TryRdP: uma Ferramenta para o Aprendizado de Modelagem de Sistemas usando Redes de Petri

John W. Soares de Lima
Departamento de Computação
UFRPE / Universidade Federal Rural de Pernambuco
Recife Pernambuco Brasil
john.lima@ufrpe.br

Taciana Pontual Falcão
Departamento de Computação
UFRPE / Universidade Federal Rural de Pernambuco
Recife Pernambuco Brasil
taciana.pontual@ufrpe.br

Ermeson Andrade
Departamento de Computação
UFRPE / Universidade Federal Rural de Pernambuco
Recife Pernambuco Brasil
ermeson.andrade@ufrpe.br

RESUMO

A exigência por sistemas computacionais de qualidade tem demandado estudo e desenvolvimento de técnicas e ferramentas que garantam alta disponibilidade e baixo custo, entre outras características. Nesse sentido, o interesse pela utilização de modelos computacionais tem aumentado nos últimos anos. Entre os diversos métodos de modelagem formal existentes, destacam-se as Redes de Petri (RdP). Rede de Petri é um formalismo matemático usado para modelar e analisar sistemas que tenham atividades paralelas, concorrentes, assíncronas e não-determinísticas. Tal formalismo é comumente abordado em cursos de graduação e pós-graduação em computação. No entanto, o modelo tradicional de ensino, baseado em exposições teóricas muitas vezes descontextualizadas das aplicações dos conceitos, possui um alto nível de abstração que pode tornar o processo de aprendizado difícil. Nesse sentido, para auxiliar na compreensão e aprendizagem significativa das Redes de Petri, este trabalho apresenta TryRdP, uma ferramenta web gratuita, interativa e baseada em exemplos práticos de aplicações reais. A interface da TryRdP apresenta três seções: introdução, modelagem básica e modelagem avançada, cada uma contendo explicações dos tópicos abordados, bem como exemplos práticos usando o formalismo das RdP. O nível dos exercícios vai crescendo em complexidade à medida que são apresentados novos elementos das RdP. Embora projetada para ser usada de forma individual e autônoma, a TryRdP adequa-se também como ferramenta de suporte ao ensino formal de graduação e pós-graduação em computação e áreas relacionadas, podendo ser usada pelo professor para atividades de reforço extraclasse ou no contexto da sala de aula invertida.

PALAVRAS-CHAVE

Recurso didático, Modelagem de sistemas, Rede de Petri.

EduComp'21, Abril 27–30, 2021, Jataí, Goiás, Brasil (On-line).

1 Introdução

Em um mundo cada vez mais tecnológico, a busca crescente por sistemas computacionais de qualidade, isto é, aqueles que tenham baixo custo, bom desempenho e sejam capazes contornar falhas, motiva o desenvolvimento de métodos para a avaliação de tais sistemas. Nesse sentido, o interesse pela utilização de métodos para modelagem de sistemas computacionais tem aumentado nos últimos anos. 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 [15]. 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) [2].

Conforme definido em [11], "um modelo é uma representação da realidade, no qual a representação é feita por meio da expressão de certas características relevantes da realidade observada e no qual a realidade consiste de objetos ou sistemas que existem, existiram ou podem existir". Essa representação é obtida usando-se métodos de abstração por meio do processo de modelagem. Redes de Petri [12], Diagrama de Confiabilidade [9], e Árvore de Falhas [4] 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 Redes de Petri (RdP) destacam-se como uma ferramenta gráfica e matemática eficiente na modelagem e análise de sistemas complexos [12]. 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) [5, 10]. As GSPN podem ser adotadas para análise de desempenho e dependabilidade de sistemas dinâmicos. Esse método formal também permite que as métricas sejam aferidas tanto via simulação, quanto pela análise do espaço de estados.

Adicionalmente, as GSPN proveem diversos recursos para modelagem, tais como funções de guarda, pesos, prioridade, entre outros.

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 sobre redes de Petri, baseado majoritariamente em exposições teóricas, nem sempre é adequado aos alunos 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 educacionais que estão se tornando cada vez mais interativas e customizadas, objetivando apoiar o ensino de conteúdos de natureza abstrata [6]. 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 [14], não existem 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.

Este artigo apresenta TryRdP - uma ferramenta baseada na Web para auxiliar no aprendizado de modelagem de sistemas por meio de redes de Petri. Este artigo está organizado da seguinte forma. A Seção 2 apresenta conceitos teóricos fundamentais sobre redes de Petri. A seção 3 abordada ferramentas existentes relacionadas ao tema. A Seção 4 apresenta como a ferramenta foi desenvolvida. A Seção 5 detalha o funcionamento da ferramenta. A seção 6 mostra o contexto de uso da ferramenta. Por fim, a Seção 7 conclui este trabalho e descreve trabalhos futuros.

2 Fundamentos sobre Redes de Petri

Segundo Arteiro *et al.* [1],

> as redes de Petri são uma família de ferramentas gráficas utilizadas para descrição formal de sistemas que possuem características como concorrência, conflito, sincronização e exclusão mútua. Uma rede de Petri é representada por meio de um multigrafo bipartite e direcionado que possui dois tipos de nós: lugares e transições (representados por círculos e retângulos, respectivamente). Semanticamente, os lugares representam os recursos, estados locais ou variáveis do sistema, enquanto as transições representam as ações possíveis. Um arco em uma rede de Petri interliga um lugar à uma transição (arco de entrada) ou uma transição a um lugar (arco de saída).

Além dos elementos básicos de uma Rede de Petri (RdP) citados acima, existem os *tokens* (também chamados de marcação). Os *tokens* são representados graficamente por círculos pequenos preenchidos, que residem em lugares e são utilizados para especificar o estado da RdP. Além disso, eles trafegam pela rede por meio dos arcos, se distribuindo na rede. A distribuição do tokens nos lugares da rede determina o estado em que o sistema se encontra em determinado momento [12].

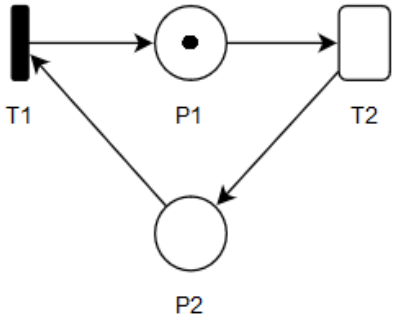

Figura 1: exemplo de uma GSPN.

A Figura 1 ilustra um exemplo de uma rede de Petri estocástica generalizada (GSPN) que é composta pelos lugares P1 e P2, por uma transição imediata T1, e por uma 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, já a transição imediata T1 não demanda tempo (isto é, ocorre de modo imediato), e ocorre de modo prioritário em relação à T2 [5, 10]. Adicionalmente, o tempo associado às transições temporizadas (e.g. T2) é distribuído exponencialmente, que é a principal característica de uma GSPN [12].

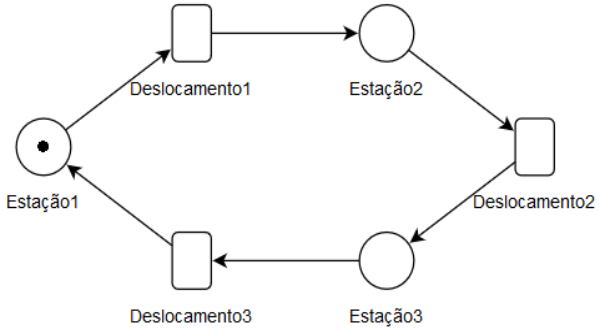

Figura 2: modelo GSPN de um sistema de transporte urbano via metrô.

Para exemplificar, o modelo GSPN da Figura 2 representa um sistema de transporte urbano via metrô. 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 GSPN, 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.

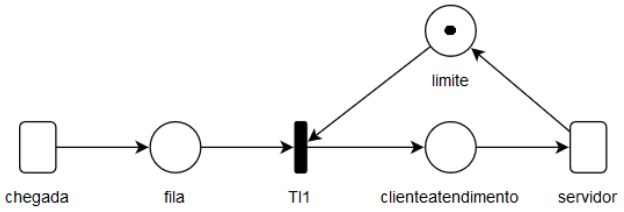

Figura 3: modelo GSPN de uma fila com um servidor.

Como outro exemplo, o modelo GSPN da Figura 3 representa uma fila com um servidor. Este modelo pode representar, por exemplo, uma fila de banco com um caixa para atender clientes, ou um sistema computacional com um servidor para atender às requisições. Neste modelo, as transições *chegada* e *servidor* representam, respectivamente, o tempo entre chegadas de clientes no sistema, e o tempo em que o cliente finaliza seu atendimento no servidor. Os lugares *fila* e *clienteatendimento* representam, respectivamente, a fila de espera de clientes, e o cliente sendo atendido pelo servidor. O lugar *limite* indica que o servidor apenas atende um cliente por vez. A transição imediata *TI1* representa a ação do cliente iniciar o atendimento, desde que o servidor esteja disponível.

Por fim, vale explanar sobre métricas nas RdP. Métricas são notações usadas para representar cálculos matemáticos ou condições dos estados nas RdP [14]. Os cálculos, representados pelas métricas, são realizados pelas ferramentas de modelagem e simulação. Por exemplo, em relação ao modelo da Figura 3, suponha que tenhamos a seguinte métrica: (E{#fila}). Essa métrica representa o cálculo da esperança matemática da quantidade de *tokens* no lugar *fila*. Em outras palavras, essa métrica representa a quantidade média de clientes à espera por atendimento na fila. Nesse mesmo modelo, também poderíamos calcular, por exemplo, a probabilidade de haver mais de um cliente na fila com a métrica: (P{#fila>1}).

3 Trabalhos relacionados

De nosso conhecimento, não existem ferramentas educacionais 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. Abaixo segue a descrição de algumas dessas ferramentas.

As ferramentas ArgoPerformance [7] e ArgoSPT [8] permitem o mapeamento automático de diagramas da UML (*Unified Modeling Language*) anotados em uma Rede de Petri Estocástica (SPN) para avaliação de desempenho. A ferramenta ADAPT [13] permite o mapeamento de AADL (*Architecture Analysis and Design Language*) em GSPNs. A ferramenta Calau [3] 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 temporizada 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 um modelo em rede de Petri por um usuário conhecedor da linguagem de alto nível utilizada por uma dessas ferramentas sem que, necessariamente, o usuário saiba modelar um sistema diretamente com rede de Petri. Por exemplo, se um usuário conhece a AADL, ele poderia usar a ferramenta ADAPT [13] para mapear uma rede de Petri. Tais ferramentas não têm nenhum objetivo didático para o aprendizado de rede de Petri. O objetivo dessas ferramentas é apenas facilitar o uso da rede de Petri.

Além dessas ferramentas citadas, existem outras usadas exclusivamente para a modelagem e a análise das redes de Petri, tais como Mercury [14], TimeNet [16], Oris [17] e SPNP [18]. Embora essas ferramentas possam ser usadas para os propósitos de modelagem e análise, as mesmas não dão suporte ao ensino das redes de Petri. Ou seja, elas requerem que os usuários já saibam e tenham experiência em modelagem de sistemas. Assim, 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 simula o modelo e a ferramenta não retorna nenhum valor.

A ferramenta educacional que estamos propondo é voltada para o ensino sobre Redes de Petri desde nível básico ao avançado. Ela mostra feedbacks em tempo real enquanto o aluno está fazendo os exercícios caso ele cometa algum erro relacionado à sintaxe do formalismo ou algum erro de digitação que comprometa a sintaxe das métricas. Além disso, caso o usuário tenha cometido erro, após a realização de cada exercício ele recebe uma mensagem com feedback indicando o possível erro para que o usuário possa corrigir.

4 Desenvolvimento da ferramenta TryRdP

A ferramenta TryRdP foi desenvolvida de acordo com o fluxograma da Figura 4, com o objetivo de auxiliar o processo de ensino e aprendizado de modelagem computacional usando Redes de Petri.

A primeira etapa consiste no levantamento do estado da arte, ou seja, revisão bibliográfica com a finalidade de organizar a literatura disponível acerca do tema do projeto, e ferramentas similares. Nessa etapa, foram feitos estudos com relação às redes de Petri, bem como a investigação de ferramentas existentes para modelagem de sistemas computacionais.

Em seguida, os requisitos e a arquitetura da ferramenta foram definidos. Em posse dos resultados obtidos na etapa anterior, foram realizados a implementação e o teste de software da ferramenta. Essas duas atividades foram executadas em paralelo, visto que a ferramenta foi desenvolvida de modo iterativo e incremental. A ferramenta TryRdP é baseada na web e foi desenvolvida usando *HTML*, *CSS*, *Bootstrap framework*, *javaScript* e três bibliotecas baseada no *javaScript* (*VEX*, *CursorJS* e *MxGraph).*

Por fim, foi feita uma avaliação da ferramenta por dois professores, sendo um especialista em cada uma das seguintes

áreas: interface de usuário e redes de Petri. Os professores foram apresentados à ferramenta e convidados a usá-la livremente. Em seguida, foram coletadas suas opiniões e sugestões, que geraram alguns ajustes na ferramenta, apresentados na próxima seção.

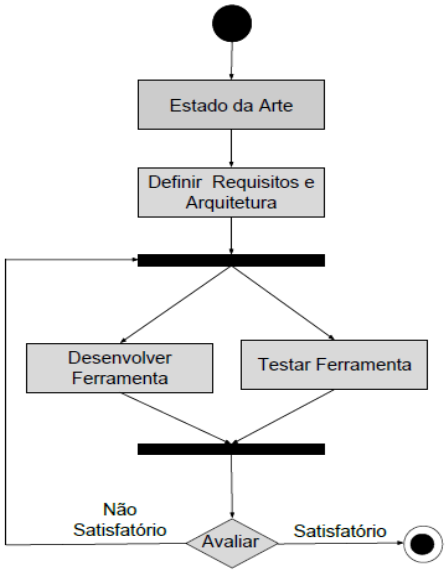

Figura 4: metodologia proposta.

5 A ferramenta TryRdP

A ferramenta foi desenvolvida de modo a ter conteúdos didáticos relacionados a modelagem usando 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 a situações familiares.

A TryRdP está dividida em três seções obrigatoriamente sequenciais: introdução, modelagem básica e modelagem avançada (Figura 5). Em cada seção, há apresentação de 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 outras seções da ferramenta, que abordam tópicos mais avançados. O aluno pode tentar fazer o exercício quantas vezes quiser, ou ver a resposta, caso não consiga responder à questão. Porém, caso o estudante opte por ver a resposta da questão, ele não receberá a pontuação correspondente.

Existem na ferramenta dois tipos de exercícios: quebra cabeça e modelagem. Nos exercícios de quebra cabeça, o aluno deve conectar os arcos nas 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, e de forma alinhada à concepção de que o erro faz parte do processo de aprendizagem.

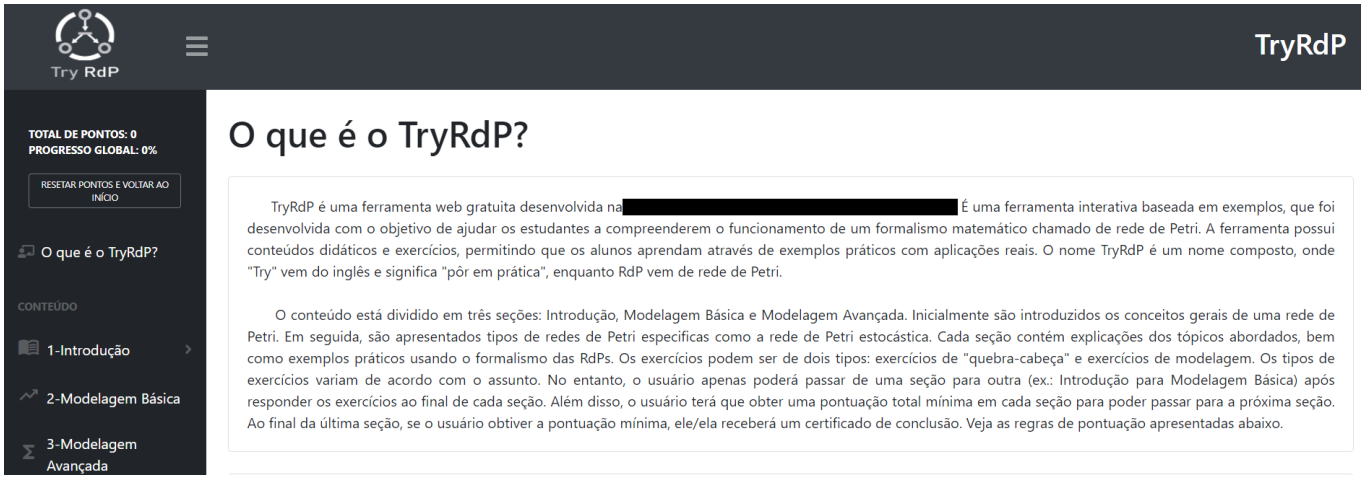

Figura 5: menu com as três seções à esquerda da imagem.

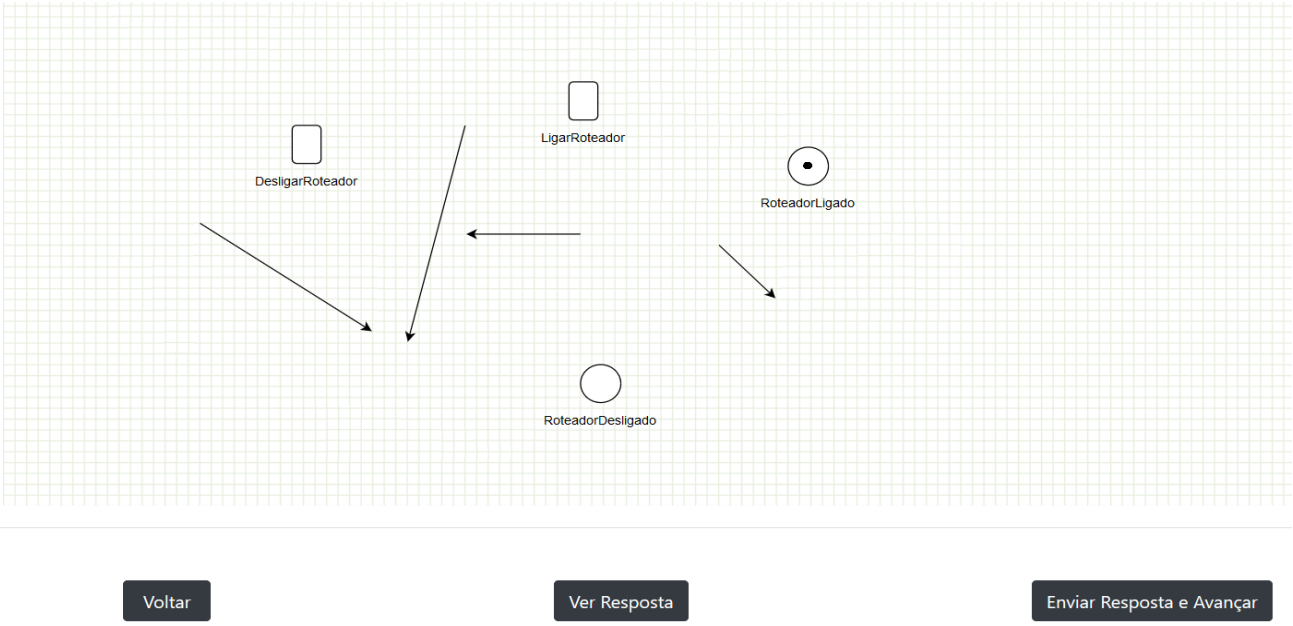

Figura 6: exemplo de exercício de quebra cabeça.

A Figura 6 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 6, o aluno poderá responder à questão ou ver a resposta.

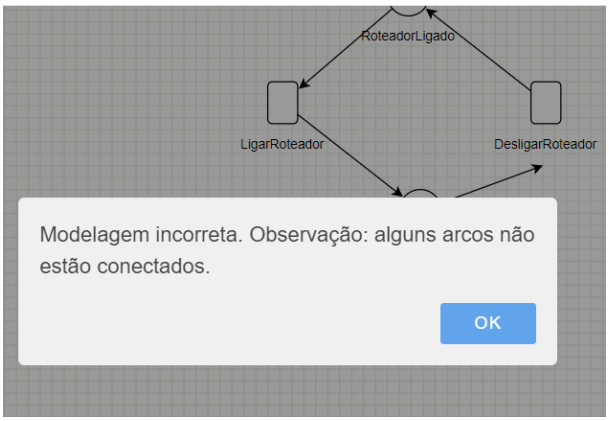

Figura 7: mensagem indicando o erro cometido.

Caso o aluno não acerte a resposta, ao clicar em enviar resposta, uma mensagem de erro irá aparecer na tela (Figura 7). Esta mensagem de erro pode indicar um possível o erro que o aluno cometeu, que ele poderá corrigir e submeter novamente a resposta da questão.

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 8), 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 8, os botões de cima para baixo representam, respectivamente: transição temporizada, transição imediata, lugar, arco, arco inibidor e métrica. O aluno pode remover ou adicionar os elementos das RdP ao editor gráfico quantas vezes quiser.

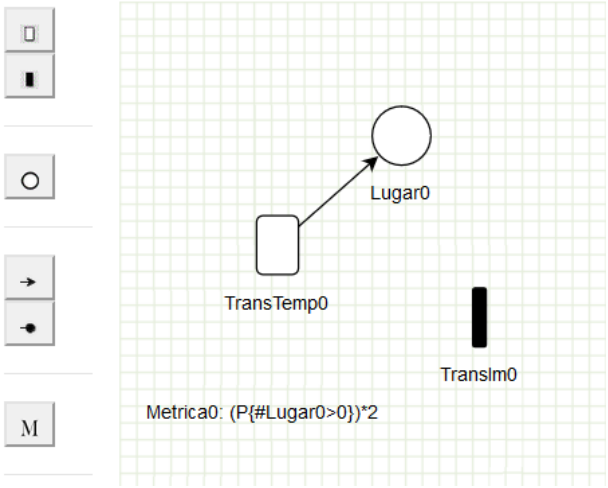

Figura 8: exemplo de exercício de modelagem.

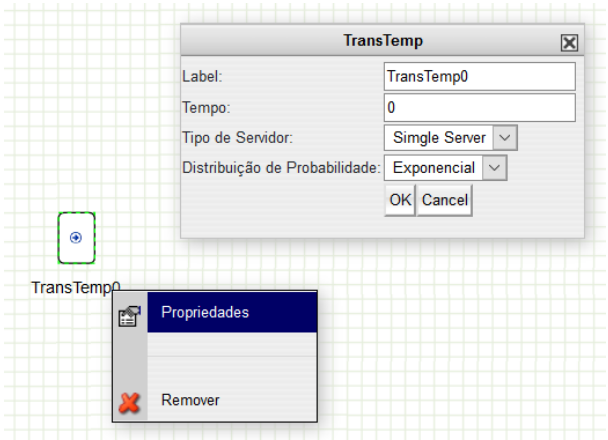

Figura 9: propriedades de uma transição temporizada em um exercício de modelagem.

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 9 mostra as propriedades de uma transição temporizada.

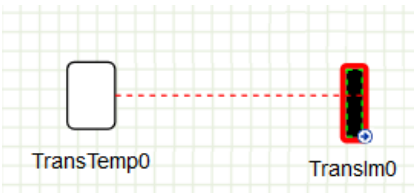

Figura 10: alerta de erro relacionado ao formalismo das RdPs.

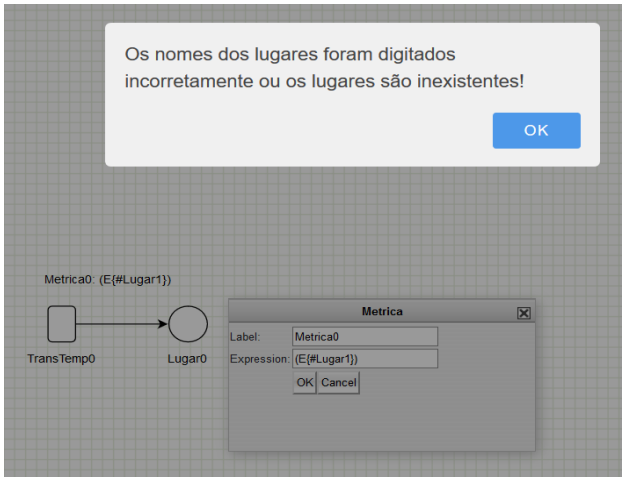

Figura 11: alerta de erro de digitação.

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 10 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 11 mostra um erro em que na métrica (E{#Lugar1}), foi digitado o nome de um lugar que não existe. Note que neste 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, e facilite o uso futuro de ferramentas computacionais de análises e simulação das RdP, como o Mercury [14] ou o TimeNet [16].

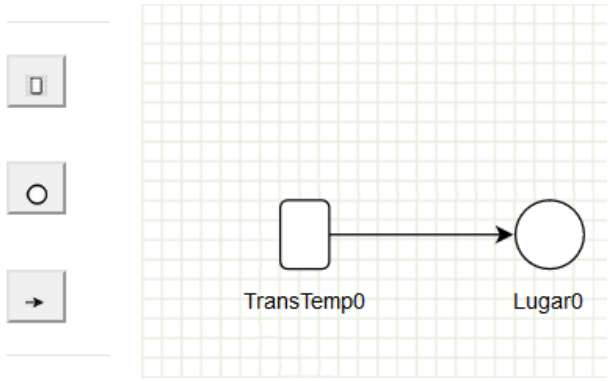

Figura 12: elementos nos exercícios de modelagem na seção 1.

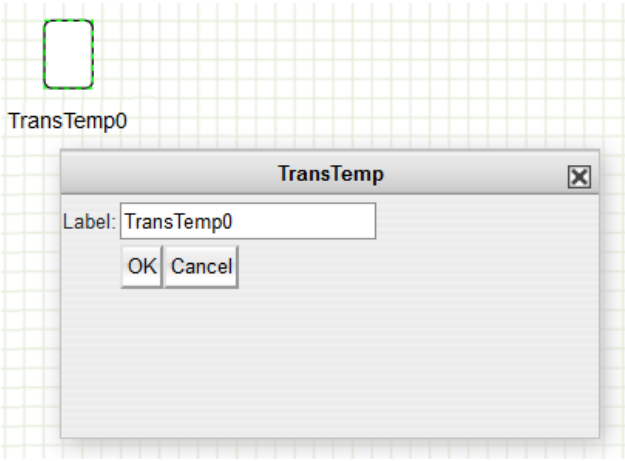

Figura 13: propriedades da transição apresentada na seção 1.

O nível e o tipo dos exercícios variam de acordo com o progresso do estudante. Nas duas primeiras seções (introdução e modelagem básica), os exercícios são do tipo quebra-cabeça e de modelagem. Na última seção (modelagem avançada) os exercícios são apenas de modelagem. No entanto, 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 13, são mostradas as propriedades de uma transição antes de introduzirmos o conceito de uma transição temporizada. Esse conceito só é apresentado na seção 2 da ferramenta. Após apresentar esse novo conceito, novos elementos são adicionados à transição (agora chamada de transição temporizada), conforme apresentado na Figura 9. Do mesmo modo, elementos das redes de Petri são mostrados conforme o progresso do aluno. Por exemplo, na Figura 12 apenas

são mostrados três elementos que compõem uma rede de Petri na seção 1: transição, lugar e arco. Já na seção 3, quando já estivermos trabalhando com arcos inibidores, transições imediatas e métricas, esses novos elementos são mostrados (Figura 8).

Para exemplificar um exercício de modelagem de nível avançado (Figura 14), considere o seguinte anunciado: *"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 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 6 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 a próxima seção. No entanto, se o aluno não conseguir o número mínimo de pontos para seguir para a próxima seção, ele receberá uma mensagem indicando que ele pode fazer os exercícios que ele não concluiu e obter a pontuação mínima para passar de seção.

A ferramenta apresentada já contempla os ajustes que foram feitos a partir da avaliação da ferramenta por dois professores:

- Adicionar exercício de quebra-cabeça: a princípio a ferramenta continha apenas exercícios de modelagem e para fins de melhorias em termos didáticos foram propostos exercícios de quebra-cabeça;
- Informar o estudante sobre que tipo de erro ele cometeu: a princípio a ferramenta apenas alertava a respeito de erros relacionados à sintaxe do formalismo. Com essa melhoria, foram adicionados também alertas indicando qual possível erro o estudante cometeu ao responder um exercício;
- Construir uma sequência didática na apresentação dos conteúdos.

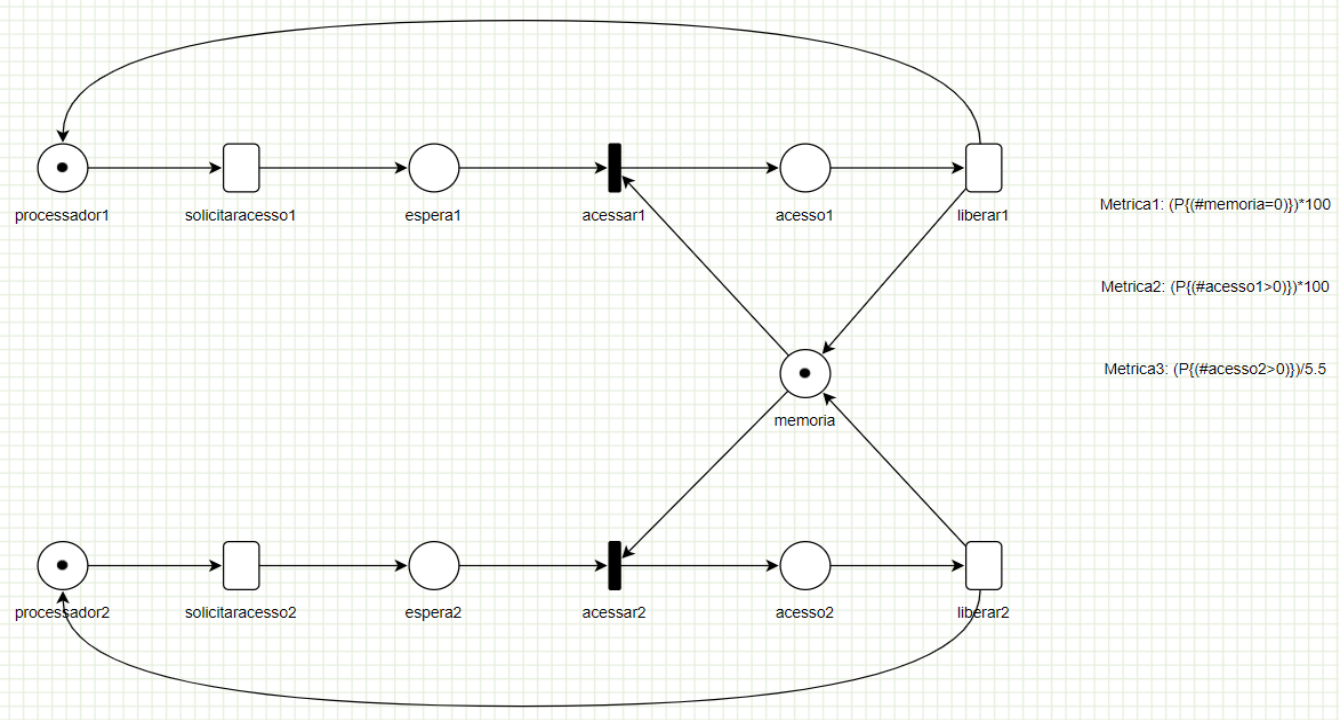

Figura 14: exemplo de exercício de modelagem de nível avançado.

6 Contexto de uso da ferramenta

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.

A ferramenta tem um viés mais prático de modo a orientar o estudante a montar as redes de Petri que representem um determinado sistema. Com isso, os estudantes podem entender previamente "por quê", "para que" e "como" usarem redes de Petri, antes de entrarem em contato com os detalhes matemáticos mais complexos do formalismo.

Assim, além do uso completamente autônomo que a ferramenta oferece, graças à sua estrutura que contempla uma sequência didática com conteúdos teóricos e exercícios para prática, TryRdP também pode ser usada como ferramenta de apoio no ensino formal, em disciplinas de graduação e pós-graduação, como complemento às aulas, para reforçar o conhecimento adquirido.

Existem algumas opções de uso. Os estudantes podem iniciar os estudos sobre modelagem com rede de Petri antes de iniciarem uma aula com um professor em uma disciplina formal de graduação ou pós-graduação. Por isso, a ferramenta apresenta tanto as explicações teóricas quanto os exercícios para praticar, tornando-se bastante versátil. Desse modo um professor, nos primeiros dias de aulas da sua disciplina pode pedir para que os alunos realizem as atividades da ferramenta TryRdP antes que ele inicie o assunto de redes de Petri e aborde detalhes matemáticos mais profundos. Ou ainda, a ferramenta pode ser usada como reforço, paralelamente às aulas, como atividade extraclasse, para revisitar os conteúdos já expostos em sala e praticar com os exercícios.

Outra opção é o uso da sala de aula invertida, em que o professor pode optar por passar o estudo dos conteúdos para momentos extraclasse, e nos encontros com os estudantes (sejam eles presenciais ou virtuais) optar por usar a ferramenta para resolução de exercícios. O professor pode acompanhar essa resolução dos exercícios da TryRdP, e esclarecer dúvidas que o feedback da ferramenta não tenha conseguido suprir. Vale ressaltar que até o momento a ferramenta não possui suporte para customização dos exercícios por parte dos professores.

7 Conclusão e Trabalhos futuros

Artefatos digitais podem funcionar como potencializadores no aprendizado, em especial, para ensino de conteúdos de natureza abstrata, como no caso deste trabalho, modelagem usando redes de Petri. Não foram encontradas na literatura 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.

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 aluno terá acesso a uma página com instruções 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 de exercícios é essencial para que o estudante seja guiado e apoiado ao longo do aprendizado.

Entretanto, para apresentarmos conclusões fundamentadas sobre a eficácia da ferramenta, é preciso avaliá-la, tanto em termos de interação e usabilidade da interface, quanto em termos pedagógicos. Assim, como trabalho futuro, pretendemos realizar avaliações formais da ferramenta com professores e alunos, por meio de testes de usabilidade e experimentos educacionais.

Além disso, para um uso da TryRdP mais eficaz no contexto do ensino formal, ou seja, associada a uma disciplina de graduação ou pós-graduação, algumas extensões da ferramenta poderiam ser feitas, como a possibilidade do professor customizar exercícios e criar novos exercícios, dando assim a possibilidade de personalizar os exercícios às necessidades dos estudantes individualmente, e de deixar a ferramenta mais aderente aos objetivos de cada professor, em sua disciplina.

AGRADECIMENTOS

Queremos agradecer a FACEPE (Nº Processo: IBPG-0370-1.03/19) pelo apoio neste e em inúmeros outros trabalhos realizados pela Universidade Federal Rural de Pernambuco.

REFERÊNCIAS

[1] Arteiro, Roberto & Souza, Fabio & Rosa, Nelson & Maciel, Paulo. (2007). Utilizando redes de Petri para modelagem de desempenho de middleware orientado a mensagem.

[2] Andrade, E.; Nogueira, B.; Matos, R.; Callou, G. Maciel, P. Availability modeling and analysis of a disaster-recoveryas-a-service solution. Computing, pages 1–26, 2017.

[3] Andrade, E. C.; Alves, M.; Nogueira, B.; Maciel, P. Calau: An environment for modeling and analyzing embedded real-time systems. In Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on, pages 3135–3140. IEEE, 2012.

[4] Bennetts, R. On the analysis of fault trees. IEEE Transactions on reliability, 24(3):175–185, 1975.

[5] Chiola, Giovanni; Marsan, Marco; Balbo, Gianfranco; Conte, Gianni. (1993). Generalized Stochastic Petri Nets: A Definition at the Net Level and Its Implications. IEEE Trans. Software Eng.. 19. 89-107. 10.1109/32.214828.

[6] Costa, D.; Teixeira, D.; Grisotto, R.; Rocha, B. 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.

[7] Distefano, S.; Paci, D.; Puliafito, A.; Scarpa, M.; Papardo, C.; Sperone, S. Design and implementation of a performance plug-in for the argouml tool. In IASTED Conf. on Software Engineering, pages 337–342, 2005.

[8] Gómez-Martínez, E.; Merseguer, J. ArgoSPE: Model-based software performance engineering. Petri Nets and Other Models of Concurrency-ICATPN 2006, pages 401–410, 2006.

[9] Hirel, C.; Sahner, R.; Zang, X.; Trivedi, K. Reliability and performability modeling using sharpe 2000. In International Conference on Modelling Techniques and Tools for Computer Performance Evaluation, pages 345–349. Springer, 2000.

[10] Marsan, Marco; Conte, Gianni; Balbo, Gianfranco. (1984). A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems. ACM Trans. Comput. Syst.. 2. 93-122. 10.1145/190.191.

[11] Martin, L.; Leslie, M.; March, L. et al. Urban space and structures. Number 1. Cambridge University Press, 1972.

[12] Murata, T. Petri nets: Properties, analysis and applications.In Proceedings of the IEEE, vol. 77, no. 4, pp. 541-580, April 1989.

[13] 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.

[14] Silva, B.; Matos, R.; Callou, G.; Figueiredo, J.; Oliveira, D.; Ferreira, J.; Dantas, J.; Junior, A.; Alves, V.; Maciel, P. 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.

[15] Trivedi, Kishor, Ermeson Andrade, and Fumio Machida. Combining performance and availability analysis in practice. *Advances in Computers*. Vol. 84. Elsevier, 2012.

[16] Zimmermann, Armin. (2017). Modelling and Performance Evaluation with TimeNET 4.4. 300-303. 10.1007/978-3-319-66335-7_19.

[17] G. Bucci, L. Carnevali, L. Ridi and E. Vicario, Oris: a tool for modeling, verification and evaluation of real-time systems, *Int. J. on Softw. Tools for Techn. Transfer*, vol. 12, no. 5, pp. 391–403, Sep. 2010.

[18] Hirel, Christophe & Tuffin, Bruno & Trivedi, Kishor. (2000). SPNP: Stochastic petri nets. Version 6.0. *Lecture Notes in Computer Science*. 1786. 354-357. 10.1007/3-540-46429-8_30.