
DESENVOLVIMENTO DE UM MODELO COMPUTACIONAL PARA SIMULA- ÇÃO DO COMPORTAMENTO DE UMA SOLUÇÃO DE INTEGRAÇÃO NA AD- MINISTRAÇÃO PÚBLICA DE HUELVA (ESPANHA) UTILIZANDO REDE DE PETRI TEMPORIZADA

FRANCIÉLI CRISTINA WELTER
UNIVERSIDADE REGIONAL DO NOROESTE DO
ESTADO DO RIO GRANDE DO SUL

DISSERTAÇÃO DE MESTRADO

ORIENTADOR:

DR. RAFAEL ZANCAN FRANTZ

COORIENTADOR:

DRA. FABRICIA CARNEIRO ROOS FRANTZ



Applied
Computing
Research Group

MARÇO, 2017

First published in March 2017 by
Applied Computing Research Group - GCA
Department of Exact Sciences and Engineering
Rua Lulu Ilgenfritz, 480 - São Geraldo
Ijuí, 98700-000, Brazil.

Copyright © MMXVII Applied Computing Research Group
<http://www.gca.unijui.edu.br>
gca@unijui.edu.br

In keeping with the traditional purpose of furthering science, education and research, it is the policy of the publisher, whenever possible, to permit non-commercial use and redistribution of the information contained in the documents whose copyright they own. You however are *not allowed* to take money for the distribution or use of these results except for a nominal charge for photocopying, sending copies, or whichever means you use redistribute them. The results in this document have been tested carefully, but they are not guaranteed for any particular purpose. The publisher or the holder of the copyright do not offer any warranties or representations, nor do they accept any liabilities with respect to them.

Universidade Regional do Noroeste do Estado do Rio Grande do Sul

A Comissão Examinadora, abaixo assinada, _____ a dissertação intitulada "Desenvolvimento de um Modelo Computacional para Simulação do Comportamento de uma Solução de Integração na Administração Pública de Huelva (Espanha) Utilizando Rede de Petri Temporizada: ", elaborada por Franciéli Cristina Welter, como requisito parcial para a obtenção do título de Mestre em Modelagem Matemática.

Dr. Rafael Zancan Frantz
UNIJUÍ
(Orientador)

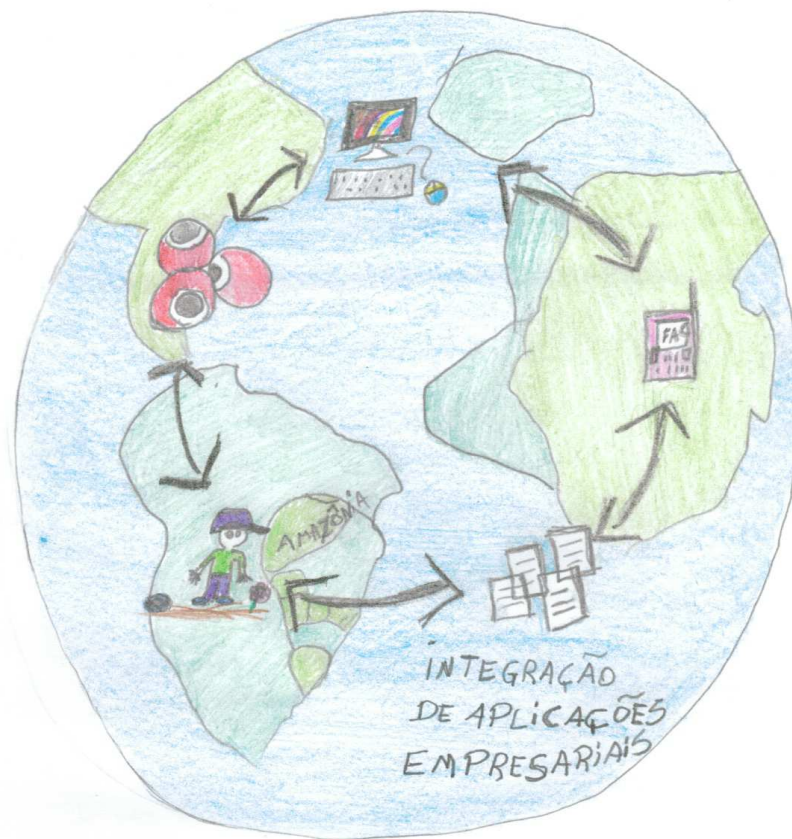
Dr. Vitor Manuel Basto Fernandes
Instituto Universitário de Lisboa
Portugal

Dr. José Antonio Gonzalez da Silva
UNIJUÍ

Dra. Fabricia Carneiro Roos Frantz
UNIJUÍ
(Co-orientador)

Dr. Sandro Sawicki
UNIJUÍ

Ijuí, ____ de _____ de _____.



Integração de Aplicações Empresariais por Henrique, 12 anos de idade.

Dedico este trabalho à minha família, que me apoiou sempre que precisei.

Conteúdo

Agradecimentos	vii
Resumo	ix
Abstract	xi
1 Introdução	1
1.1 Contexto da Pesquisa	1
1.2 Motivação	5
1.3 Objetivos	6
1.3.1 Geral	6
1.3.2 Específicos	6
1.4 Metodologia	6
1.5 Resumo das Contribuições	8
1.6 Estrutura dessa Dissertação	9
2 Revisão da Literatura	11
2.1 Integração de Aplicações Empresariais	11
2.1.1 Estilos de Integração	13
2.1.2 Topologias	17
2.1.3 Plataformas de Integração	21
2.2 Plataforma Guaraná	22
2.2.1 Linguagem de Domínio Específico	22
2.3 Simulação de Eventos Discretos	26
2.3.1 Sistema, Modelo e Simulação	26
2.3.2 Vantagens e Desvantagens da Simulação	28
2.3.3 Classificação de Sistemas de Simulação	30
2.4 Distribuições de Probabilidade	33

2.4.1	Distribuição Exponencial	34
2.4.2	Distribuição Uniforme	35
2.5	Redes de Petri	35
2.5.1	Redes de Petri Coloridas	42
2.5.2	Redes de Petri Temporizadas	43
2.6	Trabalhos Relacionados	47
2.7	Resumo do Capítulo	52
3	Modelagem	53
3.1	Caso de Estudo	53
3.1.1	Ecosistema de Software	54
3.1.2	Modelo Conceitual de Integração	55
3.2	Equivalência entre Redes de Petri e Guaraná	56
3.3	Modelo de Simulação Proposto	59
3.4	Resumo do Capítulo	64
4	Experimentação	67
4.1	Experimento	67
4.1.1	Descrição dos Experimentos e Cenários	68
4.1.2	Variáveis Observadas	71
4.1.3	Apresentação da Ferramenta	72
4.2	Resultados e Discussão	74
4.3	Verificação e Validação	78
4.3.1	Definição e Técnicas de Verificação	78
4.3.2	Definição e Técnicas de Validação	80
4.3.3	Verificação do Modelo de Simulação	81
4.4	Resumo do Capítulo	82
5	Conclusão e Trabalhos Futuros	91
	Bibliografia	95

Índice de figuras

1.1	Solução de integração de aplicações.	4
2.1	Tranferência de arquivos. (Hohpe e Woolf [32])	14
2.2	Banco de dados compartilhado. (Hohpe e Woolf [32])	15
2.3	Chamada de procedimento remoto. (Hohpe e Woolf [32])	16
2.4	<i>Messaging</i> . (Hohpe e Woolf [32])	17
2.5	Topologia <i>Point-to-Point</i> . (Martins [45])	18
2.6	Topologia <i>Hub-and-Spoke</i> . (Martins [45])	19
2.7	Topologia <i>Enterprise Service Bus</i> . (Martins [45])	21
2.8	Classificação das tarefas do Guaraná. (Frantz [30])	25
2.9	Método científico aplicado à simulação.	28
2.10	Classificação dos sistemas. (Law e Kelton [37])	31
2.11	Elementos básicos de uma Rede de Petri. (Francês [26])	36
2.12	Representação de lugares e transições. (Maciel et al. [43])	38
2.13	Redes de Petri matriciais. (Maciel et al. [43])	39
2.14	Exemplo de Rede de Petri em notação matricial. (Bressan [5])	40
2.15	Notação matricial da Rede de Petri. (Bressan [5])	41
2.16	Rede de Petri colorida. (Maciel et al. [43])	43
2.17	Rede de Petri temporizada. (Lima et al. [39])	45
3.1	Modelo conceitual da solução de integração. (Roos-Frantz et al. [52])	55
3.2	Troca de estados. (Roos-Frantz et al. [52])	61
3.3	Modelo de simulação	65
4.1	Componentes do HPSim	84
4.2	Mensagens acumuladas nos <i>slots</i> no experimento 1.	85
4.3	Mensagens acumuladas nos cenários do experimento 1.	86
4.4	Mensagens acumuladas nos <i>slots</i> no experimento 2.	87

4.5	Mensagens acumuladas nos cenários do experimento 2.	88
4.6	Mensagens acumuladas nos <i>slots</i> nos experimentos 1 e 2.	89

Índice de tabelas

2.1	Notação dos blocos construtores. (Frantz [30])	23
3.1	Equivalência entre elementos. (Roos-Frantz et al. [52])	57
3.2	Equivalência das tarefas modificadoras. (Roos-Frantz et al. [52])	58
3.3	Equivalência das tarefas roteadoras. (Roos-Frantz et al. [52])	59
3.4	Equivalência das tarefas transformadoras. (Roos-Frantz et al. [52]) ..	60
3.5	Equivalência das tarefas temporais. (Roos-Frantz et al. [52])	60
3.6	Tradução dos elementos envolvidos na solução de integração	62

Agradecimentos

Não há no mundo exagero mais belo
que a gratidão.

Jean de la Bruyere



Agradeço primeiramente a Deus, por ter me dado a oportunidade de estar realizando este sonho e por ter guiado meu caminho, dando forças para não desistir.

A minha família, pai Paulo Welter, mãe Marli T. Zaiaskoski Welter, avó Ilsa W. B. Welter, avô Ivo Welter, irmã Jéssica P. Welter, e em especial ao meu noivo Dionatan A. R. Santos. Agradeço, por terem estado a meu lado, apesar de todas as dificuldades, sempre me incentivando e acreditando em mim.

Ao meu orientador, professor Dr. Rafael Zancan Frantz, por acreditar no meu potencial, por sua dedicação, apoio, cobranças, além do esforço nas revisões e sugestões.

Aos professores Dra. Fabricia Carneiro Roos Frantz e Dr. Sandro Sawicki, membros do grupo GCA, por sua dedicação, e auxílio nesta dissertação.

Aos demais professores do Mestrado em Modelagem Matemática, que ofertaram conhecimentos para minha formação acadêmica.

Aos meus colegas do grupo GCA, pelo incentivo, apoio e troca de conhecimentos.

Aos meus colegas de mestrado, pela convivência, estudo e amizade.

A secretária Geni, pelo apoio prestado nos momentos difíceis.

Resumo

A vida sem luta é um mar morto
no centro do organismo universal.

Machado de Assis

As empresas, em seus processos de negócio desenvolvem ou compram aplicações que servem de base para apoiar a tomada de decisões e também para aperfeiçoar seus processos de negócio. Estas aplicações, que são desenvolvidas ou adquiridas pelas empresas, compõem seu ecossistema de *software*. As aplicações frequentemente são desenvolvidas sem a preocupação de integração, o que dificulta a possibilidade de reutilizar aplicações. A área da integração de aplicações empresariais (EAI) fornece metodologias, técnicas e ferramentas para que as empresas possam desenvolver soluções de integração, visando associar novas aplicações com as já existentes. Assim sendo, o problema abordado nessa dissertação visa analisar o comportamento de uma solução de integração de aplicações, no contexto da administração pública da cidade de Huelva (Espanha). Essa solução é responsável por gerar certificados digitais e unificar as bases de usuários que possuem acesso aos sistemas informáticos da administração. Essa pesquisa assume que é possível identificar gargalos de desempenho com base no modelo de simulação, obtido a partir do modelo conceitual da solução de integração ainda na fase de projeto, com auxílio da técnica matemática Redes de Petri temporizadas. Se um modelo conceitual for implementado com gargalos, poderá gerar falhas, que aumentam os custos, tempo e riscos na solução implementada. A simulação realizada possibilitou a análise de duas variáveis, o tempo associado às transições, e a quantidade de mensagens (*tokens*) em cada lugar (*slot*). Gargalos foram observados em alguns cenários de execução.

Palavras-chaves: Integração de Aplicações Empresariais, Redes de Petri Temporizadas, Simulação, Gargalos de Desempenho.

Abstract

*Life without struggle is a dead sea
in the center of the universal organism.*

Machado de Assis

Enterprises in their business processes develop or purchase applications that serve as a basis to support decision making and also to perfect their business processes. These applications, which are developed or acquired by enterprises, make up their software ecosystem. Applications are often developed without the concern of integration, which makes it difficult to reuse applications. The area of enterprise application integration (EAI) provides methodologies, techniques and tools so that enterprises can develop integration solutions, aiming to associate new applications with existing ones. Therefore, the problem addressed in this dissertation is to analyze the behavior of an application integration solution, in the context of the public administration of the city of Huelva (Spain). This solution is responsible for generating digital certificates and unifying the databases that store information used in the user access to the systems. This research assumes that it is possible to identify performance bottlenecks based on the simulation model, obtained from the conceptual model of the integration solution still in the design phase, with the aid of the mathematical technique Timed Petri Nets. If a conceptual model is implemented with bottlenecks, it can generate failures, which increase costs, time and risks in the implemented solution. The simulation made possible the analysis of the following variables, the time associated to the transitions, and the number of messages (tokens) in each slot. Bottlenecks were observed in some execution scenarios.

Key-words: Enterprise Application Integration, Timed Petri Nets, Simulation, Performance Bottlenecks.

Capítulo 1

Introdução

Não há nada melhor do que despertar
o prazer e o amor pelo estudo,
caso contrário, só se formam
bons carregadores de livros.

Michel Eyquem de Montaigne

Esta dissertação tem como proposta identificar possíveis gargalos de desempenho em soluções de integração, utilizando para isso um modelo matemático de simulação, que foi desenvolvido com o formalismo matemático Redes de Petri temporizadas. O modelo desenvolvido foi simulado com o objetivo de analisar o comportamento da solução. Dessa forma, este trabalho encontra-se organizado em 5 seções, sendo que a Seção §1.1 descreve o contexto no qual a pesquisa está inserida. A Seção §1.2 traz a motivação do trabalho. A Seção §1.3 elucida os objetivos que se pretende atingir com a presente dissertação. A Seção §1.4 aborda a metodologia utilizada na elaboração do trabalho. A Seção §1.5 apresenta o resumo das contribuições desta pesquisa, e, por fim, a Seção §1.6 finaliza este capítulo, descrevendo como está organizada a dissertação.

1.1 Contexto da Pesquisa

As empresas, nos dias atuais, vêm enfrentando um ambiente extremamente competitivo, pois estão inseridas em uma sociedade profundamente

afetada pelas novas exigências da chamada sociedade de informação [2]. Essa nova realidade exige uma reorganização por parte das empresas, de modo que consigam continuar crescendo e resistindo à grande competição existente no mercado. Para isso, é importante que as empresas utilizem sistemas de informação, visando estreitar os vínculos com seus clientes, o que auxilia seus processos de negócio.

Um sistema de informação é um conjunto de partes que interagem entre si, pretendendo atingir um objetivo [51]. Tais sistemas combinam recursos humanos e computacionais, que inter-relacionam a coleta, o armazenamento, a recuperação, a distribuição e o uso de dados, com o objetivo de realizar um bom planejamento, controlar, e comunicar a tomada de decisões, dentro das empresas [3]. Além disso, também podem ajudar os usuários a analisar problemas, criando novos produtos e serviços e buscando solucionar questões complexas, com a finalidade de fornecer suporte aos processos de negócio das empresas.

Os processos de negócio são criados ou modificados em função das necessidades das empresas. Para que essas necessidades sejam sanadas, os processos de negócio precisam estar integrados entre si, pois, desta forma, os sistemas computacionais estarão conectados. Portanto, considera-se nesta dissertação que um sistema de informação é um *software*, com a utilidade de dar suporte aos processos de negócio das empresas.

O desenvolvimento do *software* segundo Pressman [48] geralmente ocorre em 5 fases: especificação, projeto, implementação, testes e evolução. Na fase de especificação, define-se o que é esperado do *software*. Na fase de projeto, criam-se os modelos conceituais. Na fase de implementação constrói-se o *software*. Na fase de testes, encontram-se erros e na fase de evolução, realizam-se alterações, inerentes às necessidades do cliente.

Segundo Rezende [51], a engenharia de *software* visa sistematizar a produção, manutenção, evolução e recuperação de produtos de *software*, de modo que seu desenvolvimento ocorra dentro de prazos, com custos estimados e progresso controlado. A engenharia de *software* oferece suporte ao desenvolvimento do *software*, utilizando princípios, métodos, tecnologias e processos que devem estar em contínuo aprimoramento. Nesta perspectiva, surgem as aplicações empresariais, que despontam com o intuito de facilitar os processos de negócio dentro das empresas, visando agilizá-los, além de melhorar a lucratividade e competição dentro do mercado. Estas aplicações são pacotes de *software*, que geralmente são desenvolvidos pela própria empresa ou comprados de empresas terceirizadas.

As empresas possuem um ecossistema de *software*, composto por um conjunto heterogêneo de aplicações e, com o passar do tempo, necessitam tornar seus processos de negócios mais dinâmicos. Quando o ecossistema de *software* torna-se ineficiente, as empresas desenvolvem e compram novas aplicações, visando dar suporte informático a esses processos. O desenvolvimento destas aplicações, geralmente, ocorre sem a preocupação de integração.

Para suprir a necessidade de integrar as inúmeras aplicações necessárias ao funcionamento de uma empresa, surge a Integração de Aplicações Empresariais, do inglês *Enterprise Application Integration*, comumente conhecida por EAI, que proporciona metodologias, técnicas e ferramentas para projetar e implementar soluções de integração. A EAI é a solução de um problema, que existe desde a criação das primeiras aplicações, sendo este o compartilhamento irrestrito de dados e processos de negócios, entre quaisquer aplicações conectadas a fontes de dados na empresa, deste modo, ela surge devido a dificuldade do sistema, integrar aplicações que foram criadas em separado [41].

Para Linthicum [41], as empresas geralmente utilizam a EAI, pois têm necessidade de compartilhar dados e processos sem que seja necessário realizar muitas mudanças nas aplicações, portanto uma boa EAI deve ser de baixo custo e funcional. O surgimento da EAI, desponta como um novo conceito, no que tange a integração das aplicações empresariais, sendo que sua maior contribuição é a possibilidade de reutilizar aplicações.

Nesta perspectiva, uma solução de integração visa incorporar as diferentes aplicações, que constituem o ecossistema de *software* das empresas, permitindo que seus dados e funcionalidades sejam compartilhados [41]. Segundo Frantz et al. [29] uma solução de integração tem como objetivo orquestrar um conjunto de aplicações para mantê-las sincronizadas, ou proporcionar novas funcionalidades que possam ser construídas a partir das já existentes. A Figura §1.1 representa uma solução de integração que sincroniza as aplicações do ecossistema de *software*.

Hohpe e Woolf [32] afirmam que existem dois estilos diferentes de integração: compartilhamento de dados e compartilhamento de funcionalidades. Destaca-se, que ambos são utilizados, para implementar soluções de integração. O compartilhamento de dados, ocorre entre as aplicações, sendo que elas compartilham informações, transferem arquivos, e utilizam um sistema de gerenciamento de banco de dados. O compartilhamento de funcionalidades ocorre quando as aplicações precisam acessar informações umas das outras, desta forma, pode ser empregada a técnica da chamada de procedimento remoto, ou um sistema de mensagens.

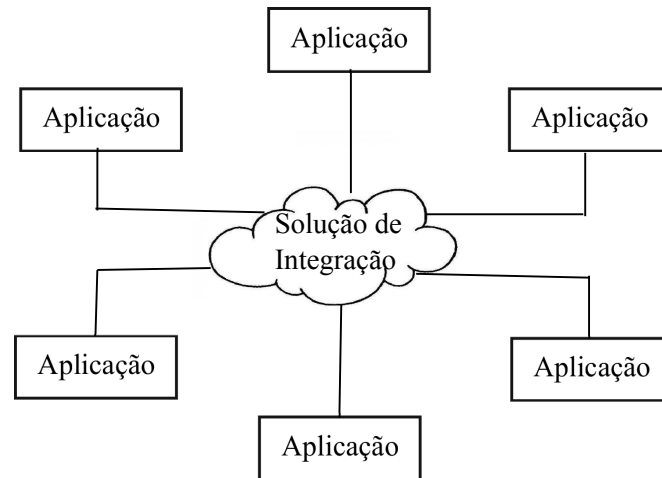


Figura 1.1: *Solução de integração de aplicações.*

Uma solução de integração pode ser representada através de um modelo conceitual, que demonstra como estão organizados os componentes do modelo, em um alto nível de abstração. Existem diferentes tecnologias, nas quais as soluções de integração podem ser implementadas, tais como: Camel [33], Mule [21], Spring Integration [24] e Guaraná [28].

A simulação é um campo de pesquisa que lida com a experimentação de modelos, de modo a fazer previsões sobre o comportamento e o desempenho de sistemas reais [54]. Simular consiste em avaliar qual teria sido o comportamento de um sistema, se este obedecesse determinadas regras de operação, onde certos impulsos houvessem ocorrido. O termo simulação, no sentido mais técnico, é usado para descrever o comportamento de um sistema que possui semelhanças com outros sistemas ou equações matemáticas. Para que a simulação possa ser realizada, é necessário utilizar uma técnica que permita simular sistemas de eventos discretos. Neste sentido, foi utilizada a técnica matemática Redes de Petri temporizadas, que permite associar tempo aos diversos componentes do sistema.

As soluções de integração podem ser caracterizadas como sistemas estocásticos, visto que a taxa de entrada de dados, não pode ser prevista, sendo aleatória, dependendo de aplicativos que estão sendo integrados. O estado no qual uma solução de integração sofre mudanças afeta o número de mensagens que estão sendo processadas em um determinado tempo de execução, além de ser influenciado por operações executadas nas mensagens.

As soluções de integração são caracterizadas como sistemas de eventos discretos, pois todos os componentes que são incluídos na solução de integração utilizam um determinado tempo de execução, no momento em que ocorre um evento. Desta forma, qualquer evento pode alterar o estado da solução de integração proposta. Com isso, tem-se um sistema discreto, sendo possível criar um modelo formal ou matemático, a partir do modelo conceitual de uma solução de integração, utilizando, para isso, técnicas e ferramentas pré-determinadas para a simulação de eventos discretos [54]. Chwif [9] complementa esta ideia, ao dizer que a simulação de eventos discretos, é uma ferramenta bem conhecida e utilizada, sendo capaz de avaliar sistemas complexos, considerando seu comportamento dinâmico.

Simulações na solução de integração são realizadas com o intuito de observar o comportamento e prever o desempenho da solução, sendo que nesta dissertação propõe-se analisar estes aspectos na fase de projeto, por este motivo, serão realizadas simulações nesta etapa. As simulações realizadas irão gerar dados, a partir dos quais busca-se encontrar gargalos de desempenho e, deste modo, evitar falhas, gerando diminuição de custos, riscos e também ganho de tempo.

1.2 Motivação

A análise do comportamento e a identificação de possíveis gargalos de desempenho nas soluções de integração de aplicações empresariais, abrangem o desenvolvimento da solução, para posterior execução em diferentes cenários críticos de funcionamento, projetados por engenheiros de *software*. Esta abordagem gera custos que costumam ser elevados e que envolvem tempo e recursos, além de riscos comumente conhecidos por *bugs*. Neste sentido, a presente pesquisa possui como motivação analisar uma solução de integração visando identificar possíveis gargalos de desempenho ainda na fase de projeto, a partir dos modelos conceituais das soluções de integração.

Nesta perspectiva, elaborou-se a seguinte hipótese:

Soluções de integração de aplicações podem ser classificadas como sistemas estocásticos, dinâmicos e de eventos discretos. Portanto é possível utilizar modelos computacionais, formalismos matemáticos e técnicas de simulação para analisar o comportamento e identificar possíveis gargalos de desempenho que possam surgir nas soluções de integração, a partir da construção de modelos formais de simulação, quando submetidos à cenários críticos de funcionamento, tendo como base seus modelos conceituais.

1.3 Objetivos

Nas subseções seguintes, são descritos os objetivos desta dissertação.

1.3.1 Geral

Propor um modelo de simulação utilizando Redes de Petri temporizadas, tendo como base o modelo conceitual da solução de integração da administração pública de Huelva (Espanha), para identificar possíveis gargalos de desempenho, ainda na fase de projeto.

1.3.2 Específicos

- Apresentar a equivalência existente entre os elementos das Redes de Petri e a tecnologia Guaraná;
- Prever o comportamento de uma solução de integração de aplicações em diversas situações críticas;
- Desenvolver um modelo de simulação utilizando Redes de Petri temporizadas, que seja equivalente ao modelo conceitual, proposto por Frantz [30] para integrar os sistemas existentes na administração pública de Huelva (Espanha);
- Analisar o modelo de simulação em diferentes experimentos e cenários, visando conhecer variáveis que permitam identificar possíveis gargalos de desempenho;
- Realizar uma análise de tempos e quantidades de mensagens nos *slots*, por meio do modelo de simulação;
- Analisar o comportamento e prever o desempenho do caso de estudo, da administração pública de Huelva (Espanha), utilizando o formalismo matemático Redes de Petri temporizadas;

1.4 Metodologia

Neste trabalho de pesquisa, a metodologia adotada baseia-se em revisão bibliográfica sobre o assunto e pesquisa experimental. Com a pesquisa bibliográfica, foi possível conhecer profundamente os conceitos inerentes ao

desenvolvimento da dissertação, sendo eles: processos de negócios das empresas, *software*, ecossistema de *software*, aplicações, solução de integração, integração de aplicações empresariais, simulação e gargalos de desempenho. Também foi realizada a análise e compreensão do caso de estudo, e elaboração de um modelo de simulação utilizando Redes de Petri temporizadas, sendo este equivalente ao modelo conceitual.

A pesquisa experimental, por sua vez, realizou-se através de análises com base na solução de integração estudada nesta dissertação. Inicialmente, foi necessário compreender como ocorre o funcionamento da solução, posteriormente definiu-se qual técnica matemática seria utilizada para realizar a equivalência e, deste modo, desenvolver o modelo de simulação. Para a elaboração do modelo de simulação, organizou-se uma tabela de equivalência entre os elementos que compõem a solução de integração desenvolvida com a tecnologia Guaraná, e sua correspondência com a técnica matemática Redes de Petri, utilizada na elaboração do modelo de simulação.

Na sequência, foram realizadas simulações, sendo necessário definir quais variáveis deveriam ser observadas nas análises realizadas. Para que a simulação pudesse ser realizada, definiram-se formas de controle e escolheram-se métodos que permitiram coletar dados, além da definição dos cenários de simulação. Deste modo, tornou-se possível adaptar o modelo de simulação, desenvolvido com auxílio da técnica matemática Redes de Petri temporizadas, visando realizar as análises consideradas na pesquisa. As simulações foram realizadas no *software* HPSim, que permite adicionar tempo aos componentes do modelo, possibilitando a análise do tempo associado às transições, e do acúmulo de mensagens nos *slots*. Posteriormente, estudaram-se técnicas de verificação, com a finalidade de escolher uma que se adapte ao caso de estudo, para desta forma, verificar o modelo de simulação.

Além disso, a dissertação incorporou o *feedback* de outros grupos de pesquisa. Para que a pesquisa fosse realizada, ocorreram reuniões entre o pesquisador e os orientadores envolvidos no projeto. Ficou definida a necessidade de elaborar um cronograma, que abrangesse todas as etapas necessárias ao desenvolvimento da pesquisa.

Também ocorreu a participação em eventos científicos e tecnológicos relacionados com os temas de pesquisa do projeto, visando obter um melhor entendimento a respeito da temática. Ao final da pesquisa, coletaram-se resultados, que foram transcritos para a dissertação, possibilitando a análise das variáveis de maneira abrangente, gerando discussões sobre os resultados encontrados com as simulações.

1.5 Resumo das Contribuições

Esta dissertação integra o projeto de pesquisa "Simulação para Predição do Comportamento de Soluções de Integração de Aplicações Empresariais", do Grupo de Pesquisa em Computação Aplicada (GCA). As principais contribuições que esta pesquisa trouxe são indicadas na continuação:

- Identificação de cenários nos quais aparecem gargalos de desempenho na solução de integração da administração pública da cidade de Huelva (Espanha).
- Identificação de gargalos de desempenho, tendo como base o modelo conceitual.
- Elaboração de um modelo de simulação, equivalente ao modelo conceitual desenvolvido com a tecnologia Guaraná.
- Verificação do modelo de simulação com auxílio de técnicas de verificação presentes na literatura.
- O estudo de diferentes técnicas matemáticas produziu informações relevantes a respeito do comportamento destas técnicas e das semelhanças e diferenças entre propriedades inerentes a elas, estudadas em um *framework* de comparação. Os resultados obtidos foram publicados na XXI Jornada de Pesquisa da UNIJUÍ, realizada na cidade de Ijuí, incorporada ao Salão do Conhecimento, entre os dias 26 a 30 de setembro de 2016, sob o título "*Framework* de comparação entre técnicas matemáticas" [60].
- Durante a revisão bibliográfica da pesquisa, tornou-se possível analisar o comportamento de uma solução de integração que gera certificados digitais e unifica as bases de usuários de acesso dos sistemas informáticos, na administração pública de Huelva (Espanha), viabilizando o desenvolvimento de um modelo de simulação, equivalente ao modelo conceitual. Este resultado foi publicado com o título "Uso de Redes de Petri para identificação de gargalos de desempenho em soluções de integração de aplicações: caso de estudo na administração pública de Huelva", na I Mostra de Pós-Graduação, promovida pelo centro universitário Univates, realizada no dia 20 de outubro de 2016 em Lajeado, RS [59].

- Durante o desenvolvimento da pesquisa, foi possível elaborar um artigo, que apresenta um modelo de simulação para o problema de integração em um contexto que envolve a necessidade de melhorar o atendimento aos alunos da universidade UNIJUÍ, este modelo de simulação foi desenvolvido utilizando o formalismo matemático Redes de Petri. O resultado desta pesquisa intitula-se "*Mathematical Model for Simulating an Application Integration Solution in the Academic Context of Unijuí University*", que foi publicado na 8ª edição do Centeris (*Conference on Enterprise Information Systems*), ocorrido nos dias 05, 06 e 07 de outubro de 2016, Porto, Portugal [36].
- O estudo que envolve a simulação demonstra ser possível analisar o comportamento e o desempenho de sistemas reais, encontrando gargalos de desempenho. Com isso, o artigo, busca fazer uma comparação das diferentes características das técnicas matemáticas Redes de Petri, Cadeias de Markov e Teoria das Filas, visando analisar os principais aspectos, na perspectiva da área de integração de aplicações. Este estudo resultou em uma publicação realizada no IV Seminário de Formação Científica e Tecnológica (SFCT), intitulada "Análise comparativa entre Redes de Petri, Cadeias de Markov e Teoria das Filas", que foi apresentado na cidade de Santa Rosa, no dia 17 de junho de 2016 [58].

1.6 Estrutura dessa Dissertação

Esta dissertação possui a seguinte estrutura:

Capítulo I: Introdução. Compreende a presente introdução.

Capítulo II: Revisão da Literatura. Proporciona ao leitor uma revisão da literatura técnica e científica relacionadas à pesquisa desenvolvida nesta dissertação. Neste capítulo será apresentado o referencial teórico, a respeito da integração de aplicações empresariais, plataforma Guaraná, simulação de eventos discretos, distribuições de probabilidade, Redes de Petri e trabalhos relacionados.

Capítulo III: Modelagem. Apresenta o trabalho desenvolvido para tratar da solução de integração. Neste capítulo serão introduzidos os trabalhos relacionados identificados ao longo dessa pesquisa.

Capítulo IV: Experimentação. Neste capítulo é introduzida a contribuição central da pesquisa desenvolvida, descrevendo o processo de experimentação adotado, apresentando o experimento, trazendo os resultados

e discutindo-os, além de tratar da validação e verificação do modelo de simulação.

Capítulo V: Considerações Finais. As conclusões e trabalhos futuros, referentes a pesquisa desenvolvida nessa dissertação são apresentadas neste capítulo.

Capítulo 2

Revisão da Literatura

As pessoas que vencem neste mundo
são as que procuram as circunstâncias
de que precisam e, quando
não as encontram as criam.

George Bernard Shaw

Para tornar possível a análise do comportamento e a identificação de possíveis gargalos de desempenho em soluções de integração de aplicações empresariais, é necessário conhecer o aporte teórico, que dá subsídios para isto. A Seção §2.1 aborda o tema da Integração de Aplicações Empresariais, estilos de integração, topologias e plataformas de integração. A Seção §2.2 discorre sobre a plataforma Guaraná. A Seção §2.3 trata sobre o conceito de simulação de eventos discretos. A Seção §2.4 aborda distribuições de probabilidade. A Seção §2.5 trata dos conceitos básicos de Redes de Petri, enfatizando as Redes de Petri temporizadas. A Seção §2.6 apresenta os trabalhos relacionados. Por fim, a Seção §2.7 resume esse capítulo.

2.1 Integração de Aplicações Empresariais

É comum a necessidade das empresas integrarem as aplicações que compõem seu ecossistema de *software*, visando reduzir os custos aliados a essa integração. Frequentemente, encontram-se empresas, que possuem um ecossistema de *software* composto por aplicações, as quais foram projetadas com

base em diversas tecnologias, sistemas operacionais e modelos de dados. Geralmente, estas aplicações não foram projetadas para compartilhar dados e menos ainda para compartilhar funcionalidades [28].

A Integração de Aplicações Empresariais, do inglês *Enterprise Application Integration* (EAI), consiste em uma resposta para décadas de criação de aplicações monolíticas, ou seja, aplicações que não foram pensadas para serem integradas, nem para trabalhar em conjunto com outras aplicações. A EAI visa solucionar um problema que existe desde o início do desenvolvimento das primeiras aplicações: a dificuldade de uma aplicação compartilhar dados, e/ou funcionalidades com outras aplicações [41].

A EAI proporciona metodologias, técnicas e ferramentas para projetar e implementar soluções de integração, sendo que o objetivo de uma solução de integração é orquestrar duas ou mais aplicações [28]. Uma solução de integração orquestra um conjunto de aplicações com a finalidade de manter seus dados em sincronia ou proporcionar novas funcionalidades, a partir das aplicações já existentes no ecossistema de *software* das empresas [32].

Frequentemente, as empresas têm como demanda compartilhar dados e processos sem serem necessárias grandes mudanças nas aplicações ou nas estruturas de dados. Por esse motivo, algumas empresas possuem seus dados armazenados em repositórios, que de tempos em tempos podem receber novos dados. Visando evitar que os dados já armazenados sejam perdidos, faz-se uma reutilização dos mesmos, podendo ser ampliados quando ocorre a implementação de uma nova aplicação, de modo que todos os dados estejam sincronizados, assim como as aplicações também devem estar. Este método corresponde às ideias de Linthicum [41], que declara como sendo um método de EAI efetivo, aquele funcional e de baixo custo.

As empresas precisam de aplicações para dar suporte aos seus processos de negócios e, por isso, costumam desenvolvê-las ou comprar de terceiros. Quando as aplicações são desenvolvidas pela própria empresa, tem-se uma maneira eficaz de diminuir custos. As empresas costumam investir muito dinheiro em integração, cerca de cinco a vinte vezes mais do que em desenvolvimento, sendo que isso ocorre porque as tecnologias de integração ainda precisam evoluir, pois existe uma carência muito grande de tecnologias de integração de aplicações empresariais eficientes [57].

As aplicações geralmente são desenvolvidas de forma independente, e precisam ser adaptadas ao sistema que já está implementado na empresa. Isto ocorre, porque muitas vezes as empresas compram aplicações já

prontas, que foram desenvolvidas por diversas empresas. Além disso, algumas aplicações foram criadas em épocas diferentes, o que acaba gerando divergências entre as mesmas, por utilizarem linguagens distintas.

Para diminuir custos e ganhar tempo, as empresas utilizam-se da tecnologia EAI para projetar e implementar soluções de integração empresariais. As soluções que são implementadas, orquestram os dados provenientes das aplicações, sincronizando-os com os novos dados, além de serem capazes de desenvolver novas funcionalidades, sem alterar o que já estava sendo utilizado [28].

Nas seções seguintes, apresentam-se os estilos de integração, geralmente usados para implementar soluções de integração; as topologias, que são utilizadas para integrar as diversas aplicações; e as plataformas de integração que possibilitam o desenvolvimento e a implementação das soluções de integração.

2.1.1 Estilos de Integração

Existe mais que um estilo de integração de aplicações, sendo que cada um deles considera tópicos inerentes à determinada operação de integração [32]. Existem quatro estilos de integração, definidos por Hohpe e Woolf [32], são eles: transferência de arquivos, banco de dados compartilhado, chamada de procedimento remoto e *messaging*. Na transferência de arquivos, as aplicações têm acesso aos arquivos de dados, produzidos por outras aplicações. No banco de dados compartilhado, todos os dados são armazenados em um banco de dados, ao qual todas as aplicações têm acesso. Na chamada de procedimento remoto, as aplicações têm acesso às funcionalidades, disponibilizadas por outras aplicações, com o auxílio de aplicativos. E, por fim, no estilo *messaging* as aplicações realizam trocas de mensagens entre si, tornando possível transferir e compartilhar dados.

Transferência de Arquivos

Para que diferentes aplicações trabalhem em conjunto, é necessário um mecanismo de transferência de dados, que possa ser utilizado por diversas linguagens e plataformas, sendo compatível com as aplicações da empresa. Neste sentido, tem-se a transferência de arquivos, representada na Figura 2.1, nela observa-se que a transferência de arquivos ocorre quando uma aplicação exporta dados que deseja compartilhar, e outra aplicação importa dados que foram disponibilizados pela primeira aplicação.

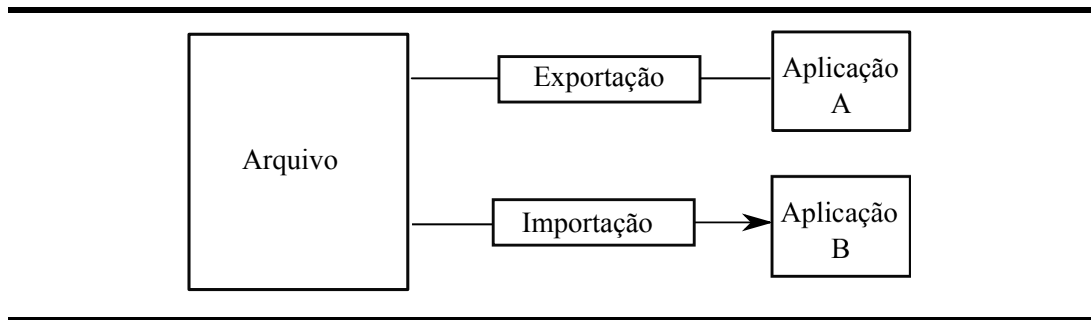


Figura 2.1: *Transferência de arquivos.* (Hohpe e Woolf [32])

A transferência de arquivos pode ser considerada simples, pois não há necessidade de adquirir ferramentas específicas para a integração. Porém, é imprescindível que os dados compartilhados estejam em um formato comum, ou seja, um formato que todas as aplicações possam acessar. No entanto, uma aplicação só pode importar um arquivo, depois que a outra já tiver exportado o mesmo. Os dados compartilhados que foram exportados podem ser importados por mais de uma aplicação, e podem algumas vezes sofrer alterações antes mesmo de serem lidos. Outro aspecto relevante é que a orquestração das aplicações geralmente é realizada manualmente, portanto as atualizações não ocorrem com grande frequência, o que pode ocasionar falta de sincronização, além disso, é necessário ter cuidado com os nomes de arquivos e diretórios, que devem ser únicos [32].

Neste estilo, a maior dificuldade consiste em gerenciar os dados, de forma que quando forem exportados, não ocorra perdas de nenhum tipo, sendo que a aplicação que gerou o arquivo, é responsável por isso. Esse estilo de integração gera custos computacionais, relacionados com o processamento dos dados, estes custos podem ser altos [32]. A Transferência de Arquivos geralmente é utilizada em cenários de integração simples, compostos por poucas aplicações, pois, em cenários mais complexos, existem dificuldades de gerenciamento, além de ser mais difícil encontrar e resolver possíveis falhas.

Banco de Dados Compartilhado

Para que o gerenciamento do banco de dados ocorra de forma satisfatória, é necessário criar uma central de armazenamento de dados que conecte todas as aplicações, que podem estar distribuídas em mais de um computador, de modo que qualquer uma delas tenha acesso aos dados sempre que for necessário [32]. Desta forma, as aplicações serão integradas, armazenando os dados

em um banco de dados compartilhado, que é utilizado por várias aplicações, que leem e podem modificar os dados, de acordo com suas necessidades.

Um banco de dados compartilhado, conforme o representado na Figura §2.2, para Hohpe e Woolf [32] tem como principal finalidade, facilitar o gerenciamento de dados, que estarão em um mesmo formato. Todas as aplicações que fazem parte da solução de integração utilizam o mesmo banco de dados e por esse motivo, quando ocorre alguma falha, é preciso corrigí-la rapidamente, para evitar que os novos dados gerados, tornem-se incompatíveis com os já existentes.

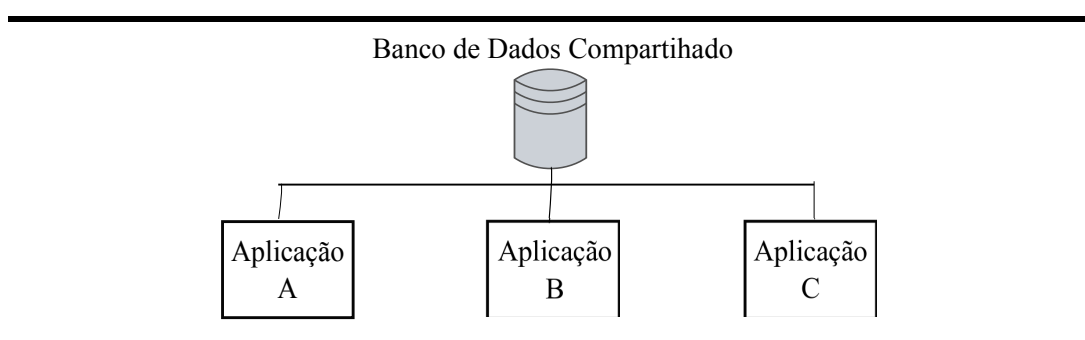


Figura 2.2: Banco de dados compartilhado. (Hohpe e Woolf [32])

Nesse estilo, existem várias aplicações integradas, e todas têm acesso ao mesmo banco de dados, o que leva à certeza de que seus dados sempre serão consistentes e compatíveis. Existem problemas decorrentes da utilização do banco de dados compartilhado, dentre eles, um dos maiores é encontrar um *design* adequado, que atenda as necessidades de toda a empresa [32]. Existem também outros tipos de dificuldades, como por exemplo, conflitos internos e aquisição de aplicações terceirizadas que foram desenvolvidas em formatos diferentes. Por outro lado, as atualizações ocorrem simultaneamente, tornando possível desenvolver sistemas de gerenciamento, nos quais os dados podem ser manipulados de forma eficiente.

Chamada de Procedimento Remoto

No estilo chamada de procedimento remoto, tem-se por objetivo integrar processos, sendo que isso ocorre quando uma aplicação invoca procedimentos de outra, deste modo, são compartilhadas funcionalidades entre as aplicações, o que ocorre quando uma aplicação necessita de dados gerados pela funcionalidade que está implementada em outra aplicação. Para que

isso ocorra, as aplicações são compostas por dados encapsulados, ou seja, dados que ficam armazenados em uma *interface* chamada função. As aplicações interagem diretamente umas com as outras e, por esse motivo, existe uma forte dependência entre elas, portanto se uma aplicação falhar, as outras poderão ser afetadas [32].

Para os autores Hohpe e Woolf [32], a chamada de procedimento remoto, conforme representada na Figura 2.3, permite que uma aplicação exiba sua *interface*, que é um conjunto de funções, de modo que as outras aplicações que compõem o ecossistema possam chamar funcionalidades da aplicação inicial. Existem diversas maneiras de facilitar a compreensão dos dados pelas aplicações, como por exemplo fornecer *interfaces*, que possuam os mesmos dados, tornando possível realizar alterações conforme seja necessário.

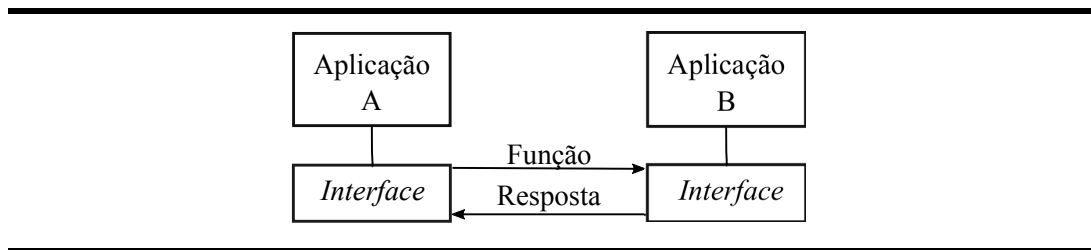


Figura 2.3: Chamada de procedimento remoto. (Hohpe e Woolf [32])

Nesse estilo, quando uma aplicação necessita de alguma informação gerada pela funcionalidade, que está implementada em outra aplicação, a mesma faz um pedido para que a aplicação lhe passe estes dados. As aplicações têm como obrigação, manter a integridade dos seus dados, além disso, cada uma delas têm a capacidade de alterar seus dados sem afetar as outras aplicações. Existem maneiras de facilitar a compreensão das funcionalidades pelas diferentes aplicações, pois elas podem fornecer inúmeras *interfaces*, utilizando os mesmos dados, o que permite realizar alterações conforme torne-se necessário, proporcionando uma maior capacidade de suportar diferentes tipos de dados [32]. Apesar do encapsulamento de dados ajudar a reduzir o acoplamento de aplicações, isso ainda ocorre inúmeras vezes, aumentando a complexidade da integração e gerando uma forte dependência, por isso é preciso criar estratégias, visando recuperar falhas.

Messaging

O estilo *messaging* é um sistema de mensagens utilizado para transferir dados e compartilhar funcionalidades imediatamente, de maneira confiável e

assíncrona, não sendo necessário preocupar-se com seu formato. Além disso, para que uma mensagem seja enviada, todas as aplicações não precisam estar sendo executadas ao mesmo tempo, o que faz com que cada aplicação conecte-se a um sistema de mensagens comum, realize trocas de dados e invoque funcionalidades através de mensagens [32]. Quanto à solução para os problemas de distribuição de sistemas, ou autores Hohpe e Woolf [32] citam o formato de mensagens assíncrono como sendo a melhor solução. O estilo *messaging* está representado na Figura §2.4.

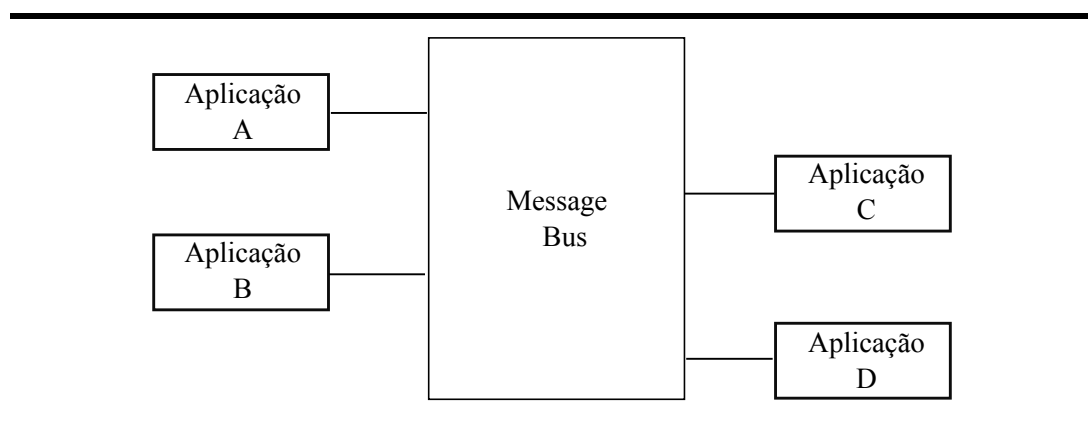


Figura 2.4: *Messaging*. (Hohpe e Woolf [32])

No estilo *messaging* as mensagens podem estar transitando, sem a necessidade de avisar remetente ou destinatário, esta dissociação permite que a integração possa transmitir mensagens para vários receptores. Como as aplicações são desenvolvidas separadamente, podem ocorrer divergências em seus modelos, o que ocasiona, algumas vezes, problemas de semântica [32]. Neste contexto, a capacidade de transformar as mensagens permite que as aplicações sejam dissociadas umas das outras.

2.1.2 Topologias

Existem três tipos de topologias que são mais utilizadas na integração de aplicações. Essas topologias indicam como as aplicações relacionam-se, estando integradas, em uma solução de integração. As topologias são subdivididas em: *point-to-point*, *hub-and-spoke* e *enterprise service bus* [56].

Point-to-Point

A topologia *point-to-point* caracteriza-se por haver um canal de comunicação direto entre uma aplicação e outra, portanto as possíveis trocas de

dados ou chamadas de funcionalidades, ocorrem diretamente entre duas aplicações, sem passar por um intermediário. A aplicação de origem converte o formato dos dados, para um formato que seja aceito pela aplicação de destino. Desta forma, duas aplicações estão integradas quando uma aplicação transfere dados ou chamadas de funcionalidades, para outra.

A Figura §2.5 representa a topologia *point-to-point*. É possível observar que a ligação é feita diretamente entre as aplicações, sem passar por nenhum intermediário. Esta topologia pode se tornar bastante complexa, quando o número de aplicações aumenta.

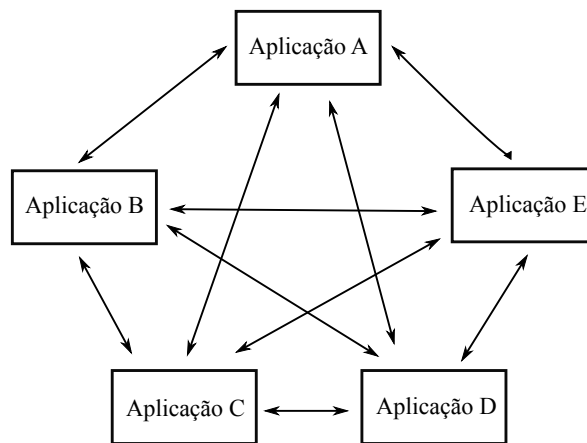


Figura 2.5: *Topologia Point-to-Point.* (Martins [45])

Pode-se calcular o número de conexões entre as aplicações, através da seguinte expressão:

$$\text{Número de conexões} = \frac{n(n-1)}{2}$$

Analisando a quantidade de conexões necessárias para n aplicações, observa-se que o número de conexões cresce de forma exponencial. Portanto, torna-se difícil administrar a integração das aplicações, com o uso da topologia *point-to-point*, quando o número de aplicações é alto. Neste sentido, a topologia deve ser utilizada quando houverem poucas aplicações, pois é uma arquitetura pouco escalável.

A topologia *point-to-point*, realiza a transferência de dados ou chamadas de funcionalidades, por isso utiliza o estilo de integração chamada de procedimento remoto. Esta utilização, faz com que uma aplicação seja alterada para chamar uma funcionalidade de outra aplicação [14].

A topologia *point-to-point* possui as seguintes características, eficiência, alto desempenho, ligações diretas entre duas aplicações, distribuição do processamento e do fluxo de dados, e crescimento uniforme, quando há acréscimo de novas aplicações [14]. Além destas características existem outras, complexidade de administração, o número de pontos de integração (conexões) aumenta sempre que são acrescentadas novas aplicações, forte acoplagem, o fato de que quando são necessárias alterações, é preciso fazê-las em todas as aplicações, além da grande dependência de integração, que ocorre por ser difícil realizar alterações rápidas, tanto de ligações, quanto de formatos.

Hub-and-Spoke

A topologia *hub-and-spoke* utiliza um intermediário para transmitir mensagens entre as aplicações. Este intermediário chama-se *hub*. O *hub* é responsável por encaminhar mensagens de uma aplicação para outra, através do *spoke*, que distribui as informações, e é representado na Figura 2.6 por raios. Destaca-se que a topologia *hub-and-spoke* possui um crescimento linear, pois o número de ligações cresce linearmente, em conjunto com o número de aplicações integradas.

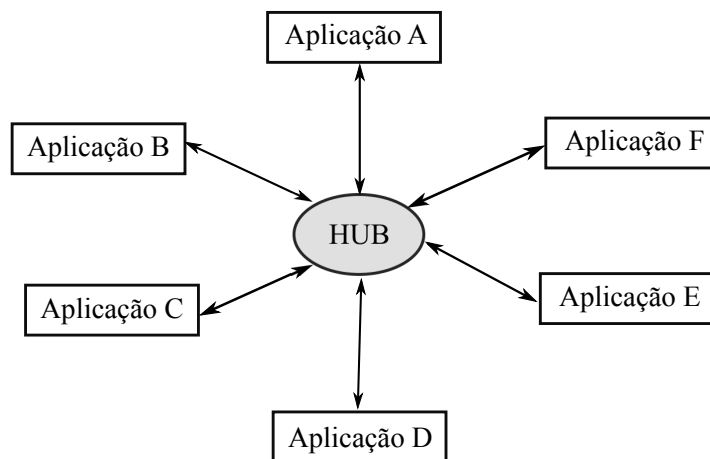


Figura 2.6: Topologia Hub-and-Spoke. (Martins [45])

A topologia *hub-and-spoke* possui um *hub*, que tem a capacidade de fazer cópias de mensagens, e de receber as mensagens enviadas pelas aplicações integradas, além disso, realiza as transformações necessárias sobre o

conteúdo, para somente então, enviar a mensagem para a aplicação de destino [14]. Dependendo da solução de integração, o *hub* realiza um trabalho de roteamento, levando em consideração, o fato de que as mensagens precisam sempre estar no mesmo formato. Nesta topologia, é fácil adicionar novas aplicações, pois estas são ligadas somente ao *hub*, que se encarrega da distribuição das mensagens entre as aplicações. A topologia *hub-and-spoke* utiliza o estilo de integração *messaging*, por ser capaz de transferir dados e chamadas de funcionalidades.

As características da topologia *hub-and-spoke* são, simplicidade conceitual, um sistema de fácil entendimento, independência, o fato de o emissor e o receptor dos dados utilizarem o *hub* para transformar mensagens, simplicidade de integração, porque as aplicações são ligadas somente ao *hub*, e fácil administração, pois o *hub* é capaz de administrar todas as aplicações [14]. Além destas, existem outras características, como por exemplo, necessidade de ter que conhecer uma tecnologia que permita implementar o *hub*, ser pouco escalável, pois todas as aplicações passam pelo *hub*, o fato de que ao acrescentar uma nova aplicação, o desempenho acaba se degradando, o que transforma o *hub* em gargalo, e pequena tolerância a falhas, pois todas as comunicações passam pelo *hub*, e se este falhar, todo o sistema irá parar até que o problema seja solucionado.

Enterprise Service Bus

As plataformas mais modernas de aplicações empresariais costumam seguir a topologia *enterprise service bus*, que geralmente apresenta um maior custo de implementação [45]. Esta topologia é uma evolução das anteriores, e tem como finalidade difundir mensagens para uma ou mais aplicações de destino, que as reconhecem subscrevendo a sua recepção [45].

A topologia possui um canal de comunicação, que recebe o nome de *BUS*, sendo responsável, pela comunicação entre os serviços de integração. As aplicações transferem mensagens, o que permite uma comunicação assíncrona, oferecendo garantias de entrega e segurança, de modo a orquestrar as aplicações. O *hub-and-spoke* é melhorado no *enterprise service bus*, pois a tarefa de configuração dos serviços é centralizada, sendo este seu único ponto de falha [56]. O estilo de integração *messaging* é utilizado, por ser capaz de transferir dados e chamar funcionalidades, sendo que esta topologia está representada na Figura §2.7.

As características desta topologia são baixo número de ligações, sendo sempre igual ao número de aplicações integradas, escalabilidade de arquitetura, além da possibilidade de disponibilizar funcionalidades remotas, por

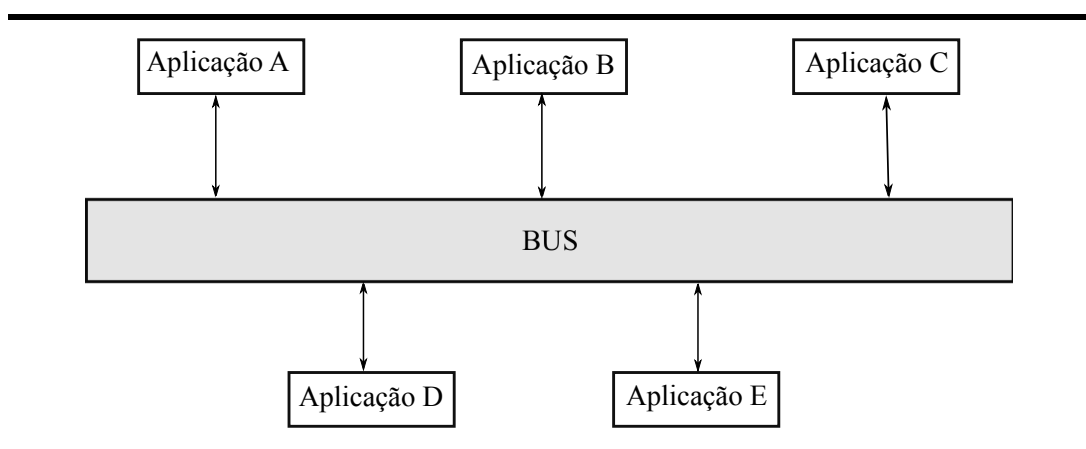


Figura 2.7: Topologia Enterprise Service Bus. (Martins [45])

meio de serviços que podem ser utilizados facilmente por outras aplicações [56]. Também existem outras características, dentre elas podem se destacar a utilização obrigatória de um componente a mais, o *BUS* que deve estar disponível, para qualquer máquina, que contenha os serviços de integração, a existência de *overhead*, que está associado à comunicação de mensagens através do *BUS*, e a necessidade das aplicações entrarem em acordo, sobre a estrutura das mensagens, pois todas precisam estar no mesmo formato.

2.1.3 Plataformas de Integração

Existem diversas plataformas para o desenvolvimento e implementação de soluções de integração. Uma plataforma é uma *interface* utilizada para desenvolver e implementar soluções de integração, sendo que deve dar suporte à decisões, considerando sempre a integração das aplicações [19]. As plataformas para desenvolvimento de soluções de integração proporcionam um conjunto de ferramentas, que normalmente incluem uma linguagem de domínio específico (DSL), uma *application programming interfaces* (API) de programação, um motor de execução e ferramentas de monitoramento.

A DSL é utilizada para criar modelos conceituais das soluções de integração, enquanto a API de programação permite aos desenvolvedores implementar modelos conceituais em código executável. O motor de execução é responsável por executar soluções de integração implementadas. As ferramentas de monitoramento permitem acompanhar o funcionamento de

uma solução de integração, sendo incluídos neste monitoramento, o seu desempenho, quantidade de mensagens recebidas, quantidade de mensagens enviadas, dentre outros aspectos. No mercado, estão disponíveis diversas plataformas de integração, dentre elas destacam-se: *Apache Camel* [33], *Mule ESB* [21], *Spring Integration* [24] e *Guaraná* [28].

Nesta dissertação, decidiu-se por utilizar a tecnologia *Guaraná*, que foi desenvolvida por pesquisadores do Grupo de Pesquisa em Computação Aplicada (GCA). Elucida-se que o estudo, será centrado no *Guaraná DSL*, pois a tecnologia *Guaraná DSL* permite projetar, implementar e executar soluções de integração. Além disso, pode ser usada como um vocabulário comum, por ter fácil comunicação, no que tange o campo da *enterprise application integration*. O *Guaraná DSL* fornece um conjunto de ferramentas de tarefas de uso geral, pois possui tarefas que fornecem as bases para muitos outros *toolkits* de tarefas, utilizados em diferentes contextos de integração. Possui uma notação gráfica que permite representar modelos de soluções de integração, com alto nível de abstração, podendo ser utilizado por engenheiros de *software* [54].

2.2 Plataforma Guaraná

A tecnologia *Guaraná* é empregada para projetar soluções de integração de aplicações empresariais. Destaca-se que ela permite aos engenheiros de *software*, manter o foco na criação de modelos, sem preocupar-se com sua implementação. Esta tecnologia possui alguns recursos, dentre estes destacam-se a Linguagem de Domínio Específico (DSL), que é utilizada para criar modelos conceituais das soluções de integração e o motor de execução que permite a implementação e execução da solução de integração.

2.2.1 Linguagem de Domínio Específico

A linguagem de domínio específico (DSL) foi desenvolvida especificamente para um domínio, de modo que tem como objetivo principal o desenvolvimento de uma semântica para aquele domínio, sendo normalmente pequena, de fácil compreensão, melhora a comunicação entre o grupo de especialistas daquele domínio, expressa soluções para o problema e tem uma expressividade limitada do domínio. O DSL geralmente é utilizado para aumentar a produtividade e comunicação entre as aplicações e possui um nível de abstração muito alto.

Neste sentido, a tecnologia *Guaraná DSL* utiliza um conjunto de ferramentas, de uso geral, que servem para o desenvolvimento de soluções de

integração, estas ferramentas recebem o nome de *toolkits* de tarefas. Além disso, possui uma notação gráfica, que possibilita representar soluções de integração, com um alto nível de abstração, sendo estes modelos independentes de plataforma. O Guaraná DSL é composto por padrões de integração de aplicações, que foram propostos por Hohpe e Woolf [32]. A tecnologia Guaraná DSL, possibilita projetar a estrutura interna de todos os processos de integração e sua comunicação com as aplicações que compõem o ecossistema de *software* das empresas.

A estrutura do Guaraná DSL é composta por blocos construtores, que estão representados na Tabela §2.1.

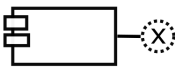

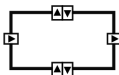
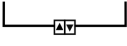

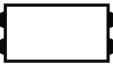


Notação	Conceito	Notação	Conceito
	Aplicação		Porta de Solicitação
	Processo de Integração		Porta de Resposta
	Porta de Entrada		Tarefa
	Porta de Saída		Slot

Tabela 2.1: Notação dos blocos construtores. (Frantz [30])

Os blocos construtores que compõem o Guaraná DSL, podem ser subdivididos em: aplicações, *slots*, portas, processos e tarefas.

As aplicações fazem parte da solução de integração, sendo que geralmente são externas, ou seja, estão localizadas externamente ao processo, inerente à solução de integração.

Os *slots* são responsáveis pela comunicação entre as tarefas, pois as interligam. O *slot*, recebe uma mensagem, proveniente de uma tarefa, e envia esta mensagem para a tarefa seguinte. Por isso, o *slot* é considerado uma unidade temporária de armazenamento. Para Frantz et al. [27], quando uma mensagem é processada e enviada para o *slot* seguinte, esta tarefa está pronta para processar outras mensagens.

As portas possibilitam o envio de mensagens. Existem diferentes tipos de portas: porta de entrada, porta de saída, porta de solicitação e porta de resposta. A porta de entrada recebe as mensagens provenientes da aplicação, e as encaminha para um *slot*, que repassa esta mensagem para a próxima tarefa. A porta de saída lê as mensagens provenientes de um *slot* e as encaminha para uma aplicação que está sendo integrada. As portas de solicitação e portas de resposta, por sua vez, possibilitam a comunicação entre as aplicações e o processo.

Uma mensagem é uma abstração da informação, que pode ser transformada, com o auxílio da solução de integração. Sua composição é cabeçalho e corpo. O cabeçalho possui algumas propriedades, como: prioridade da mensagem, identificador de mensagem e identificador de correlação. Além disso, a estrutura das mensagens, varia conforme a solução de integração.

O processo nada mais é do que um bloco, que contém a lógica de comunicação, entre as portas e as aplicações, que compõem a solução de integração. Neste sentido, os processos utilizam as portas, com o intuito de comunicar-se com as aplicações, ou com outros processos.

As soluções de integração, modeladas com o auxílio da tecnologia Guaraná DSL, frequentemente, possuem várias tarefas, sendo importante destacar que estas, podem aparecer diversas vezes em um mesmo modelo. As tarefas comunicam-se entre si, e esta comunicação, é representada por *slots*, que recebem a mensagem processada pela tarefa anterior, e a deixam disponível para ser processada pela tarefa seguinte. Assim que uma mensagem é processada, é encaminhada para o próximo *slot*, desta forma, a tarefa está habilitada para processar outra mensagem.

O Guaraná DSL, por sua vez, oferta um conjunto de ferramentas, que auxilia os engenheiros de *software*, a projetarem soluções de integração, pois possui um vocabulário de fácil compreensão [54].

As tarefas, no Guaraná DSL, podem ser representadas por um ícone, que está associado com a função, desempenhada pela tarefa. As tarefas são o principal elemento componente de um processo, e são responsáveis por modificar e também por processar mensagens. As tarefas podem ser subdivididas em: roteadoras, modificadoras, transformadoras e temporais, esta classificação pode ser observada conforme a Figura §2.8:

Abaixo as tarefas estão classificadas de acordo com a sua semântica, sendo que algumas estão sendo destacadas, pois são as que foram utilizadas no decorrer da dissertação.
































Roteadoras	Modificadoras	Transformadoras	Temporais
 Correlator	 Slimmer	 Translator	 Delayer
 Merger	 Context Slimmer	 Splitter	 Ticker
 Resequencer	 Content Enricher	 Aggregator	 Expire Checker
 Filter	 Context Content Enricher	 Chopper	
 Idempotent Transfer	 Header Enricher	 Assembler	
 Dispatcher	 Context Header Enricher	 Cross Builder	
 Distributor	 Header Promoter	 Custom Transformer	
 Replicator	 Header Demoter		
 Semantic Validator	 Set Correlation ID		
 Threader	 Set Return Address		
	 Custom Router		

Figura 2.8: Classificação das tarefas do Guaraná. (Frantz [30])

- Roteadoras: não alteram o estado das mensagens que processam e enviam as mensagens de entrada para seu destino. Destaca-se que podem multiplicar, correlacionar, direcionar e ordenar mensagens no fluxo da solução de integração. Neste grupo, é possível destacar a tarefa *filter*, o *merger*, o *correlator* e o *replicator*. O *filter* é utilizado para retirar mensagens do fluxo, de acordo com seus critérios. O *merger* serve para direcionar mensagens que são provenientes de diferentes *slots* de entrada, para um único *slot* de saída. O *correlator* analisa as mensagens de entrada e produz um conjunto de dados correlacionados, em sua saída. O *replicator*, por sua vez, é responsável por realizar cópias de uma mensagem de entrada, conforme a quantidade de *slots* de saída.
- Modificadoras: produzem uma nova mensagem ou alteram o conteúdo original de uma mensagem de entrada, podendo acrescentar informações ao conteúdo, assim, o conteúdo da mensagem é transformado para outro formato. Destaca-se a tarefa *context-based slimmer*, que recebe uma mensagem de entrada e busca mais informações, em recursos externos, levando em consideração o conteúdo da mensagem original, e então acrescenta estas novas informações, que foram obtidas, transformando a mensagem.

- Transformadoras: modificam a estrutura de uma mensagem, ou constroem mensagens novas a partir de outras. Merece destaque a tarefa *translator*, que tem a função de traduzir mensagens de um formato para outro, sempre que for necessário.
- Temporais: servem para obter um controle no tempo. Esse tempo refere-se ao fluxo de execução da solução de integração, tornando possível atrasar, ou antecipar a execução de um fluxo de integração. Destaca-se a tarefa *expire checker*, que pode ser configurada, de modo a verificar, quanto tempo uma mensagem permaneceu no fluxo da integração, podendo ainda, removê-la do fluxo.

2.3 Simulação de Eventos Discretos

A simulação é um campo de pesquisa que lida com a experimentação de modelos que permitem fazer previsões sobre o comportamento e o desempenho de sistemas reais. Existem diferentes modelos de simulação. Esta seção aborda o modelo de eventos discretos. Este modelo caracteriza-se por utilizar eventos, para modelar sistemas que mudam seu estado, em momentos distintos no tempo a partir da ocorrência de eventos [54].

Maiores detalhes envolvendo a simulação de eventos discretos são apresentados na sequência desta pesquisa, com o intuito de esclarecer ao leitor, a importância da utilização da simulação na fase de projeto, para a compreensão do comportamento do sistema. Assim sendo, este referencial está dividido em três Seções, a primeira trata dos conceitos de sistema, modelo e simulação; a segunda apresenta as vantagens e desvantagens da simulação e a última aborda a classificação dos sistemas de simulação.

2.3.1 Sistema, Modelo e Simulação

Um sistema supõe que irá ocorrer uma interação de causa-efeito entre as partes que o compõem, por isto o objetivo do sistema deve ser conhecido com clareza [25]. Segundo Chwif [9], um sistema é uma combinação de componentes que atuam de forma conjunta, visando a realização de uma função que não é possível executar com qualquer componente individual. Sobre o modelo, Chwif e Medina [11] descrevem que é uma abstração da realidade, que se aproxima do verdadeiro comportamento do sistema, mas é sempre mais simples do que o sistema real. Desta forma, um modelo busca representar as relações dos componentes de um sistema. A simulação visa retratar a realidade através de um modelo, realizado ou "materializado",

somente por equações matemáticas ou por *softwares*, com *interfaces* técnicas, ou ainda, com recursos de animação que possuem uma *interface* visual [22]. Esta realidade modelada poderá ser estudada sob condições controladas, sendo que neste ambiente poderão ser realizados experimentos, que seriam inviáveis ou extremamente caros e arriscados no mundo real.

O comportamento de um sistema de simulação, decorre da análise de um modelo de simulação, que é construído e posteriormente verificado com auxílio da simulação. A verificação do modelo permite encontrar erros, visando reduzir custos de implementação, riscos de falhas e o tempo. A simulação também é realizada com o intuito de encontrar possíveis gargalos de desempenho. Ainda sobre o tema, Chwif [9] afirma que o termo simulação, é sinônimo de simulação computacional, pois embora a simulação possa ser efetuada manualmente, diante do grande número de cálculos necessários, o tempo gasto com isso, se tornaria inviável, em termos de praticidade.

Para construir um modelo de simulação, é necessário começar com objetivos. Para isso, é indispensável que se considerem as medidas de desempenho, que são as variáveis de saída, também conhecidas por variáveis de interesse, do modelo de simulação. Além disso, quando se analisa um modelo de simulação, é preciso distinguir três elementos básicos, a entidade que é o objeto de interesse do modelo; o atributo que é a propriedade desta entidade e a atividade que é o processo que pode gerar mudanças no modelo. Destaca-se que o processo de simulação segue o método científico, no qual primeiro formulam-se as hipóteses, seguidas do preparo do modelo, para na sequência, testar as hipóteses através do modelo e descobrir se as mesmas foram validadas com base nos resultados obtidos.

Um exemplo de processo de simulação pode ser visto na Figura §2.9, em que é possível entender como funciona o processo de simulação, quando o mesmo é realizado seguindo o método científico. Observa-se que, inicialmente, formula-se o sistema, posteriormente faz-se o desenvolvimento do modelo, então, aplica-se a simulação para testar o modelo, sendo que após o teste analisam-se os resultados, que se forem coerentes com as hipóteses, tornam possível finalizar o teste, não é preciso repeti-lo. Para Chwif [9], este processo ocorre da seguinte maneira: na primeira etapa/fase, quem analisa a simulação precisa compreender o sistema que será simulado, juntamente com seus objetivos. É preciso definir as hipóteses para representar o modelo abstrato, sendo que este modelo deve ser descrito, de acordo com uma técnica de representação de modelos de simulação, para deste modo torná-lo um modelo conceitual, visando o entendimento do leitor. Na segunda etapa/fase, o modelo conceitual é convertido em um modelo de simulação, utilizando

uma linguagem de simulação, ou um simulador propriamente dito. É preciso testar a equivalência do modelo de simulação, com o modelo conceitual. Para que haja a verificação do modelo computacional, são necessários resultados. Para que existam resultados, são realizados experimentos, tendo como base o modelo de simulação. Nesta etapa, realizam-se várias simulações, e analisam-se os resultados, sendo que a partir destes, pode-se verificar o modelo de simulação, desta forma é possível obter conclusões sobre o sistema que será implementado futuramente. Caso seja necessário quando os resultados da simulação não forem satisfatórios, o modelo pode ser modificado, realizando alterações, para então reiniciar o ciclo.

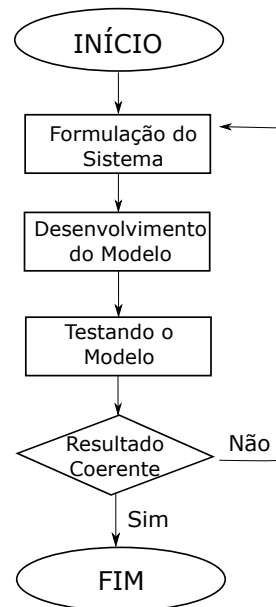


Figura 2.9: Método científico aplicado à simulação.

2.3.2 Vantagens e Desvantagens da Simulação

A simulação não é uma ferramenta capaz de substituir o trabalho de interpretação humano, mas é capaz de fornecer resultados, visando análises mais elaboradas, a respeito da dinâmica do sistema, o que permite uma interpretação mais profunda e abrangente do sistema que está sendo estudado [9]. Existem diversas áreas nas quais a simulação pode ser aplicada, podem-se destacar as seguintes: sistemas computacionais; sistemas de telecomunicações; fabricação; negócios; logística; militar; treinamento; científica;

econômica e serviços.

Inúmeros autores abordam o assunto, tendo opiniões diversas sobre a simulação. Para de Freitas Filho [17], algumas das vantagens de empregar a simulação são a possibilidade de utilizar o modelo inúmeras vezes, para avaliar projetos e políticas propostas; a possibilidade de avaliar sistemas ainda não existentes, mesmo que os dados de entrada sejam rudimentares; além da possibilidade de identificar gargalos de desempenho, sendo esta a maior preocupação, no que tange o gerenciamento operacional dos sistemas. Mas nem tudo é produtivo, o autor também cita algumas desvantagens da simulação, destacando que as desvantagens referem-se à necessidade de conhecer o *software* utilizado, o tempo requerido para elaboração dos modelos e a difícil interpretação dos resultados fornecidos pelos modelos. Isso ocorre quando não é possível saber, se determinado resultado, foi atingido devido à alteração do valor de uma variável, ou se ele deriva da interação de recursos e entidades durante a simulação. Como a simulação utiliza distribuições de probabilidade, a combinação de tais fatores pode gerar um resultado de difícil interpretação.

O autor dos Santos [20] também trata do assunto, sendo que para ele as principais vantagens da simulação são: o fato de que novas políticas, procedimentos operacionais, regras de negócio e fluxos de informação, podem ser estudados sem alterar o modelo no mundo real. Novos equipamentos, *layouts* e sistemas de transporte podem ser testados sem comprometer recursos na sua aquisição. Hipóteses sobre como e por que certos fenômenos ocorrem podem ser testadas, visando verificar sua aplicabilidade. O tempo pode ser comprimido ou expandido, permitindo acelerar ou retardar o fenômeno, que está sendo investigado. Além disso, pode-se entender a interação das variáveis do sistema, e compreender o comportamento das variáveis, na performance do sistema. Outro aspecto relevante é o fato de que um modelo de simulação, auxilia no entendimento do funcionamento do sistema, no que diz respeito ao seu modo de operar, sendo possível, validar algumas hipóteses, o que é extremamente útil na fase de *design* de um projeto.

No que diz respeito às desvantagens da simulação, dos Santos [20] considera que a construção de modelos de simulação, exige treinamento especial, por ser uma arte aprendida com tempo e experiência. Além disso, tem-se o fato de que se dois modelos forem construídos, por dois profissionais competentes, eles terão semelhanças, mas será altamente improvável que sejam iguais. Os resultados de uma simulação podem ser difíceis de interpretar, pois a maioria dos dados de saída possuem variáveis aleatórias, tornando difícil determinar se os dados resultam do relacionamento entre as variáveis

do sistema, ou são consequência da própria aleatoriedade. A construção e análise de modelos de simulação pode consumir muito tempo e, como consequência, necessitar grande quantidade de dinheiro. Muitas vezes, quando pretende-se economizar, podem ser construídos modelos incompletos. Nesse sentido, a simulação é utilizada, em casos onde uma solução analítica é possível. É importante considerar que a simulação não gera resultados exatos, ou seja, resultados iguais aos reais, porém gera resultados satisfatórios quando comparados aos reais.

Observando o que alguns autores citam sobre vantagens e desvantagens da simulação, é possível perceber que, em sua maioria, eles concordam em alguns pontos, que serão expostos na sequência. Para dos Santos [20], as principais vantagens da simulação são: reusabilidade dos modelos; possibilidade de utilização mesmo que os dados de entrada não estejam bem formulados; facilidade de aplicação, com relação aos métodos analíticos, pois possui um alto nível de detalhamento, permitindo que o modelo substitua o sistema real evitando problemas futuros; possibilidade de controlar o tempo, podendo comprimí-lo ou expandí-lo, permitir reproduzir fenômenos de maneira lenta ou acelerada, para que possam ser melhor estudados; compreender quais variáveis são mais significativas, com relação à performance do sistema, e como as mesmas interagem entre si e com outros elementos que compõem o sistema; facilitar a identificação de gargalos de desempenho; o fato de que um estudo de simulação, possibilita observar como um sistema opera, opondo-se à maneira com que imagina-se que ele opera; além do fato de que novas situações, sobre as quais têm-se poucos conhecimentos, podem ser observadas, sendo possível preparar-se, para futuros eventos. Quanto às principais desvantagens, podem-se destacar os seguintes aspectos: o fato da construção de modelos, necessitar treinamento especial; o desenvolvimento de novos sistemas envolve arte e, portanto, o aprendizado sobre os sistemas, ocorre ao longo do tempo com a aquisição de experiência; os resultados da simulação, muitas vezes são de difícil interpretação; além de que, a modelagem e a experimentação associadas aos modelos de simulação consomem muitos recursos, principalmente tempo.

2.3.3 Classificação de Sistemas de Simulação

Os sistemas possuem modelos que podem ser classificados em estocásticos, dinâmicos e discretos. Os modelos estocásticos utilizam a probabilidade para modelar sistemas reais, nos quais a incerteza está presente, tornando-os uma boa opção para representar o mundo real. Os modelos dinâmicos representam sistemas que alteram seu estado ao longo do tempo. Modelos

discretos são orientados a eventos e, desta maneira, são utilizados para modelar sistemas que mudam seu estado em momentos distintos no tempo, a partir da ocorrência de eventos.

Além disso, um modelo estocástico de simulação tem uma ou mais variáveis aleatórias como entrada. Estas entradas aleatórias levam a saídas aleatórias, que são consideradas estimativas das características verdadeiras de um modelo. Deste modo, por exemplo, uma simulação (estocástica) do funcionamento de uma agência bancária envolve variáveis aleatórias, como o intervalo entre chegadas de mensagens e a duração dos serviços prestados. Logo, variáveis como o número médio de clientes em espera, e o tempo médio de espera de um cliente, podem ser tratadas como estimativas estatísticas, das medidas reais do desempenho do sistema [20]. Os modelos de simulação dinâmicos, por sua vez, representam sistemas cujos dados variam com a passagem do tempo [20]. Todavia, em uma simulação discreta, consideram-se somente eventos onde há alteração temporal do sistema, ou seja, o tempo decorrido entre as alterações do estado do sistema, não é relevante para obtenção dos resultados da simulação, embora o tempo nunca pare [20]. A classificação dos sistemas está representada na Figura §2.10.

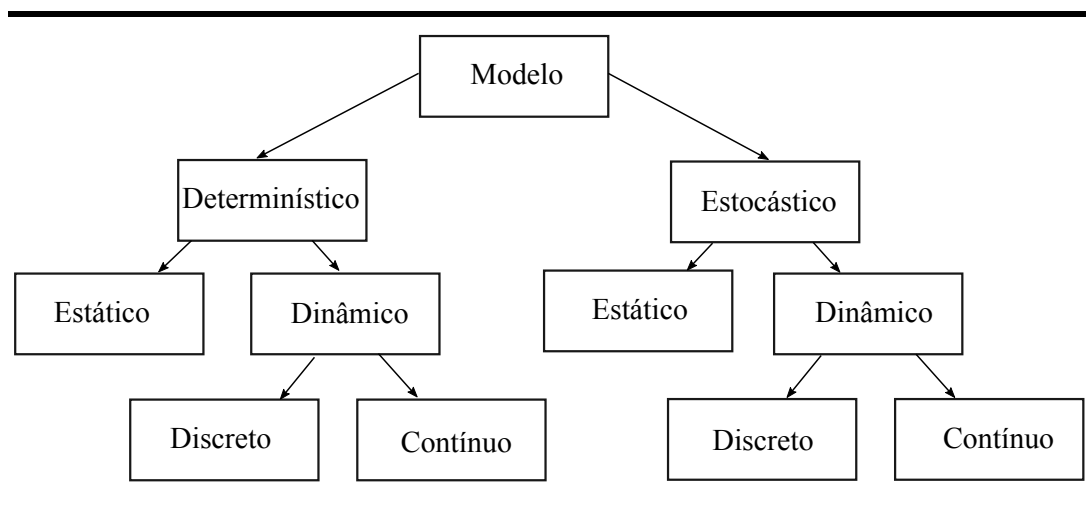


Figura 2.10: Classificação dos sistemas. (Law e Kelton [37])

A simulação de sistemas de eventos discretos é uma ferramenta muito utilizada, pois consegue avaliar sistemas complexos, considerando seu comportamento dinâmico [54]. Além disso, o modelo de eventos discretos permite prever o comportamento do sistema, pois possibilita identificar e reproduzir, as variáveis relacionadas ao desempenho, e também os meios

utilizados para interagir com outros elementos, tornando possível elaborar hipóteses por meio da observação dos modelos. Outro fator que deve ser considerado é o tempo, que pode ser controlado, visando analisar o comportamento do sistema.

Muitas aplicações pertencentes ao processo de simulação referem-se ao estudo de um sistema ao longo do tempo, caracterizando uma simulação dinâmica. Neste sentido, um aspecto importante em uma simulação refere-se à observação ao longo do tempo, dos vários serviços que compõem o modelo. Em uma simulação discreta, a passagem do tempo é feita aos saltos, entre um evento e outro, por esse motivo, é conhecida por simulação de eventos discretos. Assim sendo, supõe-se que o estado do sistema não se altera ao longo de um intervalo de tempo, compreendido entre dois eventos consecutivos [18].

São três os elementos básicos que compõem um modelo de simulação de eventos discretos: entidade, atributo e atividade [54]. Qualquer objeto envolvido com o modelo é chamado entidade. A propriedade desta entidade é nomeada atributo. Um processo que gere alterações no modelo é chamada atividade. Pode-se tomar como exemplo, um modelo de simulação de eventos discretos, que simule o funcionamento de um banco, no qual, as entidades seriam os clientes que chegam ao sistema, os atributos seriam o equilíbrio do sistema e o crédito pessoal disponível, enquanto as atividades seriam o serviço fornecido por um departamento específico do banco. É possível perceber que, desta forma, pretende-se reproduzir, as atividades de entidades envolvidas no sistema, visando descobrir algo sobre seu comportamento e desempenho. Nesse sentido, é necessário definir os estados de um sistema, e os eventos que o alteram, a partir de um estado para outro. O estado de um sistema pode ser definido por um conjunto de variáveis, utilizadas para descrever o comportamento do sistema em determinada hora. Este conjunto é chamado variáveis de estado.

Existem três maneiras que podem ser utilizadas para elaborar modelos de simulação de eventos discretos. A primeira tem como dados de entrada, a memória descritiva e a descrição do processo utilizado pelas entidades do sistema. A segunda tem como dados de entrada, a descrição completa das atividades das entidades envolvidas no sistema. A terceira maneira ocorre quando definem-se quais alterações podem ocorrer nos estados, levando em consideração o evento [54]. A partir destas estratégias, torna-se possível implementar o modelo de simulação.

2.4 Distribuições de Probabilidade

Os modelos de simulação computacionais precisam ter a capacidade de reproduzir o comportamento estocástico de um sistema real [49]. Para que seja possível solucionar este problema, utilizam-se distribuições de probabilidade, visando representar eventos que ocorrem de modo aleatório.

Uma distribuição de probabilidade é um modelo matemático que relaciona um certo valor da variável em estudo, com a sua probabilidade de ocorrência. Existem dois tipos de distribuição de probabilidade: a distribuição contínua e a distribuição discreta.

A distribuição contínua ocorre quando a variável que está sendo medida é expressa em uma escala contínua, como no caso de uma característica dimensional. Pode assumir um número infinito de valores, sendo que esses valores podem ser associados a mensurações em uma escala contínua.

A distribuição discreta ocorre quando a variável que está sendo medida só pode assumir determinados valores. Assume um número finito de valores ou tem uma quantidade enumerável de valores, como por exemplo, os números inteiros.

Algumas características numéricas possuem muita importância no que tange as distribuições de probabilidade. Estas características possuem parâmetros de distribuição, sendo estes [12]:

- Esperança matemática: conhecida por média, é denominada $E(x)$, e pode ser um número real, além de uma média aritmética;
- Variância: denominada $VAR(x)$ nada mais é que a medida do grau de dispersão (concentração) de probabilidade em torno da média. Além de conhecer a média de uma distribuição de probabilidades, é preciso uma medida que proporcione o grau de dispersão de probabilidade em torno dessa média;
- Desvio Padrão: denominado $DP(X)$, é a raiz quadrada da variância

Quando utiliza-se a modelagem no problema de pesquisa estudado, as distribuições de probabilidade são empregadas com o objetivo de realizar o disparo de transições, para que as mensagens possam seguir o fluxo. A distribuição de probabilidade utilizada mais frequentemente é a distribuição exponencial, porém também pode-se utilizar a distribuição uniforme.

Os modelos de simulação visam representar o comportamento dinâmico de sistemas, que possuam atividades concorrentes, assíncronas e não-determinísticas, através da adição do conceito de tempo no modelo, sendo que para isso são utilizadas diferentes distribuições de probabilidade [26]. O tempo também pode ser utilizado de modo probabilístico, ou seja, o disparo de transições é associado à distribuições de probabilidade.

Neste sentido, esta seção abordou as distribuições de probabilidade utilizadas para habilitar transições no *software*, utilizado para simular o modelo de simulação. O *software* empregado nesta pesquisa usa a distribuição de probabilidade para controlar o tempo, que determinada transição irá demorar para estar habilitada a disparar. Na sequência, tem-se a discussão sobre distribuição exponencial e posteriormente sobre distribuição uniforme.

2.4.1 Distribuição Exponencial

A distribuição exponencial é uma distribuição contínua de probabilidade, que tem um papel importante quando utiliza-se uma grande classe de fenômenos, a serem considerados. Esta distribuição caracteriza-se por ter uma função, com taxa de falha constante. Destaca-se que é o único tipo de distribuição que possui esta propriedade. Em termos matemáticos, considera-se que é de simples compreensão, pois descreve adequadamente, o tempo que um acontecimento demora para ocorrer [42]. A distribuição exponencial, geralmente ajusta-se bem para dados que apresentam forte assimetria, porque pode ser utilizada quando não existe correspondência entre os acontecimentos, sendo desproporcionais, ou não harmoniosos.

Definição: A variável aleatória X , tem distribuição exponencial com parâmetro λ , considerando $\lambda > 0$, quando a função da densidade de probabilidade for dada por:

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{se } x \geq 0 \\ 0 & \text{se } x < 0 \end{cases}$$

Nesta expressão, λ é o parâmetro de taxa da distribuição, que deve satisfazer $\lambda > 0$. Desta forma, λ será o tempo médio de vida, e x um tempo de falha. O parâmetro deve ter a mesma unidade de tempo que a falha x . Ou seja, se x for medido em horas, λ também deverá ser medido em horas.

A função de distribuição cumulativa de probabilidade $F(x)$ é dada por:

$$F(x) = \int_0^x f(s) ds = \begin{cases} 1 - e^{-\lambda x} & \text{se } x \geq 0 \\ 0 & \text{se } x < 0 \end{cases}$$

Utilizamos a notação $X \sim \text{Exp}(\lambda)$

2.4.2 Distribuição Uniforme

A distribuição uniforme é muito importante, sendo utilizada com grande enfoque na teoria de probabilidade. A distribuição uniforme tem uma importante característica, pois admite que todos os fenômenos, acontecerão com a mesma probabilidade.

Definição: Uma variável aleatória x possui distribuição uniforme, em um intervalo $\square a, b \square$, se a sua função da densidade de probabilidade for dada por:

$$f(x) = \frac{1}{a-b}, \text{ se } a \leq x \leq b, \text{ ou } 0$$

Deste modo, no modelo de distribuição uniforme, a probabilidade de um ponto qualquer x ser gerado em um intervalo contido no espaço amostral é proporcional ao tamanho do intervalo $\square a, b \square$.

Elucida-se que uma variável uniformemente distribuída possui uma função densidade de probabilidade, constante sobre o intervalo definido, de modo que possa satisfazer à condição $\int_{-\infty}^{\infty} f(x)dx = 1$. Esta constante precisa ser igual ao inverso do comprimento do intervalo. Além disso, uma variável uniformemente distribuída representa os resultados prováveis para qualquer intervalo $\square c, d \square$ onde $a \leq c < d \leq b$, $P(c \leq X \leq d)$, é igual para todos os subintervalos que possuam o mesmo comprimento, ou seja, $P(c \leq X \leq d) = \int_c^d f(x)dx = \frac{d-c}{b-a}$.

2.5 Redes de Petri

As Redes de Petri surgiram a partir da tese de doutorado de Carl Adam Petri, em 1962, e desde o início tinham como intuito modelar sistemas assíncronos, concorrentes, paralelos e não-determinísticos [44]. De acordo com Barroso et al. [4] uma Rede de Petri modela os estados de um sistema e suas mudanças. Na base dos sistemas, há uma forte ligação com a matemática e também uma linguagem gráfica que permite a visualização dos processos e a comunicação entre eles [43]. As Redes de Petri além de serem uma técnica de especificação formal adequada para a modelagem de sistemas, encontram aplicabilidade em diversas áreas do conhecimento, visando representar sistemas de eventos discretos.

De acordo com Cardoso e Valette [7], os elementos fundamentais que compõem as Redes de Petri são lugar, transição e *token*. O lugar é representado por um círculo, e possui vários significados, dentre os quais

merecem destaque: espera, procedimento, conjunto de recursos e estado parcial. A transição é representada por uma barra ou um retângulo e está associada a um evento que ocorre no sistema. O *token*, por sua vez, é representado por um ponto dentro de um lugar, que é um indicador e significa que a condição associada ao lugar foi verificada.

As Redes de Petri são representadas graficamente por um grafo bi-partido, que consiste em dois conjuntos distintos de nós, denominados lugares e transições. Os lugares equivalem às variáveis de estado, e as transições correspondem às ações realizadas pelo sistema. Um arco dirigido liga os lugares às transições e pode ser classificado em único ou múltiplo. Quando os lugares estão conectados a uma transição, no sentido lugar-transição, tem-se o lugar de entrada, porém quando os lugares estão conectados a uma transição, no sentido transição-lugar tem-se o lugar de saída. Os elementos básicos de uma Rede de Petri, lugar, arco e transição podem ser observados na Figura §2.11.

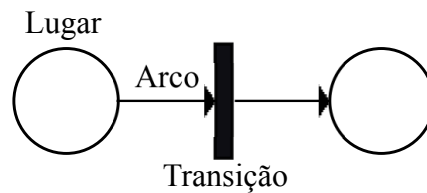


Figura 2.11: Elementos básicos de uma Rede de Petri. (Francês [26])

Uma Rede de Petri possui *tokens*, que permanecem alojados nos lugares. Durante a execução de uma Rede de Petri, uma transição será disparada somente quando estiver habilitada. Quando ocorrer o disparo da transição, ela irá consumir *tokens* dos lugares de entrada, sendo que estes *tokens* serão realocados nos lugares de saída, isto ocorre automaticamente. Os estados de um sistema, são modelados através do disparo de *tokens*, que se dirigem aos lugares, sendo a distribuição destes *tokens* denominada marcação [4].

Neste contexto, as transições podem consumir ou disponibilizar *tokens*, que irão permanecer nos lugares. Os arcos possuem origem em determinado lugar, e término em uma transição, o que significa que quando a transição disparar, um *token* será retirado do lugar. Uma transição só pode estar habilitada a disparar se houver ao menos, um *token* no lugar de entrada.

Os elementos básicos de uma Rede de Petri possuem diversas nomenclaturas e representações, sendo que os lugares podem ser chamados de estados

e representados por círculos. As transições podem ser denominadas como ações e representadas por quadrados, retângulos ou barras. Enquanto que os arcos são representados por uma seta direcionada [44]. As transições podem possuir diferentes disciplinas de disparo, como por exemplo, disparo imediato ou temporizado.

Existe mais de um tipo de Rede de Petri que pode ser utilizada na modelagem de sistemas, porém uma das mais usuais é a Rede de Petri estocástica. A Rede de Petri Estocástica, analisada por Roos-Frantz et al. [52] possui os seguintes elementos $(P, T, I, O, M_0, \Lambda)$. Quanto a estes elementos, P representa o conjunto dos lugares, T representa o conjunto das transições, I representa o conjunto dos arcos de entrada, O representa o conjunto dos arcos de saída, M_0 é a marcação inicial da rede, e Λ representa o conjunto das taxas de transição.

A taxa de transição nada mais é do que um atraso associado ao disparo das transições. Esse atraso é associado com o auxílio de uma variável aleatória X , que possui uma função chamada densidade de probabilidade exponencial negativa. A função pode ser representada por $F_{x_i}(x) = 1 - e^{-\lambda_i x}$, na qual $-\lambda_i$ é a taxa de disparo da transição t_i . A taxa de disparo depende da marcação e a média de atraso no disparo da transição t_i na marcação m_j , é representada por $\square \lambda_i(M_j) \square^{-1}$.

As Redes de Petri têm como enfoque três fundamentações diferentes, que levam a diferentes modelos formais, sendo que cada uma delas tem peculiaridades e utiliza diferentes teorias matemáticas como suporte [26]. A primeira utiliza a teoria *bag*, a segunda baseia-se nos conceitos da álgebra matricial e a última busca aporte teórico na estrutura definida por relações.

Na primeira, que utiliza a teoria *bag*, a Rede de Petri é composta por cinco partes: o conjunto de lugares P , o conjunto de transições T , o *bag* de entrada I , o *bag* de saída O e a capacidade associada a cada lugar. Para cada transição existe uma função de entrada, que é um mapeamento de uma transição t_j em um *bag* de lugares I t_j . De forma semelhante, as funções de saída mapeiam uma transição t_j em uma *bag* de lugares O t_j . Para denotarmos conjuntos, utilizamos " $\{ \}$ " e para *bags* " $[]$ ".

A segunda baseia-se nos conceitos da álgebra matricial, nesta definição a Rede de Petri R é uma quintupla $R = (P, T, I, O, K)$, onde P é um conjunto finito de lugares T é um conjunto finito de transições, $I: P \times T$ é a matriz de pré-condições, $O: P \times T$ é a matriz de pós-condições e K é o vetor das capacidades associadas aos lugares. Se o conjunto de lugares ou o conjunto de transições é vazio, a rede é dita degenerada.

A última busca aporte teórico na estrutura definida por relações, nesta definição a Rede de Petri $R = (P, T, A, V, K)$, composta pelo conjunto de lugares P , o conjunto de transições T , o conjunto dos arcos que interligam lugares às transições ou transições aos lugares, a valoração ou peso dos arcos representada por V e o conjunto das capacidades dos lugares K .

Estas teorias podem ser observadas de modo prático, através da Figura 2.12. Nesta figura, observa-se um modelo, no qual a condição atual, representando o horário do dia, é a manhã, pois o *token* (marca) está neste lugar. Quando a transição disparar, a única possibilidade é entardecer, pois o arco está ligado a esta transição. Neste sentido, quando este evento ocorrer, o *token* irá até a condição tarde.

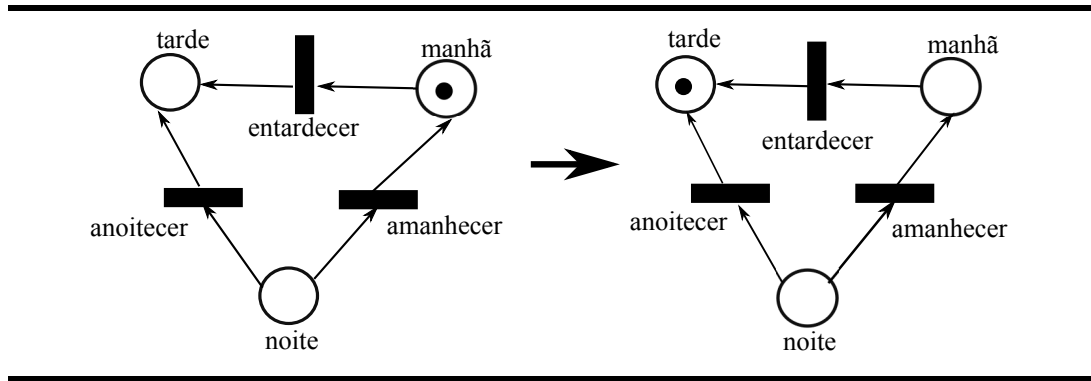


Figura 2.12: Representação de lugares e transições. (Maciel et al. [43])

Como exemplo de aplicação da teoria *bag*, pode-se utilizar a rede de períodos do dia. Esta rede pode ser formada por três lugares e três transições, que representam as variáveis de estados e as ações realizadas pelo sistema. As pré-condições para a realização das ações são representadas pelos *bags* $I(t_i)$. Desta forma, para que o amanhecer ocorra, é necessário que a condição noite seja verdadeira, sendo que $I(\text{amanhecer}) = \{\text{noite}\}$. Os lugares que são pós-condições das transições t_i , e podem ser representados pelos *bags* $O(t_i)$. A transição amanhecer tem como lugar de saída $O(\text{amanhecer}) = \{\text{manhã}\}$.

Considera-se a rede *períodos do dia*, que pode ser representada por (P, T, I, O, K) , utiliza-se o conjunto de lugares $P = \{\text{manhã}, \text{tarde}, \text{noite}\}$, o conjunto de transições $T = \{\text{amanhecer}, \text{entardecer}, \text{anoitecer}\}$, o conjunto de *bags* de entrada $I = \{I(\text{amanhecer}) = [\text{noite}], I(\text{entardecer}) = [\text{manhã}], I(\text{anoitecer}) = [\text{tarde}], \}$, o conjunto de *bags* de saída $O = \{O(\text{amanhecer}) = [\text{manhã}],$

$O(\text{entardecer})=[\text{tarde}]$, $O(\text{anoitecer})=[\text{noite}]$, e o conjunto de capacidades dos lugares $k=\{K_{\text{manhã}}=1, K_{\text{tarde}}=1, K_{\text{noite}}=1\}$.

A teoria que utiliza os conceitos da álgebra matricial pode ser representada, como no exemplo que pode ser observado na Figura §2.13. Observa-se que o conjunto de lugares é P e o conjunto de transições T . As matrizes são denominadas I e O , e representam as pré-condições e as pós-condições, das transições pertencentes à rede. A coluna amanhecer, presente na matriz I , demonstra que o lugar N , sendo a noite, é a pré-condição de amanhecer. Desta maneira, a coluna amanhecer, da matriz O , demonstra que o lugar manhã, é pós-condição da transição amanhecer. É possível observar nestas matrizes a estrutura do modelo, através da representação utilizando os conceitos da álgebra matricial.

$I=$	amanhecer	entardecer	anoitecer	manhã
	0	1	0	tarde
	0	0	1	noite
	1	0	0	

$O=$	amanhecer	entardecer	anoitecer	manhã
	1	0	0	tarde
	0	1	0	noite
	0	0	1	

Figura 2.13: Redes de Petri matriciais. (Maciel et al. [43])

A teoria que utiliza a estrutura definida por relações pode ser exemplificada no exemplo da rede períodos do dia, sendo definida como segue. Considera-se a estrutura da Rede de Petri, no exemplo da rede períodos do dia, da seguinte maneira: a rede *períodos do dia*, é composta por (P, T, I, O, K) , onde o conjunto P lugares será $\{\text{manhã}, \text{tarde}, \text{noite}\}$, o conjunto de transições T será $\{\text{amanhecer}, \text{entardecer}, \text{anoitecer}\}$, o conjunto de arcos A será $\{(\text{manhã}, \text{entardecer}), (\text{entardecer}, \text{tarde}), (\text{tarde}, \text{anoitecer}), (\text{anoitecer}, \text{noite}), (\text{noite}, \text{amanhecer}), (\text{amanhecer}, \text{manhã})\}$, o conjunto de valo-

rização dos arcos V será $\{1,1,1,1,1\}$ e o conjunto de capacidades dos lugares K será $K=\{K_{manha}=1, K_{tarde}=1, K_{noite}=1\}$.

Quanto à matemática nas Redes de Petri, Bressan [5] comenta que a notação matricial é uma das formas de representá-las. A Figura §2.14 auxilia na análise desta representação.

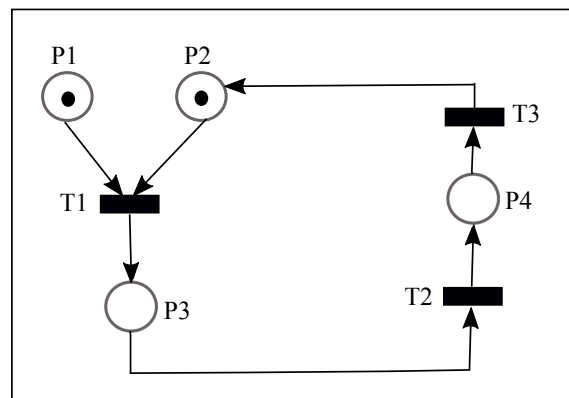


Figura 2.14: Exemplo de Rede de Petri em notação matricial. (Bressan [5])

Para compreender esta representação, é necessária a utilização de notações matriciais. As notações matriciais podem ser elaboradas com o auxílio de três tipos de matrizes, sendo elas a matriz de entrada, matriz de saída e matriz de incidência. A matriz de entrada pode ser representada por E , onde $ET \times P$ é a quantidade de arcos de entrada em cada transição. A matriz de saída pode ser representada por S , onde $ST \times P$ é a quantidade de arcos de saída em cada transição. A matriz de incidência, por sua vez, pode ser representada por I , onde $IT \times P = ST \times P$ (Matriz de Saída) $ET \times P$ (Matriz de Entrada).

A Figura §2.14, com o auxílio das notações matriciais, gerou as matrizes representadas na Figura §2.15:

A notação matricial torna possível modelar uma Rede de Petri graficamente, pois possibilita enumerar a quantidade de lugares, transições e arcos [13]. No entanto, estes elementos, não geram um modelo completo, para isso, é preciso utilizar os *tokens* e os pesos. Os *tokens* são a marcação inicial, e determinam o valor V_i . Sendo que V_i é representado por $(1, 1, 0, 0)$. O peso de cada arco é variável, mas no exemplo, este peso vale 1.

As Redes de Petri possuem uma equação fundamental, que descreve o comportamento das redes, e possibilita analisar suas propriedades comporta-

	P1	P2	P3	P4
T1	1	1	0	0
T2	0	0	1	0
T3	0	0	0	1

Matriz de Entrada

	P1	P2	P3	P4
T1	0	0	1	0
T2	0	0	0	1
T3	0	1	0	0

Matriz de Saída

	P1	P2	P3	P4
T1	-1	-1	1	0
T2	0	0	-1	1
T3	0	1	0	-1

Matriz de Incidência

Figura 2.15: Notação matricial da Rede de Petri. (Bressan [5])

mentais e estruturais [43]. A equação fundamental das Redes de Petri é a seguinte: $M'(p) = M_o(p) + C \bullet \bar{s}, \forall \pi \in P$

As Redes de Petri podem representar processos estocásticos quando existem tipos de evolução que envolvem tempo. Para analisar estes processos, é possível utilizar a probabilidade. Durante o processo de desenvolvimento de um sistema, muitas vezes é preciso representá-lo, de maneira mais ou menos detalhada [43] e [26]. Pode-se utilizar nesta representação uma forma compacta, pois, de acordo com as dimensões do sistema, sua representação pode não ser prática. As Redes de Petri podem ser classificadas de acordo com o seu nível de abstração, em ordinárias e não ordinárias [44]. As ordinárias, também conhecidas como Redes de Petri de baixo nível, caracterizam-se pelas suas marcas, que devem ser sempre do tipo inteiro e não negativo. As não ordinárias, também conhecidas como Redes de Petri de alto nível são caracterizadas pelos tipos de suas marcas, que não são elementos do tipo inteiro positivo. Este tipo de rede permite individualizar uma marca (que sempre pertence a um grupo) em um mesmo lugar, sendo que esta semântica está relacionada com o fato de serem atribuídos valores ou cores às marcas, ou utilizar tipos de dados abstratos.

As Redes de Petri de alto e baixo nível não aumentam o poder de representação de um modelo, mas permitem uma maior ou menor clareza quanto ao nível de abstração do modelo, sendo que também existem as extensões hi-

erárquicas e temporais que podem ser associadas a qualquer uma delas. Dentre estas, merecem destaque as Redes de Petri coloridas e as Redes de Petri temporizadas, que serão melhor detalhadas abaixo.

2.5.1 Redes de Petri Coloridas

Na década de noventa, as Redes de Petri tiveram como maior representante as Redes de Petri coloridas, que foram criadas por Kurt Helmer Jenses, na Dinamarca [44]. Estas redes utilizam marcas diferenciáveis e, através de aspectos temporais, possibilitam representar dados abstratos. Sua utilização tem grande importância, pois as mesmas são capazes de modelar sistemas grandes e complexos, tornando-os mais simples, pois possuem inúmeros recursos, o que gera uma redução no tamanho dos modelos, deixando-os mais claros e concisos. Isto permite que *tokens*, representados por marcas individualizadas (cores), representem diferentes processos, ou recursos em uma mesma sub-rede, as marcas são identificadas pelas cores e podem ser estruturas complexas. Nas Redes de Petri coloridas, as marcas possuem tipos de dados, e existe a possibilidade de serem efetuadas operações com os mesmos [43].

As Redes de Petri coloridas tem grande importância, em conjunto com os Objetos de Aprendizagem, conhecidos como OAs e isso é creditado ao fato de que as Redes de Petri coloridas permitem formalizar processos. Deste modo, realizam uma maior inserção de características necessárias à elaboração desses processos, além de possibilitarem a descoberta de possíveis erros, ainda na fase de modelagem [16].

Este tipo de rede possui em sua estrutura básica três partes, denominadas: estrutura, declarações e inscrições [26]. A estrutura é um grafo dirigido composto por dois tipos de nós, um denomina-se lugar (sua representação pode ser feita por círculos ou elipses) e o outro denomina-se transição (sua representação pode ser feita por retângulos). É importante ressaltar que existem arcos interconectando tipos de nós diferentes. As declarações, por sua vez, contemplam o conjunto de cores e variáveis. As inscrições podem sofrer variações de acordo com os componentes da rede.

Visando diferenciar os *tokens*, associam-se cores ao mesmos, sendo que estas cores podem ser números inteiros [7]. Portanto, cada lugar recebe um conjunto de cores que corresponde aos *tokens* deste lugar, e cada transição recebe um conjunto de cores que corresponde às diferentes maneiras, que podem servir para disparar uma transição. Em modelos mais simples, onde todos os processos possuem a mesma estrutura e são independentes uns dos

outros, as cores das transições são associadas diretamente aos processos e o conjunto de cores correspondente aos lugares e as transições são idênticos.

Para que seja possível uma melhor compreensão sobre a distribuição das cores, toma-se como exemplo, a Figura §2.16, proposta por Francês [26], que mostra a maneira mais simples de uma aplicação de Rede de Petri colorida. Pode-se observar que o arco é associado a uma determinada cor, deste modo, e o *token* será destinado ao arco que possui uma cor idêntica à da marca. Percebe-se que os *tokens* de P_0 não habilitarão a transição t_0 , porque o arco que liga P_0 a t_0 só aceita marcas do tipo "a", e o lugar P_0 só possui marcas do tipo "d", porém, o lugar P_1 possui marcas do tipo "a", podendo assim, habilitar a transição t_1 .

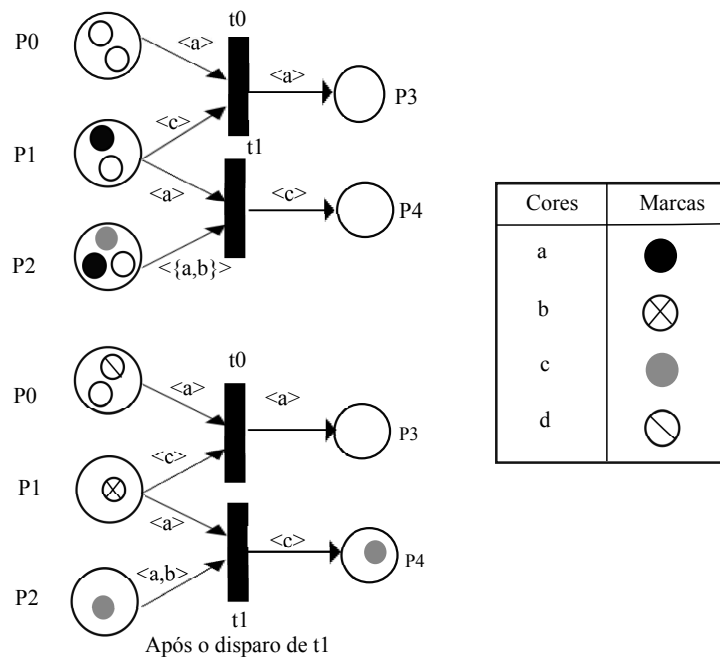


Figura 2.16: Rede de Petri colorida. (Maciel et al. [43])

2.5.2 Redes de Petri Temporizadas

As Redes de Petri temporizadas, propostas por Ramchandani [50], Merlin e Farber [46] e Sifakis [55], possuem um formalismo que possibilita expressar o comportamento dinâmico de sistemas, que desempenhem atividades concorrentes, assíncronas e não-determinísticas, considerando a variável

tempo [44]. O conceito de tempo não estava presente na definição original das Redes de Petri, porém quando levantou-se a necessidade de especificar sistemas com tempo real, assim como realizar avaliação de desempenho e problemas de escalonamento, foi necessária a utilização de retardos de tempo. A partir desta necessidade, criaram-se as Redes de Petri temporizadas.

Estudos realizados por Lima et al. [38], envolvendo as Redes de Petri temporizadas, constataram que elas são uma das mais eficientes ferramentas para modelagem de sistemas concorrentes, com restrições de tempo associadas à ocorrência de eventos. Nas Redes de Petri temporizadas, cada ocorrência de evento é associada a um intervalo de tempo, sendo que os limites inferior e superior desse intervalo de tempo, determinam o instante mínimo e máximo para a ocorrência do evento. Desta maneira, torna-se possível modelar restrições de tempo, em que o instante exato da ocorrência é desconhecido. Em sistemas reais, esta situação é bastante comum, pois na prática, especificam-se limites de tempo quando se desconhece seu valor exato. O intervalo que delimita o instante de ocorrência de cada evento será denominado domínio de tempo dessa ocorrência.

Na extensão temporizada, algumas relações podem ser descritas, dentre elas pode-se destacar que existe a possibilidade do tempo ser associado às transições, arcos, lugares e *tokens*.

Quando o tempo é associado aos *tokens*, eles transmitem uma informação que indica, se estão disponíveis para habilitar as transições. O *token* deve aguardar o *timestamp*, que indica quando uma transição pode ser disparada. Destaca-se que o *timestamp* pode ser incrementado ao disparo de uma transição.

Quando o tempo está associado aos arcos, cada arco está diretamente ligado a um determinado tempo de disparo, denominado *traveling delay*. Nesse caso, os *tokens* devem ficar indisponíveis até que a transição seja disparada.

Quando o tempo é associado aos lugares, há a indicação que o *token* deve permanecer neste lugar, antes de ser utilizado para a habilitação das transições, que ocuparão o lugar posterior. Desse modo, os *tokens* ficam disponíveis nos lugares de saída, após a passagem de um tempo pré-determinado. Os *tokens* podem ser classificados como disponíveis e indisponíveis.

Quando ocorre a associação do tempo com as transições, considera-se o tempo que a ação deverá demorar para ser executada. A transição estará

habilitada a disparar, dentro de um determinado intervalo, ou em um determinado tempo constante.

Destaca-se que, nas Redes de Petri temporizadas, para que uma transição habilitada possa ser disparada, é necessário que esta seja disparável, para isso, existe um contador, que está associado à transição e vai sendo decrementado a uma taxa constante, sendo que quando este chega a zero, a transição dispara. Além disso, é essencial que existam *tokens* no lugar de entrada, referente a esta transição.

O comportamento de uma Rede de Petri temporizada é completamente caracterizado por uma sequência, finita ou infinita, de estados, transições e atrasos, que derivam do estado inicial [39]. Sendo o tempo uma quantidade contínua, o conjunto de estados presentes em uma Rede de Petri temporizada será sempre infinito. Considerando um estado inicial, pode-se analisar através do disparo de uma sequência, possíveis transições que serão habilitadas. A Figura §2.17 apresenta um conjunto de transições, denominadas por T_h , que são habilitadas pela marcação, baseadas nas Redes de Petri temporizadas. A partir desta, é possível determinar o intervalo de tempo necessário para que as transições t_3 e t_4 sejam disparadas.

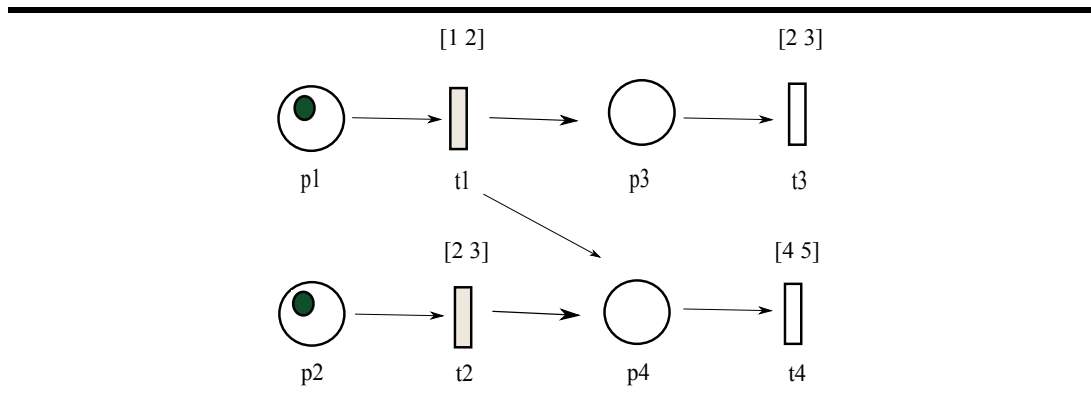


Figura 2.17: Rede de Petri temporizada. (Lima et al. [39])

Pode-se observar que, enquanto uma transição está decrementando o tempo necessário para que possa habilitar-se, os *tokens* permanecem nos lugares de entrada, até que o tempo associado à transição tenha terminado, e ela esteja habilitada. Quando isso ocorre, os *tokens* são removidos destes lugares, e são armazenados nos lugares de saída. Lima et al. [39] afirmam que apesar de uma transição estar habilitada para disparar, isto pode não ocorrer.

rer, pois o disparo de outra transição que também está habilitada para disparar, pode interferir ou impedir o disparo da outra.

As extensões de tempo, denominadas temporais, podem ser divididas em determinísticas e estocásticas [44]. As determinísticas surgiram na década de setenta e trabalham com tempos absolutos, relativos à execução dos eventos correspondentes. Já as estocásticas consideram incertezas no instante de execução dos eventos correspondentes ao sistema, associando a ele funções de probabilidade, que permitem a determinação do tempo de execução.

Lima et al. [38] tratam da definição das Redes de Petri. Para os autores, formalmente, uma Rede de Petri temporizada é uma sêxtupla $RPT = (P, T, Pre, Post, FI, M_0)$. Sendo,

- P o conjunto de lugares;
- T o conjunto de eventos (ou transições);
- $P \cap T = \emptyset$
- $Pre : P \times T \rightarrow \mathbb{N}$, é a matriz de entrada, sendo \mathbb{N} o conjunto dos números naturais (peso dos arcos de entrada das transições);
- $Pos : T \times P \rightarrow \mathbb{N}$, é a matriz de saída, sendo \mathbb{N} o conjunto dos números naturais (peso dos arcos de saída das transições);
- M_0 , é a marcação inicial da rede;
- $Fi : T \rightarrow \mathbb{IR}$, a função de intervalo estático;

A função FI associa a cada transição $t_i \in T$ da rede, um intervalo de tempo $i^s(t_i) = [\sigma^s, \Delta^s]$, denominado domínio de tempo estático da transição, sendo o vetor i^s , denominado vetor domínio de tempo estático. Os limites σ^s e Δ^s são chamados limite inferior de ocorrência (LIO) e limite superior de ocorrência (LSO), respectivamente.

No que diz respeito à habilitação de transições Lima et al. [38], tratam do assunto ao dizerem que nas Redes de Petri tradicionais, quando uma transição estiver habilitada, significa que o evento associado a ela, pode ocorrer a qualquer instante e sua ocorrência é caracterizada pelo disparo da transição. Porém, nas Redes de Petri temporizadas, o fato de uma transição t estar habilitada, não significa necessariamente que ela será disparada. Primeiro, é preciso que a transição permaneça habilitada por um tempo mínimo

igual ao LIO, a partir do qual ela pode ser disparada. Por outro lado, caso a transição permaneça habilitada até o instante igual ao LSO, então necessariamente deverá ser disparada. Dessa forma, comparada às Redes de Petri tradicionais, nas Redes de Petri temporizadas, a indeterminação de uma ocorrência é reduzida pela obrigatoriedade do disparo da transição. Portanto, para uma descrição completa do estado de uma Rede de Petri temporizada, é necessária a marcação atual da rede e uma informação a respeito do tempo de habilitação das transições. O conjunto formado por todas as transições habilitadas pela marcação M será denotado H . Outro fator importante na habilitação das Redes de Petri temporizadas, é o fato de que, se uma transição permanece habilitada após seu disparo, então sua temporização deve ser reinicializada, inviabilizando o conceito de múltipla habilitação.

2.6 Trabalhos Relacionados

As Redes de Petri, cada vez mais, tem sido estudadas em variadas áreas do conhecimento, no que tange a modelagem e a simulação de sistemas de eventos discretos. Uma solução de integração do mesmo modo, pode ser caracterizada como um sistema de eventos discretos. Isso tem ocorrido porque as Redes de Petri são uma técnica matemática flexível, no que diz respeito a situações que envolvam a modelagem. Possui uma notação gráfica, que possibilita observar o desempenho de sistemas, e dispõe de uma forte base matemática, permitindo estudos minuciosos, com relação ao comportamento dos sistemas, visando encontrar possíveis falhas, de forma a melhorá-lo. Nesse sentido, na sequência, estão descritos trabalhos relacionados à utilização das Redes de Petri, e sua relação com a simulação de sistemas de eventos discretos. Os trabalhos relacionados abaixo retratam sistemas, que possuem características específicas, sendo simulados, com o objetivo de encontrar gargalos de desempenho. Desse modo, ressalta-se que as Redes de Petri têm importância em diversos campos do conhecimento, podendo ser utilizadas de diversos modos.

O trabalho desenvolvido por Junqueira e Miyagi [34] propõe a modelagem e a simulação de sistemas produtivos distribuídos, com base nas Redes de Petri, por ser considerado um recurso fundamental para o projeto, possibilitando implementação e melhorias de desempenho em sistemas produtivos. Os autores consideram que sistemas distribuídos possuem uma grande capacidade computacional, além de disporem de estruturas diferentes, de outros sistemas utilizados para produção. Dessa forma, pretende-se por meio de simulações, e com a utilização de computadores separados fisicamente, porém integrados com o auxílio de uma rede de comunicação, avaliar o comportamento de sistemas, visando melhorar os resultados já existentes.

Neste sentido, Junqueira e Miyagi [34], propõem a utilização de um procedimento para a modelagem de sistemas produtivos em ambientes distribuídos. Destaca-se que este procedimento foi aplicado com êxito em estudos de caso, nos quais a eficácia deste procedimento foi confirmada. Este trabalho, também envolve, a proposta de utilização de um algoritmo para gerenciamento de simulações distribuídas. Neste sentido, o autor apresentou uma nova abordagem para a modelagem de sistemas distribuídos, baseada no conceito de orientação a objeto, com auxílio do formalismo Redes de Petri, associado a um procedimento de refinamento progressivo.

O trabalho desenvolvido por Facchin e Sellitto [23] apresenta um método para medição de inventário em processo, e tempo de atravessamento em um sistema de manufatura. Os autores elucidam que este método consiste em modelar a manufatura, utilizando o formalismo matemático Redes de Petri, com o intuito de simular o modelo em um computador. Este computador estará sendo alimentando com a carga inicial dos processos e com um plano de produção, de modo a obter em determinados momentos da simulação, conclusões sobre o processo de manufatura.

No trabalho desenvolvido por Facchin e Sellitto [23], foram realizados os procedimentos citados acima, sendo que depois destes, foi elaborado um diagrama de resultados. Foi calculado o valor médio simulado, referente ao inventário em processo, sobre o plano de manufatura. Além disso, ao final, realizou-se uma discussão, na qual explorou-se de que modo os resultados do método podem ser úteis em decisões de gestão, envolvendo o inventário admitido e restrições do processo de manufatura.

Em seu trabalho, de Carvalho et al. [15] aborda a importância da mão de obra multifuncional, que pode ser observada em empresas onde a automação é reduzida, e a competitividade exige uma política de desempenho diferenciada. O autor destaca que o formato das linhas de produção possui papel fundamental, no que diz respeito ao melhor aproveitamento do trabalho multifuncional. Em decorrência desse fato, o autor realizou estudos, que questionam os efeitos da aplicação de linhas de produção em forma de U, como alternativa às linhas de produção em linha reta.

No trabalho elaborado por de Carvalho et al. [15], observou-se que a modelagem do comportamento polivalente, com auxílio de ferramentas de modelagem e simulação convencionais é bastante complexa, exigindo novas abordagens. A eficácia das Redes de Petri, na modelagem de linhas de produção é comprovada, tanto pela capacidade de visualização gráfica quanto pela análise matemática facilitada. O objetivo do trabalho é aplicar o formalismo Redes de Petri temporizadas na modelagem e simulação de linhas de

produção em forma de U, com trabalho multifuncional. Para o desenvolvimento deste trabalho, foram utilizados aplicativos, que auxiliaram na construção dos modelos, análise de propriedades, simulação e avaliação de parâmetros de desempenho. Na pesquisa, considerou-se a análise da dinâmica de linhas de produção, levando em consideração os efeitos da diferença entre taxas de produção entre postos de trabalho, gerando, estoques intermediários. Observou-se que as Redes de Petri constituem uma ferramenta bastante eficiente no controle de parâmetros produtivos, o que pode resultar no aumento da produtividade dos operadores polivalentes, considerando o dimensionamento adequado de estoques intermediários.

O foco do trabalho desenvolvido por Lima et al. [38] introduz o conceito das Redes de Petri temporais, destacando o fato de serem uma das extensões das Redes de Petri, utilizadas para modelagem e análise de sistemas e eventos discretos temporizados. O autor descreve seu trabalho ao dizer que, para cada transição, um intervalo de tempo será associado, pois desta forma será possível delimitar, quando o disparo da transição irá ocorrer. Destaca-se que a análise das Redes de Petri, no trabalho desenvolvido, será realizada com o auxílio de métodos enumerativos. O autor salienta que serão utilizados recursos de álgebra e conceitos de classe de estados, de modo que seja apresentada uma nova abordagem para resolver problemas, relacionados à caracterização de intervalos de tempo de disparo, nas Redes de Petri temporais.

No trabalho elaborado por Lima et al. [38], foi utilizada uma expressão algébrica, desenvolvida com o intuito de calcular intervalos de tempo, entre duas classes de estados consecutivas. Deste modo, obtém-se uma equação intervalar, que possibilita o cálculo de intervalos de tempo, entre duas classes de estados quaisquer. Assim, o enfoque do trabalho é calcular o intervalo de tempo para ocorrência de uma sequência de eventos, com auxílio de métodos de redução da rede. Para isso, a equação intervalar assume uma forma matricial e, desta forma, amplia a classe das Redes de Petri temporais à qual a equação intervalar é aplicável. Neste sentido, foi desenvolvida uma aplicação explorando as potencialidades, da equação intervalar no cálculo de medidas características, de um sistema de eventos discretos temporizados. A partir da análise intervalar, foi possível obter resultados, tais como a redução do espaço de classes de estados de uma Rede de Petri temporal (delimitando um espaço de busca) e a definição de um conjunto de regras, para redução dessas redes, transformando a rede original em uma rede reduzida, que preserva as restrições temporais e a viabilidade de ocorrência das transições.

A dissertação proposta por Cargnin [8] está inserida na EAI, que trata da integração das aplicações que existem no ecossistema de *software* das empresas, com a utilização de uma solução de integração. A pesquisa proposta pelo

autor visa analisar o comportamento e a identificação de possíveis gargalos de desempenho em uma solução de integração, ainda na fase de projeto. O autor desenvolveu um modelo matemático de simulação, que é equivalente ao modelo conceitual da solução, sendo que para isso utilizou, as Redes de Petri estocásticas.

No trabalho desenvolvido por Cargnin [8], observa-se que é possível representar modelos conceituais, de soluções de integração de aplicações, com auxílio das Redes de Petri estocásticas. Destaca-se que foi possível desenvolver um modelo de simulação, onde indentificaram-se pontos de aglomeração de mensagens, formando filas, que são possíveis gargalos de desempenho. Além disso, analisou-se o comportamento da solução de integração ainda na fase de projeto.

O trabalho apresentado por Yamada et al. [62] tem como proposta realizar simulações nas etapas de funcionamento de uma indústria de beneficiamento de cana de açúcar, com auxílio das Redes de Petri, dando ênfase aos sistemas de recebimento, descarregamento e movimentação de matéria prima até o processo de extração. A simulação desses processos tem por objetivo analisar propostas de otimização e identificação de pontos vulneráveis sem interferir no funcionamento do sistema. O sistema de recebimento de cana de açúcar e da moagem do mesmo, possui características discretas, onde o tempo de execução de cada processo interfere no comportamento do sistema.

A simulação do modelo proposto por Yamada et al. [62] apontou a presença de paralelismo de atividades, que gera uma sobrecarga na alimentação da central de moagem, quando existem chegadas sucessivas de matéria-prima. Alguns recursos, como descarregamento, são compartilhados entre os setores, e por esse motivo, torna-se necessário sincronizar as atividades, pois é necessária uma alimentação constante de matéria-prima para beneficiamento para que não haja comprometimento na continuidade do processo. O modelo definiu as variáveis do sistema, porém não considerou suas características determinísticas e estocásticas, então, para trabalhos futuros, o autor sugere a utilização de Redes de Petri temporizadas para distinguir cada etapa do processo e consolidar a validação do modelo.

Em seu trabalho, Miyagi et al. [47] realizaram a modelagem de sistemas de saúde, sendo que as Redes de Petri foram aplicadas nos serviços do ambulatório do Hospital das Clínicas (HC) de São Paulo, visando analisar o caso de estudo e validar a proposta. A modelagem visa estudar o fluxo de pacientes que procuram atendimento no ambulatório, considerando pacientes que não possuem urgência no atendimento e, principalmente, pacientes que

procuram atendimento pela primeira vez, buscando facilitar a tomada de decisão pela gerência do sistema, e também analisar a evolução do sistema e o comportamento dos setores. O modelo foi desenvolvido utilizando Redes de Petri, que utilizam os recursos envolvidos e a movimentação dos componentes do sistema, nos quais detalhes do sistema vão sendo gradativamente inseridos no modelo. Os dados colhidos consideram a identificação do fluxo de pacientes, e com base nas informações obtidas com os funcionários, tornou-se possível desenvolver o modelo, enfatizando pacientes que procuram atendimento pela primeira vez.

A simulação do modelo proposto por Miyagi et al. [47] considerou alterações no fluxo de pacientes, além de identificar a possibilidade de eliminar alguns processos que pudessem sobrecarregar o sistema. Por fim, o trabalho de modelagem e simulação evidenciou que a teoria de Redes de Petri exige um conhecimento superficial, em comparação a outras técnicas matemáticas, pois tem-se auxílio do recurso da animação do fluxo, disponível no simulador, que permite analisar determinados setores em um instante qualquer, possibilitando avaliar a situação.

O trabalho apresentado por Roos-Frantz et al. [52] propõe a simulação de modelos conceituais de soluções de integração desenvolvidos com a tecnologia Guaraná, modelados com Redes de Petri, que possibilitou observar o comportamento do modelo anteriormente à sua implementação. A tecnologia Guaraná oferece um conjunto de ferramentas e metodologias que possibilita desenvolver modelos conceituais de soluções de integração, utilizando, para isso, uma linguagem DSL. O objetivo do trabalho é mostrar que o modelo conceitual desenvolvido no Guaraná pode ser traduzido em um modelo formal usando Redes de Petri estocásticas, e este pode ser simulado para obter informações sobre o desempenho da solução de integração, e possíveis gargalos de desempenho. Cada elemento do modelo conceitual do Guaraná é traduzido em um grafo em Redes de Petri, sendo realizada a verificação do modelo da equivalência de ambos. O estudo de caso é fundamentado em um problema real de integração que propõe automatizar o processo de registro de novos usuários de um sistema, de modo a integrar aplicações independentes, que estão implementadas em plataformas diferentes, e que geralmente são desenvolvidas sem a preocupação de integração.

A simulação é abordada por Roos-Frantz et al. [52] como maneira de minimizar custos e diminuir riscos de falhas, quando trata-se da integração de aplicações, visando analisar características da solução de integração, na fase de projeto. O trabalho demonstrou que os modelos conceituais desenvolvidos com o Guaraná podem ser traduzidos para Redes de Petri Estocásticas e

posteriormente simulados, de modo a identificar características e analisar seu comportamento, em diferentes cenários. Os autores consideram que a abordagem sobre simulação que o trabalho propõe pode melhorar a qualidade de soluções de integração de aplicações empresariais desenvolvidas com o Guaraná.

2.7 Resumo do Capítulo

Neste capítulo, foram abordados os conceitos fundamentais para o desenvolvimento desta dissertação. Neste contexto, foi realizada a revisão da literatura, na qual aprofundaram-se os estudos referentes a linguagem de domínio específico da tecnologia Guaraná, que possibilita projetar soluções de integração com um alto nível de abstração. Foram abordados os conceitos referentes à análise de um sistema por meio da simulação de eventos discretos de modelos computacionais. Em seguida, foram abordados os conceitos inerentes às distribuições de probabilidade. Por último, discutiu-se o conceito de Redes de Petri, elementos básicos e teorias matemáticas que dão suporte à elas, destacando-se a abrangência das Redes de Petri temporizadas, que foram utilizadas nesta dissertação. Por fim, fez-se um recorte de trabalhos relacionados que foram utilizados nesta dissertação.

Capítulo 3

Modelagem

*A menos que modifiquemos a nossa maneira de pensar
não seremos capazes de resolver os problemas
causados pela forma como nos
acostumamos a ver o mundo.*

Albert Einstein

Este capítulo possui como base um modelo conceitual de uma solução de integração, que corresponde à administração pública de Huelva (Espanha), desenvolvida com a tecnologia Guaraná, e propõe a modelagem formal do modelo conceitual, com o auxílio da técnica matemática Redes de Petri. A Seção §3.1 discorre acerca do caso de estudo, que abrange um modelo da solução de integração para o problema do caso de estudo. A Seção §3.2 apresenta a equivalência entre os elementos das Redes de Petri e a tecnologia Guaraná. A Seção §3.3 introduz um modelo de simulação, elaborado com o formalismo matemático Rede de Petri temporizada, equivalente ao modelo conceitual. Para concluir, a Seção §3.4 apresenta o resumo do capítulo.

3.1 Caso de Estudo

Esta seção contém um modelo da solução de integração para um problema de integração de aplicações, no contexto da administração pública

de Huelva (Espanha). O caso de estudo da administração de Huelva serviu de base para o desenvolvimento da dissertação. O objetivo da solução de integração é analisar o comportamento e o desempenho de uma solução de integração, que gera certificados digitais e unifica as bases de usuários que possuem acesso aos sistemas informáticos da administração. O caso de estudo referente à solução de integração foi transcrito em um modelo conceitual, que foi desenvolvido usando o DSL do Guaraná. Na sequência, será apresentado o ecossistema de *software* da solução de integração, e posteriormente será realizada uma descrição detalhada do modelo conceitual, considerando o fluxo de trabalho das mensagens na solução.

3.1.1 Ecossistema de Software

A solução de integração envolve seis aplicações, são elas: os usuários locais, os usuários do portal, LDAP, sistema de recursos humanos, plataforma de certificado digital e servidor de *e-mail*. Cada aplicativo é executado em uma plataforma diferente, e com exceção do LDAP, a plataforma de certificado digital e o servidor de *e-mail*, não foram projetados com preocupações de integração em mente.

Os usuários locais é a primeira aplicação, sendo que foi desenvolvida internamente e destina-se a gerenciar informações dos usuários do sistema da administração pública de Huelva (Espanha). Observa-se que esta é uma aplicação autônoma e não fornece serviços de autenticação. Os usuários do portal representam uma aplicação disponível que o Portal *Web* usa para gerenciar seus usuários. Além disso, um único repositório para os usuários foi criado usando uma aplicação baseada em LDAP, para que possa fornecer controle de acesso de autenticação, para várias outras aplicações dentro do ecossistema de *software*. O sistema de recursos humanos é um sistema legado, desenvolvido internamente para fornecer informações pessoais sobre os funcionários. É uma parte da solução de integração que solicita informações, como nome e *e-mail* para compor notificações de *e-mails*. Outra aplicação desenvolvida internamente é a plataforma de certificado digital, que visa gerenciar certificados digitais, e foi projetada com a preocupação da integração. Dentre outros serviços, essa aplicação pode ser consultada para obter uma URL que designa temporariamente um certificado digital, que os usuários podem baixar depois de estar autenticado. Finalmente, o servidor de *e-mail* executa o serviço de *e-mail* da administração, sendo utilizado exclusivamente para fins de notificação.

3.1.2 Modelo Conceitual de Integração

A solução de integração proposta foi desenvolvida utilizando o DSL da tecnologia Guaraná e é composta por um processo de orquestração, que coordena as aplicações envolvidas na solução de integração. A Figura 3.1 mostra o modelo conceitual desta solução de integração, proposto por Roos-Frantz et al. [52]. Algumas portas utilizam arquivos de texto para comunicar-se com os usuários locais, os usuários do portal, e o LDAP. O sistema de recursos humanos é consultado por meio de seu sistema de gestão de banco de dados, e a comunicação com a plataforma de certificado digital e o servidor de e-mail é efetuada por meio de APIs. As tarefas do tradutor foram usadas para traduzir mensagens do esquema canônico para esquemas com os quais as aplicações trabalham.

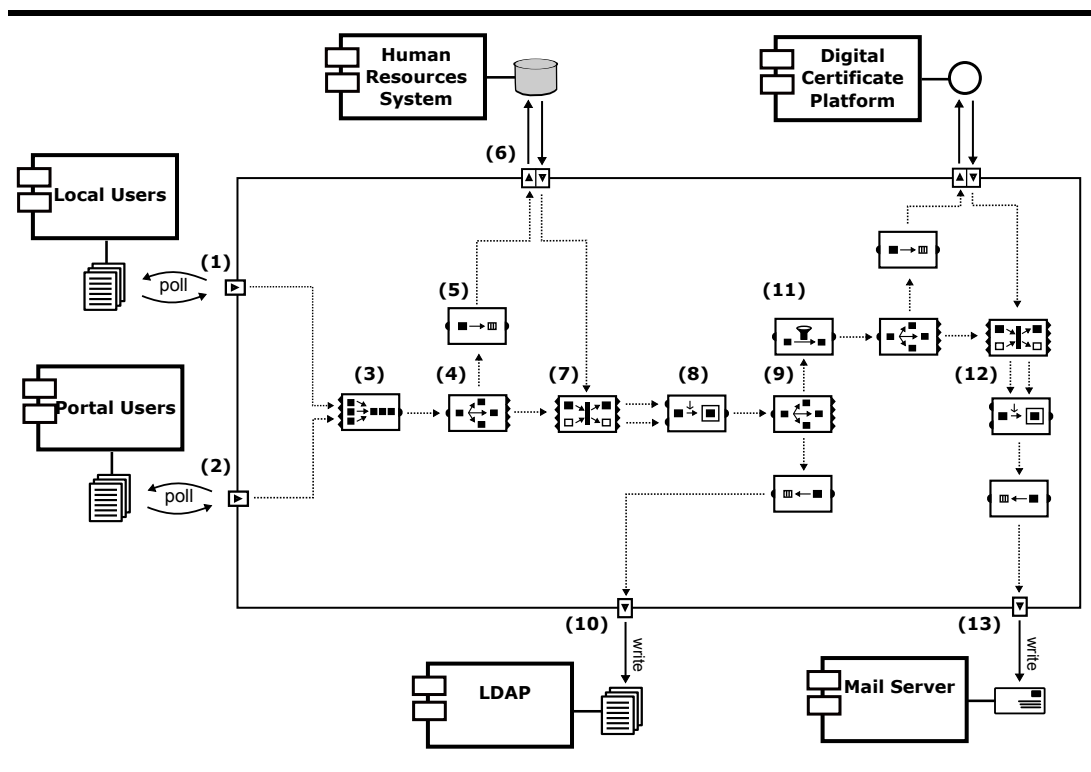


Figura 3.1: Modelo conceitual da solução de integração. (Roos-Frantz et al. [52])

O fluxo de mensagens começa nas portas de entrada (1) e (2), que periodicamente consultam registros de eventos dos usuários locais e usuários do

portal para encontrar novos usuários. Mensagens de entrada envolvendo dados que foram sondados são escritas por essas portas. Dentro do processo, a tarefa (3) recebe mensagens provenientes de ambas as portas, sendo uma tarefa de fusão, que não altera o conteúdo da mensagem. A próxima tarefa do fluxo, é a (4), ela é um *replicator* que cria duas cópias de cada mensagem recebida, de modo que uma cópia pode ser usada para consultar aplicações do sistema de recursos humanos por meio da porta de solicitação (6) para obter informações sobre o empregado que possui um registro de usuários, e a outra cópia (chamada mensagem de base) é correlacionada com a tarefa (7) e com a resposta da aplicação sistema de recursos humanos. A tarefa (5) é um *translator* que muda o esquema da mensagem para criar uma mensagem pergunta. Em seguida, a tarefa (8) enriquece a mensagem de base com as informações contidas na mensagem de resposta, e então a tarefa (9) replica esta mensagem enriquecida, com cópias para o LDAP e a plataforma de certificado digital. O novo registro de usuários é escrito para o LDAP por meio da porta de saída (10). Antes de consultar a plataforma de certificado digital, a tarefa (11) filtra as mensagens que não possuem um endereço de *e-mail* associado a elas. As mensagens que passam pela tarefa (12), são enriquecidas com o certificado digital correspondente. Finalmente, a porta de saída (13) comunica-se com a aplicação servidor de *e-mail*, para enviar o certificado digital e notificar o funcionário sobre sua inclusão no LDAP.

3.2 Equivalência entre Redes de Petri e Guaraná

Os modelos conceituais, que representam uma solução de integração, são desenvolvidos com a tecnologia Guaraná DSL, e podem ser traduzidos em um modelo de simulação através do formalismo matemático Redes de Petri. Porém, é necessário conhecer a equivalência entre seus componentes. A Tabela §3.1 demonstra a equivalência dos elementos do Guaraná DSL, sendo eles tarefas, *slots* e mensagens, com os elementos das Redes de Petri, sendo eles, transições, lugares e *tokens*.

A solução de integração considerada nesta pesquisa, foi estudada visando analisar seu comportamento e desempenho, sendo que gera certificados digitais e unifica as bases de usuários que possuem acesso aos sistemas informáticos da administração pública de Huelva (Espanha). Esta solução de integração é um sistema de eventos discretos, e seu modelo conceitual foi desenvolvido com a tecnologia Guaraná DSL. Utilizando a equivalência, foi desenvolvido nesta pesquisa um modelo de simulação, que pode ser representado com auxílio do formalismo matemático Redes de Petri temporizadas.





Guaraná DSL	Redes de Petri
Tarefa 	Transição 
Slot 	Lugar 
Mensagem 	Token 

Tabela 3.1: Equivalência entre elementos. (Roos-Frantz et al. [52])

Quanto à composição básica das Redes de Petri, em comparação com o Guaraná DSL, os arcos ligam lugares a transições, além de indicarem o sentido do fluxo das mensagens. As tarefas, possuem uma notação gráfica diferenciada, sendo possível realizar a equivalência com o Guaraná DSL, através de um grafo. Com relação à equivalência das tarefas, considerando o seu grafo, observa-se que elas possuem diferenças no que tange a sua classificação, sendo que podem ser modificadoras, roteadoras, transformadoras ou temporais.

A Tabela §3.2 apresenta as tarefas modificadoras. Uma tarefa modificadora possui como característica modificar o conteúdo de uma mensagem de entrada. A Tabela §3.3 apresenta as tarefas roteadoras. Uma tarefa roteadora é responsável por redirecionar uma mensagem de entrada, para seu destino. A Tabela §3.4 apresenta as tarefas transformadoras. Uma tarefa transformadora possui a capacidade de alterar uma mensagem, com relação a sua estrutura, ou construir nova mensagem. Por fim a Tabela §3.5 apresenta as tarefas temporais. Uma tarefa temporal pode alterar o tempo de execução de uma mensagem.

Observando a equivalência entre as Redes de Petri e o Guaraná DSL, é possível perceber que sempre que uma transição for disparada, a Rede de Petri terá seu estado alterado [52]. Com esta informação, é possível realizar a equivalência da troca de estados, conforme observa-se na Figura §3.2, no que tange uma solução de integração desenvolvida com o Guaraná DSL, e a troca de estados que ocorre em uma solução de integração que utiliza as Redes de Petri.

Quanto ao disparo de estados das Redes de Petri, observa-se que quando os *tokens* chegam até o lugar de entrada, ficam aguardando a transição ser disparada, para que então possam ser realocados no lugar de saída. A transição será disparada somente quando estiver habilitada para isso. O arco pode

Nome da Tarefa	Guaraná	Redes de Petri
Slimmer		
Context-based Slimmer		
Content Enricher		
Context-based Content Enricher		
Header Enricher		
Context-based Header Enricher		
Header Promoter		
Header Demoter		
Set Correlation ID		

Tabela 3.2: Equivalência das tarefas modificadoras. (Roos-Frantz et al. [52])

definir quantos *tokens* serão realocados no lugar de saída. No que tange ao lugar de entrada, as mensagens seguem uma disciplina de processamento. Esta dissertação está considerando a disciplina de processamento FIFO, onde a primeira mensagem que entra, é a primeira mensagem que sai.

Analisando a equivalência, quanto ao disparo de estados no Guaraná DSL, observa-se que as mensagens que chegam até o *slot* de entrada, aguardam a execução da tarefa, para então serem processadas, e realocadas no *slot* de saída. A política de disparo da tarefa determina quantas mensagens irão seguir o fluxo, cada vez que a mesma for executada. Dentro do *slot*, as mensagens também seguem uma disciplina, porém é uma disciplina de processamento.

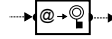
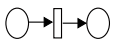

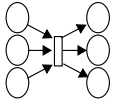

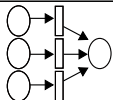

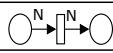

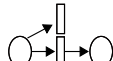

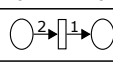

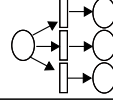

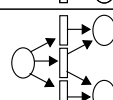

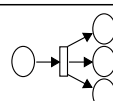
Nome da Tarefa	Guaraná	Redes de Petri
Set Return Address		
Correlator		
Merger		
Resequencer		
Filter		
Idempotent Transfer		
Dispatcher		
Distributor		
Replicator		

Tabela 3.3: Equivalência das tarefas roteadoras. (Roos-Frantz et al. [52])

3.3 Modelo de Simulação Proposto

A solução de integração foi desenvolvida com os elementos do Guaraná DSL, que possuem equivalência com os elementos presentes na técnica matemática Redes de Petri. Esta equivalência permite desenvolver um modelo de simulação, para analisar o comportamento e o desempenho de uma solução de integração, que gera certificados digitais e unifica as bases de usuários que possuem acesso aos sistemas informáticos da administração pública de Huelva (Espanha).

Nesta pesquisa foi desenvolvido um modelo de simulação com auxílio da técnica matemática Redes de Petri, sendo que para isso foi preciso conhecer as tarefas que compõem o caso de estudo, e também seu grafo em Redes

Nome da Tarefa	Guaraná	Redes de Petri
Translator		
Splitter		
Aggregator		
Chopper		
Assembler		
Cross Builder		

Tabela 3.4: Equivalência das tarefas transformadoras. (Roos-Frantz et al. [52])

Nome da Tarefa	Guaraná	Redes de Petri
Delayer		
Ticker		
Expire Checker		

Tabela 3.5: Equivalência das tarefas temporais. (Roos-Frantz et al. [52])

de Petri. Para isso utilizou-se a Tabela §3.6, que apresenta as tarefas que compõem o caso de estudo, seu grafo equivalente em Redes de Petri e a descrição do papel desempenhado por elas.

Existem algumas etapas que precisam ser seguidas no que tange o desenvolvimento de um modelo de simulação. Estas etapas são três e estão descritas na sequência. Primeiro, formula-se o modelo, visando compreender o problema de integração, para então desenvolver o modelo conceitual. Posteriormente, o modelo é implementado, para isso é necessário desenvolver o modelo computacional, além de realizar verificação e validação. Por fim, na última etapa, é realizada a experimentação do modelo, na qual é preciso analisar e interpretar os resultados obtidos através da simulação do modelo

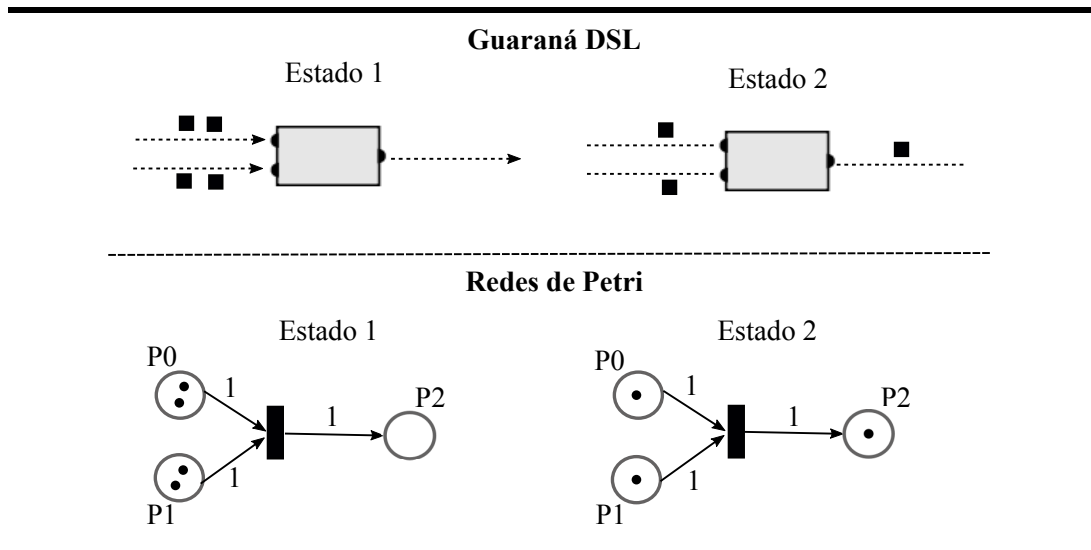


Figura 3.2: Troca de estados. (Roos-Frantz et al. [52])

computacional, além de transcrever as conclusões [61]. O processo de simulação segue o método científico, no qual primeiro são formuladas as hipóteses, seguidas do preparo do modelo, para na sequência, testar as hipóteses através do modelo de simulação desenvolvido, visando descobrir se os resultados obtidos correspondem ao comportamento do modelo.

Para que a solução de integração possa ser analisada, foi elaborado um modelo de simulação, que será simulado com o auxílio de um *software*. Nesse sentido, desenvolveu-se um modelo de simulação equivalente ao modelo conceitual da solução de integração, que foi apresentada no caso de estudo. Neste modelo, as transições P1 e P2 modelam as portas P1 e P2 respectivamente, sendo que, nestas portas, inicia-se o fluxo de *tokens*. As transições P3 e P4 representam as portas de solicitação para as aplicações sistema de recursos humanos e plataforma de certificado digital, respectivamente. A transição P5 modela a porta de saída para a aplicação LDAP, e a transição P6 modela a porta de saída, para a aplicação servidor de *e-mail*.

O fluxo de *tokens* dentro da solução ocorre como está descrito a seguir. As portas de entrada têm a função de inserir novos *tokens* na solução de integração, e suas representações em Redes de Petri equivalem às transições P1 e P2, pois quando são disparadas inserem um *token* nos lugares S1 e S2, que representam os *slots* S1 e S2. A tarefa porta de saída faz o processo inverso da porta de entrada, enviando *tokens* para as aplicações integradas,


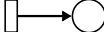
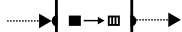
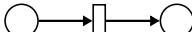
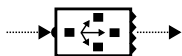
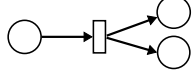

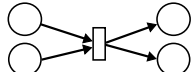

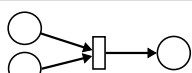



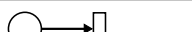



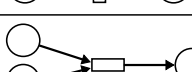
Construtor	Descrição	Guaraná	Redes de Petri
Entry Port	Local por onde as mensagens entram		
Translator	Transforma uma mensagem em outra		
Replicator	Replica as mensagens para todas as duas saídas		
Correlator	Organiza as mensagens de acordo com a sua identidade		
Context-based Content Enricher	Organiza e completa uma mensagem		
Solicitation Port	Solicita uma mensagem de uma aplicação externa		
Exit Port	Local por onde as mensagens saem		
Filter	Filtra as mensagens		
Merger	Recebe mensagens provenientes das portas, e adiciona-as para o fluxo, sem alterar o conteúdo das mensagens		

Tabela 3.6: Tradução dos elementos envolvidos na solução de integração

sendo que seu grafo equivalente em Redes de Petri indica o fluxo de *tokens* para fora do processo. A tarefa porta de solicitação (P3, P4 e P5 e P6) solicita informações de aplicações externas, que fornecem as respostas.

A tarefa *merger* (T1) recebe *tokens* provenientes das portas de entrada, e adiciona-os para o fluxo, sem alterar seu conteúdo. A tarefa *replicator* (T2, T7 e T9) é usada para replicar *tokens* para todas as suas saídas. A tarefa *translator* (T3, T8, T10 e T13) tem a função de transformar *tokens* para um formato, que a aplicação entenda.

A tarefa *correlator* (T4 e T11) é utilizada para localizar no fluxo *tokens* correlacionados, que devem ser processados em conjunto. As transições só disparam quando houver *tokens* nos lugares de entrada, referentes às transições, e projeta conjuntamente *tokens* nos seus lugares de saída. A tarefa *context-based content enricher* (T5 e T12) recebe *tokens* correlacionados

e os combina em um só *token*, sendo que a transição referente ao *context-based content enricher*, dispara quando houver ao menos um *token* nos lugares de entrada, sendo adicionado apenas um *token* ao lugar de saída. A tarefa *filter* (T6) tem a função de retirar do fluxo, *tokens* de acordo com o critério de filtragem definida. Isso acontece porque os *tokens* são removidos do fluxo, se a transição T6b disparar, e seguem o fluxo se a transição T6a disparar. Destaca-se que na política de filtragem, aproximadamente 10% dos *tokens* são filtrados, de acordo com o comportamento da solução de integração. A transição T6b retira do fluxo somente os *tokens* que não possuem endereço de *e-mail* associado.

O modelo de simulação desenvolvido é apresentado na Figura §3.3. Este modelo de simulação foi elaborado com auxílio das Redes de Petri temporizadas, sendo equivalente à solução de integração apresentada no caso de estudo.

O fluxo de *tokens* do modelo de simulação segue a descrição a seguir. O fluxo de *tokens* começa nas portas de entrada, que são representadas pelas transições P1 e P2, que adicionam os *tokens* para o fluxo. A transição T1a e T1b, recebe *tokens* provenientes de ambas as portas, sendo uma transição de fusão que não altera o conteúdo do *token*. A próxima é a transição T2, que é um *replicator*, e cria duas cópias de cada *token* recebido, uma delas consulta aplicações do sistema de recursos humanos por meio da transição P3, e a outra cópia é correlacionada com a transição T4, e com a resposta da aplicação sistema de recursos humanos. A transição T3 é um *translator* que muda o esquema do *token*. Em seguida, a transição T5 enriquece o *token*, e então a transição T7 replica este *token* enriquecido com cópias para o LDAP e a plataforma de certificado digital. O *token* que vai para a aplicação LDAP passa primeiro pela transição T8, que é tradutora, e se comunica com a aplicação, através da transição P5. Antes de consultar a plataforma de certificado digital, a transição T6b filtra os *tokens* que não possuem um endereço de *e-mail* associado, enquanto a transição T6a, faz com que os *tokens*, que possuem um endereço de *e-mail* sigam o fluxo. A transição T9 replica os *tokens*, sendo que uma cópia consulta a aplicação plataforma de certificado digital, representada pela transição P4, e a outra cópia é correlacionada com a transição T11. O *token*, que vai para a aplicação plataforma de certificado digital, passa primeiro pela transição T10, que é tradutora. Os *tokens* que passam pela transição T12, são enriquecidos com o certificado digital correspondente. Posteriormente os *tokens* passam pela transição T13, que é tradutora, para que finalmente, o *token* saia pela transição P6, que comunica-se com a aplicação servidor de *e-mail*, para enviar o certificado digital e notificar o funcionário sobre sua inclusão no LDAP.

No que diz respeito à tradução do modelo conceitual, este foi traduzido em um modelo de simulação, com auxílio do formalismo matemático Redes de Petri temporizadas, conforme a equivalência dos seus elementos. As Redes de Petri temporizadas inserem aspectos temporais ao modelo de simulação, visando considerar a semântica das tarefas. A funcionalidade de cada tarefa precisa ser mantida, quando realiza-se a transcrição das tarefas da tecnologia Guaraná para as Redes de Petri temporizadas, neste sentido, a temporização foi inserida nas transições.

A adição de tempo ocorreu da seguinte maneira: cada transição que representa uma tarefa diferente do Guaraná teria um tempo diferente associado a ela, dependendo da sua função no modelo conceitual. Elucida-se que, quando o tempo é adicionado às transições, esta só estará habilitada no tempo determinado, ou em um intervalo de tempo determinado. Outro fato que merece destaque é que se duas transições precisam de um mesmo *token* para serem disparadas, se ambas estiverem habilitadas no mesmo momento, somente uma delas irá disparar, a outra simplesmente terá decrementado seu tempo, e deverá começar novamente.

Neste modelo de simulação, não consideram-se o tamanho das mensagens, nem o tempo das aplicações. Porém, é levada em consideração a semântica do modelo, por isso associam-se tempos às transições. A entrada de mensagens no fluxo começa pelas portas P1 e P1, que possuem um intervalo de disparo associado a elas. Então, a entrada de mensagens é aleatória dentro deste intervalo. As portas adicionam mensagens uma a uma dentro do fluxo.

3.4 Resumo do Capítulo

Neste capítulo, realizou-se a transcrição do modelo conceitual, desenvolvido com o Guaraná DSL, em um modelo de simulação, utilizando o formalismo matemático Redes de Petri temporizadas. Merece destaque o fato de que as funcionalidades inerentes a cada tarefa foram preservadas, pois efetuou-se a transcrição da tecnologia Guaraná DSL para as Redes de Petri, sendo que as tarefas possuem uma semântica implementada. Além disso, foram retratadas as funcionalidades das tarefas, e seu grafo equivalente, que pode ser representado através do formalismo matemático Redes de Petri, que é capaz de manter essas funcionalidades. Desta forma, propõe-se um modelo de simulação, com auxílio do formalismo matemático Redes de Petri temporizadas.

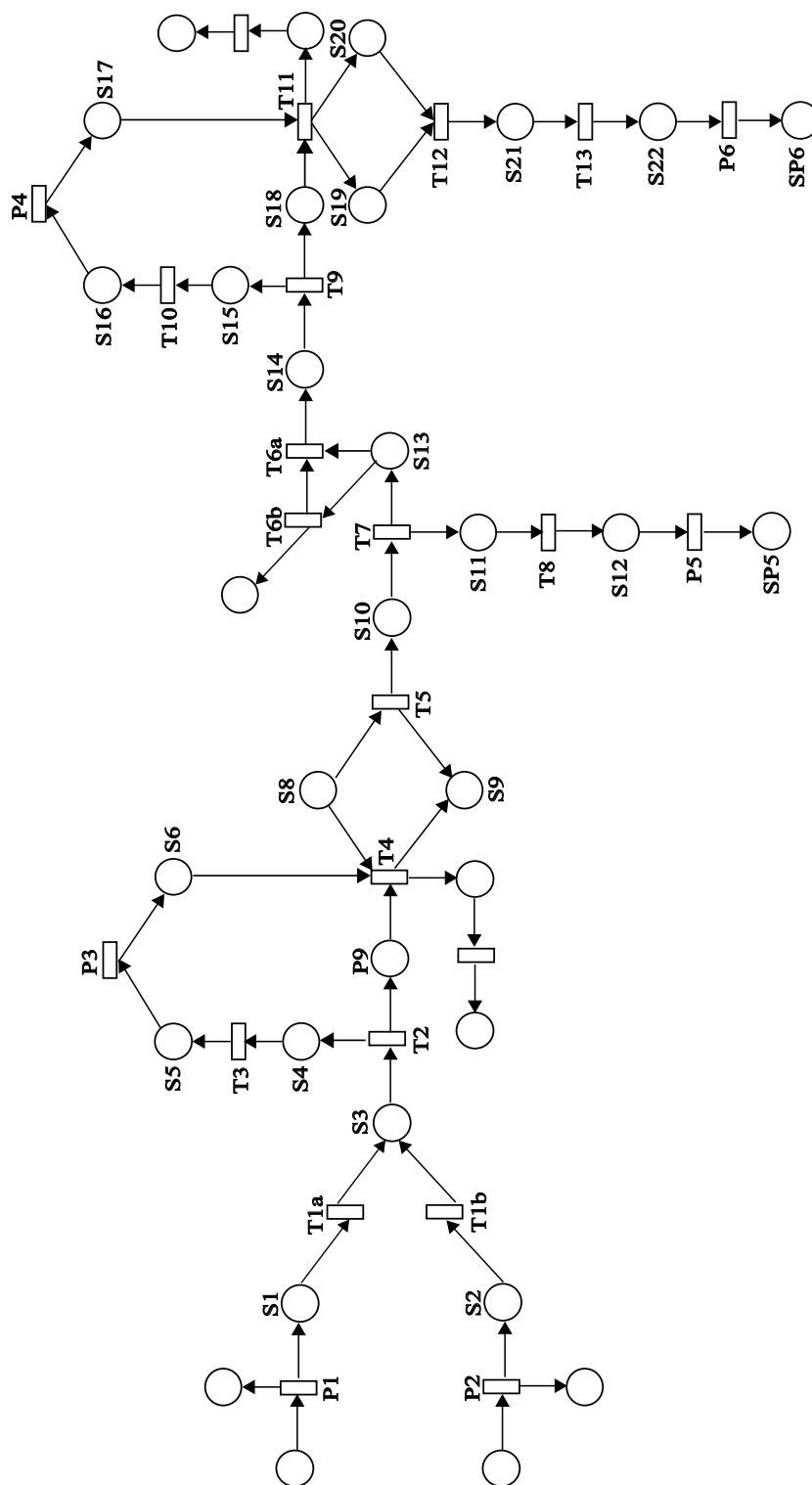


Figura 3.3: Modelo de simulação

Capítulo 4

Experimentação

*Se a educação sozinha não
transforma a sociedade
sem ela tampouco
a sociedade muda.*

Paulo Freire

Este capítulo apresenta de que modo foi realizada a experimentação, para realizar a simulação de uma solução de integração, que é o caso de estudo. A Seção §4.1 discorre acerca da descrição do experimento, cenários, variáveis que foram observadas e apresentação da ferramenta. A Seção §4.2 apresenta quais foram os resultados obtidos e trata da discussão dos mesmos. A Seção §4.3 trata da verificação e da validação do modelo de simulação. Para concluir, a Seção §4.4 apresenta o resumo do capítulo.

4.1 Experimento

A experimentação possibilita analisar o comportamento de um sistema, sob condições controladas, sendo possível comparar os resultados obtidos com a simulação do sistema, com seu desempenho no mundo real [17]. Os resultados obtidos na etapa de experimentação, chamada a partir de agora de experimento, serão comparados de modo a identificar o comportamento e o desempenho da solução estudada.

Nesta dissertação realizaram-se dois experimentos, um deles considerando o tempo fixo para as transições, e o outro considerando um intervalo de tempo para cada transição. Cada experimento será observado através de quatro cenários distintos de funcionamento.

4.1.1 Descrição dos Experimentos e Cenários

Para que o modelo de simulação proposto nesta dissertação possa considerar a semântica das tarefas, foram acrescentados tempos distintos às transições, dependendo da função que desempenham na solução de integração. Portanto para o desenvolvimento dos experimentos, estudou-se o modelo, para ver qual o tempo associado a cada transição, os tempos estão descritos na sequência:

- A transição T1 (*merger*), possui 2 unidades de tempo, ou intervalo aleatório de 1 a 3 unidades de tempo associado;
- A transição T2 (*replicator*), possui 4 unidades de tempo, ou intervalo aleatório de 3 a 5 unidades de tempo associado;
- A transição T3 (*translator*), possui 7 unidades de tempo, ou intervalo aleatório de 6 a 8 unidades de tempo associado;
- A transição T4 (*correlator*), possui 13 unidades de tempo, ou intervalo aleatório de 12 a 14 unidades de tempo associado;
- A transição T5 (*context-based content enricher*), possui 8 unidades de tempo, ou intervalo aleatório de 7 a 9 unidades de tempo associado;
- A transição T6 (*filter*), possui 3 unidades de tempo, ou intervalo aleatório de 2 a 4 unidades de tempo associado, quando as mensagens devem seguir o fluxo. E, quando as mensagens devem ser filtradas, ou excluídas do fluxo, possui 9 unidades de tempo, ou intervalo aleatório de 8 a 10 unidades de tempo associado;
- A transição T7 (*replicator*), possui 5 unidades de tempo, ou intervalo aleatório de 4 a 6 unidades de tempo associado;
- A transição T8 (*translator*), possui 5 unidades de tempo, ou intervalo aleatório de 4 a 6 unidades de tempo associado;
- A transição T9 (*replicator*), possui 5 unidades de tempo, ou intervalo aleatório de 4 a 6 unidades de tempo associado;

- A transição T10 (*translator*), possui 6 unidades de tempo, ou intervalo aleatório de 5 a 7 unidades de tempo associado;
- A transição T11 (*correlator*), possui 13 unidades de tempo, ou intervalo aleatório de 12 a 14 unidades de tempo associado;
- A transição T12 (*context-based content enricher*), possui 10 unidades de tempo, ou intervalo aleatório de 9 a 11 unidades de tempo associado;
- A transição T13 (*translator*), possui 9 unidades de tempo, ou intervalo aleatório de 8 a 10 unidades de tempo associado;

A repetição consiste em repetir a simulação do modelo, usando a mesma configuração, duração e parâmetros de entrada, porém é preciso usar uma semente de geração dos números aleatórios diferente [10]. Por esse motivo, os resultados obtidos a cada repetição são diferentes.

O número de repetições utilizado considerou a especificação proposta por Grinstead e Snell [31], que estabelece cerca de 25 repetições. De acordo com Grinstead e Snell [31], toda vez que um experimento é repetido várias vezes, com os mesmos dados, seguindo a Lei dos Grandes Números, através do número de repetições, faz-se o incremento da média amostral das variáveis consideradas no experimento, para que os resultados, sejam próximos da média populacional, teórica ou esperança matemática. Repetições são realizadas visando eliminar resultados anormais, pois trata-se de um processo estocástico.

O cálculo da média dos experimentos foi realizado seguindo as etapas descritas na sequência:

- Executar a repetição dos experimentos, seguindo a Lei dos Grandes Números;
- Utilizar um método para remover possíveis *outliers*;
- Calcular a média dos valores, já sem os *outliers*;

Estudos mostram que, em experimentos deste tipo, a média populacional é gerada quando utilizam-se, de 20 a 30 repetições [53]. O número de variações depende da estabilidade dos resultados obtidos. Podem-se, por exemplo, executar 15 repetições, e não existirem *outliers*, além disso se estes forem irrelevantes, é possível reduzir a quantidade de repetições.

Realizaram-se testes, para verificar quantas repetições eram necessárias, na ferramenta de simulação HPSim. Observou-se que 3 repetições já eram suficientes para manter os resultados estáveis, considerando-se um erro máximo de 5% para os resultados analisados.

O modelo de simulação proposto foi adaptado para 2 experimentos diferentes. Optou-se por realizar dois experimentos, visando evidenciar as possíveis divergências de valores que poderiam existir, quando o modelo de simulação fosse implementado, considerando aspectos inerentes ao ambiente externo à execução do modelo. Abaixo estão descritos estes experimentos:

- O experimento 1 considerou um tempo fixo para cada tarefa, portanto as mensagens que entram pelas portas P1 e P2, possuem sempre o mesmo tamanho. O tempo das aplicações está sendo desconsiderado, somente são contempladas as tarefas internas do processo. Assim sendo, assume-se que não há influência do ambiente externo, no funcionamento do motor de cada tarefa da solução de integração;
- O experimento 2 considerou um tempo aleatório dentro de um intervalo para cada tarefa, porém o tamanho das mensagens que entram pelas portas P1 e P2 continua sendo igual. Neste experimento, assume-se que existe a influência do ambiente externo, pois consideram-se fatores externos, que podem influenciar no tempo de execução das mensagens. Dentre estes fatores, pode-se considerar a influência do motor de execução das tarefas, o tempo das aplicações, dentre outros;

Os dois experimentos foram submetidos aos mesmos 4 cenários de simulação. Estes cenários referem-se a carga de entrada de mensagens pelas portas P1 e P2 na solução. Para cada porta de entrada, foi utilizado um intervalo de distribuição uniforme, sendo que a transição que habilita a entrada das mensagens irá disparar no intervalo de tempo associado à ela. Estes cenários estão listados na sequência:

- O cenário 1 possui uma entrada de mensagens no intervalo de [1 – 25];
- O cenário 2 possui uma entrada de mensagens no intervalo de [25 – 50];
- O cenário 3 possui uma entrada de mensagens no intervalo de [50 – 75];
- O cenário 4 possui uma entrada de mensagens no intervalo de [75 – 100];

Cada cenário possui variações na entrada de mensagens na solução de integração, para isso foi definido que cada porta de entrada terá um intervalo de tempo associado. Este intervalo controla o disparo da transição, deste modo, quando o intervalo é menor, a transição dispara mais vezes, deixando entrar mais mensagens na solução, e quando o intervalo é maior, a transição dispara menos vezes, deixando entrar menos mensagens na solução. Essa variação foi utilizada para descobrir como surgem e onde encontram-se os possíveis gargalos de desempenho. O critério de parada da simulação é o tempo de 3600000 unidades de tempo, considera-se este tempo como sendo de simulação, optou-se por este tempo pois simula uma hora de funcionamento do *software*. Em cada cenário simulado, para cada experimento, após o critério de parada, foram coletados o número de mensagens em cada *slot*.

O ambiente de execução da simulação foi um netbook Acer Aspire One 722, que possui processador com tecnologia Vision da AMD - C- Series processor C-50, 1 MB L2 cache, 1 GHz, DDR3 1066 MHz. Possui 2GB de memória, disco rígido de 500 GB, e sistema operacional Windows 7 Starter Edition 32 bits.

4.1.2 Variáveis Observadas

Inicialmente, foram realizadas discussões entre o orientador e a pesquisadora, para em conjunto definirem as variáveis consideradas na pesquisa, sendo que a simulação torna possível observar dados referentes a elas, para deste modo realizar análises dos resultados. Neste sentido, após a simulação do modelo, foram coletados os resultados obtidos para os diferentes cenários, nos diferentes experimentos, considerando duas variáveis: tempos distintos nas transições e acúmulo de *tokens* (mensagens) para cada lugar (*slot*).

A variável tempo distinto nas transições pode ser analisada, porque os lugares acumulam *tokens*, pois as mensagens vão se agrupando nos *slots*, conforme a solução de integração está sendo simulada. A quantidade de mensagens acumuladas nos *slots* possibilita a análise do comportamento e do desempenho da solução de integração. Quando existe um grande número de mensagens acumuladas em um único *slot*, é sinal de que neste ponto a solução está tendo dificuldades para processar as mensagens, gerando um possível gargalo de desempenho.

A variável acúmulo de *tokens* (mensagens) para cada lugar (*slot*), pode ser analisada pois no modelo de simulação, as mensagens disparam quando as transições tornam-se habilitadas. Para que a transição esteja habilitada, é associado a ela um tempo de disparo, que pode ser fixo ou aleatório dentro

de um intervalo. Quando o disparo ocorre, a mensagem é enviada do *slot* anterior, para o *slot* seguinte. Quando ocorre o critério de parada, as mensagens que sobraram nos *slots* da solução de integração, representam o comportamento do sistema naquele momento. Quando muitas mensagens ficam acumuladas em um mesmo *slot*, tem-se um possível gargalo de desempenho.

4.1.3 Apresentação da Ferramenta

Para que fosse possível realizar a escolha do *software* que mais se adaptava ao modelo de simulação que iria ser simulado, realizaram-se pesquisas considerando alguns *softwares* já conhecidos na literatura, e que permitiam realizar a simulação de um modelo, desenvolvido com a técnica matemática Redes de Petri temporizada. Neste sentido, foram criados pequenos modelos no *software*, visando verificar quais análises poderiam ser realizadas com os dados coletados, e também se era possível desenvolver um modelo dentro do *software* que permitisse associar tempo às transições. Deste modo, após estas análises, foi escolhida uma ferramenta de simulação, que pode ser utilizada para a análise do comportamento e do desempenho do modelo de simulação da solução de integração, denominada HPSim, que foi desenvolvida por Henryk Anschuetz [1].

O *software* HPSim apresenta uma *interface* intuitiva e de fácil utilização, que permite alterar o tempo das transições, e analisar a quantidade de *tokens* presentes em cada lugar. Dentre suas vantagens está a possibilidade de acompanhar o estado da rede de forma gráfica, o que auxilia no desenvolvimento do modelo e na detecção de possíveis erros de modelagem. O *software* também permite a gravação dos resultados da simulação e seu posterior tratamento em *softwares* como o Microsoft Excel, possibilitando uma característica essencial para a análise do sistema modelado [40].

Além do modelo de Redes de Petri lugar/transição, o *software* HPSim permite a simulação de Redes de Petri temporizadas, e também a utilização de arcos inibidores e habilitadores. Para diferenciar os tipos de Redes de Petri, utilizam-se transições diferenciadas. Nas Redes de Petri lugar/transição, as transições ocorrem instantaneamente, e são disparadas quando estiverem habilitadas. Nas Redes de Petri temporizadas, associa-se um intervalo de tempo, ou um tempo fixo, à cada transição. Quando a transição estiver habilitada é necessário aguardar o intervalo de tempo associado a ela, para que o disparo ocorra. Durante este intervalo existe a possibilidade de outra transição disparar, o que desabilita a transição atual. Após isso, a primeira transição volta a estar habilitada, e inicia-se uma nova contagem de tempo.

O simulador HPSim utiliza dois tipos de distribuição: a distribuição exponencial e a estocástica. A distribuição exponencial, é estabelecida pela taxa média de disparo, conhecida por λ , que utiliza seu tempo médio de disparo, dado através da expressão $\mu = \frac{1}{\lambda}$. As transições são disparadas, pois possuem associadas a elas a função de distribuição uniforme do tempo (distribuição estocástica), que tem sua definição dada por um limite inferior e um limite superior. Este limite delimita o tempo que a transição irá demorar para disparar, não sendo possível alterar esta função para outra.

A Figura §4.1 mostra os componentes da janela principal do HPSim. São eles:

- Janela de edição do modelo, na qual constitui-se o modelo em Redes de Petri;
- Barra principal, que contém comandos como: salvar, abrir, fechar, dentre outros;
- Barra de edição, que possui comandos para construir modelos em Redes de Petri, como: lugar, transição, arco, dentre outros;
- Barra de simulação, que contém comandos da simulação;
- Tabela de propriedades, onde encontram-se tabeladas as propriedades de cada elemento da rede, quando este for selecionado na janela de edição;

Para que os modelos sejam construídos, utiliza-se a barra de edição. Para adicionar novos itens, é preciso selecionar o item correspondente e clicar na janela de edição do modelo. Para modificar suas propriedades, seleciona-se o elemento já existente e realiza-se a modificação na tabela. É possível realizar modificações nas transições, nos lugares e nos arcos.

Depois que o modelo estiver construído, realiza-se a simulação do mesmo. É possível ajustar os parâmetros de simulação, utilizando o menu principal. Tem-se a possibilidade de definir o arquivo que será gerado, após o término da simulação, este pode ser acessado posteriormente através do Microsoft Excel. Este simulador permite que a duração máxima da simulação seja determinada através do número de passos, ou através do tempo máximo de simulação. Além disso, o incremento utilizado para a evolução do tempo, pode ser modificado, conforme seja necessário.

4.2 Resultados e Discussão

Na sequência, apresentam-se os resultados das simulações, considerando-se o tempo associado às transições, e também o acúmulo de *tokens* (mensagens) em cada lugar (*slot*). Elucida-se que o objetivo desta dissertação não é melhorar o motor de execução da plataforma, somente analisar as transições e possíveis gargalos de desempenho que possam surgir. Quanto ao desempenho das transições (tarefas), elucida-se que depende da qualidade do motor de execução da solução de integração, e também do tamanho das mensagens e da complexidade das tarefas. Salienta-se que os resultados obtidos através das simulações, não apresentaram *outliers*, o que foi comprovado estatisticamente.

Na construção dos gráficos e análises, foram desconsiderados *slots* que não possuíam um número de mensagens acumulado significativo. Por esse motivo só estão sendo analisados nos gráficos, os *slots* 1, 2, 6, 7, 17 e 18. Os *slots* 1 e 2 referem-se ao número de mensagens acumuladas nas portas de entrada. Os *slots* 6 e 7 fazem parte do primeiro correlacionador do modelo de simulação, enquanto os *slots* 17 e 18 fazem parte do segundo correlacionador presente no modelo de simulação.

Inicialmente, tem-se a análise do experimento 1, o qual associa à cada transição um tempo fixo, desconsiderando a influência do ambiente externo, nos quais os gráficos referem-se aos cenários 1, 2, 3 e 4, sendo que a entrada de mensagens na solução diminui a cada cenário. É possível observar nos *slots* 1 e 2, que, ao trocar de cenário, indo do 1 até o 4, o número de mensagens acumuladas nas portas de entrada diminui continuamente, pois entram menos mensagens na solução. O maior acúmulo de mensagens ocorre nos *slots* 6 e 7, que compõem o primeiro correlacionador do modelo, no cenário 1, acumularam-se respectivamente 154.044 e 154.045 mensagens, no cenário 2, 35.882 mensagens em ambos os *slots*, no cenário 3, 2.457 mensagens em ambos os *slots* e no cenário 4, zero e uma mensagem respectivamente. Nos *slots* 17 e 18, também existe um grande acúmulo de mensagens sendo que no cenário 1 acumularam-se 16.279 mensagens em ambos os *slots*, no cenário 2, 16.190 mensagens em ambos os *slots*, no cenário 3, 16.338 e 16.339 mensagens respectivamente, e no cenário 4, 8.316 mensagens em ambos os *slots*. Observa-se que o número de mensagens acumuladas diminui, ao trocar de cenário pois entram menos mensagens na solução de integração. É possível observar também que, nos *slots* 17 e 18, o número de mensagens é menor e mais estável, do que nos *slots* 6 e 7, isso ocorre porque os *slots*

17 e 18, compõem o segundo correlacionador, que correlaciona menos mensagens, pois algumas já foram filtradas da solução. Portanto, constata-se que os gargalos de desempenho estão presentes nos *slots* 6, 7, 17 e 18, e que quanto mais mensagens entram na solução maior é o gargalo de desempenho formado. Os resultados obtidos podem ser observados na Figura §4.2.

O experimento 1 também foi analisado de forma diferenciada, na qual cada um dos cenários foi analisado separadamente, considerando a quantidade de mensagens acumuladas nos *slots* 1, 2, 6, 7, 17 e 18 para cada um dos cenários. O cenário 1 mostra que ficaram acumuladas, 144, 32, 154.044, 154.045, 16.279 e 16.279 mensagens respectivamente. O cenário 2 mostra que ficaram acumuladas, 43, 0, 35.882, 35.882, 16.190 e 16.190 mensagens respectivamente. O que demonstra que o maior acúmulo de mensagens (gargalo) ocorreu nos *slots* 6 e 7, porém nos *slots* 17 e 18, também existe um acúmulo considerável de mensagens. O cenário 3 mostra que ficaram acumuladas, 36, 1, 2.457, 2.457, 16.338 e 16.339 mensagens respectivamente. O cenário 4 mostra que ficaram acumuladas 6, 1, 0, 1, 8.316 e 8.316 mensagens respectivamente. Observa-se que, nos cenários 3 e 4, houve diminuição no acúmulo de mensagens nos *slots* 6 e 7 e aumento no acúmulo de mensagens nos *slots* 17 e 18, isso se deve ao fato de que o critério de parada da simulação é o tempo, e, em determinado momento não entram novas mensagens na solução. Nestes cenários, o número de mensagens que entra na solução é bem menor que nos cenários anteriores, então as mensagens acabam acumulando-se no último correlacionador, pois acabaram as novas mensagens que poderiam ser correlacionadas com estas, porque para que o comportamento do correlacionador ocorra-se corretamente, adicionou-se um novo *slot*, que gera uma nova mensagem correlacionada, e este *slot* continuou gerando novas mensagens. Porém não entraram novas mensagens na solução, o que acarretou este gargalo maior nos *slots* 17 e 18, nos últimos dois cenários. Os resultados obtidos podem ser observados na Figura §4.3.

A análise do experimento 2, no qual associa-se à cada transição um tempo aleatório dentro de um intervalo, considera a influência do ambiente externo, nos quais os gráficos referem-se aos cenários 1, 2, 3 e 4, sendo que a entrada de mensagens na solução diminui a cada cenário. É possível observar, nos *slots* 1 e 2, que ao trocar de cenário, indo do 1 até o 4, o número de mensagens acumuladas nas portas de entrada diminui continuamente, pois entram menos mensagens na solução. O maior acúmulo de mensagens ocorre nos *slots* 6 e 7, que compõem o primeiro correlacionador do modelo. No cenário 1, acumularam-se respectivamente 154.217 e 154.218 mensagens, no cenário 2, 35.917 mensagens em ambos os *slots*, no cenário 3, 2.432 mensagens em ambos os *slots* e no cenário 4, zero e uma mensagem respecti-

vamente. Nos *slots* 17 e 18, também existiram um grande acúmulo de mensagens sendo que no cenário 1 acumularam-se 16.303 mensagens em ambos os *slots*, no cenário 2, 16.182 mensagens em ambos os *slots*, no cenário 3, 16.361 em ambos os *slots*, e no cenário 4, 8.301 mensagens em ambos os *slots*. Observa-se que o número de mensagens acumuladas diminui ao trocar de cenário, pois entram menos mensagens na solução de integração. É possível observar também que nos *slots* 17 e 18, o número de mensagens é menor e mais estável do que nos *slots* 6 e 7, isso ocorre porque os *slots* 17 e 18, compõem o segundo correlacionador, que correlaciona menos mensagens, pois algumas já foram filtradas da solução. Portanto, constata-se que os gargalos de desempenho, estão presentes nos *slots* 6, 7, 17 e 18, e que quanto mais mensagens entram na solução, maior é o gargalo de desempenho formado. Os resultados obtidos podem ser observados na Figura §4.4.

O experimento 2 também foi analisado de forma diferenciada, na qual cada um dos cenários foi analisado separadamente, considerando a quantidade de mensagens acumuladas nos *slots* 1, 2, 6, 7, 17 e 18 para cada um dos cenários. O cenário 1 mostra que ficaram acumuladas, 123, 568, 154.217, 154.218, 16.303 e 16.303 mensagens respectivamente. O cenário 2 mostra que ficaram acumuladas, 69, 1, 35.917, 35.917, 16.182 e 16.182 mensagens respectivamente. Esse fato demonstra que o maior acúmulo de mensagens (gargalo) ocorreu nos *slots* 6 e 7, porém nos *slots* 17 e 18, também existe um acúmulo considerável de mensagens. O cenário 3 mostra que ficaram acumuladas, 0, 53, 2.432, 2.432, 16.361 e 16.361 mensagens respectivamente. O cenário 4 mostra que ficaram acumuladas, 9, 0, 0, 1, 8.301 e 8.301 mensagens respectivamente. Observa-se que, nos cenários 3 e 4, houve diminuição no acúmulo de mensagens nos *slots* 6 e 7 e aumento no acúmulo de mensagens nos *slots* 17 e 18, isso se deve ao fato de que o critério de parada da simulação é o tempo, e em determinado momento não entram novas mensagens na solução. Nestes cenários, o número de mensagens que entram na solução é bem menor que nos cenários anteriores, então as mensagens acabam acumulando-se no último correlacionador, pois acabaram as novas mensagens que poderiam ser correlacionadas com estas, porque para que o comportamento do correlacionador ocorresse corretamente, adicionou-se um novo *slot*, que gera uma nova mensagem correlacionada, e este *slot* continuou gerando novas mensagens, porém não entraram novas mensagens na solução, o que gerou este gargalo maior nos *slots* 17 e 18, nos últimos dois cenários. Os resultados obtidos podem ser observados na Figura §4.5.

Por fim, tem-se uma comparação entre os dois experimentos, o experimento 1, que possui tempo fixo, e o experimento 2, que possui um tempo aleatório dentro de um intervalo. Novamente, analisam-se somente os *slots* 1,

2, 6, 7, 17 e 18, porém em cada cenário, foi feita uma comparação entre os dois experimentos, para que fosse possível, analisar se existem ou não divergências no número de mensagens acumuladas nos *slots*. Na sequência, tem-se a quantidade de mensagens acumuladas nos *slots*, sendo que primeiro tem-se o valor do experimento 1, e na sequência tem-se o valor do experimento 2. Para o cenário 1, observa-se que a variação no número de mensagens é: *slot* 1, possui 144 e 123; *slot* 2, 32 e 568; *slot* 6, 154.044 e 154.217; *slot* 7, 154.045 e 154.218; *slot* 17, 16.279 e 16.303; e *slot* 18, 16.279 e 16.303. Para o cenário 2, tem-se a seguinte quantidade de mensagens acumuladas, respectivamente para o experimento 1 e para o experimento 2: *slot* 43 e 69; *slot* 2, 0 e 1; *slot* 6, 35.882 e 35.917; *slot* 7, 35.882 e 35.917; *slot* 17, 16.190 e 16.182; e *slot* 18, 16.190 e 16.182. Para o cenário 3, a variação do número de mensagens nos *slots* é: *slot* 1, 36 e 0; *slot* 2, 1 e 53; *slot* 6, 2.457 e 2.432; *slot* 7, 2.457 e 2.432; *slot* 17, 16.338 e 16.361; e *slot* 18, 16.339 e 16.361. No cenário 4, a variação é: *slot* 1, 6 e 9; *slot* 2, 1 e 0; *slot* 6, 0 e 0; *slot* 7, 1 e 1, *slot* 17, 8.316 e 8.301; e *slot* 18, 8.316 e 8.301. É possível observar que a variação no número de mensagens acumuladas nos *slots*, nos diferentes experimentos é muito pequena, acredita-se que isso se deve ao fato de que o intervalo utilizado é muito próximo do valor fixo, pois ele foi variado somente um número abaixo e um número acima. Os resultados obtidos podem ser observados na Figura §4.6.

A análise da variável tempo distinto nas transições, permite observar que dependendo da função que a transição desempenha na solução, o tempo associado à ela pode ser menor ou maior. Este tempo associado refere-se ao disparo da transição, ou seja, a transição só pode disparar quando este tempo for atingido. Além disso, é possível observar que quando a transição possui um tempo menor associado a ela, o *token* irá permanecer por menos tempo no lugar que precede a transição, no entanto, se a transição possuir um tempo maior associado a ela, o *token* irá permanecer por mais tempo no lugar que precede a transição. Neste sentido, conclui-se que os gargalos irão estar presentes nas transições que possuem maior tempo associado, pois estas irão demorar mais tempo para estarem habilitadas, desta forma, nessa dissertação as transições que causam os gargalos são representadas pelo correlacionador.

A análise da variável acúmulo de *tokens* (mensagens) em cada lugar (*slot*), permite observar quantas mensagens ficaram acumuladas nos *slots* que precedem as transições. *Slots* que precedem transições que possuem menor tempo associado irão acumular menos mensagens, pois as transições serão habilitadas mais vezes, portanto irão realizar o disparo de mais mensagens, sendo que as mensagens que não foram disparadas permanecem acumuladas nos *slots*. Por outro lado, *slots* que precedem transições que possuem maior tempo associado irão acumular mais mensagens, pois as tran-

sições serão habilitadas menos vezes, desta forma, irão realizar o disparo de menos mensagens, sendo que as mensagens que não foram disparadas permanecem acumuladas nos *slots*. Portanto conclui-se que os gargalos ocorrem nos *slots* que precedem as transições com maior tempo associado, pois estas irão demorar mais tempo para estarem habilitadas, sendo que nesta dissertação estes *slots* precedem a transição que representa o correlacionador.

Observando estas variáveis, foi possível analisar o comportamento e o desempenho da solução de integração, de modo a encontrar gargalos de desempenho na solução de integração, podendo concluir-se que os gargalos estão presentes nas transições que possuem maior tempo associado, pois estas irão demorar mais tempo para estarem habilitadas, portanto o gargalo está presente nos *slots* que precedem o correlacionador, sendo o correlacionador a causa do gargalo. Deste modo, percebe-se a importância de realizar a simulação na fase de projeto, pois desta forma é possível diminuir custos de implementação, além de facilitar a realização de alterações necessárias na solução de integração.

4.3 Verificação e Validação

Uma solução de integração, para ser simulada, precisa ser transformada em um modelo de simulação equivalente ao modelo conceitual da solução. Quando realizam-se simulações, é preciso garantir que não existam erros, quando esta solução for implementada. Para que se tenha certeza de que o modelo representa a realidade e não possui erros, realizam-se duas fases, a fase de verificação e a fase de validação. Estas fases são realizadas utilizando como base técnicas formais propostas na literatura. Na sequência, trata-se da definição destes termos e das técnicas propostas na literatura, além disso tem-se a verificação do modelo de simulação. Nesta dissertação, não será realizada a validação do modelo, porque a verificação será realizada ainda na fase de projeto.

4.3.1 Definição e Técnicas de Verificação

Um modelo nada mais é do que a representação de um sistema no mundo real. É possível desenvolver um modelo de simulação de modo que esse reproduza o comportamento real da solução. Para que seja possível criar um modelo de simulação que não possua erros de modelagem, é preciso compreender como a solução comporta-se. Para garantir que os erros não existam, podem ser realizadas simulações que permitam analisar o comportamento e

o desempenho do sistema. Neste sentido, para que se tenha certeza de que não existem erros no modelo de simulação, realiza-se a verificação do mesmo.

A verificação tem como finalidade encontrar erros no modelo de simulação, e corrigi-los antes que o modelo seja implementado, sendo que existem técnicas que podem ser utilizadas para verificar o modelo. Neste sentido, a verificação visa garantir que não existam erros no modelo de simulação, pois se este modelo for implementado com erros, surgirão problemas de implementação computacional futuramente [61].

A verificação do modelo de simulação é feita por meio de técnicas de verificação, que podem ser encontradas através de pesquisa bibliográfica. Os autores Kleijen [35] e Sargent [53] tratam sobre técnicas de verificação. Estas técnicas estão descritas na sequência.

O autor Kleijen [35] trata sobre técnicas de validação e verificação, que explicitam de que maneira é possível verificar e validar modelos de simulação, utilizando técnicas estatísticas, mediante resultados quantitativos que podem ser obtidos durante as simulações. Para realizar a verificação do modelo, por sua vez, é necessário ter o conhecimento de especialistas no assunto, para que opinem sobre o comportamento do modelo de simulação, de modo a dizer se a verificação está correta.

Os dados qualitativos são colhidos com base no comportamento do modelo de simulação e servem para analisar se as características encontradas após as simulações, representam de fato o comportamento esperado pela solução de integração. Essa análise pode ser realizada, pois o comportamento da solução diante de algumas situações já é conhecido, portanto os dados obtidos deverão ter comportamento semelhante ao que já se conhece. Os dados que são colhidos com a simulação são quantitativos e permitem que seja realizada uma análise estatística do comportamento da solução. Deste modo, os resultados encontrados através do experimento, serão comparados com o conhecimento que os especialistas possuem sobre o sistema, sendo possível verificar o modelo de simulação [8].

Além disso, também podem ser utilizadas as técnicas de verificação propostas por Sargent [53], que dizem respeito ao modelo de simulação que foi desenvolvido, com base no modelo conceitual. Dentre as técnicas propostas por Sargent [53], somente duas adaptam-se ao caso de estudo desta dissertação, são elas: técnica de verificação do modelo computacional e técnica de validade de eventos.

A técnica de verificação do modelo computacional visa garantir que o modelo de simulação desenvolvido, seja equivalente ao modelo conceitual, mas

para isso é necessário utilizar uma representação gráfica específica. Neste sentido, propõe-se a verificação do modelo de simulação através de uma comparação, em que a quantidade de dados que entram no sistema, deve ser igual a quantidade de dados que saem do sistema, considerando também os dados que ficaram acumulados dentro do sistema e os dados que por algum motivo foram removidos do mesmo. Além disso, é necessário ter cuidado com os dados que possam ter sido replicados por algum motivo, dentro do modelo de simulação, sendo que sempre que houver duplicação de dados, estes dados duplicados não devem ser considerados na verificação do modelo.

A técnica da validade de eventos visa realizar uma comparação entre a ocorrência de eventos no modelo conceitual e no modelo de simulação. Assim sendo, um evento deve ocorrer aproximadamente no mesmo momento, tanto no modelo conceitual, quanto no modelo de simulação.

4.3.2 Definição e Técnicas de Validação

Um modelo representa um sistema no mundo real. Neste sentido, um modelo de simulação deve ser capaz de reproduzir o comportamento real da solução, de forma eficaz. É preciso ter certeza de que o modelo de simulação tem o comportamento correspondente ao que deve ser, para isso, é necessário validar o modelo de simulação. Existem técnicas que podem ser utilizadas para realizar esta validação.

A validação de um modelo de simulação nada mais é do que analisar se os dados encontrados com a simulação do modelo são semelhantes aos dados reais obtidos. Os dados simulados são encontrados através da simulação do modelo, que deve ser equivalente ao modelo conceitual. Os dados reais são obtidos através da implementação do modelo, neste sentido, a validação refere-se às técnicas utilizadas para assegurar que o modelo, apesar dos pressupostos, representa de fato, o sistema real [10].

Algumas técnicas de validação são propostas por Bukh e Jain [6]. Os aspectos considerados nas técnicas precisam ser validados. Estas técnicas consideram aspectos como simplificações, pressupostos adotados, parâmetros de entrada, distribuições e resultados.

Bukh e Jain [6] propõem as seguintes técnicas de verificação: teste de Turing, comparação entre modelos, medições em sistemas reais e intuição de especialistas. O teste de Turing utiliza os dados reais e os dados da simulação, de forma que quando o especialista não encontra divergências entre os valores, o modelo encontra-se validado. A comparação entre modelos realiza

comparações entre os resultados do modelo de simulação, com os resultados de outros modelos que já foram validados. As medições em sistemas reais, realizam comparações entre observações, tanto do sistema real, quanto do sistema simulado. A intuição de especialistas, por sua vez, baseia-se em reuniões, em que o desenvolvimento do modelo é discutido, considerando principalmente três aspectos: pressupostos, entrada e resultados.

Nesta dissertação, o modelo conceitual não será implementado, portanto não haverá dados reais, que possam ser comparados, de modo a realizar-se de fato a validação do modelo.

4.3.3 Verificação do Modelo de Simulação

Realizar a verificação e a validação de um modelo de simulação não é uma tarefa fácil, por isso é muito importante realizar pesquisas sobre o assunto, de modo a garantir que exista uma base teórica ao realizar de fato a validação e a verificação do modelo. Neste sentido, realizou-se uma pesquisa bibliográfica, visando obter aporte teórico sobre o tema e sobre as técnicas existentes. Além disso, é imprescindível encontrar uma técnica que se adapte a proposta do modelo de simulação que está sendo analisado nesta dissertação.

Nesta dissertação, serão realizadas simulações com o intuito de obter dados, a fim de realizar a verificação do modelo de simulação ainda na fase de projeto. Este é o diferencial que esta dissertação propõe, pois a maioria dos trabalhos realiza a validação. Pretende-se realizar a verificação do modelo, mostrando que é possível diminuir custos de implementação e também diminuir o tempo necessário para desenvolver uma solução, quando realizam-se simulações e análise de dados na fase de projeto. As técnicas propostas por Kleijen [35] e Sargent [53] serão utilizadas para a verificação do modelo.

A técnica de Kleijen [35] refere-se a análise do comportamento do sistema, com auxílio de um especialista da área. Os especialistas desta área dizem que em alguns *slots* da solução deverá ocorrer acúmulo de mensagens, os conhecidos gargalos. Os especialistas dizem que as transições que devem acumular mais mensagens são as que pertencem ao correlacionador. Portanto espera-se que a solução tenha um acúmulo nos *slots* 6, 7, 17 e 18, que alimentam os correlacionadores, ou seja, que fazem parte do correlacionador. O correlacionador tem como função correlacionar mensagens, e por esse motivo, a transição demora para estar habilitada, o que gera os gargalos de desempenho. Os dados experimentais colhidos mediante a simulação provam que os *slots* do correlacionador, de fato possuem acúmulo de mensagens, portanto são gargalos de desempenho, quando comparados aos outros *slots* da solução.

Observa-se, sob a ótica proposta por Kleijen [35], que as características esperadas no modelo, que é o acúmulo de mensagens nos *slots* que compõem o correlacionador, de fato ocorre. Portanto, a comparação que foi realizada entre o comportamento da simulação e o que era esperado pelos especialistas, mostrou semelhança, verificando o modelo por esta técnica.

Para verificação do modelo, também foi utilizada a técnica proposta por Sargent [53], que faz uma análise estatística dos dados, garantindo que o número de mensagens que entrou na solução seja igual ao número de mensagens que saíram, ficaram acumuladas nos *slots*, ou foram filtradas da solução. A aplicação da técnica proposta por Sargent [53] foi realizada inicialmente, através da contagem de quantas mensagens saíram do fluxo, pelas portas de saída, quantas mensagens ficaram acumuladas nos *slots*, e quantas foram filtradas. Considerou-se que as mensagens replicadas não devem ser contabilizadas.

Para descobrir quantas mensagens entraram no fluxo, consideraram-se as mensagens que entraram pelas portas, menos as que ficaram acumuladas nos *slots* das portas e não entraram na solução, totalizando 205.518 mensagens. Como entrada, foi considerada somente uma porta, pois depois as mensagens de entrada, são rearrumadas em uma só. Para realizar a verificação do modelo de simulação, foi necessário somar as mensagens que ficaram acumuladas nos *slots*, com as que foram filtradas e com as que saíram da solução. As mensagens acumuladas nos *slots* totalizaram 170.349 mensagens, as que saíram pela portas, somadas das que foram filtradas, totalizam 35.169 mensagens, e seu total é 205.518 mensagens. Após isso, subtraiu-se o total de mensagens que entraram na solução, com as que saíram, ficaram acumuladas, ou foram filtradas, e o resultado foi 0. Desta forma, conclui-se que, de acordo com a técnica proposta por Sargent [53], o modelo de simulação foi implementado corretamente e não apresentou erros, tornando-o verificado.

4.4 Resumo do Capítulo

Este capítulo descreve a experimentação que foi realizada, para o desenvolvimento do modelo de simulação. Foram propostos 2 experimentos, e cada um foi submetido a 4 cenários de simulação diferentes. O modelo de simulação foi implementado na ferramenta de simulação, que melhor se adaptou às necessidades do modelo, chamada HPSim. Esse *software* foi escolhido por ser capaz de realizar simulações, com o auxílio da técnica matemática Rede de Petri temporizada. Foram relatados os resultados obtidos com a simulação, e realizadas análises sobre eles. Abordou-se também

a verificação e a validação do modelo, sendo que foi trazida a definição das mesmas, e as técnicas conhecidas na literatura. Além disso, foi realizada a verificação do modelo de simulação estudado na dissertação.

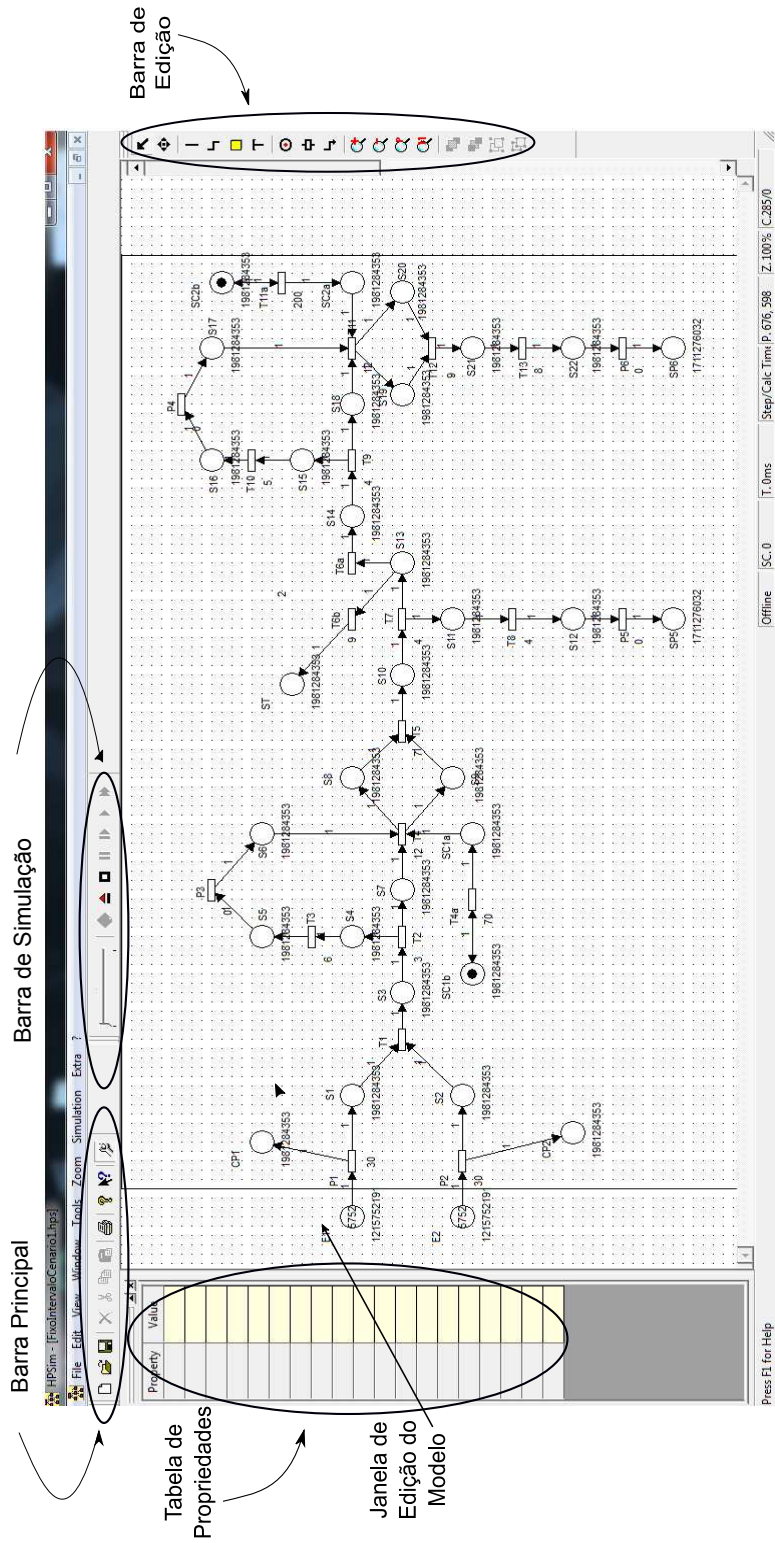


Figura 4.1: Componentes do HPSim

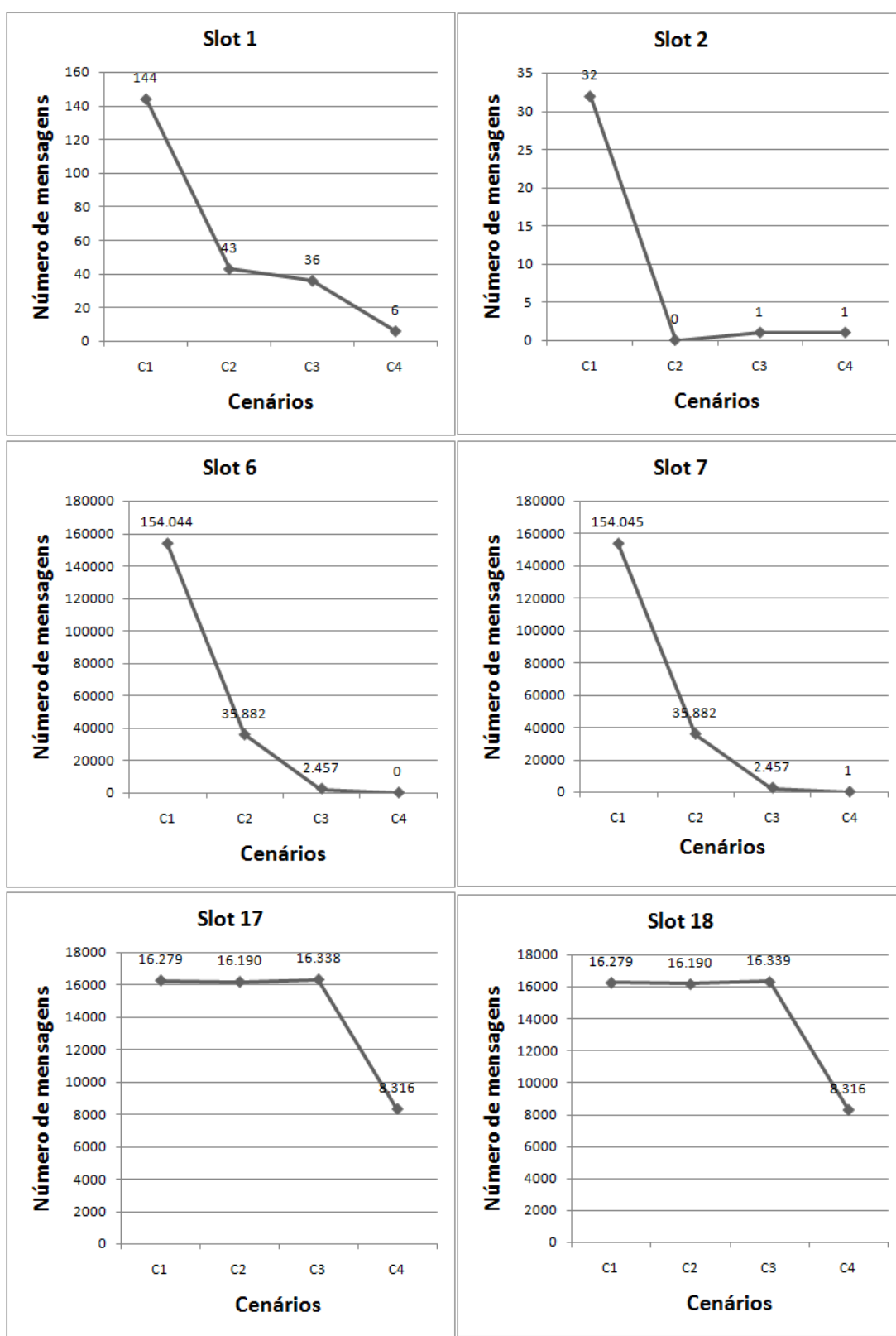


Figura 4.2: Mensagens acumuladas nos slots no experimento 1.

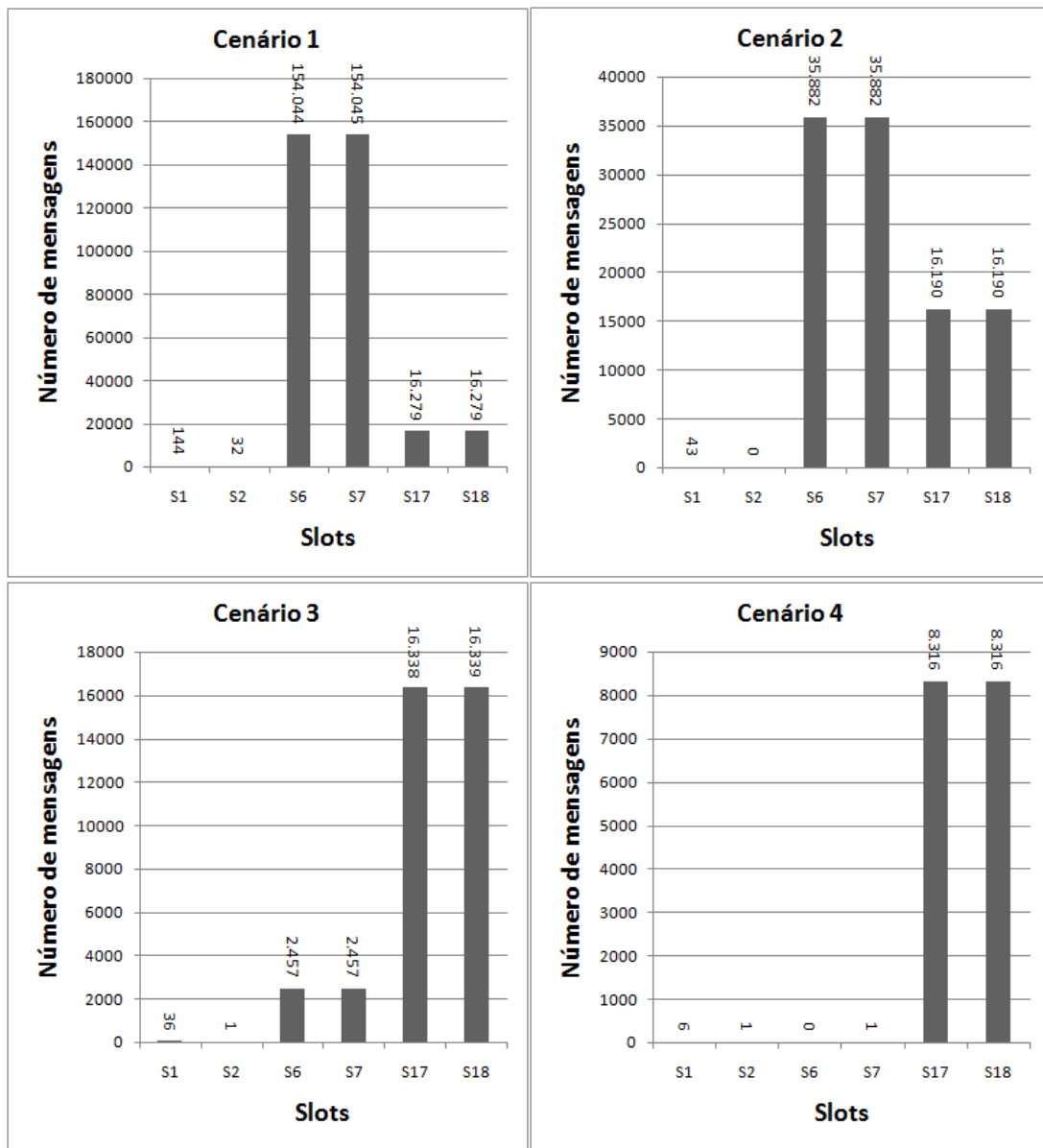


Figura 4.3: Mensagens acumuladas nos cenários do experimento 1.

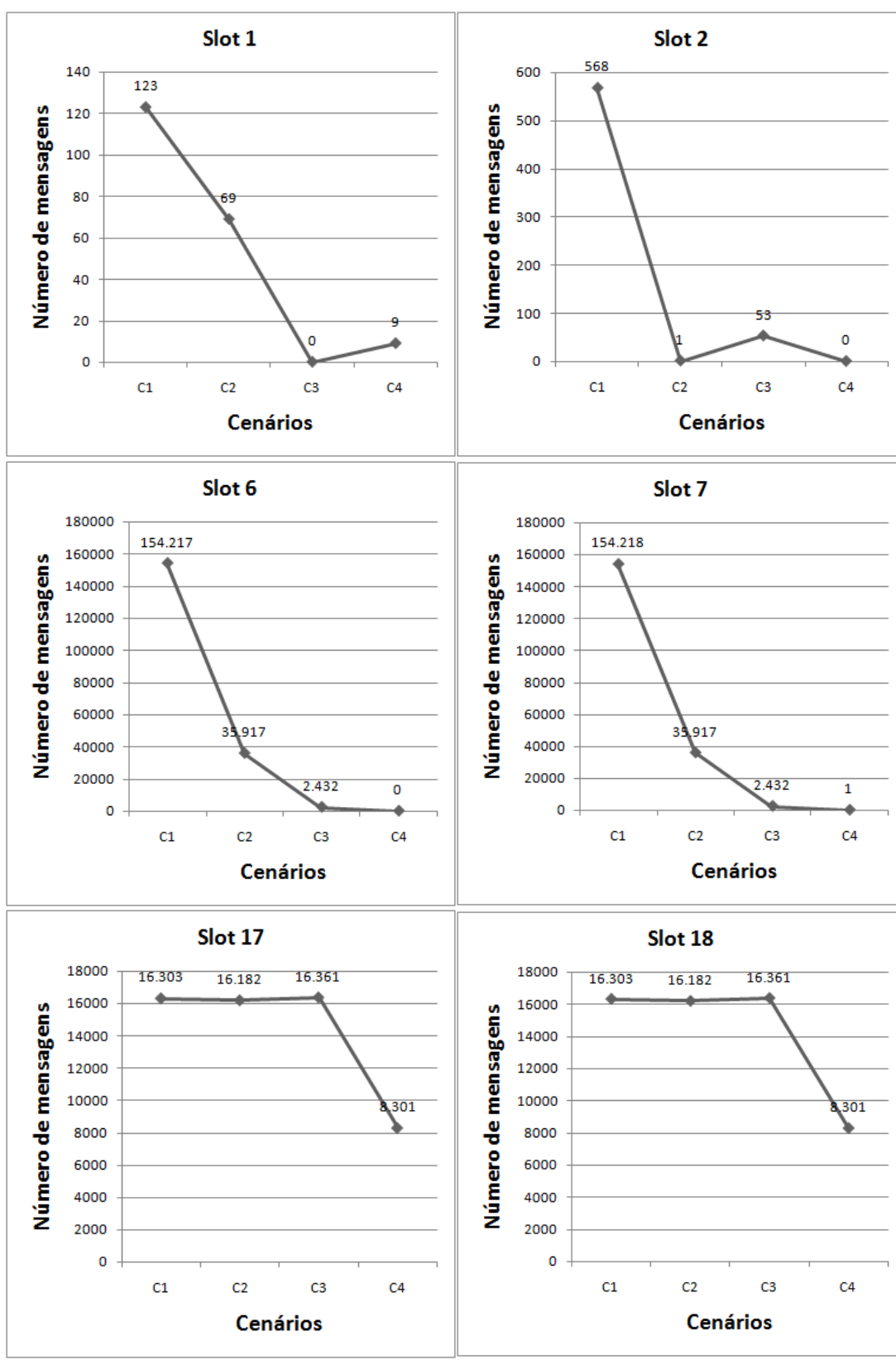


Figura 4.4: Mensagens acumuladas nos slots no experimento 2.

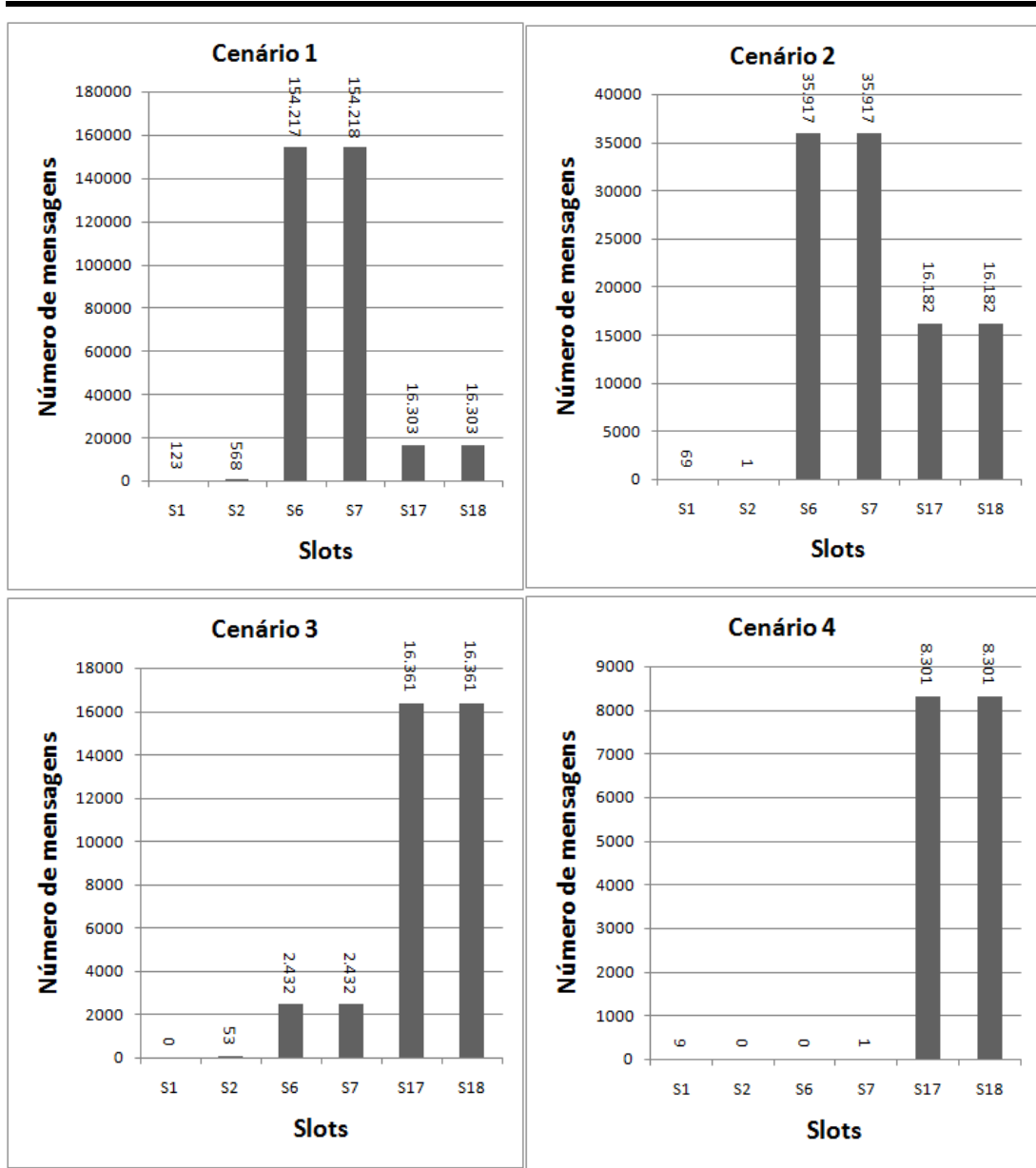


Figura 4.5: Mensagens acumuladas nos cenários do experimento 2.

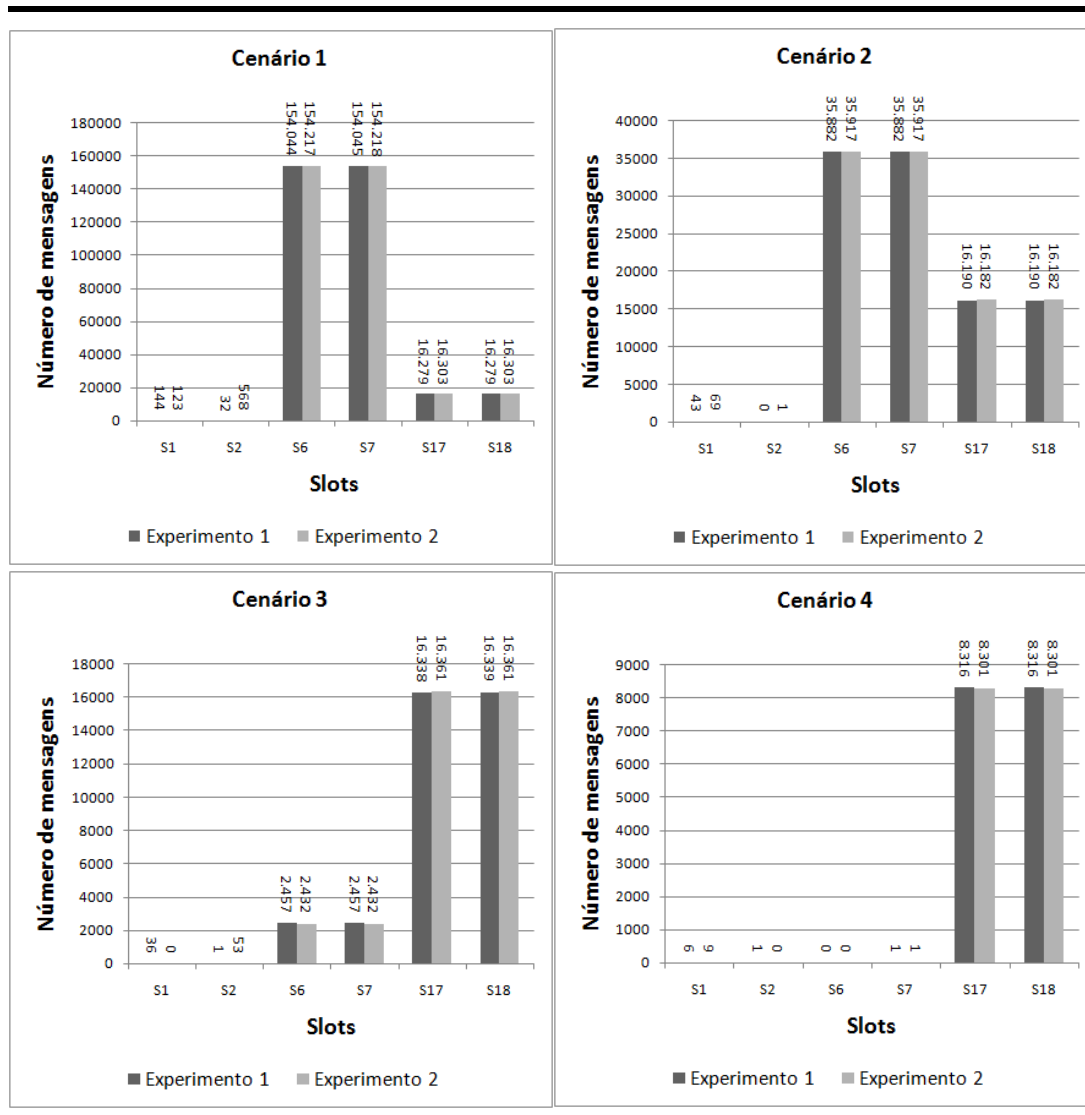


Figura 4.6: Mensagens acumuladas nos slots nos experimentos 1 e 2.

Capítulo 5

Conclusão e Trabalhos Futuros

Com o tempo, os conceitos mudam...

os sonhos mudam...

a vida muda...

Mas não se mudam princípios e valores...

Mudei e continuo igual...

Assim é o ser humano: tão coerente em suas contradições.

Jacky Correia



As empresas possuem um ecossistema de *software* composto por inúmeras aplicações, sendo que estas podem ser desenvolvidas pela própria empresa, ou adquiridas de empresas terceirizadas. Por esse motivo existe a preocupação com a integração destas aplicações, que servem de base para apoiar a tomada de decisões e também para aperfeiçoar seus processos de negócio. A integração das aplicações que compõem o ecossistema de *software* das empresas é necessária para tornar seus processos de negócios mais dinâmicos. A integração de aplicações empresariais é conhecida por EAI e proporciona metodologias, técnicas e ferramentas para projetar e implementar soluções de integração. Neste sentido, uma solução de integração visa orquestrar um conjunto de diferentes aplicações, que constituem o ecossistema de *software* das empresas, para mantê-las sincronizadas, permitindo que seus dados e funcionalidades sejam compartilhados, sem que haja a necessidade de alterar cada uma das aplicações. Para

que as soluções de integração cumpram sua função, é necessário analisar o comportamento e o desempenho das mesmas. Nessa dissertação foi analisado o comportamento e o desempenho de uma solução de integração, ainda na fase de projeto, com o auxílio da simulação. A simulação foi realizada na fase de projeto, pois deste modo tornou-se possível encontrar falhas, e corrigi-las, antes que a solução pudesse ser implementada, evitando problemas, diminuindo custos e tempo.

Uma solução de integração pode ser caracterizada como um sistema de eventos discretos, pois os componentes que estão incluídos na solução de integração utilizam um determinado tempo de execução, quando ocorre um evento, assim um evento pode alterar o estado da solução de integração proposta. Através de um sistema discreto foi possível criar um modelo de simulação, a partir do modelo conceitual da solução de integração, com auxílio da técnica matemática Redes de Petri temporizadas. Esta técnica permitiu representar sistemas e possui base matemática. O modelo de simulação desenvolvido é equivalente ao modelo conceitual que representa a solução de integração da administração pública da cidade de Huelva, na Espanha.

A equivalência utilizada para o desenvolvimento do modelo de simulação foi elaborada entre os elementos da tecnologia Guaraná DSL, e os elementos da técnica matemática Redes de Petri. A simulação da solução de integração foi realizada com o auxílio do *software* de simulação HPSim, que permitiu simular Redes de Petri temporizadas, além de fornecer as variáveis necessárias para a análise realizada durante o desenvolvimento da pesquisa. Através da simulação, foi possível analisar o comportamento e o desempenho de sistemas, de modo a encontrar os gargalos de desempenho existentes na solução.

Para garantir que o modelo de simulação proposto estivesse correto realizou-se a verificação, observando se a equivalência entre os elementos do Guaraná DSL e os elementos das Redes de Petri foi realizada corretamente. A verificação ocorreu entre o modelo conceitual e o modelo de simulação, e visou garantir que não existissem erros no modelo de simulação.

Depois de obter resultados com as simulações, realizou-se a verificação do modelo de simulação, que ocorreu comparando a quantidade de mensagens que entraram na solução, com a quantidade de mensagens que saíram pelas portas, foram filtradas, ou ficaram acumuladas nos *slots*. Esta comparação se deu através de um cálculo, onde o número de mensagens que entrou no sistema, menos as outras já citadas, resultou em zero.

Com os resultados das simulações, tornou-se viável compreender como a solução de integração poderia se comportar quando fosse implementada,

possibilitando uma análise concreta do seu comportamento, tornando possível realizar ajustes necessários antes que pudesse ser implementada. Com os resultados da simulação, foi possível observar o tempo nas transições, e através disso, analisar o acúmulo de mensagens (*tokens*) em cada lugar (*slot*).

A análise da variável tempo distinto nas transições, permitiu observar que dependendo da função que a transição desempenha na solução, o tempo associado à ela pode ser menor ou maior, sendo que este tempo refere-se ao disparo da transição. Observou-se que quando a transição possuía um tempo menor associado a ela, o *token* permanecia por menos tempo no *slot* que precedia a transição, no entanto, se a transição possuísse um tempo maior associado a ela, o *token* permanecia por mais tempo no *slot* que precedia a transição.

Já a análise da variável acúmulo de *tokens* (mensagens) em cada lugar (*slot*), permitiu observar quantas mensagens ficaram acumuladas nos *slots* que precediam as transições. *Slots* que precediam transições que possuíam menor tempo associado acumularam menos mensagens, pois as transições foram habilitadas mais vezes. Por outro lado, *slots* que precediam transições que possuíam maior tempo associado acumularam mais mensagens.

Observando-se estas variáveis, foi possível analisar o comportamento e o desempenho da solução de integração, de modo a encontrar gargalos de desempenho. Neste sentido, observou-se que os gargalos estavam presentes nas transições que possuíam maior tempo associado, pois estas demoravam mais tempo para estarem habilitadas a disparar. Portanto, conclui-se que os gargalos ocorreram nos *slots* que precediam transições com maior tempo associado. Deste modo, encontraram-se gargalos nos *slots* que precediam a transição que representava o correlacionador. Assim percebeu-se a importância de realizar a simulação ainda na fase de projeto, pois desta forma se tornou possível diminuir custos de implementação, além de facilitar a realização de alterações necessárias na solução de integração.

Como trabalhos futuros, pretende-se implementar a solução de integração e coletar dados, que permitam realizar a validação da solução de integração. Tem-se a possibilidade de realizar a simulação em outros cenários, para identificar em quais deles os gargalos de desempenho deixarão de aparecer, dependendo da carga de entrada de mensagens na solução. Além disso, tem-se a possibilidade de analisar o comportamento da solução de integração proposta, modificando o intervalo de tempo que foi associado às transições, de modo a obter resultados mais próximos da realidade, na qual a solução de integração é implementada. Ademais pode-se analisar esta solução de

integração com a utilização de outras extensões da técnica matemática Redes de Petri, como por exemplo, as Redes de Petri coloridas, pois deste modo, poderiam ser observadas diferentes variáveis considerando prioridades para as transições, diferentes tamanhos de mensagens e diferentes tipos de *tokens*, dentre outros.

Bibliografia

- [1] H. Anschuetz. *Hpsim version: 1.1. Computer software*, 1, 2001.
- [2] J. L. N. Audy e Â. F. Brodbeck. *Sistemas de informação: planejamento e alinhamento estratégico nas organizações*. Bookman Editora, 2009.
- [3] J. L. N. Audy, G. K. de Andrade e A. Cidral. *Fundamentos de sistemas de informação*. Bookman editora, 2009.
- [4] N. M. C. Barroso, J. M. Soares, G. C. Barroso, J. C. M. Mota e H. B. Neto. *Modelagem de conceitos e processos matemáticos por redes de petri coloridas: o caso da integrabilidade de funções reais*. *Boletim de Educação Matemática*, 27(45):75–95, 2013.
- [5] G. Bressan. *Modelagem e simulação de sistemas computacionais. Capítulo sobre Redes de Petri*, LARC-PCS/EPUSP, www.larc.usp.br/conteudo/universo/pcs012/modsim05.pdf, 2002.
- [6] P. N. D. Bukh e R. Jain. *The art of computer systems performance analysis, techniques for experimental design, measurement, simulation and modeling*. *Inform*s, 22(4):113–115, 1992.
- [7] J. Cardoso e R. Valette. *Redes de petri*. Editora da UFSC, 1997.
- [8] R. S. Cargnin. *Modelagem e simulação de uma solução de integração para identificação de gargalos de desempenho baseadas em formalismo matemático: uma abordagem orientada à Redes de Petri*. Dissertação de Mestrado, Universidade Regional do Noroeste do Estado do Rio Grande do Sul, 2016.
- [9] L. Chwif. *Redução de modelos de simulação de eventos discretos na sua concepção: uma abordagem causal*. Dissertação de Mestrado, Universidade de São Paulo, 1999.
- [10] L. Chwif e A. Medina. *Modelagem e simulação de eventos discretos, 4ª edição: Teoria e aplicações*. Elsevier Brasil, 2014.

- [11] L. Chwif e A. C. Medina. *Modelagem e simulação de eventos discretos*. Afonso C. Medina, 2006.
- [12] S. M. B. B. Correa. *Probabilidade e estatística*. PUC Minas Virtual, 2003.
- [13] D. O. da Penha, C. A. P. da Silva Martins e H. C. de Freitas. *Modelagem de sistemas computacionais usando Redes de Petri: aplicação em projeto, análise e avaliação*. *Anais do IV Escola Regional de Informática RJ/ES, Rio das Ostras, RJ, Brasil*, páginas 26–28, 2010.
- [14] F. O. da Silva. *Integração de sistemas e plataformas como solução para a gestão da informação de clientes*. Dissertação de Mestrado, Universidade do Porto, 2004.
- [15] H. J. R. de Carvalho, A. R. Yoshizawa, H. L. J. Pontes e A. J. V. Porto. *Análise de desempenho do trabalho multifuncional em linhas de produção, em forma de u pela modelagem e simulação usando redes de petri temporizadas*. Em *Simpósio Brasileiro de Pesquisa Operacional*, páginas 2196–2208, 2005.
- [16] M. de Fátima Costa de Souza, D. G. Gomes, G. C. Barroso, C. T. de Souza, J. A. de Castro Filho, M. C. Pequeno e R. M. C. Andrade. *LOCPN: Redes de Petri Coloridas na produção de objetos de aprendizagem*. *Revista Brasileira de Informática na Educação*, 15(4):1–14, 2007.
- [17] P. J. de Freitas Filho. *Introdução à modelagem e simulação de sistemas: com aplicações em arena*. Visual Books, 2001.
- [18] M. de Oliveira Gavira. *Simulação computacional como uma ferramenta de aquisição de conhecimento*. Dissertação de Mestrado, Universidade de São Paulo, 2003.
- [19] P. Dias. *Desenvolvimento de objectos de aprendizagem para plataformas colaborativas*. Em *Actas do VII Congreso Iberoamericano de Informática Educativa*. *Universidad de Monterrey, Monterrey*, páginas 3–12, 2004.
- [20] M. P. dos Santos. *Introdução à simulação discreta*. Editora EdUERJ, 1999.
- [21] D. Dossot, J. D’Emic e V. Romero. *Mule in action*. Manning, 2014.
- [22] R. N. Duarte. *Simulação computacional: Análise de uma célula de manufatura em lotes do setor de auto-peças*. Dissertação de Mestrado, Universidade Federal de Itajubá, 2003.

- [23] T. Facchin e M. A. Sellitto. *Medição do inventário em processo e tempo de atravessamento em manufatura por modelagem em redes de petri e diagrama de resultados*. *Revista Gestão & Produção*, São Carlos, 15 (2):307–321, 2008.
- [24] M. Fisher, J. Partner, M. Bogoevici e I. Fuld. *Spring integration in action*. Manning Publications Co., 2012.
- [25] J. W. Forrester. *Principles of systemswright allen press*, 1968.
- [26] C. R. L. Francês. *Introdução às redes de petri*. *Laboratório de Computação Aplicada, Universidade Federal do Pará*, 2003.
- [27] R. Z. Frantz, R. Corchuelo e J. González. *Advances in a DSL for application integration*. Em *Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos*, páginas 54–66, 2008.
- [28] R. Z. Frantz, A. M. R. Quintero e R. Corchuelo. *A Domain-Specific Language to Design Enterprise Application Integration Solutions*. *International Journal of Cooperative Information Systems*, 20(02):143–176, 2011.
- [29] R. Z. Frantz, S. Sawicki, F. Roos-Frantz, R. Corchuelo, V. Basto-Fernandes e I. Hernández. *Desafios para a implantação de soluções de integração de aplicações empresariais em provedores de computação em nuvem*. *XIX Jornada de Pesquisa da UNIJUI*, páginas 1–11, 2014.
- [30] R. Z. Frantz. [Enterprise Application Integration: An Easy-to-maintain Model-Driven Engineering Approach](#). University of Seville, 2012.
- [31] C. M. Grinstead e J. L. Snell. *Introduction to probability*. American Mathematical Soc., 2012.
- [32] G. Hohpe e B. Woolf. *Enterprise Integration patterns: Designing, building, and deploying messaging solutions*. Addison-Wesley Professional, 2004.
- [33] C. Ibsen e J. Anstey. *Camel in action*. Manning Publications Co., 2010.
- [34] F. Junqueira e P. E. Miyagi. [Modelagem e simulação distribuída de sistema produtivo baseados em rede de petri](#). *Sba: Controle & Automação Sociedade Brasileira de Automatica*, 20(1):1–19, 2009.
- [35] J. P. C. Kleijen. [Validation of models: statistical techniques and data availability](#). 1:647–654, 1999.

- [36] A. R. Kraisig, F. C. Welter, I. G. Haugg, R. Cargnin, F. Roos-Frantz, S. Sawicki e R. Z. Frantz. *Mathematical Model for Simulating an Application Integration Solution in the Academic Context of Unijuí University*. *Procedia Computer Science*, 100:407–413, 2016.
- [37] A. M. Law e D. W. Kelton. *Simulation modeling and analysis*, McGraw-Hill. New York, 1991.
- [38] E. A. Lima, R. Lüders e L. A. Künzle. [Uma abordagem intervalar para a caracterização de intervalos de disparo em redes de petri temporais](#). *Sba: Controle & Automação Sociedade Brasileira de Automatica*, 19 (4):379–394, 2008.
- [39] E. A. Lima, R. Lüders e L. A. Künzle. *Análise de redes de petri temporais usando tempo global*. *VII Simpósio Brasileiro de Automação Inteligente*, (7):1–8, 2005.
- [40] R. G. Lins. *Melhorias nos processos de gestão e de fabricação de uma indústria metalmecânica utilizando redes de petri auxiliada por simulação discreta*. Dissertação de Mestrado, Universidade de Taubaté, 2008.
- [41] D. S. Linthicum. *Enterprise application integration*. Addison-Wesley Professional, 2000.
- [42] G. B. Lyra, B. I. L. Garcia, S. M. D. S. Piedade, G. C. Sedyama e P. C. Sentelhas. *Regiões homogêneas e funções de distribuição de probabilidade da precipitação pluvial no estado de táchira, venezuela*. *Pesquisa Agropecuária Brasileira*, 41(2):205–215, 2006.
- [43] P. R. M. Maciel, R. D. Lins e P. R. F. Cunha. *Introdução às redes de petri e aplicações*. UNICAMP-Instituto de Computação, 1996.
- [44] N. Marranghello. *Redes de petri: Conceitos e aplicações*. Editora UNESP, 2005.
- [45] V. M. M. Martins. *Integração de sistemas de informação: Perspectivas, normas e abordagens*. Dissertação de Mestrado, Universidade do Minho, 2006.
- [46] P. Merlin e D. Farber. [Recoverability of communication protocols-implications of a theoretical study](#). *IEEE transactions on Communications*, 24(9):1036–1043, 1976.

- [47] M. M. Miyagi, P. E. Miyagi e M. Kisil. *Modelagem e análise de serviços de saúde baseados em Redes de Petri interpretadas*. *Production*, 11(2):23–39, 2001.
- [48] R. S. Pressman. *Engenharia de software*. McGraw Hill Brasil, 2011.
- [49] R. M. Protil. *Estudo na república federal da alemanha*. *Revista de Economia*, 31(2):113–134, 2005.
- [50] C. Ramchandani. *Analysis of asynchronous concurrent systems by timed petri nets*. Tese Doutoral, Massachusetts Institute of Technology, 1974.
- [51] D. A. Rezende. *Engenharia de software e sistemas de informação*. Brasport, 2005.
- [52] F. Roos-Frantz, M. Binelo, R. Z. Frantz, S. Sawicki e V. B. Fernandes. *Using petri nets to enable the simulation of application integration solutions conceptual models*. Em *17th International Conference on Enterprise Information Systems, Barcelona*, páginas 87–96, 2015.
- [53] R. G. Sargent. *Verification and validation of simulation models*. *Journal of simulation*, 7(1):12–24, 2013.
- [54] S. Sawicki, R. Z. Frantz, V. M. B. Fernandes, F. Roos-Frantz, I. Yevseyeva e R. Corchuelo. *Characterising Enterprise Application Integration Solutions as Discrete Event System*. *IGI Global*, páginas 261–288, 2015.
- [55] J. Sifakis. *Use of petri nets for performance evaluation*. *Acta Cybernetica*, 4(1978):185–202, 1980.
- [56] R. M. S. Soeiro. *Engenharia Informática e de Computadores*. Dissertação de Mestrado, Universidade Técnica de Lisboa, 2009.
- [57] J. Weiss. *Aligning relationships: Optimizing the value of strategic outsourcing*. *Global services report, IBM*, 2005.
- [58] F. C. Welter e A. R. Kraisig. *Análise comparativa entre redes de petri, cadeias de markov e teoria das filas*. Em *IV Seminário de Formação Científica e Tecnológica*, páginas 38–40, 2010.
- [59] F. C. Welter, A. R. Kraisig e R. Z. Frantz. *Uso de Redes de Petri para identificação de gargalos de desempenho em soluções de integração de aplicações: caso de estudo na administração pública de Huelva*. Em *I Mostra de Pós Graduação da Univates, (no prelo)*, 2016.

- [60] F. C. Welter, A. R. Kraisig e R. Z. Frantz. *Framework de comparação entre técnicas matemáticas*. Em *Salão do Conhecimento da UNIJUI*, páginas 1–6, 2016.
- [61] A. K. Wiesner. *Modelagem e simulação de uma solução de integração para identificação de gargalos de desempenho baseadas em formalismo matemático: uma abordagem orientada à Teoria das Filas*. Dissertação de Mestrado, Universidade Regional do Noroeste do Estado do Rio Grande do Sul, 2016.
- [62] M. C. Yamada, A. J. V. Porto e R. Y. Inamasu. *Aplicação dos conceitos de modelagem e de Redes de Petri na análise do processo produtivo da indústria sucroalcooleira*. *Revista Pesquisa Agropecuária Brasileira*. Brasília, 37(6):809–820, 2002.

This document was typeset on April 26, 2017 using class `RG-BOK` α 2.14 for $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}_{2\epsilon}$. As of the time of writing this document, this class is not publicly available. Only members of [The Distributed Group \(TDG\)](#) and the [Applied Computing Research Group \(ACR\)](#) are allowed to typeset their documents using this class.