JOHN WESLEY SOARES DE LIMA

DESENVOLVIMENTO E AVALIAÇÃO DE UMA FERRAMENTA INTERATIVA BASEADA EM EXEMPLOS PARA O APRENDIZADO DE MODELAGEM DE SISTEMAS USANDO REDES DE PETRI

RECIFE-PE

2021

Universidade Federal Rural de Pernambuco

Pró-Reitoria de Pesquisa e Pós-Graduação

Programa de Pós-Graduação em Informática Aplicada

Desenvolvimento e Avaliação de uma Ferramenta Interativa Baseada em Exemplos para o Aprendizado de Modelagem de Sistemas Usando Redes de Petri

Dissertação apresentada ao Programa de Pós-Graduação em Informática Aplicada da Universidade Federal Rural de Pernambuco como exigência parcial à obtenção do título de Mestre em Informática Aplicada.

Orientador: Prof. Dr. Ermeson Andrade

Co-orientadora: Profa. Dra. Taciana Pontual Falcão

Recife-PE

2021

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

S676d SOARES DE LIMA, JOHN WESLEY
Desenvolvimento e Avaliação de uma Ferramenta Interativa Baseada em Exemplos para o Aprendizado de Modelagem de Sistemas Usando Redes de Petri / JOHN WESLEY SOARES DE LIMA. - 2021.
95 f. : il.

Orientador: Ermeson Andrade.
Coorientadora: Taciana Pontual Falcao.
Inclui referências.

Dissertação (Mestrado) - Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, 2022.

1. Tecnologias na Educação. 2. Modelagem. 3. Rede de Petri. 4. Ferramenta. I. Andrade, Ermeson, orient. II. Falcao, Taciana Pontual, coorient. III. Título

CDD 004

Universidade Federal Rural de Pernambuco

Pró-Reitoria de Pesquisa e Pós-Graduação

Programa de Pós-Graduação em Informática Aplicada

Desenvolvimento e Avaliação de uma Ferramenta Interativa Baseada em Exemplos para o Aprendizado de Modelagem de Sistemas Usando Redes de Petri

JOHN WESLEY SOARES DE LIMA

Dissertação apresentada ao Programa de Pós-Graduação em Informática Aplicada da Universidade Federal Rural de Pernambuco e julgada aprovada para a obtenção do título de Mestre em Informática Aplicada. Data de defesa: 20/08/2021.

Orientador:

_____

Prof. Dr. Ermeson Andrade
Universidade Federal Rural de Pernambuco – UFRPE

Coorientadora:

_____

Profa. Dra. Taciana Pontual Falcão
Universidade Federal Rural de Pernambuco – UFRPE

Banca Examinadora:

_____

Prof. Dr. Ricardo André Cavalcante de Souza
Universidade Federal Rural de Pernambuco – UFRPE

_____

Prof. Dr. Jamilson Ramalho Dantas
Universidade Federal de Pernambuco - UFPE

Recife-PE
2021

AGRADECIMENTOS

- A Deus;
- Ao meu orientador Ermeson Andrade e a minha co-orientadora Taciana Pontual Falcão por terem paciência e sempre estarem dispostos a me orientar no projeto.
- Aos meus pais e a minha família;
- Aos professores, funcionários e colegas do PPGIA;
- A FACEPE pelo apoio neste e em inúmeros outros trabalhos realizados pela Universidade Federal Rural de Pernambuco.

RESUMO

Os sistemas computacionais têm se tornado cada vez mais complexos e isso tem demandado o desenvolvimento de técnicas que garantam certas características, como alta disponibilidade e baixo custo. Assim, o interesse pela utilização de modelos computacionais tem aumentado nos últimos anos, visto que eles podem ser usados em situações em que é muito caro e até mesmo impossível testar ou medir as diversas características dos sistemas computacionais (ex.: disponibilidade ou confiança). No entanto, um grande desafio no uso de tais modelos é o seu entendimento, visto que não são intuitivos e requerem um considerável esforço de aprendizado da notação utilizada. Muitas vezes, a abordagem de ensino expositivo em sala de aula, nos cursos da área de computação, é insuficiente para garantir a aprendizagem de modelagem computacional. Ferramentas educativas interativas baseadas em exemplos podem auxiliar a aprendizagem por meio de exercício e prática. Este trabalho aborda o desenvolvimento e avaliação da TryRdP, uma ferramenta *web* gratuita, interativa e baseada em exemplos para auxiliar na aprendizagem de modelagem computacional usando Redes de Petri (RdP), que é um conteúdo que integra currículos de cursos de ciência da computação, engenharia da computação e afins. As RdP são formalismos matemáticos largamente usados para modelar, analisar e projetar sistemas diversos, entre eles, sistemas computacionais. Para avaliar a eficácia da ferramenta, foram realizados experimentos com alunos e professores de cursos superiores de computação, investigando a usabilidade e utilidade da ferramenta educativa proposta. Os resultados mostraram que a ferramenta atende aos requisitos de usabilidade, utilidade e aplicação didático-pedagógica no ensino de modelagem usando as redes de Petri.

Palavras-chave: Tecnologias na Educação, Modelagem, Rede de Petri, Ferramenta.

ABSTRACT

Computational systems have become increasingly complex, and this has required the development of techniques to guarantee certain characteristics such as high availability and low cost. Thus, the interest in the use of computational models has increased in recent years, given that they can be used in situations where it is very expensive or even impossible to test or measure the various characteristics of computer systems (e.g., availability or reliability). However, a great challenge in the use of such models is their understanding, since they are not intuitive and require a considerable effort to learn the notation used. Often, the lecture approach in the classroom, in computer science majors, is insufficient to guarantee the learning of computational modeling. Interactive educational tools based on examples can assist learning through drill and practice. This work addresses the development and evaluation of TryRdP, a free, interactive, and example-based web tool to assist the learning of computational modeling using Petri Nets, which is a content that integrates curricula in computer science, computer engineering, and other courses. Petri Nets are mathematical formalisms widely used to model, analyze and design several systems, including computer systems. In order to evaluate the effectiveness of the tool, experiments were carried out with students and instructors from higher education computer science majors, investigating the usability and utility of the proposed educational tool. The results showed that the tool meets the requirements of usability, utility and didactic-pedagogical application in the teaching of modeling using Petri nets.

Keywords: Educational Technology, Modeling, Petri net, Tool.

LISTA DE FIGURAS

Figura 1: elementos básicos de uma RdP. 20
Figura 2: modelo de estoque com loja de produtos e seus estados. 23
Figura 3: exemplo de uma RdP e seu formalismo matemático. 25
Figura 4: estrutura básica da sequência. 26
Figura 5: estrutura básica da distribuição. 26
Figura 6: estrutura básica da junção. 27
Figura 7: estrutura básica da escolha não-determinística. 27
Figura 8: modelo que representa o envio de sinais em paralelo. 28
Figura 9: modelo de teste de míssil. 30
Figura 10: modelo de um sistema de transporte urbano via metrô. 31
Figura 11: exemplo de uma GSPN e arco inibidor. 32
Figura 12: GSPN com quatro transições imediatas. 33
Figura 13: falha e reparo de um sensor. 34
Figura 14: sistema de fila com dois servidores. 35
Figura 15: metodologia proposta. 47
Figura 16: arquitetura da TryRdP. 52
Figura 17: menu com os três tópicos (barra vertical à esquerda). 55
Figura 18: exemplo de exercício de quebra cabeça. 56
Figura 19: mensagem indicando o erro cometido. 57
Figura 20: exemplo de exercício de modelagem. 58
Figura 21: propriedades de uma transição temporizada em um exercício de modelagem. 58
Figura 22: alerta de erro relacionado ao formalismo das RdP. 59
Figura 23: alerta de erro de digitação. 59
Figura 24: propriedades da transição apresentada na tópico 1. 60
Figura 25: elementos nos exercícios de modelagem na tópico1. 61
Figura 26: exemplo de exercício de modelagem de nível avançado. 62
Figura 27: sistema de estoque. 64
Figura 28: exemplo básico de um estoque sem limite de capacidade. 64
Figura 29: modelo de estoque com limite de 50 produtos. 66
Figura 30: exemplo de um sistema de fila. 67

Figura 31: modelo que representa o comportamento de falha e reparo de um computador. 68
Figura 32: modelo *Hot Standby*. 69
Figura 33: fila com um servidor e entrada a partir de três fontes diferentes. 70
Figura 34: modelo de disponibilidade com *NoBreak.* 71
Figura 35: Resultados da avaliação feita pelos professores. 73
Figura 36: resultados das avaliações feitas pelos estudantes. 81

LISTA DE TABELAS

Tabela 1: Comparação das notas dos 15 estudantes (graduação e pós-graduação) antes e depois de usarem a ferramenta. 79
Tabela 2: Comparação das notas dos 11 estudantes de graduação antes e depois de usarem a ferramenta. 79
Tabela 3: Comparação das notas dos 7 estudantes da graduação que não tiveram contato com o assunto de redes de Petri antes de serem convidados. 80
Tabela 4: Comparação das notas dos 4 estudantes de pós-graduação antes e depois de usarem a ferramenta. 80
Tabela 5: Teste de normalidade dos dois grupos de dados. 80

LISTA DE QUADROS

Quadro 1: caracteres reservados relacionados aos cálculos das métricas. 37
Quadro 2: caracteres reservados relacionados às operações nas métricas. 37
Quadro 3: tópicos usados na avaliação de professores e estudantes. 50
Quadro 4: sugestões/opiniões dos professores quanto à aprendizagem do conteúdo e nível de dificuldade..... 74
Quadro 5: sugestões/opiniões dos professores quanto à visualização do progresso 74
Quadro 6: sugestões/opiniões dos professores quanto à facilidade e satisfação de uso da ferramenta 75
Quadro 7: sugestões/opiniões dos professores quanto à completude do conteúdo . 76
Quadro 8: sugestões/opiniões dos professores quanto à clareza das mensagens de erro e feedbacks..... 77
Quadro 9: sugestão dos estudantes quanto à liberdade (ou flexibilidade) na navegação..... 82
Quadro 10: sugestões dos estudantes quanto à facilidade de uso da ferramenta 83

LISTA DE ABREVIATURAS

AADL	*Architecture Analysis and Design Language*
CIO	*Chief Information Officer*
CPS	*Cyber-Physical System*
CSS	*Cascading Style Sheets*
GSPN	*Generalized Stochastic Petri Nets*
HTML	*HyperText Markup Language*
IDE	*Integrated Development Environment*
IoT	*Internet of Things*
OECD	*Organisation for Economic Co-operation and Development*
POO	Programação Orientada a Objetos
RdP	Redes de Petri
SO	Sistemas Operacionais
SPN	*Stochastic Petri Net*
SysML	*Systems Modeling Language*
UML	*Unified Modeling Language*
TI	Tecnologia da Informação
XML	*Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO 14
1.1 CONTEXTO 14
1.2 MOTIVAÇÃO 16
1.3 OBJETIVOS 17
1.4 ESTRUTURA DA DISSERTAÇÃO 18
2 FUNDAMENTOS TEÓRICOS 19
2.1 REDES DE PETRI 19
2.1.1 Elementos Básicos de Uma RdP 20
2.1.2 Definição Formal de Uma RdP 22
2.1.3 Redes Elementares 25
2.1.4 Exemplos com as Redes Elementares 28
2.1.5 Rede de Petri Estocástica Generalizada (GSPN) 31
2.1.6 Métricas 36
2.2 INFORMÁTICA NA EDUCAÇÃO 38
3 TRABALHOS RELACIONADOS 42
3.1 FERRAMENTAS EDUCATIVAS PARA O ENSINO DE DISCIPLINAS DA COMPUTAÇÃO 42
3.2 FERRAMENTAS DE MAPEAMENTO DE RDP 43
3.3 FERRAMENTAS DE MODELAGEM E ANÁLISE 44
4 METODOLOGIA 46
4.1 AVALIAÇÃO DA FERRAMENTA 48
5 A FERRAMENTA TRYRDP 52
5.1 ARQUITETURA DA TRYRDP 52
5.2 FUNCIONAMENTO DA FERRAMENTA 54
5.3 CONTEÚDO ABORDADO NA FERRAMENTA 63
5.3.1 Tópico 1 – Introdução 63
5.3.2 Tópico 2 – Modelagem Básica 65
5.3.3 Tópico 3 – Modelagem Avançada 69
6 AVALIAÇÃO DA FERRAMENTA 72
6.1 AVALIAÇÃO COM PROFESSORES 72

6.2 AVALIAÇÃO COM ESTUDANTES 78
6.2.1 Avaliação de Aprendizagem 78
6.2.2 Avaliação de Usabilidade 81
7 CONCLUSÕES 84
7.1 LIMITAÇÕES 85
7.2 TRABALHOS FUTUROS 86
REFERÊNCIAS 87

1 INTRODUÇÃO

1.1 CONTEXTO

O desenvolvimento tecnológico está associado com mudanças sociais e econômicas, além de possibilitar uma maior competição global entre países mediante a criação de novos produtos e serviços (OECD/EUROSTAT, 2005; KORMANN e SUBERG, 2021). Nesse sentido, cientistas e engenheiros têm juntado esforços na pesquisa e desenvolvimento de novos produtos e serviços, difundindo a inovação na economia global (DIETZ et al., 2000). Entre as diversas áreas de tecnologia existentes, a área da Tecnologia da Informação (TI) vem se destacando pelo seu rápido crescimento (U.S. BUREAU OF LABOR STATISTICS, 2013, 2020). Segundo as projeções estatísticas mais recentes mostradas pelo U.S. Bureau of Labor Statistics (2020), a quantidade de empregos na área de TI crescerá 11% entre os anos de 2019 e 2029 e esse crescimento é maior do que a média das outras áreas de ocupação. Vale ressaltar que a projeção mostrada não levou em consideração o impacto causado na economia global pela pandemia devida ao vírus COVID-19, que iniciou no final do ano de 2019 na cidade chinesa de Wuhan (MO et al., 2020; MEI et al., 2020). No entanto, segundo a *Harvey Nash / KPMG CIO Survey 2020*, logo no início da pandemia as empresas gastaram 15 bilhões de dólares a mais por semana com TI para permitir trabalho doméstico seguro, e isso foi considerado como “um dos maiores surtos de investimento em tecnologia da história” (KPMG, 2020). Além disso, 70% dos CIOs (*Chief Information Officer*) acreditam que essa mudança positiva para com a área de TI será permanente (TECH MONITOR, 2020).

Em paralelo a esse rápido crescimento, sistemas computacionais têm se tornado cada vez mais complexos. Sistemas como Ciberfísicos (*Cyber-Physical System* - CPS) e Internet das Coisas (*Internet of Things* - IoT) vêm sendo largamente estudados e prometem impactar a área de TI (ROSSITER, 2018; RHO; VASILAKOS e CHEN, 2016). Tais sistemas estão associados, por exemplo, à chamada “Indústria 4.0”. O termo “Indústria 4.0” surgiu na Alemanha em 2011 e está relacionado ao que há de mais moderno na área de TI aplicada à indústria (FALANI, AGUIAR e FORNO, 2020). Um sistema industrial 4.0 é complexo e tem por base quatro pilares: sistema ciberfísico,

internet das coisas, computação em nuvem e *big data* (ZHOU K; LIU e ZHOU L, 2015). Além dos sistemas citados, existem também aqueles voltados para saúde e que são bem estudados atualmente. Os termos “Saúde Eletrônica” (*e-health*) e “Saúde Móvel” (*m-health*) são largamente utilizados e se referem, respetivamente, ao uso da TI (sistemas distribuídos, IoT, computação em nuvem e etc.) aplicada à saúde, e ao uso de aplicativos móveis no auxílio ao diagnóstico e na prática da medicina em geral (CAVALCANTI et al., 2020; MARCOLINO et al., 2018; FARAHANI et al., 2017).

Sistemas computacionais complexos, como os citados acima, requerem um alto nível de qualidade, desempenho, disponibilidade e tolerância a falhas. Assim, a busca por tais sistemas tem motivado o desenvolvimento de métodos de modelagem. Em muitas áreas da computação como, por exemplo, sistemas embarcados e distribuídos, os ambientes computacionais não são analisados diretamente, mas sim, por meio de modelos (TRIVEDI, ANDRADE e MACHIDA, 2012). Esses modelos podem ser usados para situações em que é muito caro e até mesmo impossível testar ou medir as diversas características dos sistemas computacionais (ex.: disponibilidade ou confiança) (ANDRADE et al., 2017).

Segundo Lima, Callou e Andrade (2021, p. 2):

> A vantagem de se usar modelos é que eles nos permitem abstrair um problema complexo de modo a torná-lo mais simples. Isso consiste em desconsiderar detalhes sem importância para se concentrar nas características essenciais do problema. De maneira mais específica, a modelagem de sistemas é um estudo multidisciplinar que visa representar os sistemas através de modelos, de modo que os mesmos possam ser analisados sem a implementação real do sistema.

Essa representação é obtida usando-se métodos de abstração por meio do processo de modelagem. Redes de Petri - RdP (MURATA, 1989), Diagrama de Confiabilidade (HIREL et al., 2000), e Árvore de Falhas (BENNETTS, 1975) são exemplos de métodos de modelagem formais que têm sido largamente usados na modelagem e análise dos mais variados tipos de sistemas computacionais. Tais métodos são apoiados por fundamentos matemáticos sólidos, que suportam sua semântica precisa, estimulam a avaliação quantitativa e fornecem suporte para verificações de propriedades qualitativas.

Entre os vários métodos de modelagem formal existentes, as RdP destacam-se como uma ferramenta gráfica e matemática eficiente na modelagem e análise de sistemas complexos (MURATA, 1989). As RdP têm a capacidade de modelar com facilidade sistemas nos quais há ocorrência de paralelismo, sequenciamento, escolha não-determinística, sincronismo, compartilhamento de recursos, concorrência e exclusão mútua. Mais especificamente, adotamos neste trabalho uma extensão das redes de Petri denominada de Redes de Petri Estocásticas Generalizadas (*Generalized Stochastic Petri Nets* – GSPN) (CHIOLA et al., 1993; MARSAN; CONTE e BALBO, 1984). As GSPN podem ser adotadas para análise de desempenho e dependabilidade de sistemas dinâmicos. Esse formalismo matemático também permite que métricas sejam aferidas tanto via simulação, quanto pela análise do espaço de estados. Adicionalmente, as GSPN provêm diversos recursos para modelagem, tais como funções de guarda, pesos, prioridade, entre outros.

1.2 MOTIVAÇÃO

As redes de Petri são um assunto abordado em disciplinas nos cursos de graduação e pós-graduação em computação e engenharia. No entanto, o processo de compreensão das redes de Petri, independentemente de suas extensões, nem sempre é fácil, visto que tal formalismo é baseado em representações matemáticas que podem não ser intuitivas. O modelo tradicional de aulas baseado majoritariamente em exposições teóricas, nem sempre é adequado para ensinar redes de Petri por conta do alto nível de abstração do assunto, o que pode tornar o processo de aprendizado difícil e desmotivador. Nesse sentido, para auxiliar na compreensão das redes de Petri, é possível recorrer a ferramentas educativas que estão se tornando cada vez mais interativas e customizadas, objetivando apoiar o ensino de conteúdos de natureza abstrata (COSTA et al., 2017). Com a utilização de ferramentas desse tipo, o aluno tem a possibilidade de explorar visualizações e interagir com modelos, beneficiando-se do feedback proporcionado pela ferramenta, e assim engajando-se em um processo de aprendizagem autônoma e personalizada.

Embora existam várias ferramentas para modelagem e análise de redes de Petri (SILVA et al., 2015), não identificamos ferramentas que auxiliem o processo de ensino e aprendizado de modelagem computacional usando tal formalismo. A literatura

existente foca principalmente no mapeamento de linguagens de alto nível em redes de Petri com a finalidade de facilitar o seu uso. Desse modo, o presente trabalho visa suprir essa falta de ferramentas educativas voltada para o aprendizado de modelagem usando redes de Petri.

1.3 OBJETIVOS

Este trabalho apresenta TryRdP[1] - uma ferramenta baseada na *web* para auxiliar no aprendizado de modelagem de sistemas por meio de redes de Petri. A ferramenta foi desenvolvida de modo que ela possa ser usada tanto de modo autônomo quanto em um curso formal mediado por um professor. Esta dissertação também almeja avaliar a eficácia da ferramenta através de experimentos realizados com alunos e professores da área de computação.

De modo mais específico os objetivos deste trabalho são:

- Propor e desenvolver uma ferramenta educativa baseada em exemplos para o ensino e a aprendizagem de modelagem usando as RdP;
- Avaliar a usabilidade da ferramenta considerando a percepção do público-alvo;
- Avaliar a aplicação didático-pedagógica da ferramenta.

Partes dos resultados desta pesquisa foram apresentadas nas seguintes publicações:

- *TryRdP: uma Ferramenta para o Aprendizado de Modelagem de Sistemas usando Redes de Petri.*[2] Esse artigo foi publicado nos anais do Simpósio Brasileiro de Educação em Computação (EduComp) ocorrido no ano de 2021.
- *Teoria de Filas e Rede de Petri Estocástica: Um tutorial.*[3] Esse artigo foi publicado na revista *Research, Society and Development* (RSD).

[1] Ferramenta disponível em: https://tryrdp.herokuapp.com/apres.html

[2] DOI: https://doi.org/10.5753/educomp.2021.14504

[3] DOI: https://doi.org/10.33448/rsd-v10i3.12826

- *Desenvolvimento e Avaliação de uma Ferramenta Interativa Baseada em Exemplos para o Aprendizado de Modelagem de Sistemas Usando Redes de Petri*[4]. Esse artigo foi publicado na Revista Brasileira de Informática na Educação (RBIE).

1.4 ESTRUTURA DA DISSERTAÇÃO

Esta dissertação tem sete capítulos e, além do presente capítulo, ela está organizada como segue. O Capítulo 2 apresenta conceitos teóricos fundamentais sobre as redes de Petri e informática na educação. O Capítulo 3 aborda ferramentas existentes relacionadas ao tema. O Capítulo 4 apresenta como a ferramenta TryRdP foi desenvolvida. O Capítulo 5 detalha o funcionamento da ferramenta. O Capítulo 6 detalha a avaliação da ferramenta realizada com alunos e professores voluntários, bem como os resultados obtidos. Por fim, o Capítulo 7 conclui este trabalho e descreve trabalhos futuros.

[4] DOI: https://doi.org/10.5753/rbie.2021.2072

2 FUNDAMENTOS TEÓRICOS

Neste capítulo serão mostrados os fundamentos teóricos deste trabalho: redes de Petri e informática na educação.

2.1 REDES DE PETRI

As redes de Petri originaram-se da tese de doutorado intitulada *Kommunikation mit Automaten* (Comunicação com Autômatos), de Carl Adam Petri em 1962 na *Darmstad University* (PETRI, 1962). A partir deste trabalho, esse formalismo foi sendo utilizado em diferentes áreas, tais como Ciência de Computação, Engenharia Elétrica, Química e Administração (MACIEL; LINS e CUNHA, 1996). A respeito do desenvolvimento histórico das RdP, Maciel, Lins e Cunha (1996, p. 1) afirmam:

> O trabalho de Petri atraiu a atenção de A. W Holt que, em conjunto com outros pesquisadores, desenvolveu muito a teoria, notação e representação das redes de Petri. De 1970 a 1975, o grupo de estrutura da computação do MIT foi o mais ativo na condução da pesquisa sobre redes de Petri. Em 1975, houve uma conferência sobre redes de Petri, no entanto não houve publicação nos anais. Em 1979, pesquisadores de vários países europeus reuniram-se em Hamburgo para um curso avançado sobre a Teoria Geral das Redes de Processos e Sistemas. Seguiram-se diversos outros trabalhos propondo alterações ao modelo original, tais como redes com arco inibidor, e redes temporizadas determinísticas e estocásticas. Hoje em dia, rede de Petri é considerada uma técnica para especificação de sistemas concorrentes consolidada. Grupos de pesquisa em todo o mundo têm redes de Petri como tema, desenvolvendo estudos sobre seus aspectos teóricos e suas aplicações.

Entre as diversas vantagens no uso das RdP, podemos destacar as seguintes (GIRAULT e VALK, 2001):

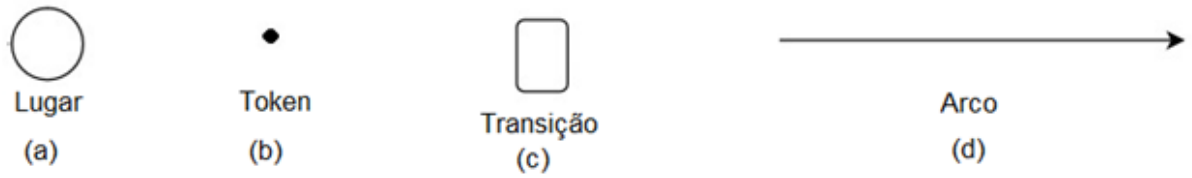
- São um formalismo de modelagem que permite representação gráfica e são fundamentados matematicamente;

- Fornecem mecanismos de abstração que são de grande importância para projetos de sistemas complexos;
- Por terem sido utilizadas em diversas áreas, vários resultados são encontrados na literatura para os diversos domínios da sua aplicação;
- Existe uma grande variedade de ferramentas para modelagem, análise e verificação disponíveis para as RdP.

As RdP são um formalismo matemático usado para descrever sistemas que possuem características como concorrência, conflito, sincronização e exclusão mútua. Uma rede de Petri é graficamente desenhada como um multigrafo bipartido direcionado e que possui apenas dois tipos de nós: lugares e transições representados, respectivamente, por círculos e retângulos (ARTEIRO et al., 2007). Abaixo segue a descrição dos elementos básicos de uma RdP.

2.1.1 Elementos Básicos de Uma RdP

Figura 1: elementos básicos de uma RdP.



Fonte: o autor.

Os elementos básicos de uma RdP podem representar elementos passivos ou ativos de um sistema real. A seguir, cada um desses elementos é detalhado (MURATA, 1989; MACIEL; LINS e CUNHA, 1996):

- **Lugares** (ver Figura 1 (a)). São graficamente representados por círculos. Semanticamente, representam os estados possíveis de um sistema. Esses elementos representam componentes passivos da rede e têm como principal função o armazenamento de *tokens* (também conhecidos como marcas), os quais são removidos e adicionados na medida em que as transições são disparadas. O lugar cuja marcação influencia o disparo de uma transição é chamado de pré-condição daquela transição. O lugar que tem sua marcação afetada pelo disparo de uma transição é chamado de pós-condição daquela

transição. Os lugares representam recursos, estados ou variáveis do sistema. Em sistemas computacionais, lugares podem representar recursos como servidores ou componentes de um servidor. A existência de um ou mais tokens em um lugar pode representar a disponibilidade do recurso.

- **Tokens** (ver Figura 1 (b)). São representados graficamente por círculos pequenos preenchidos (também podem ser representados por números), onde os mesmos residem em lugares e são semanticamente utilizados para especificar o estado da RdP. A distribuição de *tokens* nos lugares da rede determina o estado em que o sistema se encontra em um dado momento.
- **Transições** (ver Figura 1 (c)). São graficamente representadas por barras. Semanticamente, representam elementos ativos da rede, isto é, as ações ou os eventos realizados pelo sistema. Uma transição só estará habilitada se todas as suas pré-condições estiverem satisfeitas. Uma vez que a transição satisfaz todas as pré-condições, essa transição poderá disparar, removendo uma determinada quantidade de *tokens* dos lugares e colocando em outros, gerando assim as pós-condições. Em sistemas computacionais, as transições podem representar as falhas e reparos dos servidores.
- **Arcos** (ver Figura 1 (d)). São representados graficamente por setas e simbolizam o fluxo de *tokens* pela rede. Um arco interliga um lugar a uma transição ou uma transição a um lugar, mas nunca ocorre interligação entre elementos semelhantes, ou seja, lugar com lugar ou transição com transição. Quando os lugares estão conectados a uma transição no sentido lugar-transição, denominam-se lugar de entrada daquela transição. Porém, quando os lugares estão conectados a uma transição no sentido transição-lugar, denominam-se lugar de saída. Os arcos também possuem pesos. Os pesos indicam a quantidade de *tokens* que uma transição irá remover de um lugar e/ou a quantidade de *tokens* que a transição irá colocar em um outro lugar. Por exemplo, observe a RdP na Figura 3 (b), a rede contém um arco de peso 3 (note o rótulo no arco com o número 3) ligando o lugar *p0* à transição *t0* (sentido lugar-transição) e, também, a rede contém um arco de peso 1 ligando a transição *t0* ao lugar *p2* (sentido transição-lugar). Vale ressaltar que

para arcos com peso 1, não há necessidade de rotular o arco com o número 1.

2.1.2 Definição Formal de Uma RdP

Segundo Murata (1989), uma RdP é uma 5-tupla, $PN = (P, T, F, W, M_0)$, onde:

- $P = \{p_1, p_2, \dots, p_n\}$, é um conjunto finito de lugares;
- $T = \{t_1, t_2, \dots, t_n\}$, é um conjunto finito de transições;
- $F \subseteq (P \times T) \cup (T \times P)$, é um conjunto de arcos (relação de fluxo);
- $W: F \rightarrow \{1,2,3,4 \dots n\}$, é a função de atribuição de pesos aos arcos;
- $M_0: P \rightarrow \{0,1,2,3 \dots\}$, é a marcação inicial;
- Os conjuntos P e T devem ter a seguinte relação: $P \cap T = \emptyset$ e $P \cup T \neq \emptyset$.

Ainda, segundo Murata (1989), os disparos de uma transição devem obedecer às seguintes regras:

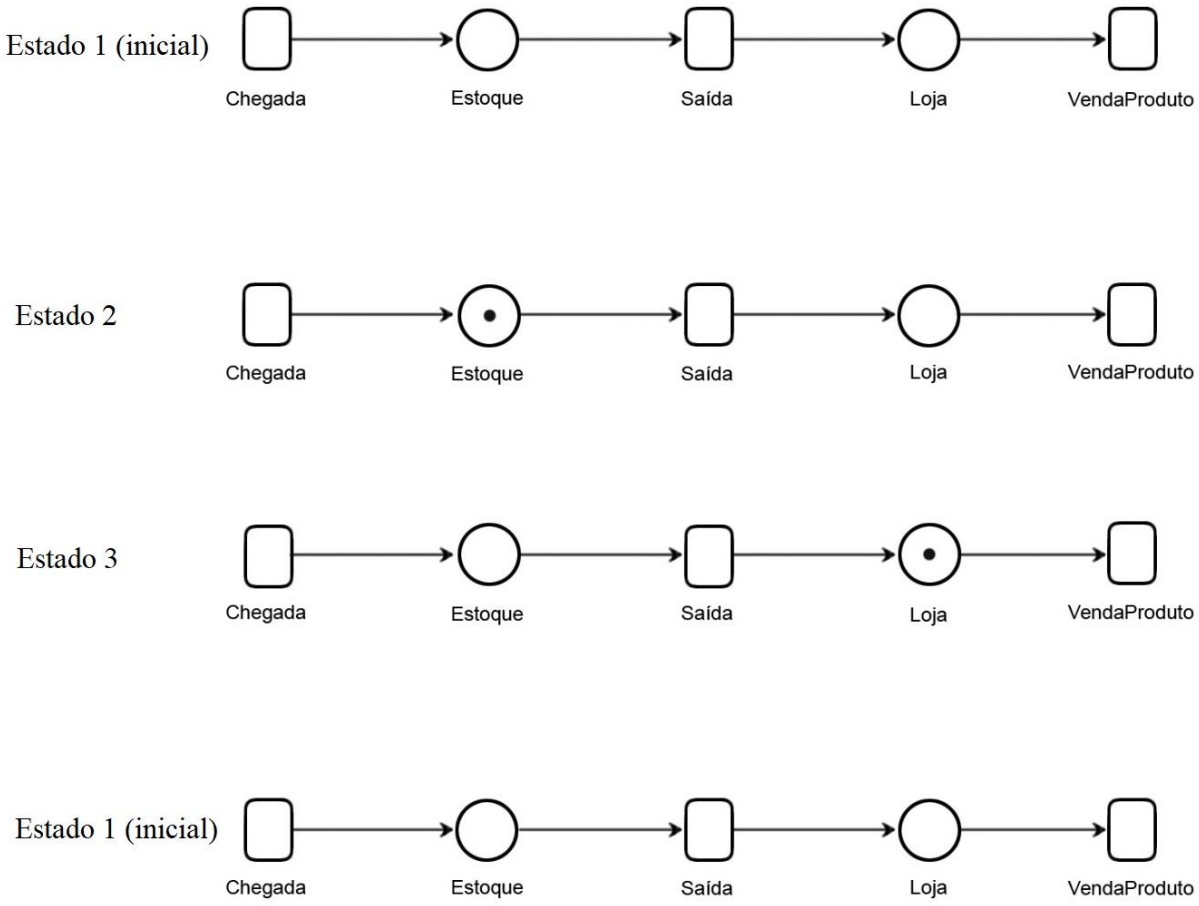
- Diz-se que uma transição *t1* é habilitada para o disparo se cada lugar de entrada *p1* de *t1* é marcado com pelo menos *w(p1, t1) tokens*, onde *w(p1, t1)* é o peso do arco que conecta *p1* e *t1* no sentido lugar-transição.
- Uma transição habilitada pode ou não disparar. Isso depende de um ou vários eventos acontecerem ou não.
- Um disparo de uma transição habilitada *t1* remove *w(p1, t1) tokens* de cada lugar de entrada *p1* de *t1*, e adiciona *w(t1, p1) tokens* a cada lugar de saída *p2* de *t1*, onde *w(t1, p2)* é o peso de o arco que conecta *t1* e *p2* no sentido transição-lugar.

Exemplo de uma RdP e Funcionamento de Disparo

Suponha que em um sistema simples um produto passa pelas seguintes etapas: 1º- entre no estoque; 2º- sai do estoque e vai para uma loja; 3º- e, por fim, ele é vendido. O modelo de RdP para esse sistema pode ser descrito por três estados, os quais são apresentados na Figura 2. A transição *Chegada* representa o evento da chegada de um produto no estoque. O lugar *Estoque* representa o estoque. A transição

Saída representa o evento de saída de um produto para a loja. O lugar *Loja* representa a loja. A transição *VendaProduto* representa o evento da venda de um produto.

Figura 2: modelo de estoque com loja de produtos e seus estados.



Fonte: o autor.

No primeiro estado, não há produtos no estoque e nem na loja. Se observarmos o modelo é possível ver que as transições (Estado 1 (inicial)) *Saída* e *VendaProduto* não podem ser executadas (disparadas) porque elas necessitam de um produto (*token*) nos lugares de saída (*Estoque* e *Loja*). A necessidade de haver um *token*, em um lugar previamente conectado, para que uma transição seja disparada é chamada de "pré-condição" de uma transição. No Estado 1 apenas a transição *Chegada* pode ser disparada, pois a mesma não possui lugares de entrada. Assim a pré-condição também é satisfeita. Note que, uma transição que não possui nenhum lugar de entrada é chamada de “transição de origem” e incondicionalmente estará sempre habilitada. Por

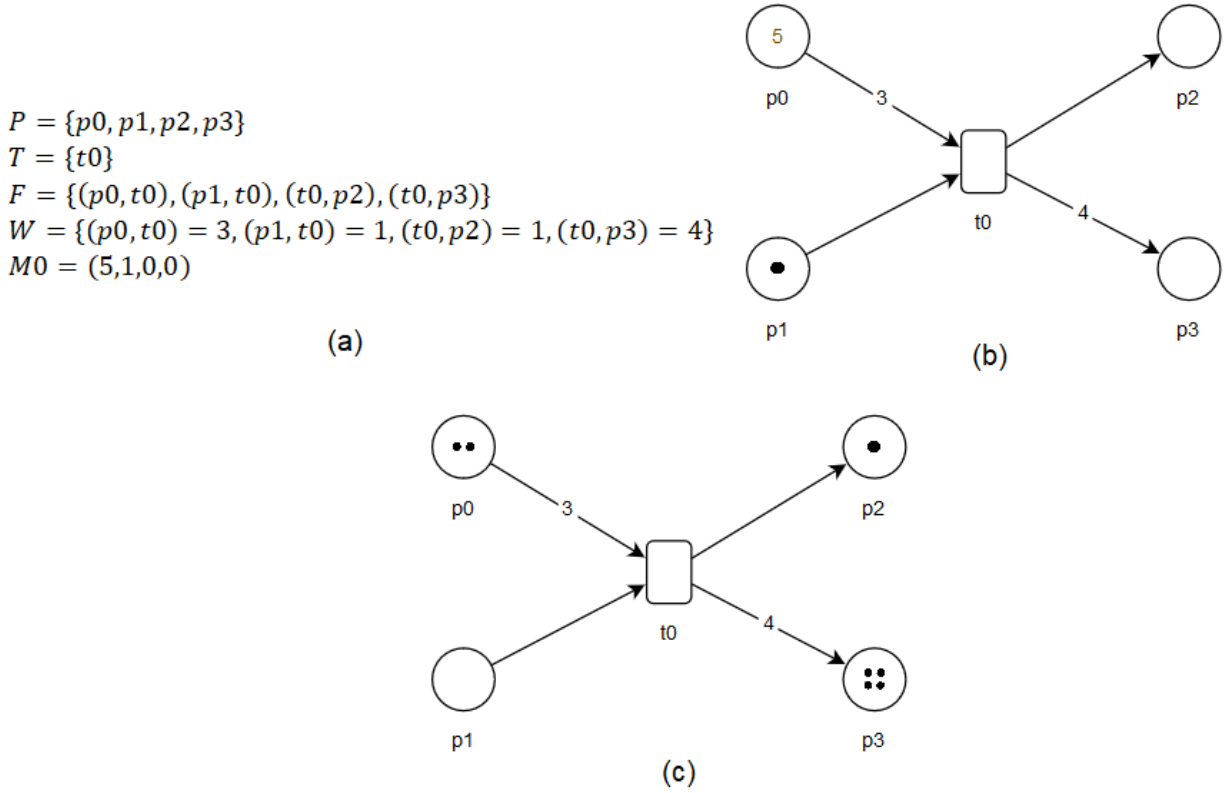
outro lado, uma transição que não possui lugar de saída é chamada de “sorvedouro” e nunca produz *tokens*, apenas os consome.

O Estado 2 ocorre quando a transição *Chegada* é disparada, de modo que um produto é adicionado no estoque (representado pelo lugar *Estoque*). Esse deslocamento de *tokens* que ocorre após o disparo de uma transição é chamado de "pós-condição". Após isso, a transição *Saída* poderá ser disparada deslocando um produto (*token*) para a loja (representado pelo lugar *Loja*), resultando no Estado 3. Por fim, a transição *VendaProduto* pode ser disparada representando a venda de um produto da loja, resultando no retorno ao estado inicial.

Exemplo de uma RdP e Representação Formal

A partir de um exemplo, vamos mostrar como construir a representação formal de uma RdP. A Figura 3 (a) mostra o formalismo matemático da RdP graficamente representada na Figura 3 (b). Observando essas duas figuras, notamos que a RdP possui quatro lugares ($P = \{p0, p1, p2, p3\}$), uma transição ($T = \{t0\}$), conjunto de arcos ($F = \{(p0, t0), (p1, t0), (t0, p2), (t0, p3)\}$), pesos dos arcos ($W = \{(p0, t0) = 3, (p1, t0) = 1, (t0, p2) = 1, (t0, p3) = 4\}$) e marcação inicial ($M0 = (5,1,0,0)$). Note que uma relação existente no conjunto de arcos, por exemplo, *(p0,t0)*, indica que o arco conecta *p0* e *t0* no sentido lugar-transição. Caso a relação fosse *(t0,p0)*, isso indicaria que o arco conecta *t0* e *p0* no sentido transição-lugar. As relações mostradas no conjunto *W* indicam os pesos nos arcos. Por exemplo, *(p0,t0)=3*, indica que o arco que conecta *p0* e *t0* no sentido lugar-transição tem peso 3. A marcação inicial *M0* indica a quantidade de *tokens* que os lugares da RdP contêm em seu estado inicial. Desse modo, $M0 = (5,1,0,0)$ indica que os lugares *p0, p1, p2 e p3* (seguindo a ordem do conjunto $P = \{p0, p1, p2, p3\}$)), respectivamente, contêm 5, 1, 0 e 0 *tokens* em seu estado inicial. Agora, supondo que a transição *t0* seja disparada, a RdP sairá do estado inicial e entrará em seu próximo estado (ver Figura 3 (c)). Nesse caso a marcação será $M1 = (2,0,1,4)$.

Figura 3: exemplo de uma RdP e seu formalismo matemático.



Fonte: o autor.

2.1.3 Redes Elementares

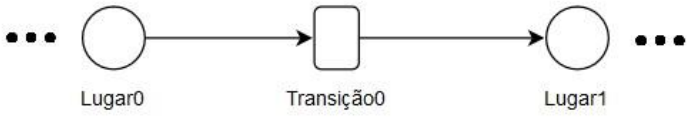
Segundo Maciel, Lins e Cunha (1996, p. 25): “as redes elementares são blocos básicos que permitem a modelagem de sistemas mais complexos”. Desse modo, serão apresentadas algumas redes elementares nessa seção: sequência, distribuição, junção e escolha não-determinística.

Sequência

A sequência é uma rede que representa ações consecutivas, desde que uma determinada condição seja satisfeita. Isso significa que, após a execução de cada ação, uma nova condição é satisfeita e uma nova ação é executada. A Figura 4 mostra a estrutura da sequência. Um *token* no *Lugar0* habilita a transição *Transição0* e com o disparo desta transição é estabelecida uma nova condição (*Lugar1* marcado). Desse

modo, uma nova condição pode permitir o disparo de uma transição associada ao lugar *Lugar1* (SPOLON, 2001).

Figura 4: estrutura básica da sequência.

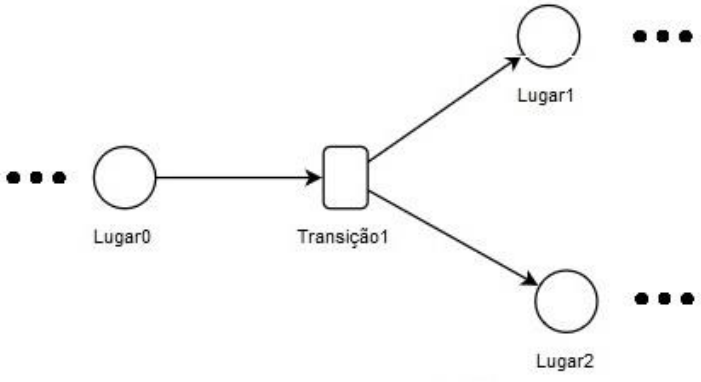


Fonte: o autor.

Distribuição

A rede de distribuição possibilita a criação de processos paralelos a partir de um processo prévio. A Figura 5 mostra a estrutura de uma rede de distribuição. Um *token* no *Lugar0* habilitaria o disparo da transição *Transição1*. O disparo dessa transição adicionaria simultaneamente um *token* no *Lugar1* e outro no *Lugar2*. Essas novas condições estabelecidas a partir da marcação dos lugares *Lugar1* e *Lugar2* permitem a execução de novos processos paralelos (SPOLON, 2001).

Figura 5: estrutura básica da distribuição.



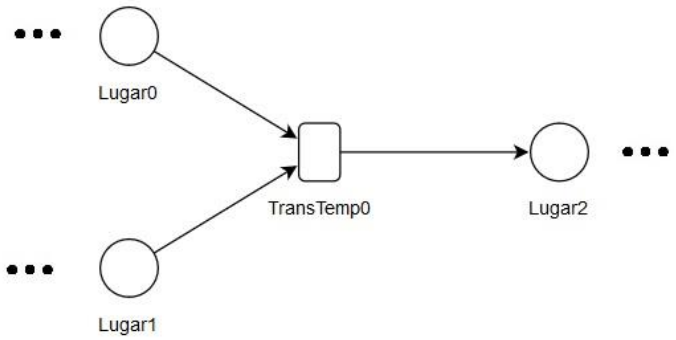
Fonte: o autor.

Junção

A Figura 6 mostra a estrutura básica de uma rede de junção. Essa rede elementar representa a sincronização entre atividades concorrentes. Ela combina dois ou mais processos, de modo que só é permitido que outro processo continue sua execução após o término de todos os outros processos paralelos que o antecedem.

Assim, a transição *TransTemp0* só estará habilitada para disparo se ambos os lugares *Lugar0* e *Lugar1* estiverem marcados (MACIEL; LINS e CUNHA, 1996).

Figura 6: estrutura básica da junção.

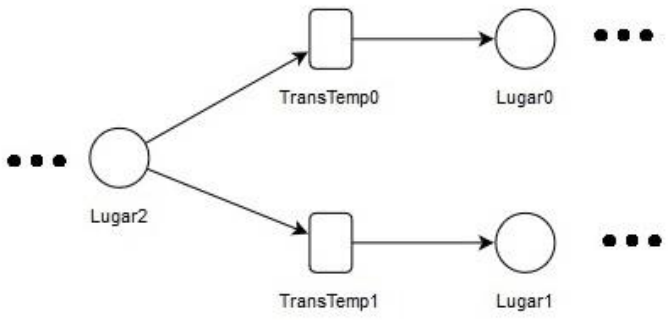


Fonte: o autor.

Escolha Não-Determinística

A escolha não-determinística se caracteriza pelo fato de duas ou mais transições possuírem um mesmo lugar de entrada, ou seja, ao se disparar uma transição, inabilita-se as outras. Esse tipo de rede elementar também é denominada de conflito ou decisão. A Figura 7 mostra a estrutura dessa rede elementar. Note que, se houvesse um *token* no lugar *Lugar2,* ambas as transições *TransTemp0* e *TransTemp1* estariam habilitadas, entretanto, o disparo de qualquer uma dessas transições inabilitaria a outra (MACIEL; LINS e CUNHA, 1996).

Figura 7: estrutura básica da escolha não-determinística.



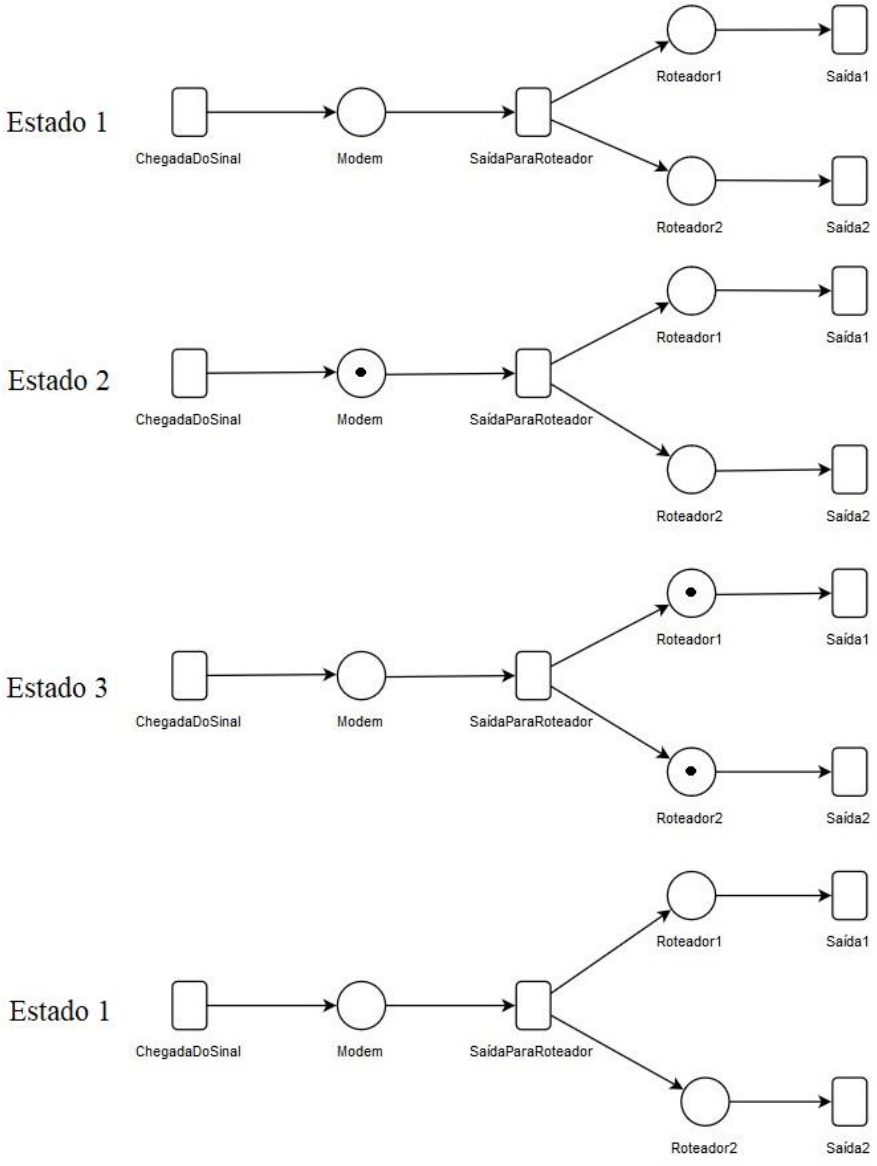
Fonte: o autor.

2.1.4 Exemplos com as Redes Elementares

A seguir são apresentados alguns exemplos de modelos criados a partir das redes elementares.

Exemplo 1

Figura 8: modelo que representa o envio de sinais em paralelo.



Fonte: o autor.

A partir de um exemplo mostraremos um modelo que usa uma rede elementar de distribuição. Imagine um cenário onde um modem recebe um sinal e esse mesmo

sinal é transmitido simultaneamente para dois roteadores. Esses roteadores, por sua vez, transmitem esses sinais para computadores. O modelo que representa esse cenário é mostrado na Figura 8.

No Estado 1, o modem (representado pelo lugar *Modem*) ainda não recebeu o sinal. No Estado 2, o modem recebe o sinal (representado por um *token*) após o disparo da transição *ChegadaDoSinal*. No Estado 3, ocorre a execução da transição *SaídaParaRoteador*. Isto é, o modem envia o sinal simultaneamente para o roteador 1 e o roteador 2 que são representados, respectivamente, pelos lugares *Roteador1* e *Roteador2*. Vale lembrar que a pré-condição para que a transição *SaídaParaRoteador* seja executada é haver um *token* no lugar *Modem* (Estado 2). Quando a transição *SaídaParaRoteador* é disparada, como resultado (pós-condição), ela consome um *token* do lugar *Modem* e deposita um *token* nos lugares *Roteador1* e *Roteador2* (Estado 3). Por fim, as transições *Saída1* e *Saída2* podem ser executadas, indicando que os roteadores enviaram os sinais para computadores. Note que, após isso, o modelo retorna para o Estado 1.

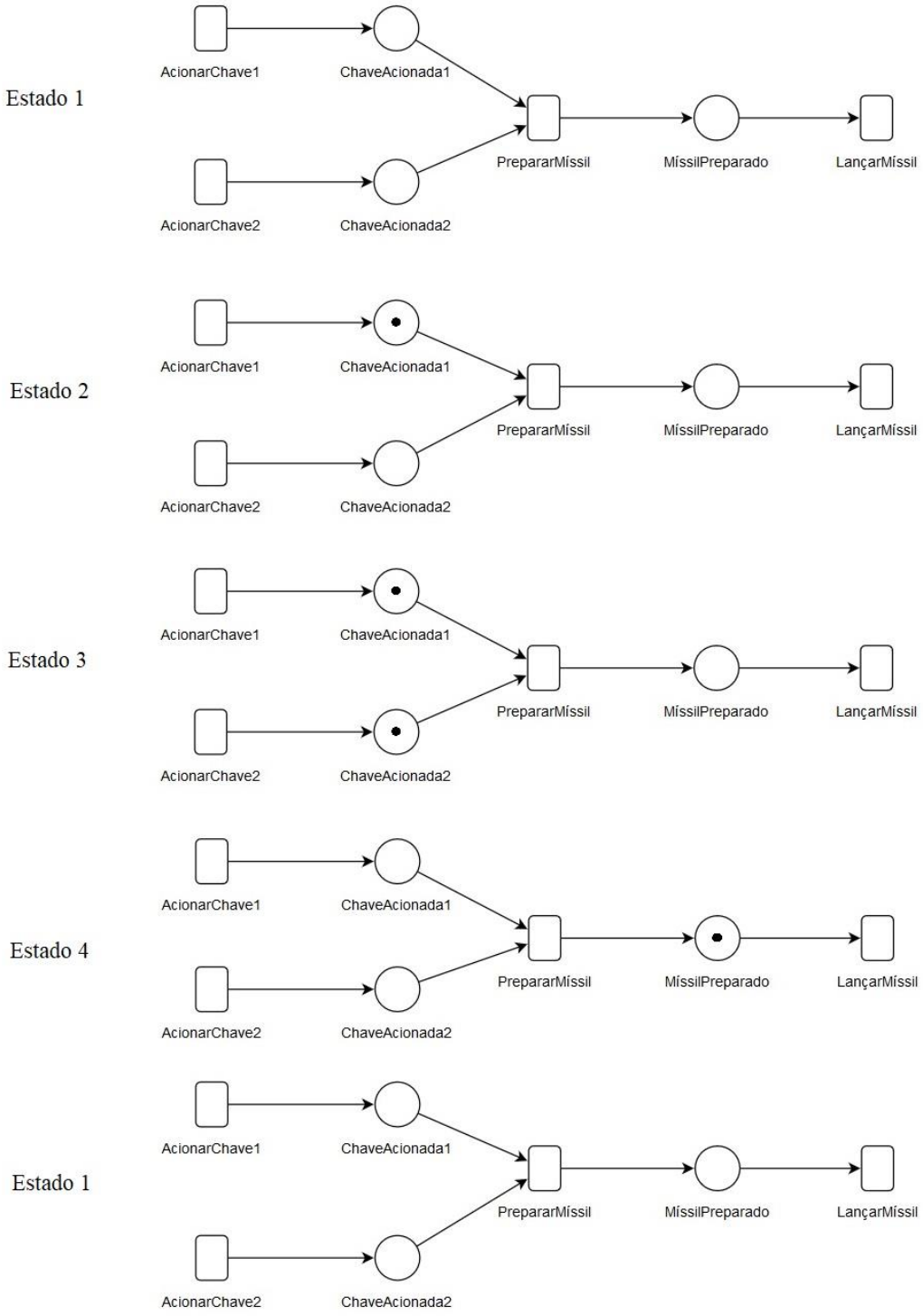
Exemplo 2

Suponha que em uma base militar estejam realizando testes nucleares e que, em seu sistema de lançamento de míssil, seja necessária a ativação de duas chaves diferentes. O sistema funciona da seguinte forma: duas chaves distintas são acionadas por dois oficiais e, após a ativação dessas chaves, o sistema permite a preparação do míssil para o lançamento. Estando o míssil preparado, ele poderá ser lançado para a realização do teste.

Na Figura 9 é mostrado o modelo que representa o sistema de teste de míssil. Note que será usada a rede elementar de junção. No Estado 1, o sistema está em espera, aguardando que algum oficial acione uma das chaves. No Estado 2, a transição *AcionarChave1* é disparada, significando que um oficial acionou a primeira chave. Assim, um *token* é gerado no lugar *ChaveAcionada1*. No Estado 3, ocorre o acionamento da segunda chave. Isto é, a transição *AcionarChave2* é disparada e um *token* é gerado no lugar *ChaveAcionada2*. No Estado 4, ocorre o disparo da transição *PrepararMíssil*, liberando um *token* para o lugar *MíssilPreparado*. Vale ser ressaltado que a pré-condição da transição *PrepararMíssil* é que os lugares

ChaveAcionada1 e *ChaveAcionada2* tenham pelo menos um *token* cada. Quando ocorre o disparo da transição *LançarMíssil*, a rede volta para o Estado 1.

Figura 9: modelo de teste de míssil.

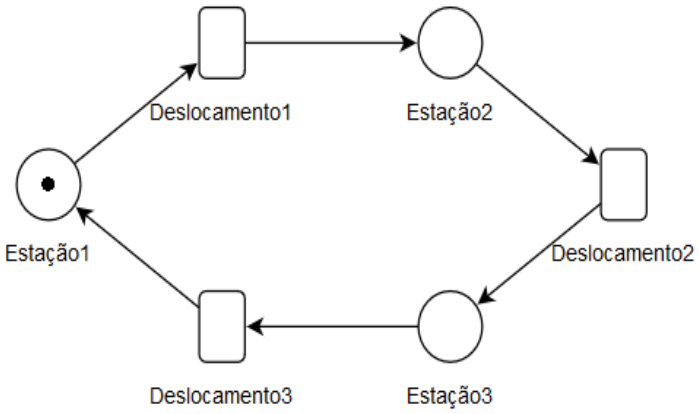


Fonte: o autor.

Exemplo 3

Figura 10 representa um sistema de transporte urbano via metrô. Note que tal modelo usa a rede elementar de sequência. Neste sistema, existem três estações e um trem circulando entre as estações. O trem inicia seu percurso na primeira estação e se desloca para a segunda e depois para a terceira estação. A terceira estação é conectada à primeira, formando uma linha circular. Além disso, o percurso é feito em apenas um sentido. No modelo, os elementos do sistema são representados da seguinte maneira: as estações são representadas pelos lugares *Estação1*, *Estação2* e *Estação3*; o trem é representado pelo *token* que está contido no lugar *Estação1*; as transições *Deslocamento1*, *Deslocamento2* e *Deslocamento3* representam o tempo de deslocamento entre uma estação e outra; por fim, os arcos representam o sentido de deslocamento do trem.

Figura 10: modelo de um sistema de transporte urbano via metrô.



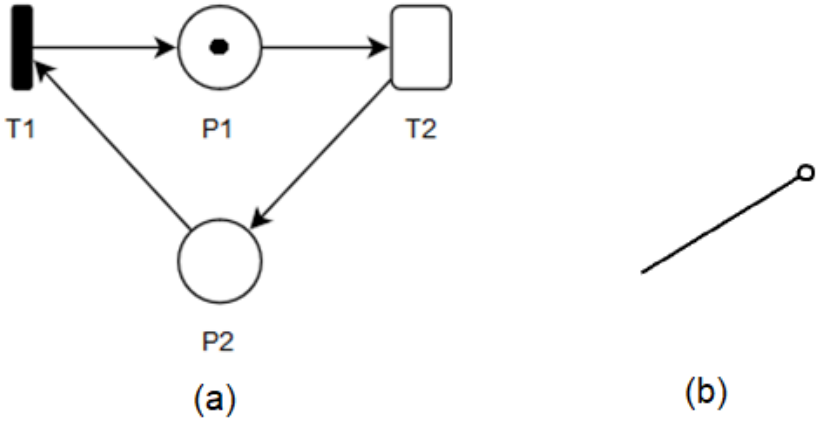
Fonte: o autor.

2.1.5 Rede de Petri Estocástica Generalizada (GSPN)

Diversas variantes de RdP têm sido desenvolvidas ao longo do tempo, tais como redes de Petri temporizada (MERLIN e FABER, 1976), redes de Petri estocástica (MARSAN, 1989), redes de Petri de alto-nível (JENSEN, 1991) e redes de Petri orientadas a objetos (JANOUSEK, 1998). Como já mencionado na introdução, este trabalho adota uma extensão das redes de Petri denominada de Rede de Petri Estocástica Generalizada (GSPN). A GSPN, graficamente, é representada por arcos

(setas), arcos inibidores (ver Figura 11 (b)), transições temporizadas, transições imediatas, lugares e *tokens*. A Figura 11 (a) ilustra um exemplo de uma GSPN que é composta pelos lugares *P1* e *P2*, pela transição imediata *T1* e pela transição temporizada *T2*. No lugar *P1* existe um *token* associado. A diferença entre as transições *T2* e *T1* é que a transição *T2* demanda um tempo (ou *delay*) para ser executada. A transição imediata *T1*, por outro lado, não demanda tempo e ocorre de modo prioritário em relação à *T2*, isto é, ocorre de modo imediato ou com tempo equivalente a zero (0). Os tempos associados às transições temporizadas (ex. *T2*) são distribuídos exponencialmente.

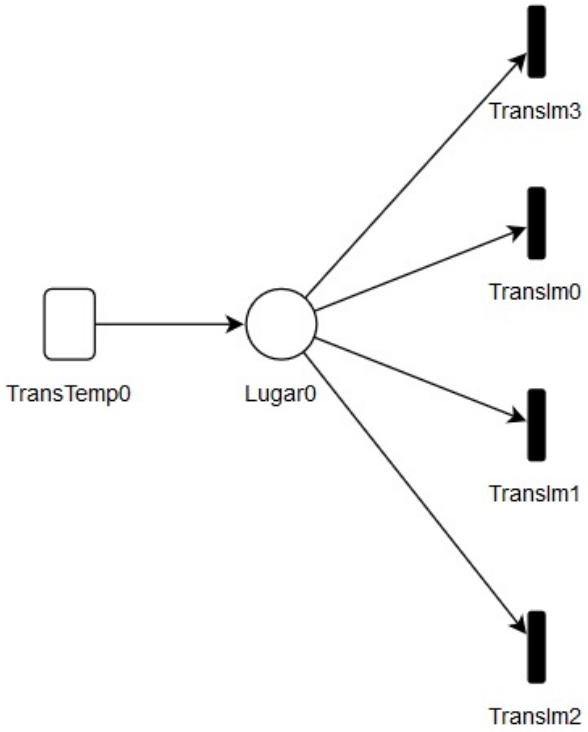
Figura 11: exemplo de uma GSPN e arco inibidor.



Fonte: o autor.

A transição imediata pode possuir um peso associado, o qual indica a probabilidade da transição disparar (MURATA, 1989; TRIVEDI, 2008). Agora, observe a Figura 12: existe um lugar *Lugar0* cuja origem de quatro arcos inicia nele e conecta quatro transições imediatas. Na modelagem de sistemas com RdP, quando um lugar cuja origem dos arcos inicia nele e se conecta a transições imediatas, é necessário indicar qual a probabilidade (peso) de disparo dessas transições. Se um lugar se conecta a apenas uma transição imediata, o peso dessa transição será 1. De outro modo, quando um lugar se conecta a mais de uma transição imediata, o peso dessas transições deve ser entre 0 e 1. No entanto, como o peso é uma probabilidade (varia de 0% a 100%, ou 0 a 1), a soma dos pesos das transições imediatas conectadas a um mesmo lugar deve ser igual a 1 (BALBO, 2001). Na Figura 12, cada transição imediata tem peso igual a 0.25, sendo a soma desses pesos igual a 1.

Figura 12: GSPN com quatro transições imediatas.



Fonte: o autor.

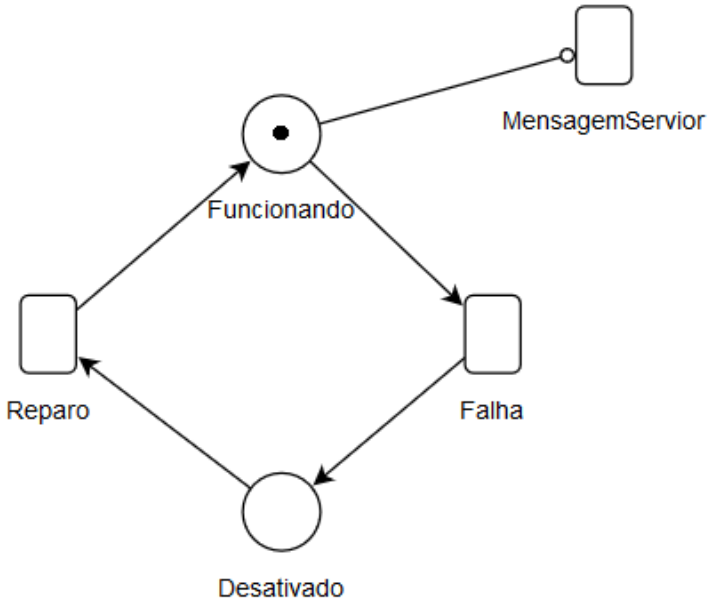
Exemplo de uma GSPN com Arco Inibidor

Os arcos (setas), que foram apresentados até agora, conectam transições a lugares ou lugares a transições, e habilitam transições para o disparo apenas quando existirem *tokens* nos lugares de entrada das mesmas. Já o arco inibidor desabilita a transição quando o lugar de entrada tem um *token*. Quando ocorre o contrário, isto é, quando o lugar de entrada não possui *tokens*, a transição é habilitada para o disparo. Os arcos inibidores sempre têm sua origem em um lugar e o final em uma transição e não vice-versa. Isso ocorre devido ao objetivo prático do uso do arco inibidor que é desabilitar ou habilitar a ação de uma transição. O lugar que é conectado à transição por meio de um arco inibidor é chamado de lugar inibidor (MURATA, 1989).

Suponha que em uma região exista um sensor que capte a temperatura do ambiente. A Figura 13 apresenta o modelo desse sistema. O sensor pode estar em dois estados, “desativado” ou “funcionado”, representados, respectivamente, pelos lugares *Funcionando* e *Desativado*. O estado “desativado” ocorre quando o sensor falha (evento representado pela transição *Falha*). Já o estado “funcionando” ocorre quando o sensor é reparado (evento representado pela transição *Reparo*). Quando o sensor falha, uma mensagem acusando o erro poderá ser enviada para o servidor que está monitorando a temperatura. O evento de envio da mensagem é representado pela transição *MensagemServidor.* Note que existe um arco inibidor ligando o lugar *Funcionando* à transição *MensagemServidor*.

Inicialmente o sensor está funcionando, logo, um *token* reside no lugar *Funcionando*. Note que um *token* determina o estado em que o sistema se encontra em um dado momento. Assim, quando o sensor falha, o *token* que residia no lugar *Funcionando* é deslocado para o lugar *Reparo*. Desse modo, o lugar *Funcionando* não conterá *tokens* e, assim, a transição *MensagemServidor* é habilitada.

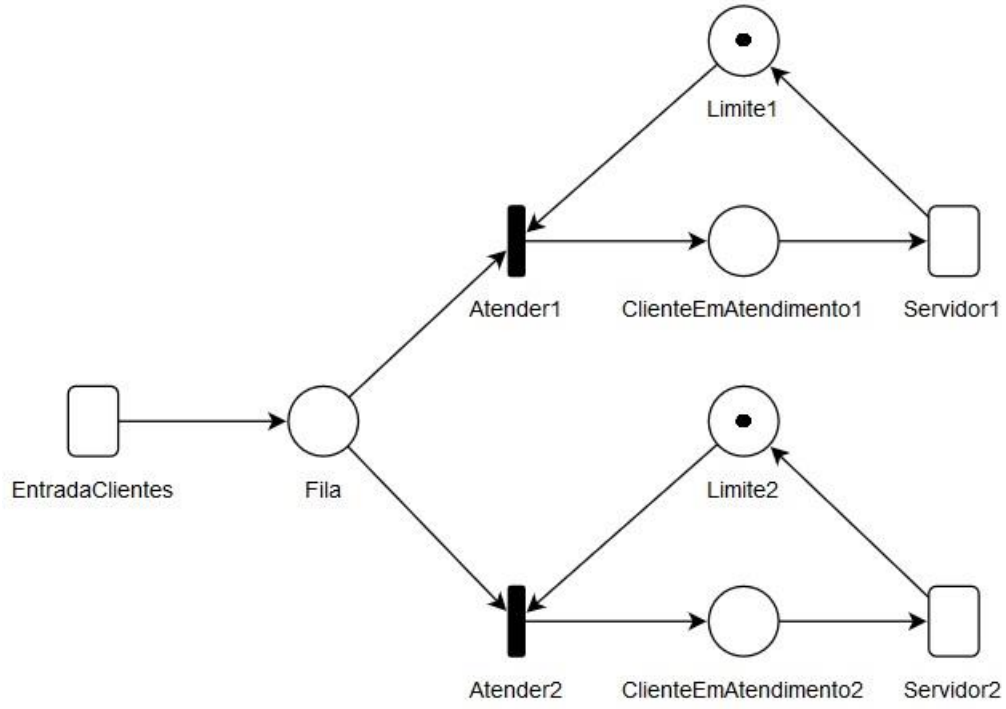
Figura 13: falha e reparo de um sensor.



Fonte: o autor.

Exemplo de uma GSPN com Transição Imediata

Figura 14: sistema de fila com dois servidores.



Fonte: o autor.

No modelo GSPN da Figura 14, é representado um sistema de fila em que clientes entram no sistema, esperam na fila e depois podem ser atendidos por um dos dois servidores (note a estrutura de escolha não-determinística). Nesse sistema, a transição *EntradaClientes* representa o tempo médio de chegada de um cliente no sistema. As transições *Servidor1* e *Servidor2* podem representar o tempo médio de atendimento em um caixa de banco ou um atendimento num hospital, por exemplo. A fila de espera é representada pelo lugar *Fila* e não há limites de clientes nela. Além disso, não há limites de clientes dentro do sistema. Existe apenas um limite de atendimento de um cliente por vez para cada um dos servidores. Note que os lugares *Limite1* e *Limite2* (marcados com um *token*) indicam que apenas um cliente pode ser atendido por vez para cada um dos servidores. As transições *Atender1* e *Atender2* são imediatas, pois quando um dos servidores estiver disponível, o atendimento ao cliente se inicia imediatamente.

Agora, suponha que o tempo médio de chegada de um cliente seja de 20 minutos e que o tempo médio de atendimento de ambos os servidores seja em média de 10 minutos. Nesse caso, os tempos associados às transições temporizadas *EntradaClientes*, *Servidor1* e *Servidor 2* serão, respectivamente, 20 minutos, 10 minutos e 10 minutos. Além disso, se considerarmos que a probabilidade de um cliente ser atendido pelo servidor 1 seja igual à de ser atendido pelo servidor 2, então a probabilidade de atendimento para cada um dos servidores será de 50%. Assim, o peso associado às transições imediatas *Atender1* e *Atender2* será 0.5 para ambas.

2.1.6 Métricas

No estudo das redes de Petri, métricas são notações usadas para representar cálculos matemáticos ou condições dos estados. Os cálculos, representados pelas métricas, são realizados pelas ferramentas de modelagem e simulação, por exemplo, o Mercury (SILVA et al., 2015). Vale ressaltar que a sintaxe de escrita das métricas varia de acordo com a ferramenta de modelagem e simulação que está sendo utilizada. Aqui e na ferramenta TryRdP foi adotada uma sintaxe semelhante à sintaxe apresentada no Mercury.

Os Quadros 1 e 2 descrevem os caracteres reservados usados nas métricas. O Quadro 1 mostra os caracteres relacionados aos cálculos das métricas e outros caracteres que são usados na sintaxe. Note que existem basicamente dois tipos de cálculos: a esperança matemática e a probabilidade. O Quadro 2 mostra os caracteres que podem ser usados para representar as operações matemáticas (aritmética, comparação e lógica). Tanto os operadores matemáticos quanto os caracteres próprios da sintaxe são semelhantes àqueles usados em linguagens de programação.

Agora, a partir do modelo da Figura 14, vamos exemplificar o uso das métricas. Suponha que queremos obter as seguintes métricas do modelo: 1) a quantidade média de clientes em espera na fila; 2) a quantidade média total de clientes no sistema; 3) a probabilidade de não haver 4 clientes esperando na fila; 4) a probabilidade de haver menos que 5 e mais que 2 clientes na fila em espera; 5) a probabilidade de que o servidor 1 esteja atendendo pelo menos um cliente (o valor desse cálculo é chamado de fator de utilização ou taxa de utilização do servidor). A seguir será descrito cada uma dessas métricas.

Quadro 1: caracteres reservados relacionados aos cálculos das métricas.

Caractere Especial	Significado
E	Calcula a quantidade média de *tokens* em um lugar – esperança matemática.
P	Calcula a probabilidade de haver determinada quantidade de *tokens* em um lugar.
#	Referencia o número de *tokens* em um lugar na rede de Petri.
{ }	Usado com os caracteres E e P. Apenas faz parte da sintaxe das métricas.
( )	Usado para separar cálculos ou expressões lógicas.

Fonte: o autor.

Quadro 2: caracteres reservados relacionados às operações nas métricas.

Operadores Aritméticos	Operadores de Comparação	Operadores Lógicos
+	=	AND
	<	
-	<=	OR
	>	
*	>=	NOT
	/=	
/		

Fonte: o autor

Para calcular a quantidade média de clientes em espera na fila devemos usar a esperança matemática (E). Nesse caso, calculamos a quantidade média de *tokens* no lugar *Fila*. Assim, temos a métrica 1 dada por: *(E{#Fila})*. Como pode ser observado, o cálculo da métrica (seja esperança ou probabilidade) deve estar entre parênteses '( )'.

No cálculo da esperança, fazemos referência ao número de *tokens* em um lugar. Para o caso da métrica 1, usamos o lugar *Fila*, consequentemente, escrevemos *#Fila*. Os colchetes '{ }' são usados logo após os caracteres 'E' ou 'P', e englobam a referência do número de *tokens* ou expressões lógicas. A quantidade média total de clientes no sistema significa a soma da quantidade média de *tokens* nos lugares *Fila*, *ClienteEmAtendimento1 e ClienteEmAtendimento2*. Assim, a métrica 2 consiste em: *(E{#Fila})+(E{#ClienteEmAtendimento1})+(E{#ClienteEmAtendimento2})*. Observe que nessa métrica cada cálculo de cada esperança está envolvido com parênteses.

Para calcular a probabilidade de não haver 4 clientes esperando na fila, que é a métrica 3, devemos considerar o cálculo de probabilidade (P). Assim, temos a métrica 3: *(P{(#Fila/=4)})*. A escrita das métricas de probabilidade é feita com expressões lógicas. Nesse caso, temos a expressão *(#Fila/=4)* com o operador de comparação de diferença '/='. Observe que a expressão lógica deve estar envolvida com parênteses. A métrica da probabilidade de haver menos que 5 e mais que 2 clientes na fila em espera, se faz com o uso de expressões lógicas e operador lógico 'AND'. Consequentemente, a métrica 4 é dada por: *(P{(#Fila>2)AND(#Fila<5)})*. A probabilidade de que o servidor 1 esteja atendendo pelo menos um cliente (taxa de utilização), isto é, que haja ao menos um *token* no lugar *ClienteEmAtendimento1,* se faz com uso da métrica 5: *(P{(#ClienteEmAtendimento1>0)}).*

2.2 INFORMÁTICA NA EDUCAÇÃO

Os primeiros trabalhos acadêmicos voltados para o tema de “Informática na Educação” no Brasil são datados da década de 70 (BITTENCOURT e ISOTANI, 2018). De um modo geral, esse tema de estudo envolve a pesquisa da inserção do computador no processo de aprendizagem dos conteúdos curriculares de qualquer nível e modalidade de educação (VALENTE, 2000).

Segundo Valente (2000, p. 2), “O computador pode ser usado na educação como máquina de ensinar ou como máquina para ser ensinada”. A partir dessa afirmação Valente (1998, 2000) mostra duas abordagens de utilização do computador na educação:

- Abordagem Instrucionista: nessa abordagem o computador é usado como máquina de ensinar. Isso consiste em informatizar os métodos de ensino tradicionais. Ou seja, um *software* será implementado e usado no computador com uma série de informações (conteúdo didático) que serão passadas ao aluno. Além disso, o *software* poderá ser implementado de modo a fazer perguntas e receber respostas para verificar se o aluno reteve a informação. As categorias de *softwares* mais comuns na modalidade instrucionista são tutorial, exercício-e-prática, jogo e simulação (VALENTE, 1998):
  - Tutorial: segundo Vieira (2015, p. 1), esse tipo de software se caracteriza por “transmitir informações pedagogicamente organizadas, como se fossem um livro animado, um vídeo interativo ou um professor eletrônico”.
  - Exercício-e-Prática: normalmente são usados para revisar o material visto em classe. Esse tipo de software enfatiza a apresentação dos exercícios que requerem a resposta do aluno e propiciam *feedback* de modo imediato (VALENTE, 1998).
  - Jogo: tem a finalidade de divertir, motivar e desafiar o aluno conforme ele vai recebendo as informações (conteúdo didático) (MIASHIRO et al., 2020).
  - Simulação: segundo Miashiro *et al.* (2020, p. 4), esse tipo de software “possibilitam a vivência de situações difíceis ou até perigosas de serem reproduzidas em aula”. Por exemplo, permite a realização de experimentos em áreas com risco à saúde humana como química e balística.

- Abordagem Construcionista: nessa abordagem o computador é usado como máquina para ser ensinada. Ainda segundo Valente (1998), nessa abordagem o computador é usado como uma ferramenta educacional, ou seja, o estudante constrói, por intermédio do computador, o seu próprio conhecimento. Como exemplo, podemos citar a ferramenta LOGO (VALENTE, 1998). LOGO é uma linguagem de programação voltada para o ambiente educacional e foi desenvolvida por Seymour Papert no *Massachussets Institute of Technology*. A linguagem LOGO sintetiza de modo prático a teoria do construcionismo visto que

o próprio Seymour Papert foi o criador dessa teoria (VALENTE, 1998, 2000). Segundo Schneider (2021, p. 1):

> LOGO propõe uma metodologia de ensino que busca, através de uma linguagem semelhante à natural, facilitar a comunicação entre o usuário e o computador e proporcionar a criação de modelos através de formas geométricas e do raciocínio lógico. Propõe também, que o aluno seja ativo construtor de seus próprios conhecimentos, desenvolvendo assim sua capacidade intelectual. O professor deve permitir a reflexão do aluno, ao contrário do modelo tradicional onde reina o autoritarismo. O aluno através do erro, é condicionado a refletir novas formas de resolução do problema, ou seja, ele tem a chance de aprender com seus próprios erros e é estimulado a tentar.

Além da abordagem que define como o artefato é usado no processo de ensino e aprendizagem, é importante também distinguir o que vem a ser um *software* educacional e/ou um *software* educativo (OLIVEIRA, JOSÉ e MOREIRA, 2001):

- *Software* Educacional: é um *software* usado no contexto escolar e não necessariamente tem a função de educar. Por exemplo, *Microsoft PowerPoint* pode ser utilizado no contexto escolar para fazer apresentações.
- *Software* Educativo: é também um *software* educacional. No entanto, ele é desenvolvido com a finalidade de educar. Um *softwar*e educativo possui quatro características:
  i. Apresenta uma fundamentação pedagógica.
  ii. Possui uma finalidade didática.
  iii. Possui uma interação entre aluno e *software*, mediada por um professor.
  iv. É fácil de usar.

A ferramenta que estamos propondo é voltada para a aprendizagem sobre redes de Petri do nível básico ao avançado, apresentando como características de software educativo:

- Fundamentação pedagógica: a ferramenta segue uma abordagem instrucionista, combinando os formatos de tutorial e exercício e prática (VALENTE, 2000).

- Finalidade didática: tem o objetivo de levar o aluno a construir o conhecimento relacionado à modelagem de sistemas usando redes de Petri.
- Interação aluno-software educativo-professor: a ferramenta foi desenvolvida de modo que ela possa ser usada tanto de modo individual quanto em um curso formal mediado por um professor.
- Facilidade de uso: esse foi um dos aspectos avaliados por professores e estudantes durante a presente pesquisa. De forma geral, a ferramenta foi bem avaliada. Os resultados detalhados são apresentados no Capítulo 6.

3 TRABALHOS RELACIONADOS

Embora existam vários softwares educativos para auxiliar o aprendizado, não foram encontrados softwares educativos com objetivo de auxiliar o aprendizado de modelagem de sistemas com redes de Petri. As ferramentas existentes focam principalmente no mapeamento de linguagens de alto nível em redes de Petri com a finalidade de facilitar o seu uso para modelagem de sistemas. Nesta seção, descrevemos algumas dessas ferramentas. Vale ressaltar que a busca por tais ferramentas se deu através da ferramenta de busca *Google Scholar*[5]. Durante as buscas foram utilizadas palavras chaves que envolviam os temas de informática na educação e de modelagem de sistemas usando redes de Petri, por exemplo, “redes de Petri”, “ferramenta de apoio ao aprendizado” e “modelagem de sistemas”.

3.1 FERRAMENTAS EDUCATIVAS PARA O ENSINO DE DISCIPLINAS DA COMPUTAÇÃO

Ferramentas educativas dinâmicas e interativas nas disciplinas de computação têm sido propostas nos últimos anos a fim de tornar o aprendizado mais envolvente e significativo para os alunos. A seguir iremos apresentar um breve resumo de algumas dessas ferramentas. A ferramenta IO Simulator é um ambiente de simulação cujo objetivo é mostrar de maneira prática os conceitos sobre os processos de gerenciamento do sistema de entrada e saída de um Sistema Operacional (SO). Tal ferramenta pode servir de apoio às disciplinas que abordam sistemas operacionais nos cursos de graduação e pós-graduação em computação (MEDEIROS et al. 2011). O WebJuvia é um software educativo baseado na *web* com o propósito de simular o funcionamento dos principais algoritmos de alocação dinâmica de memória por meio de representação gráfica. Tal assunto é abordado na disciplina de SO (SILVA; VALENTE; CARMONA, 2021). Já NeanderSIM é um simulador gráfico cujo objetivo é apoiar o

[5] https://scholar.google.com/

ensino do conteúdo acerca dos processos de execução de instruções em um computador na disciplina de Arquitetura de Computadores (ULLMANN et al. 2014).

Costa et al. (2017) apresentam a ferramenta LpT para auxiliar o estudante na compreensão do conteúdo referente a conversão de linguagens de alto nível (desde da linguagem natural) para linguagens de baixo nível (até a linguagem de máquina). A ferramenta PEEF (ARAUJO; BITTENCOURT; CHAVEZ, 2021) é um IDE (*Integrated Development Environment*) *on-line* que visa auxiliar no processo de ensino-aprendizagem de programação. Segundo os autores, aprender a programar é um processo complexo que envolve pensamento lógico, interpretação de texto e habilidades matemáticas. Desse modo, a ferramenta PEEF, a partir de *feedbacks,* auxilia os estudantes nesse propósito. Já a ferramenta PuzzlEdu visa auxiliar no ensino-aprendizado de Programação Orientada a Objetos (POO) (DANTAS, 2011).

Note que as ferramentas citadas, embora sejam voltadas para algumas disciplinas da área de computação, não têm foco na área de modelagem de sistemas. Desse modo, este trabalho aborda o desenvolvimento e avaliação da TryRdP, uma ferramenta *web* gratuita, interativa e baseada em exemplos para auxiliar na aprendizagem de modelagem de sistemas usando redes de Petri.

3.2 FERRAMENTAS DE MAPEAMENTO EM RDP

Vários trabalhos propõem ferramentas para o mapeamento de modelos de alto nível em modelos formais com o objetivo da análise quantitativa. Em (DISTEFANO et al., 2005), os autores propõem uma ferramenta chamada ArgoUML, cujo objetivo final é mapear um modelo UML (*Unified Modeling Language*) em uma rede de Petri para avaliar o processo de desenvolvimento de software. Já a ferramenta ArgoSPE (GÓMEZ-MARTÍNEZ e MERSEGUER, 2006), permite o mapeamento automático de diagramas da UML anotados em uma Rede de Petri Estocástica (SPN) para a avaliação de desempenho de sistemas computacionais.

A ferramenta ADAPT (RUGINA, KANOUN e KAÂNICHE, 2008) permite o mapeamento da linguagem AADL (*Architecture Analysis and Design Language*) em um modelo GSPN com o objetivo de fazer a avaliação de confiabilidade em processos de engenharia de sistemas. A ferramenta Calau (ANDRADE et al., 2012) auxilia projetistas no mapeamento do diagrama de estados da SysML (*Systems Modeling Language*),

anotado de acordo com o *profile* MARTE, em uma rede de Petri, com o intuito de realizar análises de caminhos críticos para o tempo de execução e o consumo de energia.

O propósito dessas ferramentas é facilitar o mapeamento de representações em alto nível em modelos de RdP sem que, necessariamente, o usuário saiba modelar um sistema diretamente com tal formalismo. Por exemplo, se um usuário conhece a AADL, ele/ela poderia usar a ferramenta ADAPT para descrever alguma arquitetura e depois realizar o mapeamento dessa arquitetura em um modelo de rede de Petri. Uma vez obtido o modelo, o usuário poderia realizar análises e verificações que não são possíveis com a AADL. Nota-se que tais ferramentas focam no mapeamento de linguagens de alto nível em redes de Petri com a finalidade de facilitar o seu uso. Assim, tais ferramentas não tem foco no ensino de modelagem.

3.3 FERRAMENTAS DE MODELAGEM E ANÁLISE

Além das ferramentas já citadas, existem outras usadas exclusivamente para a modelagem e a análise das redes de Petri, tais como Mercury (SILVA et al., 2015), TimeNet (ZIMMERMANN, 2017), Oris (BUCCI et al., 2010) e SPNP (HIREL, TUFFIN e TRIVEDI, 2000). Vale ressaltar que tais ferramentas são bem conhecidas e são comumente usadas por pesquisadores da área de modelagem de sistemas computacionais. Embora essas ferramentas possam ser usadas para os propósitos de modelagem e análise, as mesmas não dão suporte à aprendizagem das redes de Petri. Ou seja, elas requerem que os usuários já saibam e tenham experiência em modelagem de sistemas.

O uso de tais ferramentas com o propósito de modelagem e análise é um desafio para um usuário iniciante ou com pouco conhecimento em modelagem, visto que um pequeno erro na sintaxe do formalismo ou algum erro de digitação quando, por exemplo, o usuário digita as métricas, compromete toda a análise ou simulação de um modelo. Adicionalmente, quando o usuário comete algum erro na construção das redes de Petri, no geral, nenhuma mensagem de erro com feedback é apresentada, e quando é apresentado algo, a mensagem contém informações vagas sem precisão sobre qual erro o usuário cometeu. O usuário, muitas vezes, só vai saber se houve ou não um erro quando ele/ela simula o modelo e a ferramenta não retorna nenhum valor. Assim,

percebe-se que essas ferramentas não possuem objetivos de ensino ou aprendizagem de modelagem de sistemas, de modo que não se caracterizam como softwares educativos (OLIVEIRA, JOSÉ e MOREIRA, 2001).

4 METODOLOGIA

A ferramenta TryRdP foi desenvolvida de acordo com o fluxograma da Figura 15, com o objetivo de auxiliar o processo de ensino e aprendizado de modelagem computacional usando redes de Petri. Primeiramente, foram buscadas e analisadas ferramentas existentes para modelagem de sistemas computacionais (apresentadas no capítulo de trabalhos relacionados). Em seguida, os requisitos pedagógicos, requisitos de *software* e a arquitetura da ferramenta foram definidos de modo paralelo.

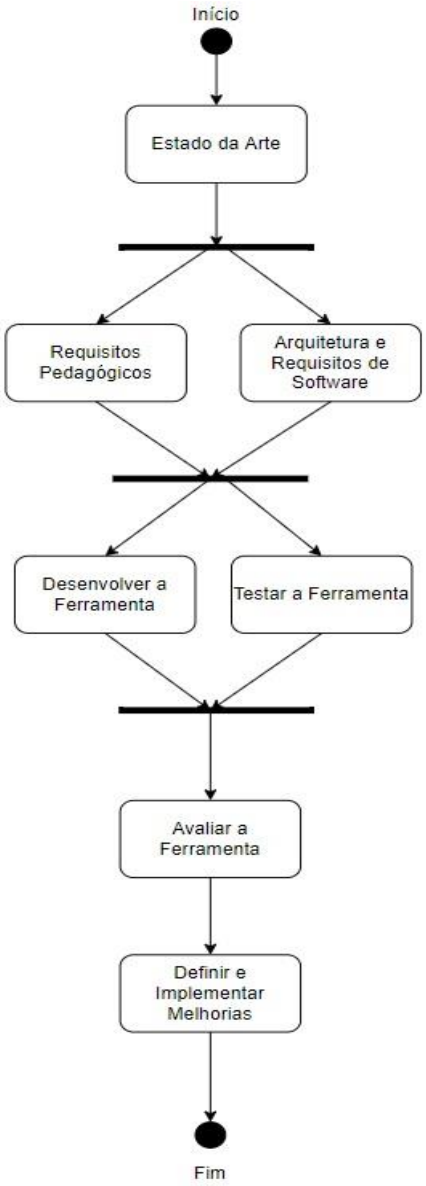
Os requisitos básicos definidos para a ferramenta foram:

- Ser gratuita e baseada na *web*.
- Conter exercícios de modelagem e de quebra cabeça.
- Adequar-se tanto ao uso autônomo e interativo pelos estudantes, como ao uso mediado por um professor.
- Apresentar *feedbacks* para os estudantes de modo a auxiliá-los na compreensão do conteúdo.
- Apresentar uma sequência lógica/didática na apresentação dos conteúdos.
- Conter elementos básicos de gamificação como pontuação e barra de progresso.

A arquitetura (detalhada na Seção 5.1) foi projetada de modo a incluir os requisitos básicos definidos. Vale ressaltar que a equipe que participou da definição dos requisitos e da arquitetura, bem como do desenvolvimento da ferramenta, era composta pelo autor deste trabalho, a co-orientadora (especialista em interface de usuário) e o orientador (especialista em redes de Petri).

Após a etapa da definição dos requisitos e da arquitetura deu-se início ao desenvolvimento da ferramenta. A TryRdP foi desenvolvida de modo iterativo e incremental, com implementação e testes ocorrendo de forma paralela. A TryRdP foi desenvolvida usando *PHP*, *HTML*, *CSS*, *Bootstrap framework*, *javaScript* e seis bibliotecas baseadas no *javaScript* (*jQuery*, *html2pdf*, *filesaver*, *VEX*, *CursorJS* e *MxGraph*).

Figura 15: metodologia proposta.



Fonte: o autor.

Como não foram encontradas ferramentas com objetivo similar, a TryRdP foi projetada essencialmente com base na fundamentação teórica sobre as RdP e as diretrizes didáticas seguidas por professores e profissionais da área. Também vale mencionar que os exemplos contidos em alguns livros como Maciel *et al*. (1996) e Trivedi (2008) serviram de inspiração para a construção do conteúdo didático da ferramenta. Adicionalmente, destacamos que a ferramenta segue o paradigma de software educativo baseado em tutorial e em exercício e prática.

Após o desenvolvimento da ferramenta, foi feita a avaliação da TryRdP. Como essa parte é uma das principais contribuições deste trabalho, detalhamos a mesma na Seção 4.1. A partir dos *feedbacks* dos professores e estudantes obtidos da avaliação, foram implementadas e definidas as melhorias na ferramenta. No Capítulo 6 são apresentados detalhes das melhorias propostas e implementadas.

4.1 AVALIAÇÃO DA FERRAMENTA

A avaliação da ferramenta foi feita com 8 professores especialistas em rede de Petri e 15 estudantes da área de TI, sendo 11 estudantes da graduação e 4 estudantes da pós-graduação. Os participantes foram recrutados por e-mail, e participaram de maneira voluntária. Aqueles que demonstraram interesse em participar receberam esclarecimentos completos sobre as etapas da pesquisa, assim como a garantia de anonimato, e tinham o direito de desistir da participação a qualquer momento sem nenhum tipo de consequência. No caso dos estudantes, não havia nenhuma consequência em seu curso de graduação ou pós-graduação, decorrente de sua participação na pesquisa e da nota obtida nos testes.

Vale ressaltar que a avaliação de um software educativo deve contemplar aspectos de usabilidade e critérios pedagógicos, como abordagens complementares (JUNIOR e AGUIAR, 2018). Assim, foram usados critérios de interface inspirados nas heurísticas de Nielsen (1994), que são um conjunto tradicional de heurísticas da área de Interação Humano-Computador, e questões didático-pedagógicas a partir da prática docente dos pesquisadores. Além disso, para avaliação de aprendizagem a partir do uso da ferramenta, foi usado o método de pré e pós-teste – esses instrumentos não foram padronizados, e sim elaborados pelos próprios pesquisadores (RAABE e BOMBASAR, 2020), a partir de seu conhecimento na área de estudo (redes de Petri) e docência. A significância estatística da diferença entre os resultados dos testes foi avaliada usando-se o teste não paramétrico de Wilcoxon (TRIOLA, 2017).

A avaliação com os professores foi feita com 8 doutores especialistas na área de sistemas computacionais e modelagem de sistemas com redes de Petri, os quais participaram de maneira voluntária. Tais professores exercem atividades de docência e pesquisa em universidades e institutos federais. *E-mails* foram enviados aos professores mostrando do que se tratava a pesquisa e perguntando se eles poderiam

participar da avaliação da ferramenta. A avaliação com professores objetivou avaliar a usabilidade da ferramenta e sua aplicação didático-pedagógica. Os professores receberam um link para usar a ferramenta livremente por um período de 10 dias. Em seguida, foi disponibilizado um formulário[6] desenvolvido usando o *Google Forms*[7], onde os professores classificaram as suas opiniões sobre a ferramenta de acordo com 13 tópicos que cobrem questões de interface e questões didático-pedagógicas (ver Quadro 3). A avaliação de cada tópico se deu por meio de escala Likert (JOSHI, 2015) de cinco níveis: “péssimo”, “ruim”, “regular”, “bom” e “ótimo”. Além disso, os professores puderam escrever um *feedback* com pontos positivos ou sugerindo melhorias na ferramenta. Os 10 primeiros tópicos do Quadro 3 foram apresentados tanto aos professores quanto aos estudantes. No entanto, os 3 últimos tópicos, que abordam temas voltados ao conteúdo da ferramenta e a utilidade para o ensino, foram apresentados apenas aos professores.

Já a avaliação com os estudantes objetivou avaliar a usabilidade e o aprendizado decorrente do uso da ferramenta. Houve a participação de 15 voluntários, sendo 11 de graduação e 4 de pós-graduação em áreas relacionadas a tecnologia da informação. Os estudantes foram recrutados de dois modos: i) enviando *e-mails* para grupos de graduação explicando sobre a pesquisa e chamando por voluntários; ii) solicitando os professores que aceitaram participar da pesquisa que convidassem seus orientandos de graduação ou pós-graduação.

Houve duas formas de avaliação com os estudantes: i) uma avaliação de aprendizagem sobre o conteúdo apresentado na ferramenta, por meio de um teste prévio e um teste posterior ao uso da ferramenta; e ii) uma avaliação sobre a usabilidade da ferramenta similar ao que foi feito com os professores. Assim, a avaliação ocorreu em 4 etapas seguindo a seguinte ordem: 1º) um teste prévio com conteúdo sobre modelagem usando as redes de Petri foi aplicado antes dos estudantes usarem a ferramenta TryRdP; 2º) após o teste, os estudantes foram apresentados à TryRdP e puderam usá-la livremente por um período de 10 dias; 3º) em seguida, um outro teste sobre redes de Petri foi aplicado; 4º) por fim, os estudantes foram

[6] Testes e formulários para professores e estudantes disponível em: https://www.dropbox.com/sh/uqo2rt7turwydsg/AAAsPj8FwYMl24LhmYSoOXica?dl=0

[7] https://www.google.com/forms/about/

convidados a darem sugestões e opiniões sobre a ferramenta, respondendo um formulário de avaliação contendo 10 tópicos (Quadro 3).

Quadro 3: tópicos usados na avaliação de professores e estudantes.

10 tópicos de avaliação apresentados aos professores e estudantes (inspirados nas heurísticas de Nielsen (1994))	
Tópico 1	Facilidade de uso da ferramenta.
Tópico 2	Visual da ferramenta.
Tópico 3	Tempo de resposta da ferramenta (demora ou rapidez na apresentação dos resultados).
Tópico 4	Liberdade (ou flexibilidade) na navegação.
Tópico 5	Visualização do progresso (o usuário consegue perceber o quanto falta para concluir o curso? - pontuação e progresso).
Tópico 6	Clareza dos textos explicativos.
Tópico 7	Clareza das mensagens de erro ou alerta.
Tópico 8	Satisfação de uso.
Tópico 9	Eficácia para a aprendizagem.
Tópico 10	Feedback a erros nos exercícios.
3 tópicos de avaliação apresentados apenas aos professores (didático-pedagógicos)	
Tópico 11	Completude do conteúdo.
Tópico 12	Utilidade para uso no ensino formal (sala de aula).
Tópico 13	Utilidade para uso autônomo pelos estudantes.

Fonte: o autor.

Os testes continham 10 questões de múltipla escolha e abordavam os assuntos apresentados na ferramenta. No entanto, essas questões são diferentes dos exercícios mostrados na ferramenta. As questões foram elaboradas de modo que elas envolvessem mais de um assunto apresentado na ferramenta. Ou seja, para que o estudante respondesse uma questão ele deveria dominar mais de um assunto. Acreditamos que o modo como foram feitas as questões (diferentes dos exercícios da ferramenta e exigindo o entendimento de mais de um assunto) evita viés causado pelos testes.

Para a avaliação da aprendizagem, as notas do primeiro e segundo teste (com escala de 0 a 100 pontos) foram comparadas mediante o uso dos parâmetros de estatística descritiva: média e mediana. Como havia dois grupos de estudantes com nível de formação distinta (graduação e pós-graduação), a comparação foi feita de três modos: com todos os 15 estudantes, apenas com os 11 estudantes de graduação e apenas com os 4 estudantes de pós-graduação. Vale ressaltar que 4 estudantes da graduação e 4 estudantes da pós-graduação tiveram algum contato com o assunto de redes de Petri antes de serem convidados para participarem da avaliação. Desse modo, a comparação também foi feita com o grupo de 7 estudantes que não tiveram contato com o assunto de redes de Petri antes de serem convidados para a avaliação. Por fim, foi feito um estudo de estatística inferencial (teste não paramétrico de Wilcoxon) a fim de identificar se o uso da ferramenta contribuiu para o aumento das notas dos estudantes.

Tanto os testes quanto o formulário foram implementados com o uso do *Google forms*, e enviados por *e-mail* aos estudantes. A partir dos *feedbacks* dos professores e estudantes, foram implementadas e definidas as melhorias na ferramenta. No Capítulo 6 são apresentados os resultados das avaliações.

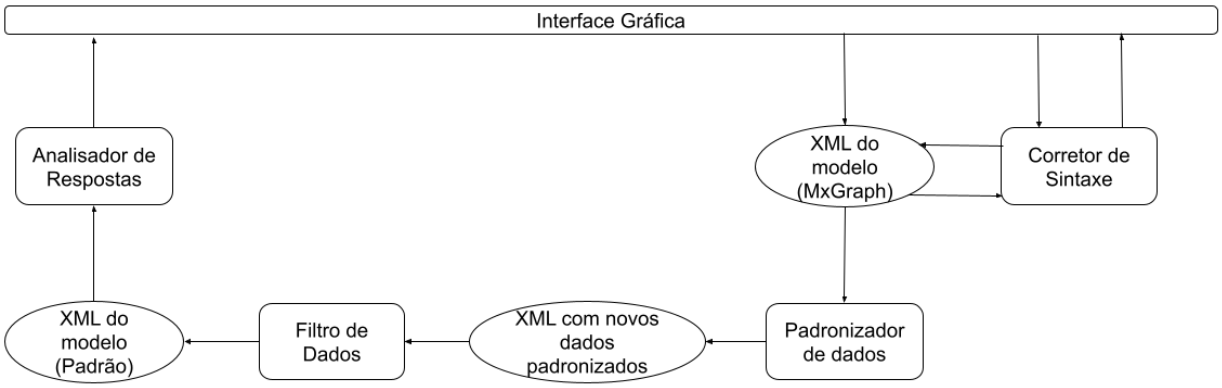
5 A FERRAMENTA TRYRDP

Este capítulo está dividido em três seções. A primeira aborda a arquitetura da ferramenta. A segunda apresenta o funcionamento e a interface gráfica. A terceira apresenta os assuntos abordados nos tópicos da ferramenta.

5.1 ARQUITETURA DA TRYRDP

A TryRdP foi criada para permitir o uso autônomo e interativo por parte dos estudantes, de modo a possibilitar uma melhor compreensão dos assuntos. Assim, ela foi desenvolvida para permitir não somente a modelagem e a análise de modelos, mas também para prover *feedbacks* em tempo real para ajudar os estudantes em cada etapa do processo de aprendizagem. A Figura 16 mostra o fluxograma da arquitetura adotada. A partir da interface gráfica, o fluxograma apresenta a execução de cada módulo de acordo com o sentido das setas. Segue a descrição de cada módulo:

Figura 16: arquitetura da TryRdP.



Fonte: o autor.

- **Módulo 1 - Interface Gráfica:** nesse módulo, o usuário da ferramenta interage com a interface gráfica que será apresentada na próxima seção. Esse módulo foi desenvolvido com o uso da biblioteca gráfica *MxGraph*. A biblioteca *MxGraph* possibilita criar um arquivo em XML e o salva na memória temporária do computador cliente. Esse arquivo é usado para

armazenar dados dos objetos gráficos introduzidos pelo usuário em tempo real. A biblioteca possui um sistema que armazena dados gerais sobre os objetos gráficos e suas relações, por exemplo, dados sobre a posição (em *pixel*) dos elementos no plano da interface gráfica e o seu *id* (um número único que identifica cada objeto). Em termos de RdP, esse sistema pode ser programado para armazenar dados sobre a rede e sobre os elementos da rede, ou seja, ele armazena quais os elementos da rede foram usados (transições, lugares ou arcos) e as propriedades desses elementos. Esse arquivo XML será utilizado nos Módulos 2 e 3 para obtenção de informações a respeito da RdP.

- **Módulo 2 - Corretor de Sintaxe:** aqui vale destacar que os Módulos 1, 2 e 3 são executados de modo conjunto e, também, são executados em tempo real. O Módulo 2 foi desenvolvido para ser um corretor da sintaxe da RdP. Desse modo, conforme o usuário vai montando a rede na interface gráfica, o Módulo 2 vai sendo executado e faz o trabalho de checar se a sintaxe da RdP está sendo corretamente colocada. Esse módulo também é responsável pelas mensagens indicando os erros de sintaxe para que os usuários corrijam. O Módulo 2 também é importante pelo fato de que é necessário que a RdP esteja com a sintaxe correta antes da padronização dos dados que é executado pelo Módulo 3.
- **Módulo 3 - Padronizador de Dados:** esse módulo é o responsável por obter o arquivo XML gerado pelo *MxGraph* e criar um padrão de armazenamento de dados. Embora o arquivo XML gerado pelo *MxGraph* traga as relações e propriedades dos elementos, ele não é amigável para os propósitos do uso da RdP devido à complexidade desse formalismo. Isso ocorre pois esse arquivo XML não armazena todos os detalhes existentes nas relações dos elementos de uma RdP. O arquivo XML gerado pelo *MxGraph* armazena dados sobre os objetos tomando por base a origem e a chegada de um determinado arco. As RdP são complexas e as relações binárias do XML gerado pelo *MxGraph* não expressam de maneira acurada esse formalismo. Desse modo, o Módulo 3 traz uma forma mais amigável de armazenamento dos dados dos elementos e isso inclui adicionar novas *tags* com novos dados ao arquivo XML gerado pelo *MxGraph.* Essas *tags* referenciam dados que

trazem informações como, por exemplo, os pesos ou outras propriedades das transições imediatas conectadas a um lugar ou a diversos lugares e suas relações. A padronização de dados é importante para que seja feita uma análise das repostas (Módulo 5) de um modo mais simples.

- **Módulo 4 - Filtro de Dados:** esse módulo é executado logo após o usuário enviar a sua resposta para a análise. Esse módulo cria um novo arquivo XML contendo apenas uma parte das *tags* adicionadas no Módulo 3. O Módulo 4 apenas guarda as informações relevantes para a RdP e exclui as outras. Note que informações como posição (em *pixel*) dos elementos no plano da interface gráfica são excluídas. Além disso, uma parte dos dados padronizados, que foram adicionados no módulo 3, também podem ser excluídos. Esse novo arquivo XML será usado para a análise das respostas (Módulo 5).
- **Módulo 5 - Analisador de Respostas:** esse módulo é o responsável por verificar se a resposta enviada pelo usuário está correta ou não e, consequentemente, entregar pontos caso o usuário acerte a resposta. A partir do arquivo XML gerado no Módulo 4, é feita a análise e comparação dos dados do arquivo XML contendo a resposta do usuário com o arquivo XML da resposta correta. Arquivos XML das respostas corretas dos exercícios ficam armazenados no servidor e são utilizados assim que o Módulo 5 é executado.

A seguir iremos detalhar a interface gráfica da TryRdP, bem como o seu funcionamento básico, visto que é a parte principal da ferramenta.

5.2 FUNCIONAMENTO DA FERRAMENTA

A ferramenta TryRdP foi desenvolvida de modo a ter conteúdos didáticos relacionados à modelagem de sistemas usando as RdP, com exemplos de aplicações reais e exercícios a serem realizados em um editor de gráfico. A contextualização provida pelos exemplos de situações reais permite uma aprendizagem mais significativa, facilitando a compreensão dos estudantes ao situar os conteúdos em

situações familiares. Vale ressaltar que, como já mencionado na introdução, o escopo do assunto abordado na ferramenta engloba apenas as GSPN.

A TryRdP está dividida em três tópicos obrigatoriamente sequenciais: introdução, modelagem básica e modelagem avançada (ver Figura 17). Em cada tópico, são apresentados conteúdos teóricos sobre o formalismo e, posteriormente, exercícios correspondentes. Nos exercícios, a ferramenta TryRdP verifica se o modelo criado pelo estudante está correto, fornecendo feedback imediato. Cada exercício possui uma pontuação de acerto, de modo que o estudante precisa atingir uma quantidade mínima de pontos para poder passar para outros tópicos da ferramenta, os quais abordam tópicos mais avançados. Além disso, o aluno pode tentar fazer os exercícios quantas vezes quiser, ou ver a resposta, caso não consiga responder uma questão. Porém, caso o estudante opte por ver a resposta de uma questão, ele/ela não receberá a pontuação correspondente.

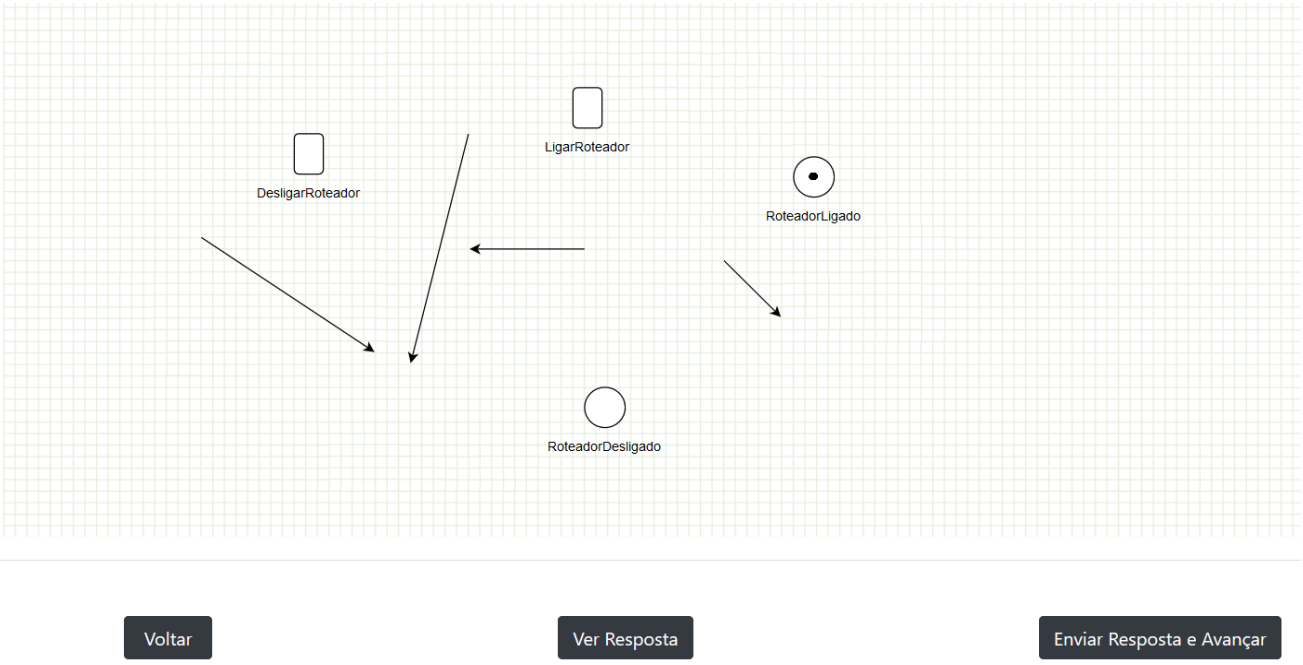
Figura 17: menu com os três tópicos (barra vertical à esquerda).

Fonte: o autor.

Existem na ferramenta dois tipos de exercícios: quebra cabeça e modelagem. Nos exercícios de quebra cabeça, o aluno deve conectar os arcos as transições e lugares (arrastando os elementos com o *mouse*), de modo a representar o modelo do sistema proposto. Os exercícios de quebra-cabeça são mais simples, pois fornecem todos os elementos necessários para o aluno montar a rede de Petri que modela o sistema apresentado – basta descobrir como conectá-los. Assim, a ferramenta é

baseada no modelo de aprendizagem por prática de exercícios por tentativa e erro, mas com espaço para exploração de ideias até chegar à solução ao longo desse processo. Desse modo, a ferramenta é alinhada à concepção de que o erro faz parte do processo de aprendizagem.

Figura 18: exemplo de exercício de quebra cabeça.

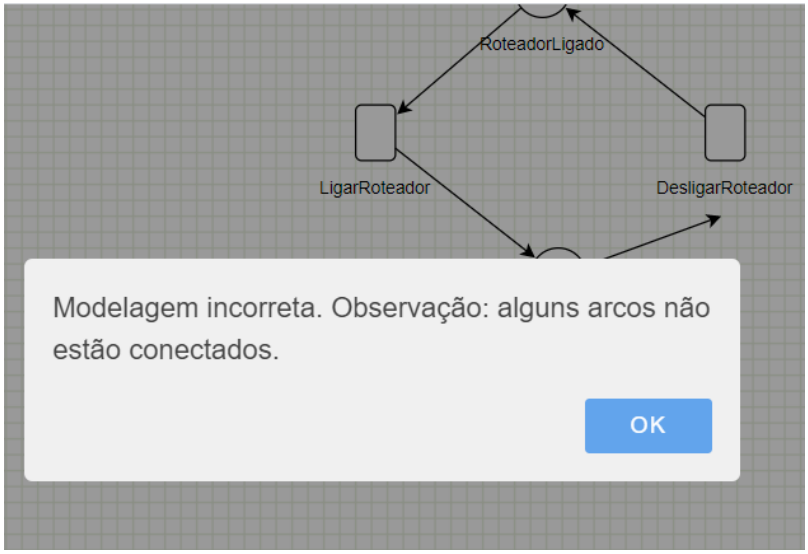


Fonte: o autor.

A Figura 18 mostra um exemplo de exercício de quebra cabeça, com o seguinte enunciado: "*Monte uma rede de Petri que representa o comportamento de ligar e desligar de um roteador. Os lugares abaixo representam estados do roteador que podem ser ligado ou desligado. Inicialmente, o roteador está ligado, de modo que existe um token no lugar 'RoteadorLigado' indicando tal condição inicial. As transições representam as ações de desligar ou ligar o roteador. Monte o modelo de modo que, após o roteador ser desligado, ele possa ser ligado novamente (e vice-versa), formando uma rede cíclica com arcos apontando para um único sentido*". Como se pode observar pelos botões na parte inferior da Figura 18, o aluno poderá responder à questão ou ver a resposta. Caso o aluno não acerte a resposta, ao clicar no botão "Enviar Resposta e Avançar", uma mensagem de erro irá aparecer na tela (ver Figura 19). Consequentemente, o aluno poderá corrigir o modelo e submeter novamente a

resposta da questão. Vale ressaltar que, como nos exercícios de quebra cabeça os elementos já são dados, os *feedbacks* dos erros são mais simples do que nos exercícios de modelagem. Nos exercícios de modelagem os *feedbacks* são mais detalhados e específicos, conforme explicado a seguir.

Figura 19: mensagem indicando o erro cometido.

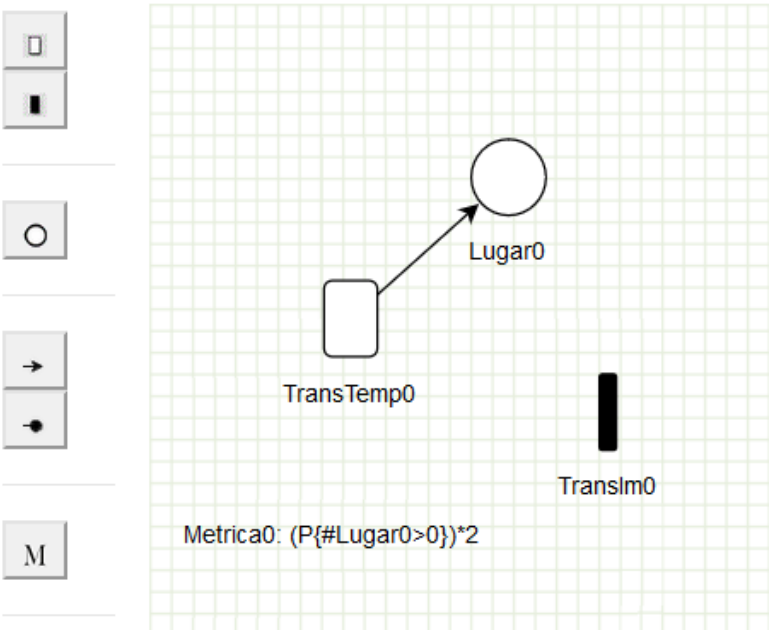


Fonte: o autor.

Os exercícios de modelagem são mais complexos, pois o estudante deve selecionar os elementos para modelar o sistema (usando os botões do lado esquerdo da Figura 20). Ou seja, esses elementos não são apresentados ao estudante como no quebra-cabeça. Ao clique no botão, o respectivo elemento é exibido no editor gráfico, para manipulação por parte do estudante. Na Figura 20, os botões de cima para baixo representam, respectivamente, a transição temporizada, a transição imediata, o lugar, o arco, o arco inibidor e a métrica. O aluno pode remover ou adicionar os elementos das RdP ao editor gráfico quantas vezes quiser. Além disso, diferentemente dos exercícios de quebra cabeça, nos exercícios de modelagem os elementos possuem propriedades, e de acordo com o enunciado do exercício, o aluno deverá adicionar valores às propriedades de alguns dos elementos. A Figura 21 mostra quatro propriedades de uma transição temporizada: propriedade “Label” é um nome ou título que o usuário pode colocar no elemento; a propriedade “Tempo” é o *delay* associado à transição temporizada; a propriedade “Tipo de Servido” indica se uma determinada transição pode fazer disparos simultâneos (no TryRdP usamos apenas o *Single Server*); já a

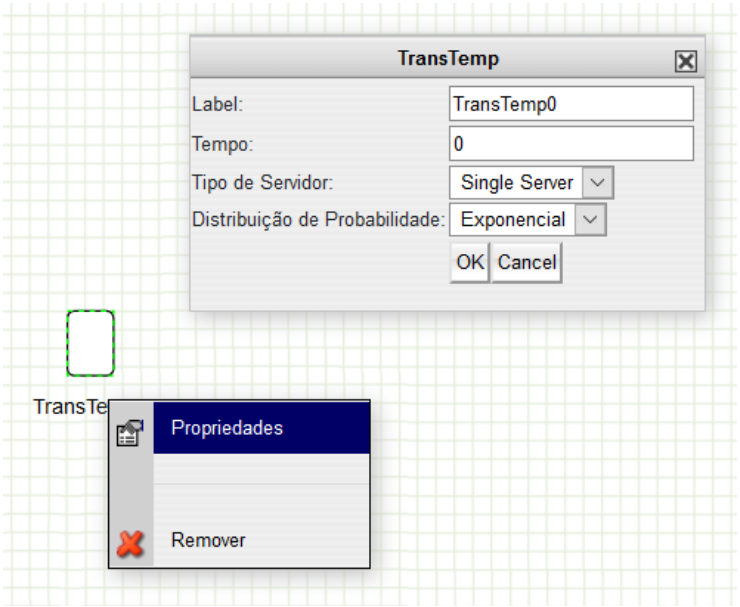
propriedade “Distribuição de Probabilidade” é a distribuição adotada (no TryRdP usaremos apenas a distribuição exponencial).

Figura 20: exemplo de exercício de modelagem.



Fonte: o autor.

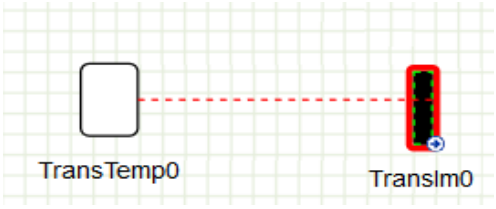
Figura 21: propriedades de uma transição temporizada em um exercício de modelagem.



Fonte: o autor.

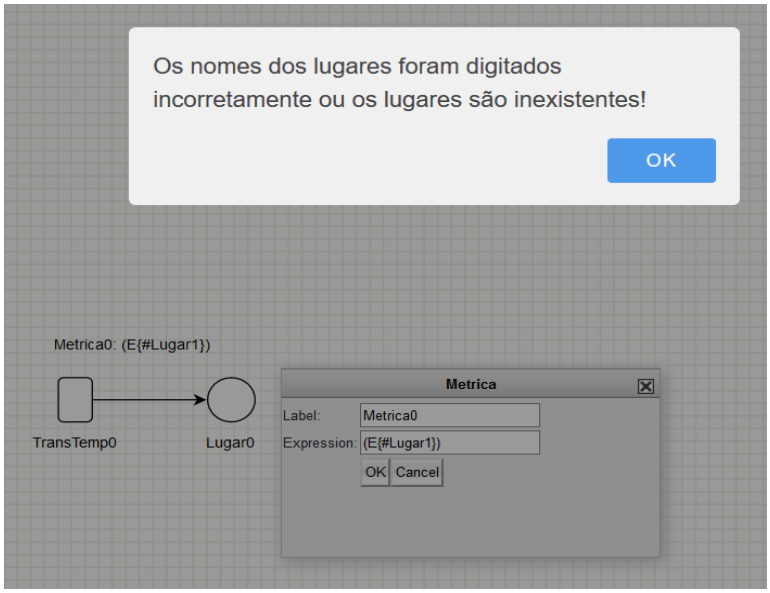
Os exercícios de modelagem também possuem recursos que corrigem certos erros em tempo real, isto é, enquanto o aluno está fazendo o exercício. Por exemplo, a Figura 22 mostra um erro que foi a tentativa de conectar duas transições. Nas RdP, uma transição só pode se conectar com um lugar e vice-versa. Além desses erros, existem os erros relacionados à digitação incorreta das propriedades dos elementos e das métricas do modelo. A Figura 23 mostra um erro em que na métrica (E{#Lugar1}), foi digitado o nome de um lugar que não existe. Note que, nesse caso, a métrica correta seria (E{#Lugar0}), visto que o nome correto do lugar é *Lugar0*. Vale ser ressaltado que essas correções servem para auxiliar os estudantes quanto ao uso correto das RdP, permitindo que eles foquem no modelo conceitual, de modo a facilitar o uso futuro de ferramentas computacionais de análises e simulação das RdP, como o Mercury (SILVA et al., 2015) ou o TimeNet (ZIMMERMANN, 2017).

Figura 22: alerta de erro relacionado ao formalismo das RdP.



Fonte: o autor.

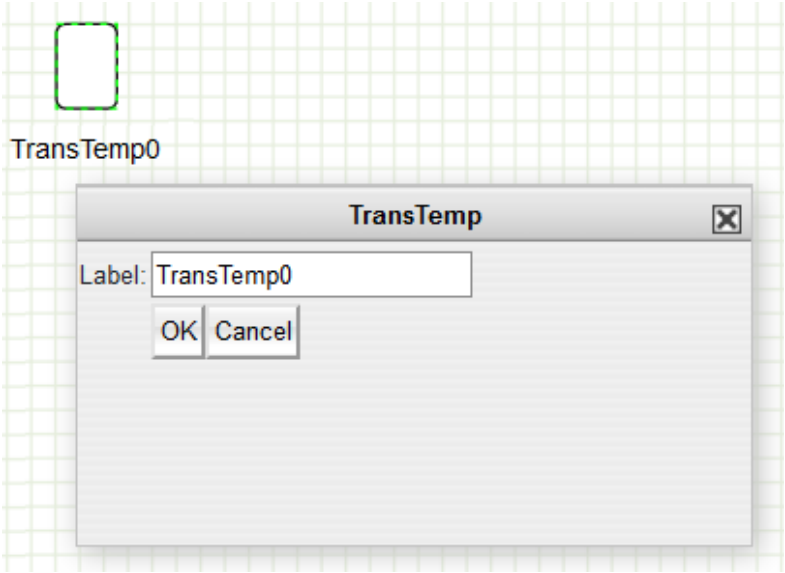
Figura 23: alerta de erro de digitação.



Fonte: o autor.

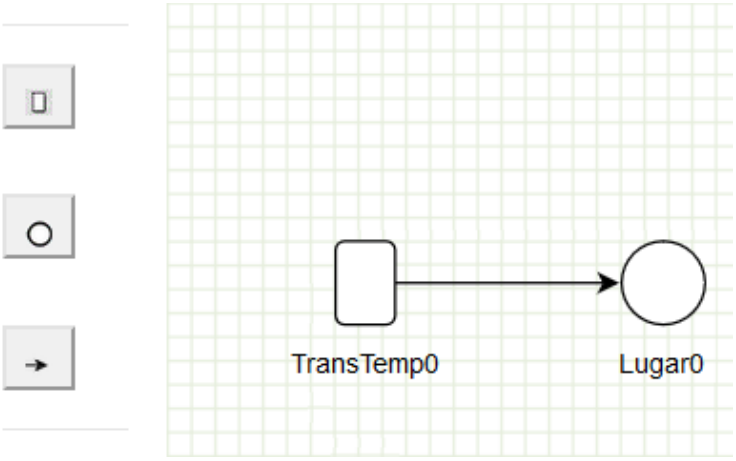
O nível e o tipo dos exercícios variam de acordo com o progresso do estudante. Nos dois primeiros tópicos (introdução e modelagem básica), os exercícios são do tipo quebra-cabeça e modelagem. No último tópico (modelagem avançada), os exercícios são apenas de modelagem. Apenas para os exercícios de modelagem, os elementos que compõem a rede de Petri e as propriedades de certos elementos variam conforme o progresso do estudante. Por exemplo, na Figura 24, são mostradas as propriedades de uma transição antes de introduzirmos o conceito de uma transição temporizada. Esse conceito só é apresentado no tópico 2 (modelagem básica) da ferramenta. Após apresentar esse novo conceito, novos elementos são adicionados à transição (agora chamada de transição temporizada), conforme apresentado na Figura 21. Do mesmo modo, elementos das redes de Petri são mostrados conforme o progresso do aluno. Por exemplo, na Figura 25 são mostrados apenas três elementos que compõem uma rede de Petri no tópico 1: transição, lugar e arco. Já no tópico 3, que aborda arcos inibidores, transições imediatas e métricas, esses novos elementos são mostrados (ver Figura 20).

Figura 24: propriedades da transição apresentada na tópico 1.



Fonte: o autor.

Figura 25: elementos nos exercícios de modelagem na tópico1.

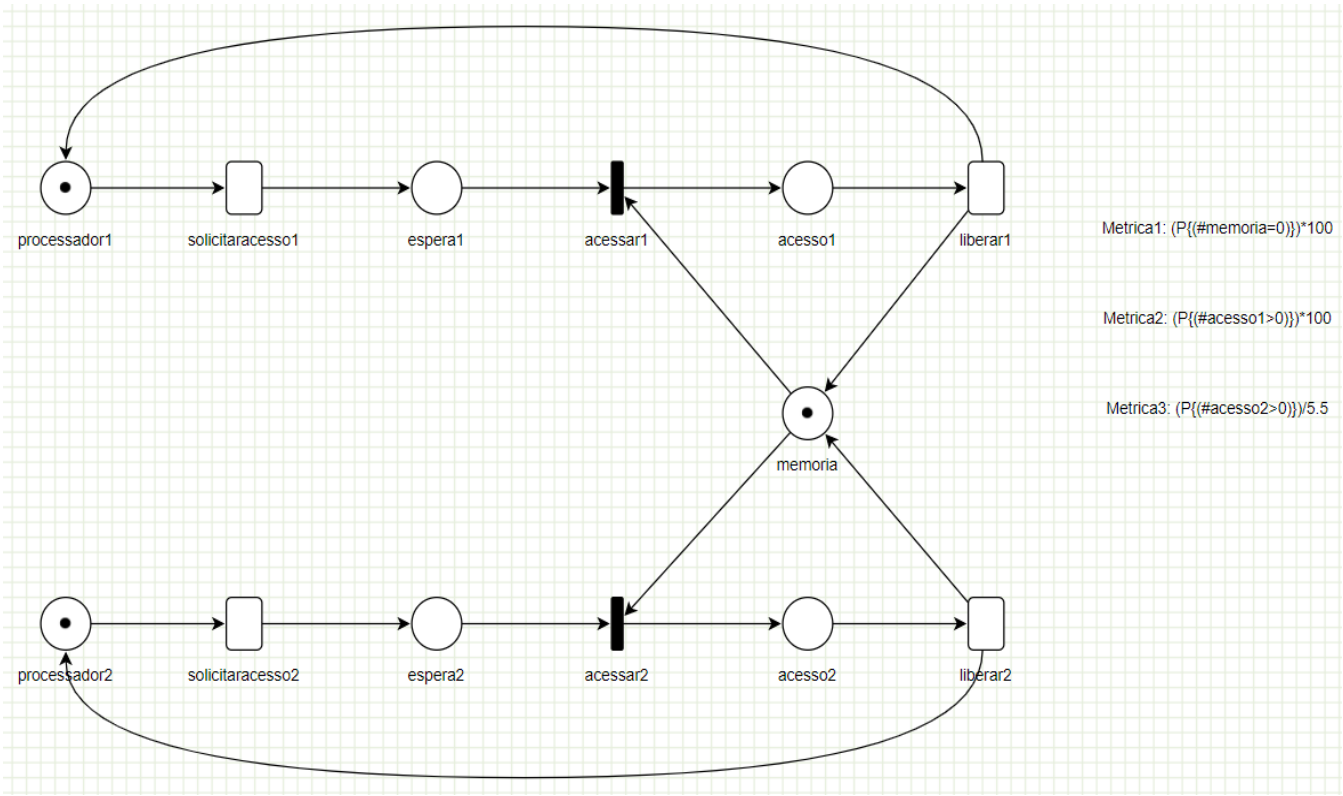


Fonte: o autor.

Para exemplificar um exercício de modelagem de nível avançado, considere o seguinte enunciado: *"Suponha que dois processadores desejam acessar uma memória compartilhada. Ambos têm o mesmo comportamento. Eles trabalham localmente por algum tempo, depois solicitam o acesso à memória compartilhada, acessam a memória e finalmente liberam a memória. O processador que estiver acessando a memória não pode fazer uma nova solicitação de acesso sem que antes tenha terminado o seu acesso e liberado a memória compartilhada. A memória compartilhada é um recurso exclusivo. Isto é, enquanto um processador acessa a memória compartilhada, o outro tem que esperar até que o primeiro tenha finalizado seu acesso. Assumimos que, se a memória não está em uso, um processador pode acessar imediatamente. Adicionalmente, assumimos que, em média, o processador 1 solicita acesso a memória a cada 25 nanosegundos, enquanto o processador 2 solicita acesso a cada 20 nanosegundos. Também, em média, o processador 1 gasta 5 nanosegundos acessando a memória. Já o processador 2 gasta, em média, 5.5 nanosegundos. Construa uma rede de Petri que representa esse sistema. Além disso, escreva três métricas que calculam: i) a taxa média (em porcentagem) de utilização geral da memória compartilhada; ii) a taxa média (em porcentagem) de utilização da memória pelo processador 1; iii) e a taxa média (solicitação/nanosegundo) que o processador 2 solicita acesso a memória."* A resposta desse exercício é mostrada na Figura 26 e com as seguintes métricas seguindo o anunciado, respectivamente: i) *Metrica1: (P{#memoria=0})*100*; ii) *Metrica2: (P{#acesso1>0})*100*; iii) *Metrica3: (P{(#acesso2>0)})/5.5*.

Observe que o exercício de nível mais avançado requer que o estudante já tenha domínio dos conceitos fundamentais sobre rede de Petri, diferentemente do exercício mostrado na Figura 18 da seção de introdução da ferramenta TryRdP. Além disso, conforme já mencionado, quando um aluno responde e acerta uma questão, ele recebe uma pontuação e, dependendo do total de pontos, o aluno pode seguir para o próximo tópico. No entanto, se o aluno não conseguir o número mínimo de pontos para seguir para a próximo tópico, ele receberá uma mensagem indicando que ele pode fazer os exercícios que ele não concluiu e obter a pontuação mínima para passar de tópico. Ao final do último tópico, se o aluno conseguir atingir a pontuação mínima, ele/ela obterá um certificado de participação.

Figura 26: exemplo de exercício de modelagem de nível avançado.



Fonte: o autor.

5.3 CONTEÚDO ABORDADO NA FERRAMENTA

O conteúdo didático apresentado na TryRdP foi dividido em três tópicos: introdução, modelagem básica e modelagem avançada. A seguir iremos descrever e dar exemplos dos assuntos de cada um desses tópicos.

5.3.1 Tópico 1 – Introdução

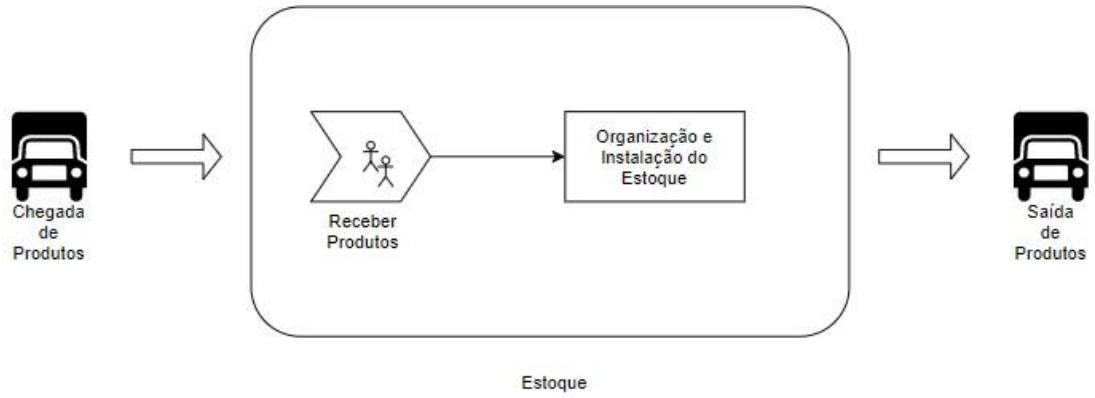
A introdução visa promover os seguintes conhecimentos nos estudantes:

- Conceitos gerais a respeito de modelagem com RdP;
- Elementos básicos de uma RdP e o que eles podem representar;
- Funcionamento das regras de disparos de uma transição.

A introdução possui exercícios simples de quebra cabeça e de modelagem (por exemplo, o exercício mostrado na Figura 18). Esse tópico visa formar a base do entendimento sobre modelagem e RdP, a qual é fundamental para que os estudantes possam compreender os assuntos abordados nos tópicos seguintes. Abaixo seguem dois trechos do tópico 1. Esses trechos são mostrados aos estudantes logo após a apresentação do conteúdo acerca dos elementos básicos de uma RdP.

Note que esse trecho originário da ferramenta é relativo à Figura 27: "*A fim de exemplificar o uso das redes de Petri, iremos modelar um exemplo de um sistema real usando os elementos básicos de uma RdP. Considere um sistema de estoque, o qual é composto por caminhões para entregar e distribuir produtos, bem como funcionários para receber e colocar os produtos nas instalações do estoque. Abaixo é apresentado um pequeno esquema desse sistema. Como é possível observar, um sistema de estoque pode possuir muitos detalhes. No entanto, a nível de modelagem, é possível considerar apenas o que é relevante para o estudo. Isto é, concentrar nas características relevantes e ignorar as irrelevantes. Esse processo chama-se abstração. Assim, para o sistema de estoque iremos considerar apenas que ocorrem duas ações: (i) a chegada de um produto para ser armazenado e (ii) a saída de um produto. Além disso, iremos considerar que o estoque está inicialmente sem produtos e ele não tem limite de capacidade.*"

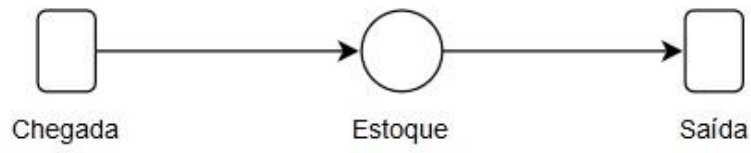
Figura 27: sistema de estoque.



Fonte: o autor.

O trecho a seguir é a continuação do trecho anterior e é relativo à Figura 28: *“Na Figura abaixo é apresentada uma RdP que representa um sistema de estoque sem limite de capacidade. No entanto, primeiramente, é necessário entender quais são os estados e as ações do sistema. Conforme explicado anteriormente, o sistema possui duas ações principais que consistem na chegada e saída de um produto. Além disso, o sistema pode assumir dois estados: com estoque e sem estoque. Uma vez levantados os estados e as ações do sistema, é possível modelá-lo usando as redes de Petri. Visto que ações e os estados são representados por transições e lugares respectivamente, o modelo resultante possui duas transições chamadas de Chegada e Saída que representam a chegada e a saída dos produtos, e um lugar chamado de Estoque que representa o estoque do sistema. Por fim, os produtos são representados por tokens, pois eles são usados para especificar o estado do sistema (ex.: com ou sem estoque). Para representar o fluxo dos produtos, os lugares e as transições são ligados através de arcos. Note que inicialmente não há produtos no estoque, então não há tokens no lugar Estoque.”*

Figura 28: exemplo básico de um estoque sem limite de capacidade.



Fonte: o autor.

5.3.2 Tópico 2 – Modelagem Básica

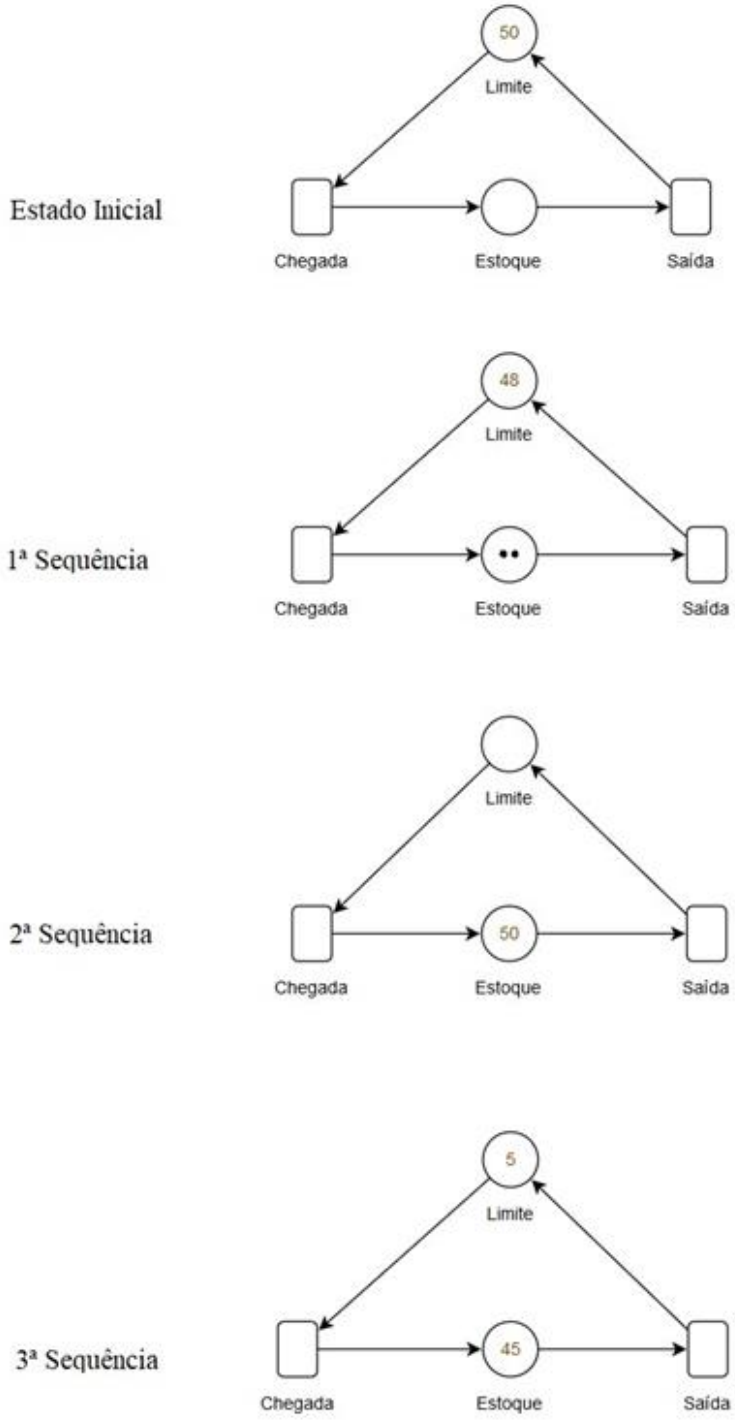
O objetivo desse tópico é aumentar o leque de exemplos e abordar os seguintes assuntos:

- Construções elementares da RdP, ou seja, sequência, distribuição, junção e escolha não-determinística;
- Modelagem com *Buffer*;
- Modelagem com arcos inibidores;
- Modelagem com arcos de peso maior que um;
- Rede de Petri estocástica;
- Conceito básico sobre transição imediata;
- Exemplos de modelos de filas básicos;
- Modelo de falha e reparo.

Esse tópico de modelagem básica conta com exercícios de quebra cabeça e de modelagem. Ele visa aprofundar todos os conceitos básicos a respeito das RdP que foram introduzidos no tópico anterior. Vale ser ressaltado que os exemplos do tópico 2 são mais realistas. A seguir serão mostrados trechos originais do tópico 2 relacionados aos assuntos de modelagem com *buffer*, modelo básico de filas e modelo de falha e reparo.

O trecho a seguir é parte do assunto sobre modelagem com buffer e está relacionado à Figura 29: *“No tópico 1 mostramos um modelo de estoque em que não havia limite de capacidade. Aqui, como mostrado na figura abaixo, temos um modelo com limite de 50 produtos no estoque. No Estado Inicial, o estoque está vazio; em seguida, ocorre a chegada de dois produtos. Isto é, a transição Chegada é disparada duas vezes (Sequência 1). Posteriormente, a transição Chegada é disparada mais 48 vezes (Sequência 2), completando assim, o limite do estoque em 50 produtos. Ao se atingir o limite de 50 produtos no estoque, a transição Chegada não poderá ser mais executada, visto que todos os tokens do lugar Limite foram consumidos. Por fim, a transição Saída é disparada 5 vezes (Sequência 3) liberando espaço para que sejam armazenados mais 5 produtos no estoque. Essas estruturas (modelos semelhante ao apresentado abaixo) das redes de Petri são chamadas de "buffers".”*

Figura 29: modelo de estoque com limite de 50 produtos.

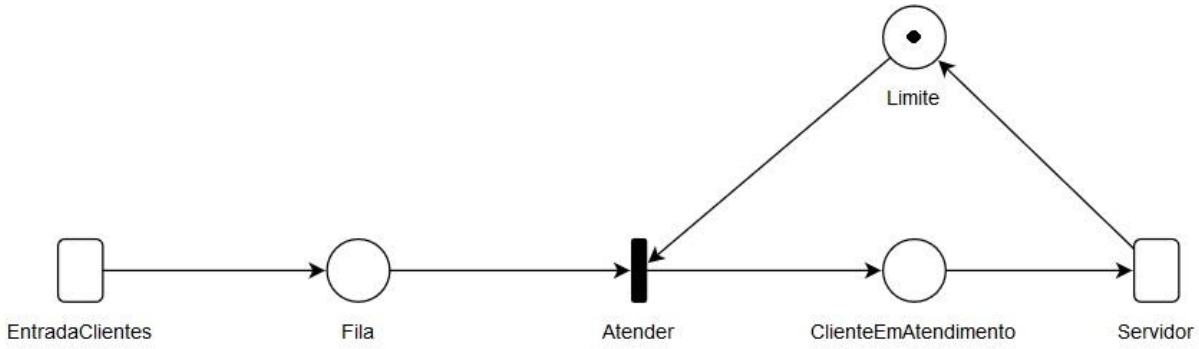


Fonte: o autor.

O trecho a seguir está relacionado a Figura 30 e faz parte do assunto de modelos básicos de filas: *“Sistemas de filas são sistemas que exigem que os clientes ou objetos esperem para que sejam atendidos ou recebidos por algum serviço. As filas*

estão presentes em diversos setores de nossas vidas, pessoas esperando no caixa do supermercado ou pacotes que chegam a um roteador para serem encaminhados. No modelo abaixo, é representado um sistema de fila em que clientes entram no sistema, esperam na fila de espera e depois são atendidos um por vez pelo servidor. Nesse sistema, a fila de espera é representada pelo lugar Fila e não há limites de clientes nela. Além disso, não há limites de clientes dentro do sistema. Existe apenas, um limite de atendimento de um cliente por vez. Note que a transição Atender é imediata, pois quando o servidor estiver disponível, o atendimento ao cliente se inicia imediatamente. O lugar Limite indica que apenas um cliente pode ser atendido por vez. Vale ser ressaltado que o servidor pode ser um caixa de um banco, um servidor computacional ou um atendimento num hospital, por exemplo."

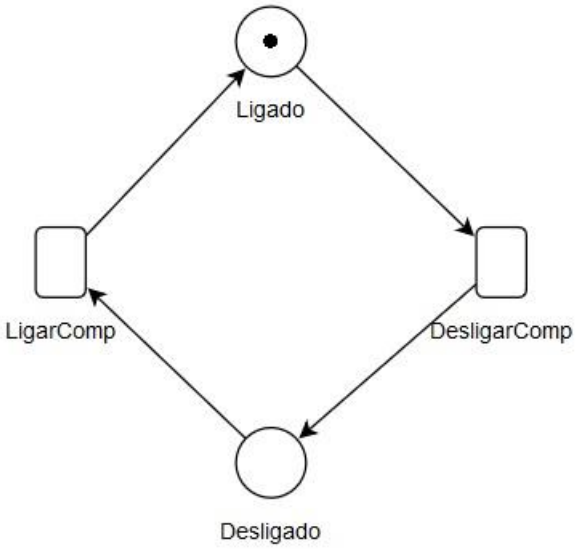
Figura 30: exemplo de um sistema de fila.



Fonte: o autor.

Esse trecho descreve um modelo de falha e reparo e, também, está relacionado ao assunto sobre disponibilidade que será mostrado no tópico 3 (ver Figura 31): *"De forma sucinta, quando falamos em "Falha e Reparo" estamos nos referindo a sistemas cujos elementos podem estar, por exemplo, ligados ou desligados, funcionando ou não funcionando, disponíveis ou indisponíveis. Abaixo é mostrada uma rede que representa o comportamento de falha e reparo de um computador, onde ele pode assumir dois estados: ligado e desligado. Adicionalmente, o modelo inicia no estado ligado, devido à presença de um token no lugar Ligado."*

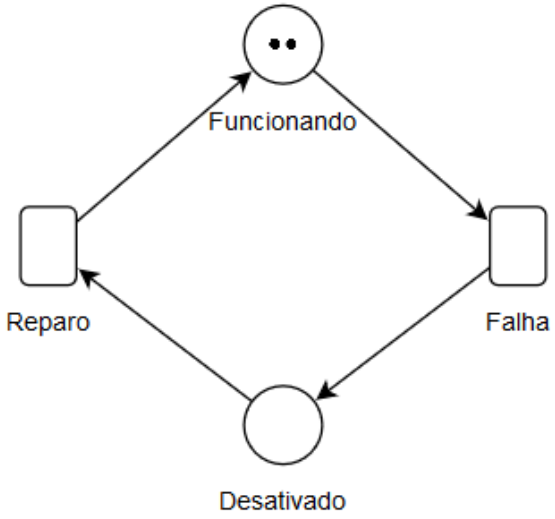
Figura 31: modelo que representa o comportamento de falha e reparo de um computador.



Fonte: o autor.

O trecho a seguir dá continuidade ao assunto do trecho anterior e mostra outro exemplo do comportamento de falha e reparo. Esse trecho é original da ferramenta e é relativo à Figura 32: *“Em tecnologia da informação, redundância é uma abordagem que visa aumentar a tolerância a falha dos sistemas. Um dos tipos de redundância usados em sistemas distribuídos é o "hot standby". Essa redundância consiste de duas infraestruturas redundantes, sendo uma a primária e a outra a secundária. A primária é a infraestrutura onde os processos estão sempre ocorrendo, enquanto a secundária é a infraestrutura de redundância, isto é, caso a infraestrutura primária falhe, a secundária é ativada. Mas note que pelo fato de ser uma redundância do tipo "hot-standby", a infraestrutura secundária está sempre ligada. Ou seja, ela está pronta para receber a carga de trabalho caso a primária falhe, visto que seus dados são mantidos em sincronia com a infraestrutura primária. O modelo abaixo representa a técnica de redundância "hot standby", onde as duas infraestruturas são representadas pelos dois tokens no lugar Funcionando.”*

Figura 32: modelo *Hot Standby*.



Fonte: o autor.

5.3.3 Tópico 3 – Modelagem Avançada

O tópico de modelagem avançada procura estimular os estudantes a compreenderem conceitos mais avançados, de modo que possam aumentar a sua capacidade de abstração e uso das RdP. Esse tópico visa promover os seguintes conhecimentos:

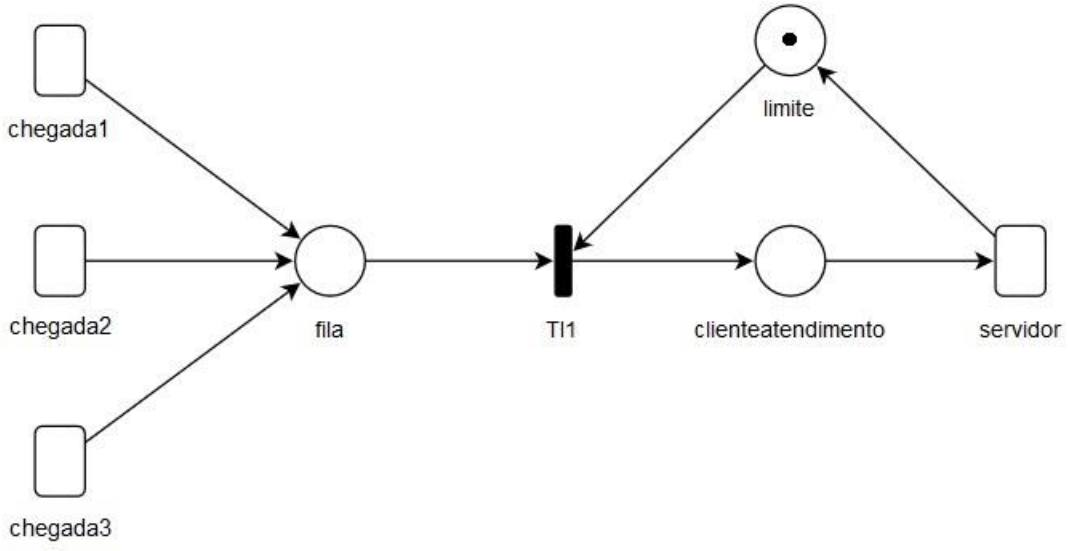
- Entendimento e uso de métricas;
- Entender e usar as propriedades das transições imediatas;
- Aprofundar os conceitos sobre teoria de filas e lei de Little.
- Apresentar uma introdução rápida sobre os conceitos de performance, disponibilidade e performabilidade.

Diferentemente dos outros tópicos, o tópico 3 apresenta apenas exercícios de modelagem. Além disso, os exercícios são bem mais complexos que nos tópicos anteriores (note o exercício mostrado na Figura 26). Nesse tópico é apresentado uma introdução bem rápida sobre os conceitos de performance, disponibilidade e performabilidade. Esses conceitos são bastante extensos e discutidos na literatura acadêmica. Assim, vale ressaltar que esses conceitos são introduzidos de modo rápido e visa fazer o estudante ter uma noção básica sobre eles. O objetivo da ferramenta é auxiliar no aprendizado de modelagem de sistemas, logo, detalhes teóricos sobre

esses conceitos não são apresentados, visto que, também, a ferramenta ficaria muito extensa e cansativa para os estudantes. A seguir serão mostrados trechos originais do tópico 3 e que estão relacionados aos assuntos de teoria de filas e disponibilidade.

Esse trecho é mostrado no assunto relacionado à teoria de filas e exemplifica um sistema de filas simples onde os clientes chegam a partir de três fontes diferentes (ver Figura 33): *"Neste ponto, é necessário que vocês entendam que uma das principais métricas é aquela relativa ao cálculo da taxa média efetiva de chegada de clientes (vazão), λef, pois o seu cálculo pode ser feito de diferentes formas a depender do modelo criado. Aqui na ferramenta TryRdP, para uma fila com um servidor, vamos apenas calcular λef como (P{(#clienteatendimento>0)})*μ, visto que tal cálculo é generalista (serve para a maioria das situações) e é mais intuitivo de se fazer quando estamos trabalhando com uma RdP complexa. Apesar disso, vamos mostrar alguns exemplos com diferentes formas de cálculo para λef. Agora, observe a RdP abaixo, ela representa um sistema de fila com um servidor e entrada a partir de três fontes diferentes. Supondo que as taxas médias nominais de chegada de clientes a partir de cada uma das fontes fossem λ1, λ2 e λ3, o cálculo de λef seria λef = λ1 + λ2 + λ3."*

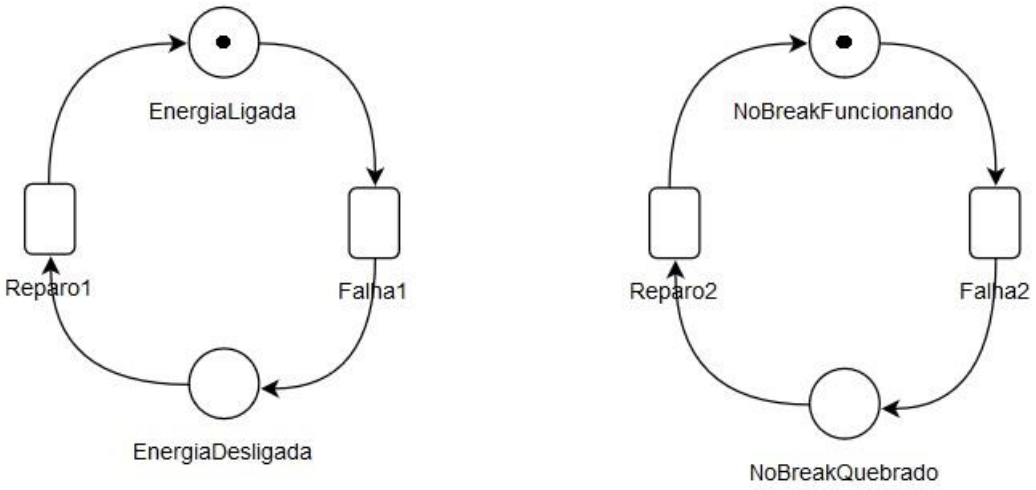
Figura 33: fila com um servidor e entrada a partir de três fontes diferentes.



Fonte: o autor.

O trecho a seguir foi retirado do assunto sobre disponibilidade e está relacionado a Figura 34: *“A disponibilidade de um sistema é a probabilidade de que ele esteja operacional. No subtópico 2.12, falamos a respeito de modelos de falha e reparo, no entanto, não havíamos falado a respeito de cálculo de métricas e por isso não citamos o termo "disponibilidade". Considerando o uso de métricas para cálculo da disponibilidade nos exemplos do subtópico 2.12, tais modelos seriam chamados de modelos de disponibilidade. De qualquer forma, através de um exemplo vamos mostrar como escrever uma métrica que calcula a disponibilidade de um sistema. Nas figuras abaixo, temos duas RdP que representam, por exemplo, um computador ou qualquer outro equipamento que está ligado na energia elétrica e, caso a energia seja desligada, um NoBreak irá suprir o equipamento. Observe que o NoBreak pode não funcionar. Assim, a disponibilidade desse sistema deve levar em conta a energia ligada ou o NoBreak funcionando. Desse modo, a métrica seria (P{(#EnergiaLigada>0)OR(#NoBreakFuncionando>0)}).”*

Figura 34: modelo de disponibilidade com *NoBreak.*



Fonte: o autor.

6 AVALIAÇÃO DA FERRAMENTA

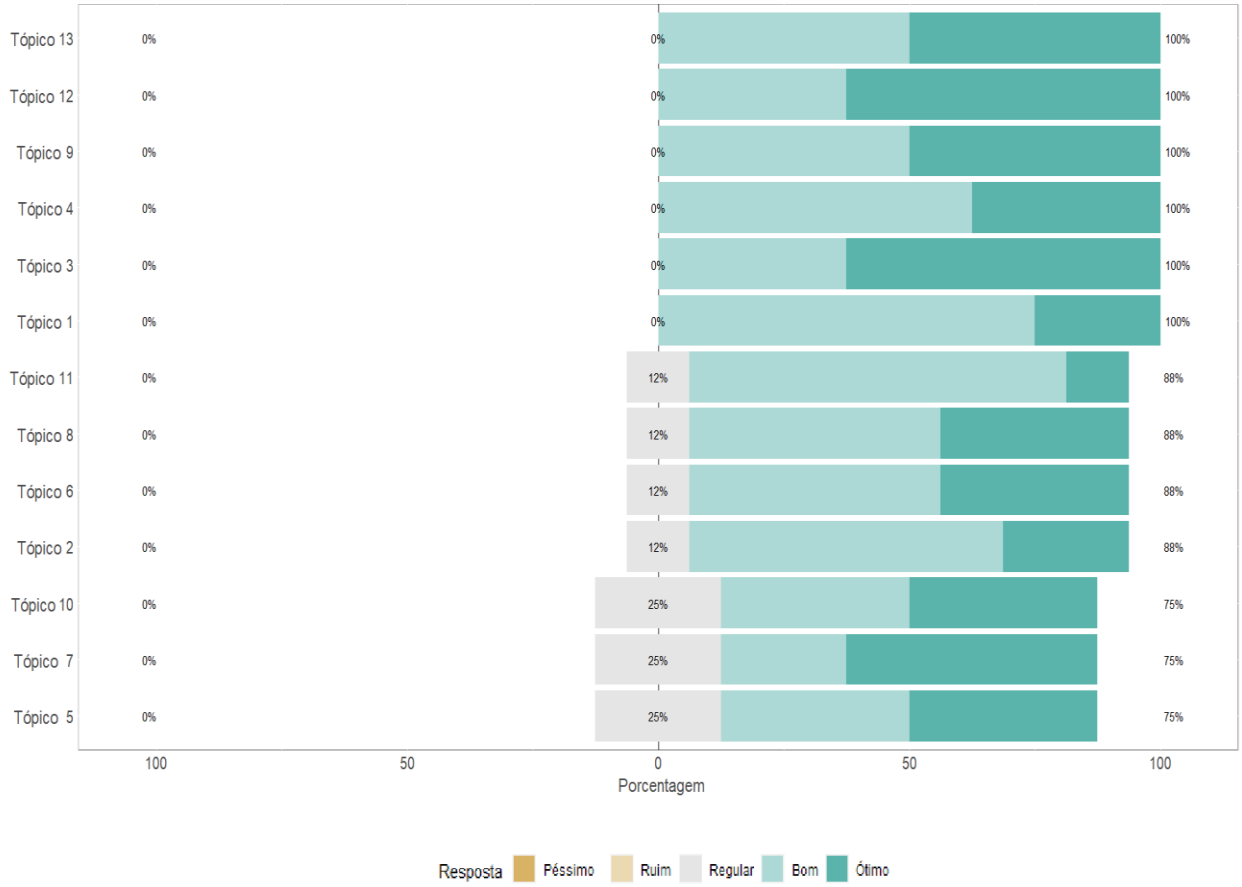
Uma etapa importante no desenvolvimento de uma ferramenta educativa é a avaliação pelo público-alvo. A partir dela, é possível obter indícios sobre a eficácia para o aprendizado e a usabilidade da interface. Abaixo seguem as descrições dos resultados das avaliações com professores e estudantes, conforme detalhadas na Seção 4.1.

6.1 AVALIAÇÃO COM PROFESSORES

A Figura 35 mostra o resultado das avaliações dos professores sobre cada um dos 13 tópicos (Quadro 3). O gráfico é dividido em três partes: esquerda, meio e direita. A parte da esquerda mostra a proporção em porcentagem da quantidade de avaliações do tipo “ruim” (cor amarelo) ou “péssimo” (cor amarelo escuro). A parte do meio mostra a proporção da quantidade da avaliação “regular” (cinza). Já a parte da direita mostra a proporção da quantidade de avaliações do tipo “bom” (cor verde claro) ou “ótimo” (cor verde escuro). Além disso, as avaliações dos tópicos são mostradas em ordem decrescente. Isto é, as avaliações que têm a maior soma de avaliações como “bom” e “ótimo” aparecem na parte superior, e as que têm mais avaliações negativas aparecem na parte inferior. Como pode ser observado, as avaliações foram muito boas e em nenhum dos tópicos houve as avaliações “péssimo” ou “ruim”.

Como se vê na Figura 35, os tópicos 1, 3, 4, 9, 12, e 13 foram os mais bem avaliados. Em especial, vale destacar o tópico 9, que trata da eficácia para a aprendizagem, aliado ao 12 e 13, que tratam da utilidade da ferramenta tanto para uso autônomo quanto mediado, indicando uma ótima percepção dos professores quanto à sua utilidade na aprendizagem. Nos campos abertos, os professores afirmaram que a ferramenta é um excelente instrumento de apoio ao ensino de modelagem de sistemas usando rede de Petri. Em particular, destacaram o tipo de exercício de quebra cabeça, que segundo eles, ajuda o estudante iniciante a compreender o processo de modelagem.

Figura 35: Resultados da avaliação feita pelos professores.



Fonte: o autor.

Por outro lado, os tópicos 5, 7 e 10 foram os menos bem avaliados, indicando que a percepção do progresso e o *feedback* provido pela ferramenta, embora considerados bons, poderiam melhorar. Esse é um aspecto importante visto que o *feedback* é essencial para a aprendizagem, principalmente nas situações de uso autônomo, em que a ferramenta será a fonte principal para compreensão do erro, e não há mediação de um professor para fornecer explicações. A seguir, são mostradas opiniões e sugestões de melhorias dadas pelos professores nos campos abertos, no formulário de avaliação. Nos quadros 4, 5, 6, 7 e 8, as sugestões e opiniões são apresentadas agrupadas por tema (foram omitidas sugestões de melhorias linguísticas e correção de erros ortográficos). Para cada opinião ou sugestão, apresentamos um comentário discutindo a questão.

Quadro 4

Sugestões/opiniões dos professores quanto à aprendizagem do conteúdo e nível de dificuldade	
Opinião	O exercício 11 foi considerado um pouco difícil para quem está iniciando os estudos em rede de Petri. Comentário dos autores: estruturalmente o modelo desse exercício é simples. Ele contém apenas duas transições temporizadas, duas transições imediatas e três lugares. No entanto, o raciocínio que o usuário deve ter para se chegar no modelo requer um pouco de trabalho. Vale ressaltar que o modelo desse exercício é usado na modelagem de sistemas computacionais e por isso foi escolhido. Além disso, caso o estudante tenha dificuldade com a questão, ele poderá ver a resposta. A ferramenta foi construída se baseando na concepção de que o erro faz parte do processo de aprendizagem.

Fonte: o autor.

Quadro 5

Sugestões/opiniões dos professores quanto à visualização do progresso	
Sugestão	Alguns professores indicaram que a visualização do progresso não estava muito evidente. Comentário dos autores: na nova versão da interface, essa questão foi melhorada, com uma barra colorida.

Fonte: o autor.

Quadro 6

Sugestões/opiniões dos professores quanto à facilidade e satisfação de uso da ferramenta	
Opinião	Alguns professores consideraram um pouco trabalhosa a ação de conectar os arcos com o movimento de arrastar (usando mouse ou *trackpad*), nos exercícios de quebra cabeça. Comentário dos autores: dadas as tecnologias usadas, esta foi a melhor forma encontrada para se fazer o quebra cabeça, porém outras formas de interação podem ser estudadas em trabalhos futuros.
Sugestão	Adicionar o botão *tokengame* nos exercícios de modelagem. Comentário dos autores: o *tokengame* é um recurso que gera uma animação no modelo de RdP construída pelo usuário. Esse recurso normalmente é apresentado em ferramentas de simulação e análise (ex.: Mercury). Embora represente uma visualização bastante interessante, a implementação da mesma é complexa e demanda bastante tempo. Consequentemente, devido a limitação de tempo desta pesquisa, o *tokengame* ficou de fora do escopo. No entanto, isso pode ser estudado futuramente e, talvez, deva ser utilizada outra tecnologia para se criar o *tokengame*.

Fonte: o autor.

Quadro 7

Sugestões/opiniões dos professores quanto à completude do conteúdo	
Sugestão	Alguns professores sugeriram como trabalho futuro a tradução para o inglês e espanhol da ferramenta. Comentário dos autores: de fato, almejamos no futuro traduzir os textos da ferramenta para outros idiomas.
Sugestão	Alguns professores sugeriram adicionar referências de artigos em formato de *links* para que o estudante, caso ele deseje, busque informações mais aprofundadas. Comentário dos autores: foram adicionadas referências conforme a sugestão.
Sugestão	Incluir vídeos com um professor explicando alguns conceitos, além do texto. Comentário dos autores: pretendemos fazer isso no futuro com o auxílio de outros professores.
Sugestão	Abordar outros tipos de rede de Petri. Comentário dos autores: o assunto sobre as redes de Petri é bastante extenso, sendo impraticável em apenas uma ferramenta cobrir outros tipos de redes de Petri.

Fonte: o autor.

Quadro 8

Sugestões/opiniões dos professores quanto à clareza das mensagens de erro e feedbacks	
Opinião	No exercício 4, quando a resposta é submetida de modo incorreto, a mensagem de erro que é mostrada é genérica. Comentário dos autores: o exercício 4 é do tipo quebra cabeça, em que os elementos são mostrados e devem ser organizados pelo estudante. Como os exercícios de quebra cabeça são mais simples que os de modelagem, as mensagens de erro são menos precisas. Se as mensagens nos exercícios fossem mais precisas, a ferramenta acabaria mostrando a resposta da questão.
Opinião	Alguns professores notaram que em alguns dos exercícios mais complexos de modelagem, a precisão das mensagens de erros diminui. Comentário dos autores: nos exercícios de modelagem, as mensagens de erros são mais precisas que nos exercícios de quebra cabeça. No entanto, na parte de modelagem avançada, quando é abordado o assunto de métricas e função de guarda, a precisão tende a diminuir. Isso ocorre, pois a ferramenta trabalha não apenas com o modelo gráfico, mas também, com reconhecimento de *strings* e caracteres escritos nas métricas, funções de guarda e *labels* dos elementos da rede. Para realizar tal reconhecimento, a ferramenta usa estruturas complexas de "*if-else*" e "*switch-case*". Tais estruturas, por mais complexas que sejam, apresentam limitações e, nesse caso, técnicas mais avançadas de inteligência artificial deveriam ser utilizadas para que a precisão aumentasse. Isso é uma limitação da ferramenta.

Fonte: o autor.

6.2 AVALIAÇÃO COM ESTUDANTES

A seguir serão apresentados os resultados da avaliação de aprendizagem com os estudantes, bem como serão mostrados os resultados da avaliação de usabilidade com os mesmos.

6.2.1 Avaliação de Aprendizagem

As Tabelas 1, 2, 3 e 4 mostram os parâmetros estatísticos descritivos e o intervalo (mínimo e máximo) das notas dos testes realizados antes e depois do uso da ferramenta, respectivamente, de todos os 15 estudantes; os 11 estudantes da graduação; os 7 estudantes da graduação que não tiveram contato com o assunto de redes de Petri antes de serem convidados; e os 4 estudantes da pós-graduação. Como mostrado nas tabelas, os estudantes obtiveram progresso nas notas. Na comparação realizada com todos os 15 estudantes (ver Tabela 1), a média das notas teve um aumento de quase 160% enquanto a mediana teve um aumento de 700%. Comparando os grupos de estudantes da pós-graduação e da graduação (ver Tabelas 2 e 4), é possível observar que ambos obtiveram progresso em suas notas. No entanto, os alunos de graduação obtiveram um progresso superior, tendo a média aumentada em mais de 270% e sua mediana aumentada em 800%. Já os estudantes da pós-graduação, aumentaram a sua média em mais de 40% e a sua mediana em mais de 54%. Analisando apenas o grupo dos estudantes que não tiveram contato com o assunto de redes de Petri antes de serem convidados (ver Tabela 3), observamos que o progresso deles foi muito superior ao progresso dos outros grupos. Nesse caso, a média aumento em 4759% evoluindo de 1,42 pontos para 69 pontos.

A fim de verificar se de fato existe uma diferença estatística significativa entre as notas dos alunos antes de depois do uso da ferramenta, foi realizado um teste de hipótese, com o uso da linguagem R (IHAKA e GENTLEMAN, 1996). Primeiramente, realizou-se um teste de normalidade dos dois grupos de notas dos pré e pós testes dos 15 estudantes. A Tabela 5 mostra os resultados do teste de normalidade de *Shapiro-Wilk* com significância de 0,05. Segundo esse teste, se o resultado do cálculo de *p-value* for maior que 0,05, considera-se que o grupo de dados segue uma distribuição normal. Conforme mostrado na Tabela 5, os dois grupos de notas não seguem uma

distribuição normal, pois a condição *p-value* > 0,05 não é satisfeita. Dessa forma, como a premissa de normalidade não é atendida, devemos realizar um teste não paramétrico a fim de verificar se o uso da ferramenta contribui no melhoramento das notas. Como os dois grupos de notas são dependentes, ou seja, foram extraídos de uma pesquisa com a mesma amostra de estudantes, consideramos realizar o teste não paramétrico de *Wilcoxon* para dados emparelhados com significância de 0,05 (Triola, 2017).

O teste de *Wilcoxon* compara dois grupos de dados emparelhados e considera duas hipóteses: hipótese nula – as medianas dos dois grupos são iguais; hipótese alternativa – as medianas dos dois grupos são diferentes. Se *p-value* > 0,05, então aceitamos a hipótese nula. Caso contrário, ou seja, *p-value* <= 0,05, aceitamos a hipótese alternativa. Do ponto de vista prático, se *p-value* <= 0,05, afirmamos que as medianas das notas dos alunos são diferentes. Ou seja, a ferramenta contribuiu para o aumento das notas. Desse modo, calculamos o *p-value* e verificamos que ele é igual a 0,001. Assim, a partir do teste de *Wilcoxon*, concluímos que a ferramenta contribuiu para o aprendizado dos estudantes.

Tabela 1: Comparação das notas dos 15 estudantes (graduação e pós-graduação) antes e depois de usarem a ferramenta.

	Estatísticas das notas dos 15 estudantes (graduação e pós-graduação) antes e depois de usarem a ferramenta		
	Média (Pontos)	**Mediana (Pontos)**	**Mínimo – Máximo (Pontos)**
Antes	28	10	0 - 90
Depois	72	80	30- 100

Fonte: o autor.

Tabela 2: Comparação das notas dos 11 estudantes de graduação antes e depois de usarem a ferramenta.

	Estatísticas das notas dos 11 estudantes de graduação antes e depois de usarem a ferramenta		
	Média (Pontos)	**Mediana (Pontos)**	**Mínimo – Máximo (Pontos)**

Antes	19	0	0 - 70
Depois	71	80	50 - 90

Fonte: o autor.

Tabela 3: Comparação das notas dos 7 estudantes da graduação que não tiveram contato com o assunto de redes de Petri antes de serem convidados.

	Estatísticas das notas dos 11 estudantes de graduação antes e depois de usarem a ferramenta		
	Média (Pontos)	**Mediana (Pontos)**	**Mínimo – Máximo (Pontos)**
Antes	1,42	0	0 - 10
Depois	69	80	50 - 90

Fonte: o autor.

Tabela 4: Comparação das notas dos 4 estudantes de pós-graduação antes e depois de usarem a ferramenta.

	Estatísticas das notas dos 4 estudantes de pós-graduação antes e depois de usarem a ferramenta		
	Média (Pontos)	**Mediana (Pontos)**	**Mínimo – Máximo (Pontos)**
Antes	52,5	55	10 - 90
Depois	75	85	30 - 100

Fonte: o autor.

Tabela 5: Teste de normalidade dos dois grupos de dados.

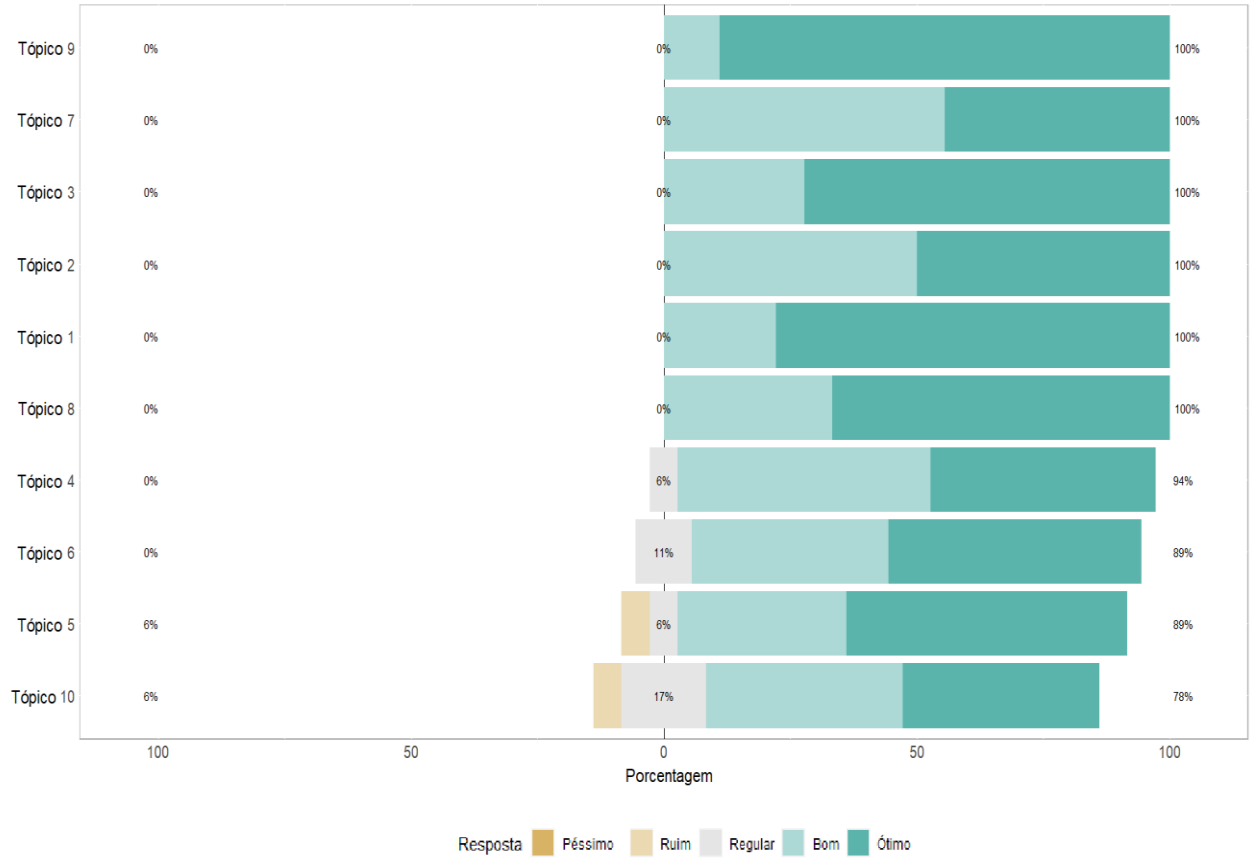
	Teste de *Shapiro-Wilk* com significância de 0,05	
	p-value	**As notas seguem uma distribuição normal, ou seja, *p-value* > 0,05 ?**
Antes	0,001	não
Depois	0,0232	não

Fonte: o autor.

6.2.2 Avaliação de Usabilidade

A seguir, serão mostrados os resultados da avaliação de usabilidade com os estudantes (ver Figura 36). As avaliações dos estudantes foram muito boas e em nenhum dos tópicos houve a avaliação como “péssimo”. Os tópicos 1, 2, 3, 7, 8 e 9 foram os mais bem avaliados. A boa avaliação dos tópicos 1, 2 e 3 indica uma boa usabilidade da ferramenta e a satisfação com a interface. Vale destacar também o tópico 9, que trata da eficácia para o aprendizado, que foi muito bem avaliado tanto pelos professores quanto pelos estudantes. Nos comentários, os estudantes elogiaram a ferramenta e afirmaram que conseguiram obter um maior domínio no assunto de modelagem usando as redes de Petri. Outro tópico também interessante de destacar é o tópico 7, que trata da clareza das mensagens de erro ou alerta. Apesar desse tópico ter sido bem avaliado pelos estudantes, ele foi um dos tópicos menos bem avaliados pelos professores.

Figura 36: resultados das avaliações feitas pelos estudantes.



Fonte: o autor.

Por outro lado, os tópicos 4, 5, 6 e 10 foram os menos bem avaliados pelos estudantes. Os tópicos 5 e 10 merecem destaque pois ambos foram os menos bem avaliados tanto pelos professores quanto pelos estudantes, reforçando que a visualização do progresso e o *feedback* dos erros podem melhorar. As sugestões e opiniões dos estudantes para os tópicos 5 e 10 foram semelhantes às dos professores (mostradas nos Quadros 5 e 8). Os Quadros 9 e 10 mostram outras sugestões escritas pelos estudantes sobre a ferramenta agrupadas por tema, com os comentários dos autores deste trabalho para cada uma dessas sugestões.

Quadro 9

Sugestão dos estudantes quanto à liberdade (ou flexibilidade) na navegação	
Sugestão	Foi sugerido que o usuário possa navegar por todos os tópicos mesmo sem ter concluído o anterior de modo a dar mais liberdade na navegação. Comentário dos autores: os tópicos seguem uma sequência didática e lógica. Além disso, para se entender um tópico posterior, é necessário entender ou ter estudado o(s) tópico(s) anterior(es). Assim, por uma questão de aprendizagem, é preferível que a ferramenta não fique totalmente aberta.

Fonte: o autor.

Quadro 10

Sugestões dos estudantes quanto à facilidade de uso da ferramenta	
Sugestão	Foi sugerido que ferramenta tenha um *layout* responsivo para o acesso a partir de dispositivos móveis como celular ou *tablets.* Comentário dos autores: a ferramenta foi desenvolvida usando *Bootstrap framework* e, portanto, possui elementos de responsividade. Embora seja possível construir modelos usando celular ou *tablet*, a interação com a interface é mais difícil. Nesse caso, em um trabalho futuro, pretendemos estender a ferramenta TryRdP com tecnologias específicas para esses dispositivos.
Sugestão	Considerar a questão de acessibilidade para usuários com certas deficiências. Comentário dos autores: para a ferramenta TryRdP, a questão de acessibilidade deve levar em conta não apenas os textos escritos, mas também, a construção e o controle dos modelos das redes de Petri. Isso torna o trabalho mais complexo e, nesse caso, um estudo futuro mais específico deve ser realizado.

Fonte: o autor.

As avaliações indicam que a ferramenta é um bom instrumento de auxílio ao aprendizado de modelagem de sistemas. A ferramenta TryRdP foca na modelagem gráfica e em conteúdos que estimulem a capacidade de abstração dos estudantes. Ao término das atividades e reflexões propostas na ferramenta, é esperado que um estudante tenha uma melhora na sua habilidade de abstração. Ao se observar as avaliações, nota-se que os estudantes melhoraram sua capacidade de abstração e compreenderam a sua importância.

7 CONCLUSÕES

Artefatos digitais podem funcionar como potencializadores no aprendizado, em especial, para o ensino de conteúdos de natureza abstrata, como no caso deste trabalho, a modelagem usando as redes de Petri (MORAES e LIMA, 2019). Não foram encontradas ferramentas que auxiliem no processo de ensino e aprendizagem de modelagem por meio das redes de Petri, e a ferramenta TryRdP contribui para suprir essa lacuna. A ferramenta tem um viés prático de modo a orientar o estudante a construir as redes de Petri que podem representar um determinado sistema. Com isso, os estudantes podem entender previamente “por quê”, “para que” e “como” usar as redes de Petri, antes de entrarem em contato com os detalhes matemáticos mais complexos do formalismo.

A ferramenta TryRdP foi desenvolvida para que o estudante a use individualmente e de maneira autônoma. Embora seja voltada, a princípio, para estudantes de graduação ou pós-graduação em computação, pode ser de interesse também a estudantes de qualquer área das ciências exatas que estejam motivados a aprender o assunto.

O ambiente da ferramenta permite que o estudante explore o assunto de modelagem de modo que ao abrir um determinado tópico (como modelagem de disponibilidade de um sistema computacional), o estudante terá acesso a uma página com explicações e exemplos sobre esse tópico. Em seguida, com base nos conceitos apresentados, o aluno é encorajado a reutilizar os modelos de RdP para a resolução de exercícios. O *feedback* imediato fornecido pela ferramenta durante a resolução dos exercícios é essencial para que o estudante seja guiado e apoiado ao longo do aprendizado. Note que a ferramenta mostra os *feedbacks* em tempo real enquanto o aluno está fazendo os exercícios, de modo que caso ele/ela cometa algum erro relacionado à sintaxe do formalismo ou algum erro de digitação que comprometa a sintaxe das métricas uma mensagem será apresentada. Além disso, caso o usuário tenha cometido algum erro, após a realização de um exercício, ele/ela receberá uma mensagem com um *feedback* sugerindo o possível erro para que o usuário possa corrigir.

Além do uso autônomo que a ferramenta oferece, graças à sua estrutura que contempla uma sequência didática com conteúdo teóricos e exercícios práticos, a TryRdP também pode ser usada como uma ferramenta de apoio no ensino formal em disciplinas de graduação e pós-graduação como complemento às aulas para reforçar o conhecimento adquirido. Adicionalmente, a ferramenta pode ser usada em abordagens de sala de aula invertida (PAVANELO e LIMA, 2017). Vale ser ressaltado que o professor pode acompanhar a resolução dos exercícios da TryRdP e esclarecer possíveis dúvidas que o *feedback* da ferramenta não tenha conseguido suprir.

Os *feedbacks* obtidos através das avaliações mostraram que a percepção do público alvo (professores e estudantes) é bastante positiva em relação à ferramenta. De modo geral, os professores afirmaram que acreditam que a ferramenta pode contribuir para o aprendizado dos estudantes. Já os estudantes relataram que conseguiram compreender o assunto de modelagem usando a ferramenta. Em relação à aplicação didático-pedagógica, as avaliações mostraram que a ferramenta é eficaz no seu objetivo de auxiliar os estudantes no aprendizado de modelagem usando as RdP. Desse modo, podemos concluir que, a partir da percepção do público alvo, a ferramenta cumpre o seu objetivo de ser eficiente na sua usabilidade e na aplicação didático-pedagógica.

7.1 LIMITAÇÕES

Os resultados das avaliações com professores e alunos mostraram que a ferramenta tem grande potencial enquanto um instrumento de apoio ao ensino formal sobre modelagem de sistemas usando as redes de Petri, assim como, a ferramenta é, também, um bom instrumento para o estudo individual desse assunto. Apesar das avaliações positivas, o tamanho da amostra obtida foi pequeno contando apenas com 8 professores e 15 estudantes. Isso pode ser considerado uma limitação do experimento, visto que se fosse utilizada uma amostra maior, o resultado das avaliações teria maior confiabilidade.

Também, podemos citar as limitações técnicas relacionadas à usabilidade. A ferramenta não permite a acessibilidade para pessoas com certos tipos de deficiência como a deficiência visual. Nesse caso, uma versão da ferramenta com tecnologias

mais específicas deve ser criada para atender essa demanda. Além disso, embora a ferramenta tenha sido desenvolvida com certos elementos de responsividade, o seu uso tem uma melhor experiência via computadores do tipo *desktop* e *laptops*. O uso da ferramenta via *tablets* ou *smartphones* não trazem uma boa experiência para o usuário, visto que é trabalhoso conectar os elementos gráficos da RdP usando esses dispositivos.

Por fim, embora nos exercícios de modelagem, as mensagens de erros sejam mais precisas que nos exercícios de quebra cabeça, na parte de modelagem avançada, quando é abordado o assunto de métricas e função de guarda, a precisão tende a diminuir. Isso ocorre, pois a ferramenta trabalha não apenas com o modelo gráfico, mas também, com reconhecimento de *strings* e caracteres escritos nas métricas, funções de guarda e *labels* dos elementos da rede. Para realizar tal reconhecimento, a ferramenta usa estruturas complexas de "*if-else*" e "*switch-case*". Tais estruturas, por mais complexas que sejam, apresentam limitações e, nesse caso, técnicas mais avançadas de inteligência artificial deveriam ser utilizadas para que a precisão aumentasse.

7.2 TRABALHOS FUTUROS

Possíveis trabalhos futuros, coletados a partir das sugestões dos participantes da pesquisa, incluem: tradução do texto da ferramenta para o inglês e o espanhol; desenvolvimento de uma versão da ferramenta para dispositivos móveis; investigar tecnologias para implementar o *tokengame* e melhorar a precisão das mensagens de erros; adicionar vídeos explicativos com especialistas; e por fim, investigar tecnologias para aumentar a acessibilidade para pessoas com deficiência. Além disso, uma nova avaliação da ferramenta com uma amostra maior também poderia ser feita.

REFERÊNCIAS

ANDRADE, E. et al. Availability modeling and analysis of a disaster-recoveryas-a-service solution. **J Grid Computing 17**, pages 603–621, 2017.

ANDRADE, E. et al. Calau: An environment for modeling and analyzing embedded real-time systems. In: **Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference**, pages 3135–3140, 2012.

ARAUJO, L. G. J.; BITTENCOURT, R. A.; CHAVEZ, C. V. F. G. Python Enhanced Error Feedback: Uma IDE Online de Apoio ao Processo de Ensino-Aprendizagem em Programação. **SIMPÓSIO BRASILEIRO DE EDUCAÇÃO EM COMPUTAÇÃO (EDUCOMP)**, 1. On-line. Porto Alegre: Sociedade Brasileira de Computação, p. 326-333. 2021.

ARTEIRO, R. et al. Utilizando redes de Petri para modelagem de desempenho de middleware orientado a mensagem. In: **XXVII Congresso da SBC***,* 595-614, 2007.

BALBO, G. Introduction to Stochastic Petri Nets. **Lectures on Formal Methods and Performance Analysis**, Vol. 2090 of LNCS, pp. 88-155. 2001.

BENNETTS, R. On the analysis of fault trees. **IEEE Transactions on reliability**, 24(3):175–185, 1975.

BITTENCOURT, Ig Ibert; ISOTANI, Seiji. Informática na educação baseada em evidências: Um manifesto. **Revista Brasileira de Informática na Educação**, v. 26, n. 03, p. 108, 2018.

BUCCI, G. et al. Oris: a tool for modeling, verification and evaluation of real-time systems. **Int J Softw Tools Technol Transfer 12,** 391–403, 2010.

CAVALCANTI, T.C. et al. Smartphonebased spectral imaging otoscope: System development and preliminary study for evaluation of its potential as a mobile diagnostic tool. **J. Biophotonics**.13:e201960213. 2020.

CHIOLA, G. et al. Generalized Stochastic Petri Nets: A Definition at the Net Level and Its Implications. **IEEE Trans. Software Eng.** 1993.

COSTA, D. et al. Lpt: ferramenta educacional para auxiliar o ensino/aprendizagem de traduções de diferentes níveis de linguagens de programação. In: **Anais do Workshop de Informática na Escola**, volume 23, page 695, 2017.

DIETZ, J. et al. Using the Curriculum Vita to Study the Career Paths of Scientists and Engineers: An Exploratory Assessment. **Scientometrics**, *49*(3), 419-442, 2000.

DISTEFANO, S. et al. Design and implementation of a performance plug-in for the argouml tool. In: **IASTED Conf. on Software Engineering**, pages 337–342, 2005.

DANTAS, E. et al. PuzzlEdu: uma Ferramenta para Apoiar o Ensino Aprendizagem de Orientação a Objetos. 17º Workshop de Informática na Escola - **22º Simpósio Brasileiro de Informática na Educação**. Aracajú-SE. 2011.

FALANI, L.; AGUIAR, C.; FORNO, A. Mapeamento da literatura sobre as tecnologias da indústria 4.0 no segmento têxtil brasileiro. **Brazilian Journal of Development.** 6. 42437-42452. 2020.

FARAHANI, B. et al. Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare. **Future Generation Computer Systems**. 2017.

GÓMEZ-MARTÍNEZ, E.; MERSEGUER, J. ArgoSPE: Model-based software performance engineering. **Petri Nets and Other Models of Concurrency-ICATPN 2006**, pages 401–410, 2006.

GIRAULT, C.; VALK, R. **Petri Nets for System Engineering: A Guide to Modeling, Verification, and Applications.** Springer-Verlag New York, Inc. Secaucus, NJ, USA. 2001.

HIREL, C. et al. Reliability and performability modeling using sharpe. In: **International Conference on Modelling Techniques and Tools for Computer Performance Evaluation**, pages 345–349. Springer. 2000.

HIREL, C.; TUFFIN, B.; TRIVEDI, K. SPNP: Stochastic petri nets. Version 6.0. **Lecture Notes in Computer Science**. 1786. 354-357, 2000.

IHAKA, R.; GENTLEMAN, R. R: A Language for Data Analysis and Graphics. **Journal of Computational and Graphical Statistics**, 5:3, 299-314, 1996.

JUNIOR, Ozonias; AGUIAR, Yuska Paola Costa. Taxonomia de Critérios para Avaliação de Software Educativo - TaCASE. **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE)**, [S.l.], p. 298, out. 2018. ISSN 2316-6533.

JANOUSEK, C.V. **Modelling Objects by Petri Nets.** PhD thesis, PhD thesis, Department of Computer Science and Engineering, Technical University of Brno, Czech Republic, 1998.(In Czech). CiteSeer. IST-Copyright Penn State and NEC.

JENSEN, K. Coloured Petri Nets: A High Level Language for System Design and Analysis. **Advances in Petri Nets 1990**, 1991.

JOSHI, A. et al. Likert Scale: Explored and Explained**. Current Journal of Applied Science and Technology**, *7*(4), 396-403, 2015.

KORMANN, H.; SUBERG, B. Innovation Strategy for Renewal and Growth. In: **Kormann H., Suberg B. (eds) Topics of Family Business Governance. Management for Professionals**. Springer, Cham. 2021.

KPMG. **COVID-19 forces one of the biggest surges in tech investment in history, finds world's largest tech leadership survey**, **2020.** Disponível em: <https://home.kpmg/xx/en/home/media/press-releases/2020/09/covid-19-forces-one-of-the-biggest-surges-in-technology-investment-in-history-finds-worlds-largest-technology-leadership-survey.html#:~:text=The%20largest%20technology%20leadership%20survey,10%20IT%20leaders%20report%20that > Acesso em: 25 de fev. de 2021.

LIMA, J. W. S.; CALLOU, G. R. A. ; ANDRADE, E.C. Teoria de Filas e Rede de Petri Estocástica: Um tutorial. **Research, Society and Development**, v. 10, p. e2810312826. 2021.

MACIEL, P.; LINS, R.; CUNHA, P. **Uma Introdução às Redes de Petri e Aplicações**. Campinas, SP: Sociedade Brasileira de Computacão, v. 1. 213 p, 1996.

MARCOLINO, M. et al. The Impact of mHealth Interventions: Systematic Review of Systematic Reviews. **JMIR mHealth and uHealth**. 6. e23. 2018.

MARSAN, M.; CONTE, G.; BALBO, G. A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems. **ACM Trans. Comput. Syst**. 1984.

MARSAN, M.A. Stochastic Petri Nets: An Elementary Introduction. **Advances in Petri Nets**, 424:1–29, 1989.

MEDEIROS, T. R. et al. IO Simulator: Um Simulador de Dispositivos de Entrada e Saída para Auxiliar o Ensino de Sistemas Operacionais. Workshop de Educação e Informática. **XXXI Congresso da Sociedade Brasileira de Computação.** 2011.

MERLIN, P.; FABER, D. J. Recoverability of communication protocols: Implicatons of a theoretical study. **IEEE Transactions on Communications**, 24(9):1036–1043, Sept. 1976.

MEI, H. et al. Managing patients with cancer during the COVID-19 pandemic: frontline experience from Wuhan. **The Lancet. Oncology**, *21*(5), 634–636, 2020.

MIASHIRO, G. et al. O USO DE SOFTWARES EDUCATIVOS NO ENSINO DE MATEMÁTICA NOS ANOS INICIAIS DO ENSINO FUNDAMENTAL. **Anais do CIET: EnPED: 2020-(Congresso Internacional de Educação e Tecnologias| Encontro de Pesquisadores em Educação a Distância).** 2020.

MO, P. et al. Clinical Characteristics of Refractory COVID-19 Pneumonia in Wuhan, China**. Clinical Infectious Diseases**, ciaa270, 2020.

MORAES, DA; LIMA, CM. Os artefatos digitais como ferramentas mediadoras das atividades cognitivas dos estudantes: possibilidades para novos cenários de aprendizagem. **Educar em Revista.** 2019.

MURATA, T. Petri nets: Properties, analysis and applications. In: **Proceedings of the IEEE***, vol. 77*, no. 4, pp. 541-580, 1989.

MURATA T. Petri Nets and their Application an Introduction. In: **Chang SK. (eds) Management and Office Information Systems***.* Springer, Boston, MA. 1984.

NIELSEN, J. Enhancing the explanatory power of usability heuristics. In: **Proceedings of the SIGCHI conference on Human Factors in Computing Systems**, 152-158, 1994.

OECD/EUROSTAT. **Oslo Manual**: Guidelines for Collecting and Interpreting Innovation Data, Paris, 2005.

OLIVEIRA, C. C.; JOSÉ W.C., MOREIRA M. **Ambientes Informatizados de Aprendizagem: Produção e Avaliação de software Educativo**. Campinas, SP: Papirus, 2001.

PAVANELO, E; LIMA, R. Sala de Aula Invertida: a análise de uma experiência na disciplina de Cálculo I. **Bolema: Boletim de Educação Matemática.** 2017.

PETRI C. A. **Kommunikation mit Automaten**. PhD Dissertation, Darmstad University, Germany, 1962.

RAABE, A. L. A.; BOMBASAR, J. R. Mensuração e testes em Informática na Educação. In: **JAQUES, Patrícia Augustin; SIQUEIRA; Sean; BITTENCOURT, Ig; PIMENTEL, Mariano. (Org.) Metodologia de Pesquisa Científica em Informática na Educação: Abordagem Quantitativa. Porto Alegre: SBC, 2020.** (Série Metodologia de Pesquisa em Informática na Educação, v. 2) Disponível em: <https://metodologia.ceie-br.org/livro-2>.

RHO, S.; VASILAKOS, A.; CHEN, W. Cyber physical systems technologies and applications. **Future Generation Computer Systems**. 2016.

ROSSITER, D. Past, present & future of information technology in pedometrics. **Geoderma**. 2018.

RUGINA, A. E.; KANOUN, K.; KAÂNICHE, M. The adapt tool: From aadl architectural models to stochastic petri nets through model transformation. In: **Dependable Computing Conference, 2008. EDCC 2008**. Seventh European, pages 85–90. IEEE. 2008.

SCHNEIDER, Bruno de Oliveira. **O que é LOGO.** Disponível em: <http://algol.dcc.ufla.br/~bruno/wxlogo/docs/oquee.html>. Acesso em: 9 de dez. de 2021.

SILVA, B. et al. Mercury: An integrated environment for performance and dependability evaluation of general systems. In: **Proceedings of Industrial Track at 45th Dependable Systems and Networks Conference (DSN)**, 2015.

SILVA, E. O.; VALENTE, J. W. M.; CARMONA, J. V. C. WebJuvia: Simulador Web de Apoio ao Ensino de Gerência de Memória na Disciplina de Sistemas Operacionais. **SIMPÓSIO BRASILEIRO DE EDUCAÇÃO EM COMPUTAÇÃO (EDUCOMP)**, 1. On-line. Porto Alegre: Sociedade Brasileira de Computação, p. 343-351. 2021.

SPOLON, R. **Um método para avaliação de desempenho de protocolos de sincronização otimistas para simulação distribuída**. 2001. Tese (Doutorado em Física Aplicada) - Instituto de Física de São Carlos, University of São Paulo, São Carlos, 2001. doi:10.11606/T.76.2001.tde-19042002-162157.

TECH MONITOR. **CIOs Wielding More Influence in the Boardroom as IT Spending Surges by $15 Billion a Week, 2020.** Disponível em: <https://techmonitor.ai/techonology/data/cio-survey-2020-kpmg >. Acesso em: 25 de fev. de 2021.

TRIOLA, M. F. **Introdução à Estatística**. 12a Edição. Editora LTC, 2017.

TRIVEDI, K.; ANDRADE, E.; MACHIDA, F. Combining performance and availability analysis in practice. **Advances in Computers**. Vol. 84. Elsevier, 2012.

TRIVEDI, K. S. **Probability & Statistics with Reliability, Queuing and Computer Science Applications**. John Wiley & Sons. 2008.

U.S. BUREAU OF LABOR STATISTICS. **Economic News Releases: Employment Projections. 2013.** Disponível em: <https://www.bls.gov/opub/btn/volume-2/careers-in-growing-field-of-information-technology-services.htm#ednref4 >. Acesso em: 25 de fev. de 2021.

U.S. BUREAU OF LABOR STATISTICS. **Economic News Releases: Employment Projections. 2020.** Disponível em: <https://www.bls.gov/bls/newsrels.htm#OEP>. Acesso em: 25 de fev. de 2021.

ULLMANN, M. et al. Neandersim: Simulador gráfico de apoio ao ensino de arquitetura de computadores. **XXII Workshop sobre Educação em Computação**. SBC, 2014.

VALENTE, J. A. **Computadores e conhecimento: repensando a educação.** 2ª Edição. Campinas: UNICAMP. 1998.

VALENTE, J. A. Informática na educação: instrucionismo x construcionismo. ISSN: 1984-6290, **Revista Educação Pública**. 2000.

VIEIRA, Fábia Magali Santos. Classificação de softwares educacionais. **Mídias na Educação**, v. 12, n. 1, p. 1-3, 2015.

ZHOU, K.; LIU, T.; ZHOU, L. Industry 4.0: Towards future industrial opportunities and challenges. **12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)**, pp. 2147-2152, 2015.

ZIMMERMANN, A. Modelling and Performance Evaluation with TimeNET 4.4. In: Bertrand N., Bortolussi L. (eds) Quantitative Evaluation of Systems. QEST 2017. **Lecture Notes in Computer Science***, vol 10503*. Springer, Cham. 2017.