré-definidos. As threads dos monitores executam a operação de registrar e armazenar as informações num arquivo e voltam a ficar inativas até o próximo período de execução (FRANTZ; REINA QUINTERO; CORCHUELO, 2011).

Para configurar o funcionamento do *Scheduler*, é utilizado um arquivo no formato XML, no qual é determinado o número de atendentes (*threads*), o arquivo no qual os monitores escrevem as estatísticas geradas, a periodicidade do monitoramento e o arquivo de log para reportar advertências e exceções que possam ocorrer durante a execução. O *Scheduler* é iniciado a partir do carregamento e análise do arquivo de configuração, da inicialização do sistema de log e da criação da fila de tarefas. Desta forma, o motor de execução é inicializado no momento que o engenheiro de software decidir. Quando o motor de execução é iniciado, os monitores e *threads* também são inicializados (FRANTZ; REINA QUINTERO; CORCHUELO, 2011).

A partir da descrição do *Scheduler*, é possível na Figura 2 visualizarmos o funcionamento do motor de execução das plataformas de integração. Nela podemos identificar que a fila de tarefas recebe de entrada diversas *work units*, as quais são provenientes das tarefas de uma solução de integração, e seguem uma distribuição de entrada aleatória, ou seja, novas entradas podem ocorrer a qualquer momento. Neste exemplo, a fila de tarefas possui disponível três *threads* que podem executar as *work units*. Assim, o processo segue com novas entradas de *work units* sendo atendidas pelas *threads* (FRANTZ; REINA QUINTERO; CORCHUELO, 2011).

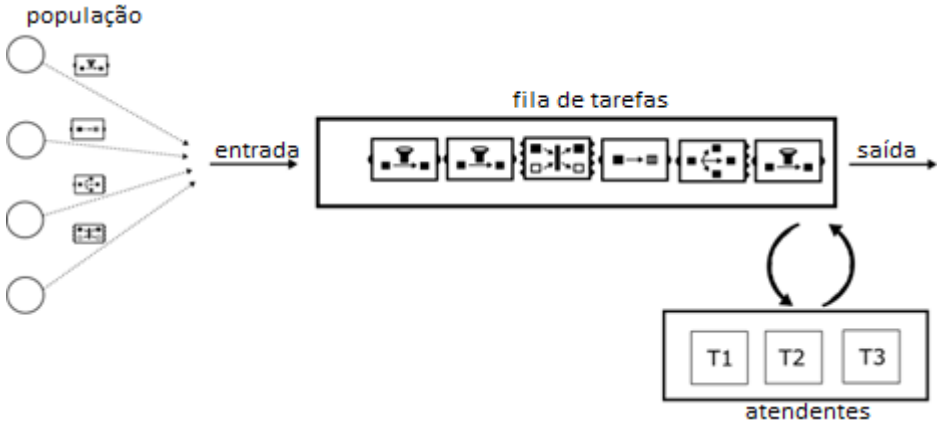

Figura 2. Modelo conceitual do motor de execução das plataformas de integração.

CONCLUSÃO

Com a realização deste trabalho de pesquisa, foi possível identificar que os motores de execução das plataformas de integração não podem ser formalmente descritos através de teoria das filas com abordagem matemática. Apesar de o motor de execução utilizar um sistema de filas, onde as tarefas que precisam ser executadas são armazenadas em uma fila, e os atendentes executam

essas tarefas através de uma disciplina, neste caso não se aplica a teoria das filas com abordagem matemática, pois o sistema não é estável, ou seja, as entradas do sistema não seguem um ritmo constante.

Neste caso, é necessário a utilização da simulação computacional, onde através de um programa de computador, será desenvolvido um modelo que represente o funcionamento do motor de execução.

REFERÊNCIAS

MESSERSCHMITT, David G. et al. **Software ecosystem: understanding an indispensable technology and industry**. MIT Press Books, v. 1, 2005.

FRANTZ, Rafael Z.; CORCHUELO, Rafael; ROOS-FRANTZ, Fabricia. **On the design of a maintainable software development kit to implement integration solutions**. Journal of Systems and Software, v. 111, p. 89-104, 2016.

HOHPE, Gregor; WOOLF, Bobby. **Enterprise integration patterns: Designing, building, and deploying messaging solutions**. Addison-Wesley Professional, 2004.

LINTHICUM, David S. **Enterprise application integration**. Addison-Wesley Professional, 2000.

IBSEN, Claus; ANSTEY, Jonathan. **Camel in action**. Manning Publications Co., 2010.

FISHER, Mark et al. **Spring integration in action**. 2012.

DOSSOT, David; D'EMIC, John; ROMERO, Victor. **Mule in action**. Manning, 2014.

INDRASIRI, Kasun. **Introduction to WSO2 ESB**. In: Beginning WSO2 ESB. Apress, 2016. p. 1-16.

KONSEK, Henryk. **Instant Apache ServiceMix How-to**. Packt Publishing Ltd, 2013.

GOSLING, James. **The Java language specification**. Addison-Wesley Professional, 2000.

ADAN, Ivo; RESING, Jacques. **Queueing theory**. 2002.

PRADO, Darci Santos do. Teoria das Filas e da Simulação. **Belo Horizonte, MG: Editora de Desenvolvimento Gerencial**, v. 5, 2014.

HILLIER, Frederick S.; LIEBERMAN, Gerald J. **Introdução à pesquisa operacional**. McGraw Hill Brasil, 2013.

FOGLIATTI, Maria Cristina; MATTOS, Neli Maria Costa. **Teoria de filas**. Rio de Janeiro: Interciência, p. 1-290, 2007.

CHWIF, Leonardo; MEDINA, Afonso Celso. **Modelagem e simulação de eventos discretos**. Afonso C. Medina, 2006.

FRANTZ, Rafael Z.; REINA QUINTERO, Antonia M.; CORCHUELO, Rafael. **A domain-specific language to design enterprise application integration solutions**. International Journal of Cooperative Information Systems, v. 20, n. 02, p. 143-176, 2011.