CINTHIA KAORI YAMADA
MARCELO FERREIRA MORAIS SILVA

MODELAGEM DA REDE DE TRENS URBANOS EM REDE DE PETRI UTILIZANDO O GOOGLE MAPS

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do título de Engenheiro Mecatrônico

São Paulo
2012

CINTHIA KAORI YAMADA
MARCELO FERREIRA MORAIS SILVA

MODELAGEM DA REDE DE TRENS URBANOS EM REDE DE PETRI UTILIZANDO O GOOGLE MAPS

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do título de Engenheiro Mecatrônico

Área de Concentração:
Engenharia Mecatrônica

Orientador:
Prof. Dr. Fabrício Junqueira

São Paulo
2012

FICHA CATALOGRÁFICA

Yamada, Cinthia Kaori
Modelagem da rede de trens urbanos em Rede de Petri utilizando o Google Maps / C.K. Yamada, M.F.M. Silva. -- São Paulo, 2012.
64 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1. Redes de Petri (Modelagem) 2. Redes de computadores 3. Transporte ferroviário I. Silva, Marcelo Ferreira Morais II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos III. t.

Aos meus pais Ana Maria e Raimundo Nonato e aos meus irmãos Marcos e Milena.

Marcelo Ferreira Morais Silva

Aos meus pais Nair e Ricardo e ao meu irmão Lucas.

Cinthia Kaori Yamada

AGRADECIMENTOS

Agradecemos primeiramente aos nossos familiares por todo apoio e incentivo durante o curso.

Agradecemos também ao Prof. Dr. Fabrício Junqueira, que foi fundamental para o desenvolvimento do projeto.

Finalmente, agradecemos aos amigos e a todos que de alguma forma contribuíram ao nosso desenvolvimento acadêmico e pessoal.

RESUMO

Diariamente, mais de sete milhões de pessoas utilizam os trens da CPTM e do METRÔ. Porém, devido à falta de organização dos transportes públicos e o aumento no número de passageiros, o serviço prestado é ineficiente, levando a um caos urbano nos horários de maior demanda. Por isso, há uma necessidade de criar ferramentas que levem a uma maior organização e planejamento, auxiliando na tomada de decisões para melhorar a qualidade dos serviços prestados. O objetivo do presente projeto foi utilizar o *Google Maps* para implementar uma ferramenta que auxilie a modelagem, em Rede de Petri, da malha ferroviária da região metropolitana da cidade de São Paulo. Foram exploradas as modelagens de trens, linhas e estações, identificando horários de pico, distâncias entre estações, quantidade de trens disponíveis, entre outros. Ou seja, foi mapeada cada uma destas entidades em Rede de Petri para futura geração automática de código. O *Google Maps* serviu como uma interface com o usuário onde são escolhidos os parâmetros desejados para a modelagem. A partir destes parâmetros uma Rede de Petri é criada e enviada para um simulador. Foram utilizadas a API do *Google Maps*, e a linguagem Java para programação do aplicativo.

ABSTRACT

Every day, more than seven million people use the CPTM and METRO trains. However, due to public transport´s lack of organization and the increase in the number of passengers, the service is inefficient, leading to an urban chaos in peak hours. Therefore, there is a need for tools that lead to more organization and planning, assisting in making decisions to improve the quality of services provided. The goal of this project was to implement a tool that assists the modeling of the railway network in the metropolitan region of São Paulo in Petri Nets using Google Maps. The modeling of trains, stations and lines, identifying peak times, distances between stations, number of trains available, among others was explored. Each of these entities was mapped in Petri Nets for future automatic code generation. Google Maps was used as a user interface where the desired parameters are chosen for modeling. From these parameters a Petri net is created and sent to a simulator. We used the Google Maps API and the Java language for application programming.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

Diariamente, mais de sete milhões de pessoas utilizam os trens da Companhia Paulista de Trens Metropolitanos (CPTM) e do Metrô. No entanto, a demanda pelo transporte ferroviário metropolitano na grande São Paulo é maior que o disponibilizado pela atual estrutura existente, resultando em um serviço pouco eficiente, levando a um caos urbano nos horários de maior demanda.

Por causa de fatores como a expansão das linhas do metrô e a criação do bilhete único, o volume de passageiros transportados cresceu cerca de 20% em 2011, quando comparado a 2010 (PEREIRA, 2012). Além disso, o governo tem planos de expandir a rede do metrô em quase três vezes o tamanho atual até 2018 (SILVA, 2012).

Portanto, para auxiliar em um crescimento ordenado da malha de trens urbanos e também melhorar o conforto dos passageiros, há uma necessidade de criar ferramentas que levem a uma maior organização e planejamento, auxiliando nas tomadas de decisões para melhorar a qualidade dos serviços prestados.

A modelagem é um recurso que pode auxiliar nesta tarefa, pois, por meio da representação de uma aproximação simplificada do sistema real, permite o estudo de suas características, suas variáveis, assim como permite a previsão dos impactos que possíveis mudanças podem gerar.

A Rede de Petri é uma ferramenta que pode ser utilizada para a modelagem da rede de trens, pois este sistema pode ser classificado como a eventos discretos. Em Aalst e Odijk (1995) uma Rede de Petri do tipo ITCPN (*interval timed coloured Petri Nets*) foi utilizada no intuito de modelar uma rede de trens de carga. Castelain e Mesghouni (2002) utilizaram uma Rede de Petri de alto nível para modelar uma rede de transporte público de passageiros multimodal considerando tanto o comportamento do meio de transporte em si, quanto de seus passageiros.

No entanto, nem todas as pessoas estão familiarizadas com as formalidades da Rede de Petri, dificultando a modelagem por um usuário comum. Para tornar a criação dos modelos mais amigável e dessa forma, mais acessível, torna-se necessário a criação de uma ferramenta que facilite a implementação de modelos em Rede de Petri.

1.1. OBJETIVO

O objetivo do presente trabalho foi utilizar o *Google Maps* no projeto e implementação de uma ferramenta que auxilie a modelagem, em Rede de Petri, da malha ferroviária da região metropolitana da cidade de São Paulo. Foram exploradas as modelagens de trens, linhas e estações, identificando horários de pico, distâncias entre estações, quantidade de trens disponíveis, entre outros. Ou seja, é mapeada cada uma destas entidades em Rede de Petri para geração automática da Rede de Petri correspondente à malha ferroviária modelada. O *Google Maps* faz a interface com o usuário onde são escolhidos os parâmetros desejados para a modelagem, a partir destes parâmetros uma Rede de Petri é criada e enviada para um simulador.

1.2. ORGANIZAÇÃO DO TRABALHO

Para organizar a apresentação do trabalho realizado, este documento está dividido em quatro seções: primeiramente, apresenta-se a revisão bibliográfica realizada sobre os temas abordados: transporte ferroviário urbano na região metropolitana de São Paulo, Rede de Petri, *Google maps* API e UML. Em seguida, apresenta-se a parte do projeto, modelagem da malha ferroviária em Rede de Petri e a modelagem do *software* em UML. Depois, é mostrado o desenvolvimento e implementação do *software* proposto, bem como seu código fonte. Por fim, são apresentados os resultados do trabalho.

2. REVISÃO BIBLIOGRÁFICA

2.1. O TRANSPORTE FERROVIÁRIO URBANO

Devido a crescente produção de café em São Paulo em meados de 1850 surgiu a necessidade do aperfeiçoamento do escoamento da produção, tanto para os novos centros urbanos quanto principalmente aos portos exportadores. Neste período, o agressivo empreendedor Irineu Evangelista de Sousa, o Barão de Mauá, deu início à construção da ferrovia que ligava Jundiaí - São Paulo - Santos. Após a conclusão deste projeto, muitos outros foram executados gerando a complexa malha ferroviária que envolve o Estado de São Paulo. Neste processo houve um distanciamento entre a expansão com enfoque no escoamento de produtos, manufaturados e matéria prima, e o transporte urbano de passageiros sendo este último o enfoque do presente estudo.

2.1.1. REDE DE TRENS URBANOS DO ESTADO DE SÃO PAULO

A Região Metropolitana do Estado de São Paulo atualmente possui sua rede de trens urbanos gerenciados pela Companhia Paulista de Trens Metropolitanos (CPTM), sendo uma sociedade de economia mista vinculada à Secretaria dos Transportes Metropolitanos do Estado de São Paulo. Como ilustrado na Tabela 1 (CPTM, 2011), a média de usuários por dia útil em 2011 foi de 2,3 milhões de usuários, número que representa mais que o triplo de passageiros registrados em 1994, início das operações. Como o crescimento exponencial dos usuários não foi acompanhado pela expansão significativa da quantidade de novos trens e novas estações além da falta de obras antienchentes, problemas relacionados com a superlotação vem gerando descontentamentos cada vez maiores dos usuários com a CPTM. A depredação da estação Francisco Morato em 29/03/2012 (FOLHA, 2012), foi o estopim da contínua insatisfação da população com o transporte publico.

2.1.2. REDE METROVIÁRIA DO ESTADO DE SÃO PAULO

Além da CPTM a cidade de São Paulo conta com a Companhia do Metropolitano de São Paulo (Metrô) que integra o transporte urbano sobre trilhos. O Metrô como a CPTM é uma empresa de capital mista sendo a maior parte do controle acionário assumido pelo governo do estado (METRÔ, 2012). A primeira viagem ocorreu em 1972, entre as estações Jabaquara e Saúde, porém somente em 1974 começou a operar em escala comercial no trecho Jabaquara – Vila Mariana, esta linha se estendeu até Tucuruvi e foi denominada de Linha

Azul, ligando o Norte e o Sul da cidade de São Paulo como observado na Figura 1. Em 1979, a linha com maior circulação de passageiros nos dias de hoje, a Linha Vermelha, iniciou suas operações. Atualmente, une as estações Barra Funda – Itaquera que correspondem às regiões Leste - Oeste. A conhecida "Linha da Paulista" foi a terceira em entrar em operação em 1991, seguida pela implementação em 2002 da Linha – Lilás no extremo oeste da cidade, sendo a menor em extensão e que atende uma das regiões mais carentes da cidade, Capão Redondo. Em julho de 2010 a primeira linha metroviária a ser operada em regime de concessão privada pelo Consórcio ViaQuatro iniciou suas atividades, ainda em estágio inicial estão funcionando apenas 6 das 11 em projeto mas já conta com um fluxo de muito grande passageiros ainda não mensurável.

Tabela 1 – Dados das linhas controladas pela CPTM (CPTM, 2011)

Linha	Itinerário	Usuários (mil)	Extensão Total (km)
Linha 7	(Luz – Jundiaí)	391.27	60.5
Linha 8	(Júlio Prestes – Itapevi)	430.56	35.3
Linha 9	(Osasco – Grajaú)	364.49	31.8
Linha 10	(Brás – Rio Grande da Serra)	366.16	37.2
Linha 11	(Luz – Estudantes)	558.77	50.8
Linha 12	(Brás – Calmon Viana)	208.72	38.8
Total		2 319.97	254.40

2.2. REDE DE PETRI

2.2.1. CONCEITOS BÁSICOS

Rede de Petri é uma ferramenta gráfica e matemática que permite o projeto, modelagem e análise e projeto de Sistemas a Eventos Discretos (SED) e foi proposta na tese de doutorado de Carl Adam Petri, em 1962. Ao longo dos anos, sua teoria e formas de representação foram estabelecidas e suas aplicações foram se expandindo. Atualmente, é uma ferramenta amplamente utilizada na representação de Sistemas a Eventos Discretos.

Como aplicações típicas da Rede de Petri, pode-se citar: automação de manufatura, automação de escritórios, avaliação de desempenho, banco de dados, circuitos integrados, protocolos de comunicação, sistemas distribuídos e sistemas de produção (MARRANGHELLO, 2005).

Figura 1 – Mapa do Transporte Ferroviário Metropolitano de São Paulo

Os elementos que compõem graficamente a Rede de Petri podem ser classificados em (CARDOSO e VALETTE, 1997):

- **Transições**: são representadas por retângulos, se ocos são transições temporizadas e se sólidos são transições instantâneas. São elementos ativos da rede, são associados a um evento que ocorre no sistema e podem provocar a mudança de estado;
- **Lugares**: são representados por um círculo. São elementos passivos que armazenam informações, podendo ser interpretados como uma condição, um estado parcial, uma espera, um procedimento, um conjunto de recursos, um estoque, uma posição geográfica;
- **Marcas**: são representados por um ponto. São indicadores de recursos ou informações, significando que a condição associada ao lugar é verificada;
- **Arcos orientados**: são representados por linhas com ponta final em forma de flecha. São elementos que indicam a direção e sentido do fluxo de informação ou recurso e correspondem ao caminho que as marcas seguem.

Como vantagens na utilização de Rede de Petri (CARDOSO e VALETTE, 1997), tem-se principalmente:

- Flexibilidade, pois permite a representação da dinâmica e da estrutura do sistema no detalhamento desejado;
- Os estados e eventos são mostrados explicitamente;
- Uma única ferramenta é utilizada em todas as etapas (especificação, modelagem, análise, avaliação de desempenho e implementação), bem como para os diversos níveis hierárquicos de controle e permite representar elementos de diferentes significados utilizando a mesma representação;
- É possível descrever de forma natural as características dos Sistemas a Eventos Discretos.

Exemplos de características de SED modeladas em Rede de Petri (MARRANGHELLO, 2005):

- **Conflito** – Na Figura 2, ao surgir uma marca no lugar P0, tanto a transição T0 quanto a T1 poderiam ocorrer. Mas somente uma delas ocorrerá, pois só há uma marca para duas transições. Não é possível determinar em Rede de Petri qual das duas transições ocorrerá, a escolha é aleatória.

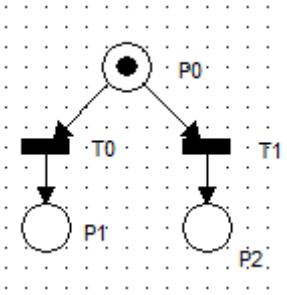

Figura 2 – Conflito

- **Paralelismo** – Na Figura 3, a transição T0 ocorrerá e ambos os lugares, P1 e P2, receberão marcas. As transições seguintes aos lugares P1 e P2 ocorrerão em paralelo.

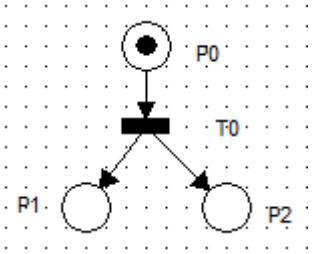

Figura 3 – Paralelismo

- **Sequência** – Na Figura 4, ao surgir uma marca em P0, as transições ocorrerão em sequência, ou seja, primeiro ocorrerá T0 e depois ocorrerá T1.

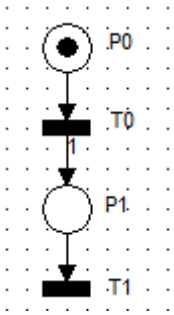

Figura 4 – Sequência

- **Sincronização** – Na Figura 5, a condição necessária para que a transição T0 ocorra é que os lugares P0 e P1 possuam marcas, ou seja, se P0 receber uma marca antes de P1, terá que esperar até que P1 receba uma marca também, sincronizando o processo.

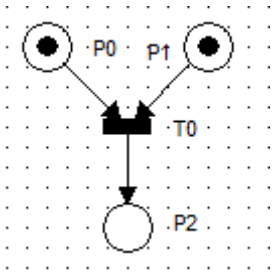

Figura 5 – Sincronização

- **Compartilhamento de recurso** – Na Figura 6, as transições T0 e T1 dependem da marca no lugar P2 para ocorrer. A marca em P2 representa um recurso compartilhado por duas atividades diferentes, representadas por P3 e P4. Ao término da atividade que estiver utilizando o recurso, este é devolvido, permitindo que a outra atividade seja realizada.

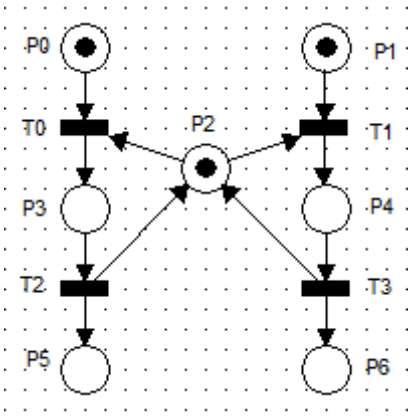

Figura 6 – Compartilhamento de Recurso

2.2.2. TIPOS DE REDE DE PETRI

Uma forma muito utilizada para classificar Rede de Petri é segundo o seu grau de abstração. As Redes de Petri de baixo nível de abstração não possuem distinção entre suas marcas, podendo ser divididas entre a Rede Condição – Evento e a Rede Lugar - Transição. Já nas redes de alto nível suas marcas podem ser diferenciadas através de sua associação a cores, valores ou dados abstratos (MARRANGHELLO, 2005).

A. Rede Condição – Evento

A Rede de Petri Condição – Evento, também conhecida com Rede de Petri elementar, permite apenas uma marca por vez em cada lugar. Usualmente os lugares são chamados de condições e as transições de eventos. A ocorrência de um evento está limitada a pré-requisitos que ou estão satisfeitos ou não. É utilizada no estudo de compartilhamento de recurso e no desenvolvimento teórico da Rede de Petri (MARRANGHELLO, 2005). Um comportamento importante da Rede de Petri Condição – Evento é que ela possui um comportamento não determinístico em caso de conflito, ou seja, quando dois eventos possuem suas condições satisfeitas e a ocorrência de um acarreta na não ocorrência do outro (MIYAGI, 1996).

B. Rede Lugar – Transição

A principal diferença entre o tipo anterior e a Rede de Petri Lugar – Transição é que esta última permite mais de uma marca em cada lugar ao mesmo tempo, evidenciando quantidades e não mais apenas a ocorrência de uma condição. Agora, os arcos orientados podem ser ponderados, simbolizando que mais de uma marca será retirada quando ocorre uma transição (no caso o valor da ponderação) e os lugares possuem indicações de capacidades. Com estas modificações em relação à rede elementar, foi possível a criação de modelos com menor número de elementos.

C. Rede Colorida

A Rede de Petri Colorida permite marcas coloridas na rede, adicionando a possibilidade de diferenciar as marcas entre si. Assim, uma marca azul pode representar um objeto A, enquanto que uma marca vermelha, o objeto B. As marcas também podem estar

associadas a atributos, como tempo e quantidades. Nela, é preciso definir os tipos de marcas que são aceitos nos lugares. Nesta Rede de Petri, arcos orientados aceitam declarações mais complexas, como lógicas e operações matemáticas.

2.2.3. SOFTWARE PARA MODELAGEM EM REDE COLORIDA – CPN TOOLS

Para a modelagem em Rede de Petri Colorida, será utilizado o *software* CPN Tools versão 3.4. Para a definição e manipulação de dados é utilizada a linguagem de programação CPN ML, que é baseada no padrão *ML* (*Markup Language*) (JENSEN, KRISTENSEN e WELLS, 2007)

A interface do *software* pode ser vista na Figura 7. O modelo é um exemplo explicado passo a passo no artigo de Jensen, Kristensen e Wells (2007), e representa o funcionamento de um protocolo de comunicação.

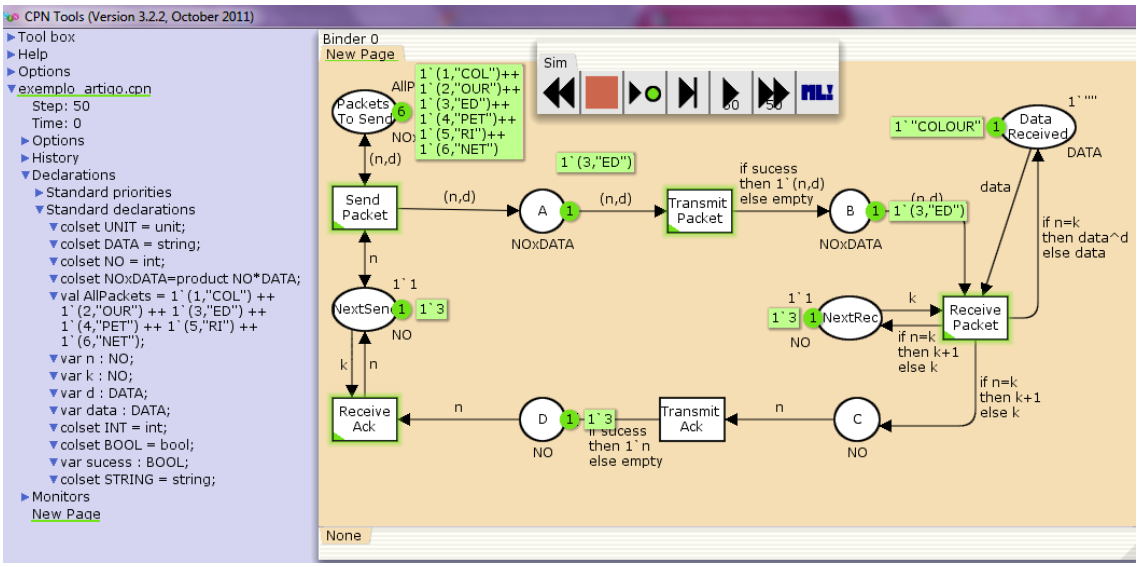

Figura 7 – Interface do *software* CPN Tools

O *software* CPN Tools não permite a limitação da capacidade de um lugar. Assim, para conseguir realizar esta restrição é preciso utilizar a seguinte construção que permite a limitação de capacidade: é necessário criar um lugar complementar ao lugar original que deverá ter um número de marcas iniciais igual à capacidade máxima desejada ao lugar. Assim, para que uma transição ligada ao lugar ocorra, o lugar complementar necessita ter uma marca, que é devolvida após a ocorrência da transição posterior ao lugar. No exemplo da Figura 8, é desejado que o “place_2” possua no máximo 2 marcas. Assim, o seu lugar complementar “anti_place” possui inicialmente 2 marcas. Ao adquirir sua primeira marca,

uma marca é consumida no lugar complementar e esta marca só retornará quando uma marca for retirada do “place_2”.

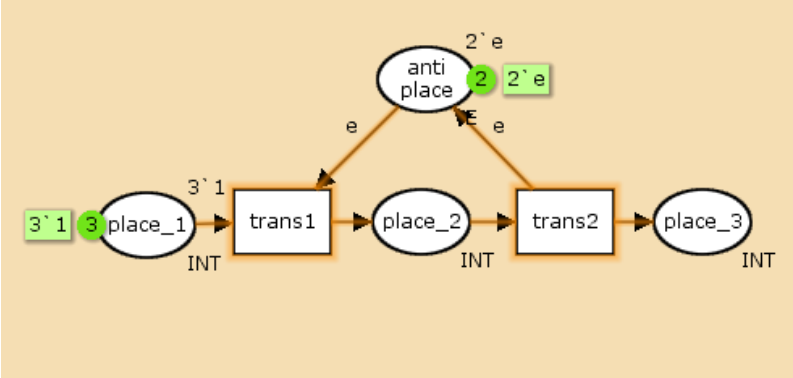

Figura 8 – Exemplo de lugar complementar

O formato do arquivo de entrada do CPN Tools é o XML ISO 8856-1, que pode ser aberto diretamente no bloco de notas.

O código sempre contém um cabeçalho, contendo informações sobre a versão do programa, definição das cores e variáveis utilizadas no modelo. As cores do modelo podem ser definidas como *int*, *string*, *boolean*, *unit* ou então, pode-se relacionar uma cor à uma letra ou um número, declarando da seguinte forma:

```
colset E = with e;
```

Neste caso, a cor E foi definida para aceitar somente a letra “e”.

No código, cada elemento é definido por um id único, seu formato e posição, cores e variáveis. Para a definição de um lugar, primeiramente tem-se a definição de seu Id, como pode ser visto no trecho de código abaixo.

```
<place id="ID1412862538">
```

Em seguida, é preciso definir a localização do lugar no modelo, o texto opcional e o tamanho da elipse que representa o lugar.

```
<posattr x="335.000000" y="388.000000"/>
<text>anti-place</text>
<ellipse w="38.000000" h="40.000000"/>
```

Depois, é definida a localização da marca e de sua inscrição em relação ao lugar. Como pode ser visto abaixo, é possível ocultar ou não, o texto com a inscrição da marca.

```
<token x="-10.000000" y="0.000000"/>
<marking x="0.000000" y="0.000000" hidden="false"></marking>
```

Também é declarado qual o tipo permitido no lugar, seu id e posição, conforme pode ser visto abaixo:

```
<type id="ID1412862539">
  <posattr x="355.000000" y="364.000000"/>
  <text tool="CPN Tools" version="3.4.0">E</text>
</type>
```

Por fim, tem-se a declaração da marca inicial no lugar. A marca inicial possui um id e também é preciso definir sua localização no espaço do modelo.

```
  <initmark id="ID1412862540">
    <posattr x="355.000000" y="411.000000"/>
    <text tool="CPN Tools" version="3.4.0">e</text>
  </initmark>
</place>
```

A definição de uma transição também começa com a declaração de sua id.

```
<trans id="ID1412680237">
```

Em seguida, é definida a posição da transição no espaço do modelo, o texto opcional e o tamanho do retângulo que representa a transição.

```
<posattr x="17.000000" y="-3.000000"/>
<text></text>
<box w="60.000000" h="40.000000"/>
```

Quando a transição é temporizada, também é preciso definir sua variável tempo, com uma id, posição, e o valor da temporização.

```
<time id="ID1413248133">
  <posattr x="69.500000" y="28.000000"/>
  <text tool="CPN Tools" version="3.4.0">@+60</text>
</time>
```

É possível definir uma lógica na transição manipulando as variáveis de entrada e saída da transição. No código, a lógica é definida no trecho denominado “code” que necessita da definição de uma id e posição. No “code” primeiro deve-se definir os *inputs* e *outputs* da transição. Para auxiliar na construção da lógica, é possível definir variáveis auxiliares internas através do comando *let*. No exemplo, M foi definido como uma distribuição de Poisson de média 35.

```
<code id="ID1412680240">
  <posattr x="-122.000000" y="-159.000000"/>
  <text tool="CPN Tools" version="3.4.0">
    input (a, c);
    output (b, d);
    action
    let
      val M = poisson(35.0)
    in
      if a = 0 then (a, c)
      else if a &gt;= M then (a - M, c + M)
      else (0, c + a)
    end
  </text>
</code>
</trans>
```

A definição do arco também começa com a declaração do número de sua id, seguida da sua direção, que pode ser do lugar para a transição - PtoT (*place to transition*), da transição para o lugar - TtoP (*transition to place*) ou bidirecional - BOTHDIR (*both directions*).

```
<arc id="ID1412680314" orientation="PtoT">
```

A localização do arco no modelo é sempre considerada como em x = 0 e y = 0 porque a existência do arco está ligada ao lugar e à transição e, portanto, sua localização também depende da localização destes dois elementos.

```
<posattr x="0.000000" y="0.000000"/>
```

Em seguida, é preciso determinar qual o lugar e qual a transição que estão relacionados ao arco. Isto é determinado pelo id do lugar e da transição.

```
<transend idref="ID1412680237"/>
<placeend idref="ID1412680141"/>
```

Por fim, define-se a inscrição do arco, seu id e sua posição. A inscrição pode ser apenas uma variável (que deve ser do mesmo tipo que a cor do lugar) ou pode conter uma lógica. A inscrição também pode ser temporizada, bastando acrescentar à ela "@+(tempo)"

```
<annot id="ID1412680315">
  <posattr x="3.000000" y="61.000000"/>
  <text tool="CPN Tools" version="3.4.0">c</text>
</annot>
</arc>
```

2.3. GOOGLE MAPS API

2.3.1. INTERFACE ENTRE APLICAÇÕES E PROGRAMAÇÃO

O advento da internet renovou o espírito empreendedor e a forma de fazer negócios, assim o mundo fechado dos *softwares* onde as empresas gerenciavam e criavam seus códigos de forma exclusiva vem perdendo espaço para formas cooperativas de programação, onde há uma interação do programador de qualquer parte do mundo com outros programas.

Esse modelo de negócio permite a exploração de um aplicativo sem que seja necessariamente proprietária dele. A conhecida API (Interface entre Aplicações e Programação) é um poderoso recurso, criado a partir desta perspectiva de criação cooperativa, que permite a interação de um aplicativo com outro sem que seja necessário conhecer os detalhes da implementação do outro *software* a ser integrado, mas somente utilizar seus serviços. Assim, a API é um conjunto de instruções e padrões de programação para acesso a um aplicativo de *software*.

2.3.2. API GOOGLE MAPS

A API Google Maps permite ao desenvolvedor integrar os serviços que o Google Maps oferece às suas próprias páginas *web* possibilitando a interação com mapas além de adicionar conteúdo a eles por meio de uma variedade de funções. A partir deste recurso o desenvolvedor pode criar aplicações de mapas extremamente complexos sem desenvolver integralmente o *software*.

O código *JavaScript* para a API do Google Maps carrega os principais objetos *Javascript* e símbolos para o uso da API que permitem a geolocalização, localização do idioma, localização da região e outras funcionalidades (GOOGLE, 2012). As ferramentas da API estão disponíveis também em diversas bibliotecas entre elas a *geomery*, que possui funções capazes de fazer cálculos geométricos com objetivo de encontrar área ou distâncias de determinadas regiões, a biblioteca *adsense* que permite a inclusão de anúncios de texto além de diversas outras bibliotecas.

Há também a API Static Maps que permite integrar uma imagem do Google Maps em uma página web, sem a necessidade de JavaScript ou de qualquer carregamento de página dinâmica.

2.4. UML

O ponto inicial da criação de um *software* complexo deve ser a modelagem, que consiste em construir modelos que refletem as características e/ou o comportamento do programa a ser escrito. A modelagem proporciona uma visão global do *software* facilitando a codificação além de auxiliar no entendimento do programa por parte de outras pessoas que não o criador do código. Em geral, a modelagem de *software* utiliza alguma notação gráfica.

Dentre as linguagens de modelagem, a UML (*Unified Modeling Language* ou Linguagem Unificada de Modelagem) será utilizada neste projeto. A UML é utilizada para documentação, especificação, visualização e desenvolvimento de sistemas orientados a objetos (OMG, 2005). A UML privilegia a descrição de um sistema seguindo três perspectivas:

- Dados estruturais: Os diagramas de classes;
- Operações funcionais: Os diagramas de casos de uso;
- Eventos temporais: Os diagramas de sequência, atividades e transição de Estados.

2.4.1. CASOS DE USO

O diagrama de caso de uso lista o conjunto de funcionalidades do sistema, por meio do elemento sintático "caso de uso" representado por um círculo, e os elementos externos que interagem com o sistema por meio do elemento sintático "ator". Os casos de uso e os atores são unidos por meio de relacionamentos de dependência, generalização e associação. Os casos de usos são aplicados para ilustrar o comportamento pretendido do sistema a ser desenvolvido, sem ser necessário explicar como esse comportamento é implementado.

2.4.2. DIAGRAMA DE CLASSES

O diagrama de classe especifica os atributos, métodos e os relacionamentos entre as entidades, assim este diagrama produz uma descrição próxima da estrutura de código do programa. Classes e relacionamento constituem os elementos sintáticos básicos deste diagrama.

2.4.3. DIAGRAMA DE SEQUÊNCIA

Este diagrama representa a sequência de processos de uma situação especifica por meio da troca de mensagens entre diversos objetos, as mensagens representam os serviços solicitados entre os objetos. O diagrama de sequência enfatiza a ordenação temporal do processo e permite entender de forma intuitiva como e quando os objetos interagem entre si.

3. PROJETO

O projeto foi dividido em 2 partes. A primeira consiste na modelagem dos elementos que compoem a rede ferroviária em Rede de Petri para servir de gabarito na geração automática de código. A segunda etapa, consiste no desenvolvimento do *software* que permitirá a modelagem da rede utilizando o Google Maps.

3.1. MODELAGEM EM REDE DE PETRI

O tipo de Rede de Petri escolhido para modelar o problema proposto foi a Rede de Petri Colorida pois possui uma funcionalidade que permite a associação de atributos às marcas da rede. Esta característica é importante para a representação do fluxo de passageiros nos trens e nas estações e não seria possível com a utilização de Rede de Petri de baixo nível de abstração.

A modelagem em Rede de Petri Colorida foi feita utilizando-se o *software* CPN Tools, de acordo com as considerações que podem ser vistas abaixo.

No início, um primeiro modelo foi construído a partir de uma abordagem simplificada da rede de trens. Nela, o fluxo de passageiros não é representado e, portanto, o que diferencia os trechos das linhas e as estações será apenas o tempo de ocorrência da transição.

Um trecho de linha pode ser representado por um lugar e uma transição temporizada. De acordo com dados fornecidos pela CPTM (Tabela 2), a velocidade comercial dos trens varia de 34 a 52 km/h, com uma média de 41 km/h. Como a distância de segurança utilizada nos trens da CPTM é de 450m, as linhas foram divididas em intervalos de 500m e, portanto tem-se que o tempo para a ocorrência de cada transição será de 45 segundos, que representa o tempo que um trem leva para percorrer aproximadamente 500m.

Tabela 2 – Informações Técnicas e Operacionais (CPTM, 2011)

Nome da linha		LINHA 7 LUZ/FMO	LINHA 7 (Extensão) FMO/JUN	LINHA 8 JPR/IPV	LINHA 9 OSA/GRA	LINHA 10 BAS/RGS	LINHA 11 (Expresso) LUZ/GUA	LINHA 11 (Extensão) GUA/EST	LINHA 12 BAS/CVN
Velocidade comercial (km/h)		39	52	34	45	34	45	43	37
Intervalo entre trens (minutos)	Pico Manhã	8	11	6	4	5	4	9	6
	Vale	8	14	8	7	8	7	8	8
	Pico Tarde	6	11	7	4	5	4	8	6
Tempo de parada nas estações		O tempo de parada nas estações pode variar entre 20s/30s ou 1 min, dependendo da estação, sentido e horário							

Nesta abordagem simplificada, uma estação também pode ser representada por um lugar e uma transição. O trem fica parado na estação por aproximadamente 30 segundos durante os horários de maior demanda. A CPTM considera como horários de maior demanda, o período das 04h00 às 8h30 e das 16h30 às 20h00.

A distância considerada entre a primeira estação e a segunda estação é de aproximadamente 1km, por isso, no modelo há duas transições temporizadas de 45 segundos. Na Figura 9, pode-se visualizar o início da simulação, com o trem partindo.

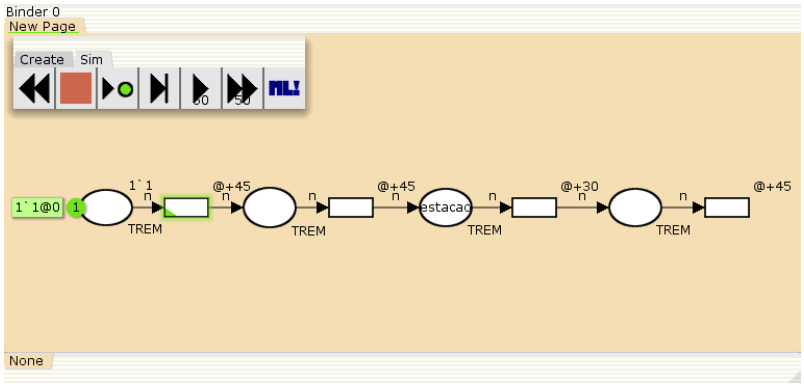

Figura 9 – Modelagem - configuração inicial

Como pode ser visto na Figura 10, o trem levou 90 segundos para percorrer 1 km e chegar à estação, por isso a marca que representa o trem possui um atributo de tempo 90.

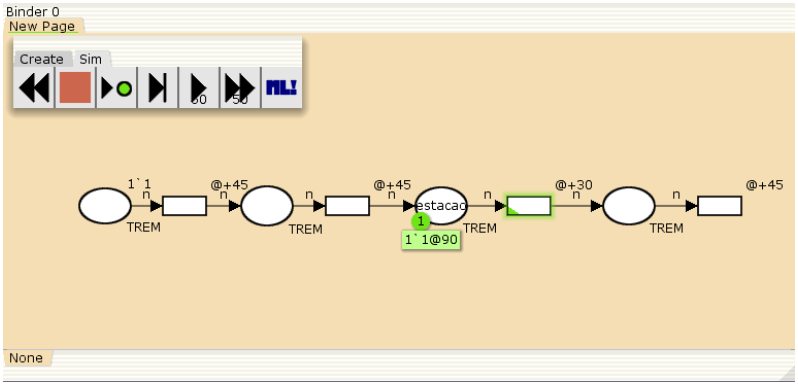

Figura 10 – Trem ao chegar à estação

De acordo com a Figura 11, ao chegar à estação, o trem aguarda por 30 segundos para o desembarque e embarque de passageiros e no tempo = 120, parte da estação.

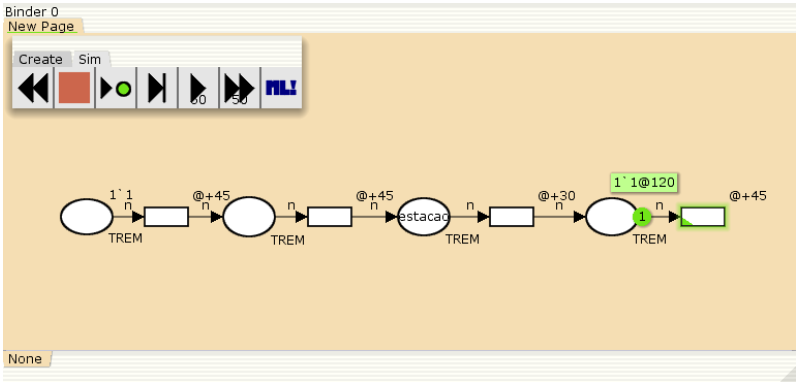

Figura 11 – Trem partindo da estação

Após esta abordagem simplificada, um segundo modelo foi proposto. Neste caso, o fluxo de passageiros também foi modelado e, para isso, o trem passou a ter um atributo do tipo INT que representa a quantidade de passageiros que estão sendo transportados pelo trem.

Uma plataforma pode ser representada pelos elementos na Figura 12. Agora, é preciso uma lógica para determinar a quantidade de passageiros que desembarcam e embarcam no trem.

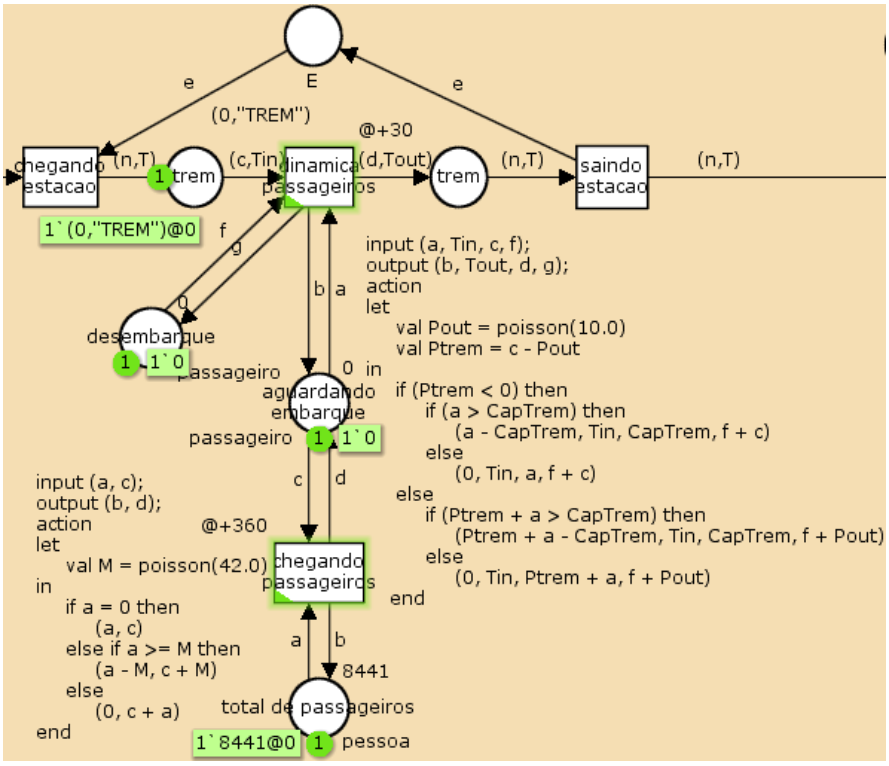

Figura 12 – Modelagem de 1 plataforma

A dinâmica de passageiros começa com a lógica de desembarque de passageiros, seguido pela de embarque. Para o desembarque de passageiros, tem-se que o número de

passageiros que desembarcam na estação é igual a uma distribuição de Poisson de média 10, representado pela variável “Pout”. Como não há dados sobre desembarque de passageiros, esta variável foi definida arbitrariamente. Se o número de passageiros que já estavam no trem for maior que “Pout”, um número igual a “Pout” desembarca. Se o número de passageiros for inferior, todos os passageiros desembarcam. Todos os passageiros que desembarcaram na estação passam para o lugar "desembarque".

O exemplo da Figura 12 foi feito com base nos dados da estação Hebraica – Rebouças no ano de 2011 (Tabela 5), que podem ser vistos no Anexo A. No ano de 2011, ocorreram 6.077.682 embarques, com uma média de 16.882 embarques por dia. No lugar denominado "total de passageiros" há o total de passageiros que deverão embarcar em cada plataforma em 1 dia e como cada estação possui 2 plataformas, 8441 passageiros.

Foi considerado que uma estação funciona 20 horas por dia, das 4h00 às 24h00 e, portanto, há uma média de 422 passageiros por plataforma a cada hora. De acordo com a CPTM, um trem passa aproximadamente a cada 360 segundos, ou seja, uma média de 10 trens por hora. Para estimar quantas pessoas chegam à estação, foi suposto que a cada 360 segundos chegam uma média de 42 pessoas em cada plataforma. No modelo, isso está definido pelo tempo que a transição “chegando passageiros” leva para ocorrer, no caso 360 segundos e pela variável “M” que representa um número correspondente a uma distribuição de Poisson de média 42. Em seguida, os passageiros passam a aguardar o embarque no lugar denominado "aguardando embarque". Para determinar o embarque de passageiros há a seguinte lógica: se o número de passageiros não excede o limite do trem “Captrem”, então todos os passageiros aguardando embarcam, se não, somente o número de passageiros igual ao número de lugares vagos embarcam.

Um ponto importante da modelagem, é que o fluxo de trens não está dependente da existência de passageiros na estação. Isto é obtido por meio de uma lógica que estabelece que quando não há passageiros disponíveis para embarcar, uma marca com 0 passageiros fica disponível para disparar a transição da estação.

Da mesma forma que na primeira modelagem simplificada, um trem demora 30 segundos na estação, para o desembarque e embarque de passageiros, e 45 segundos em cada trecho de linha.

Na Figura 13, pode-se ver a modelagem que representa duas estações separadas por 1 km de distância. Cada estação possui 2 plataformas, pois recebem 2 linhas de trens. Os trens percorrem um caminho circular, saindo da primeira estação, chegando até a última estação pela linha de ida e voltando novamente à primeira estação pela linha da volta.

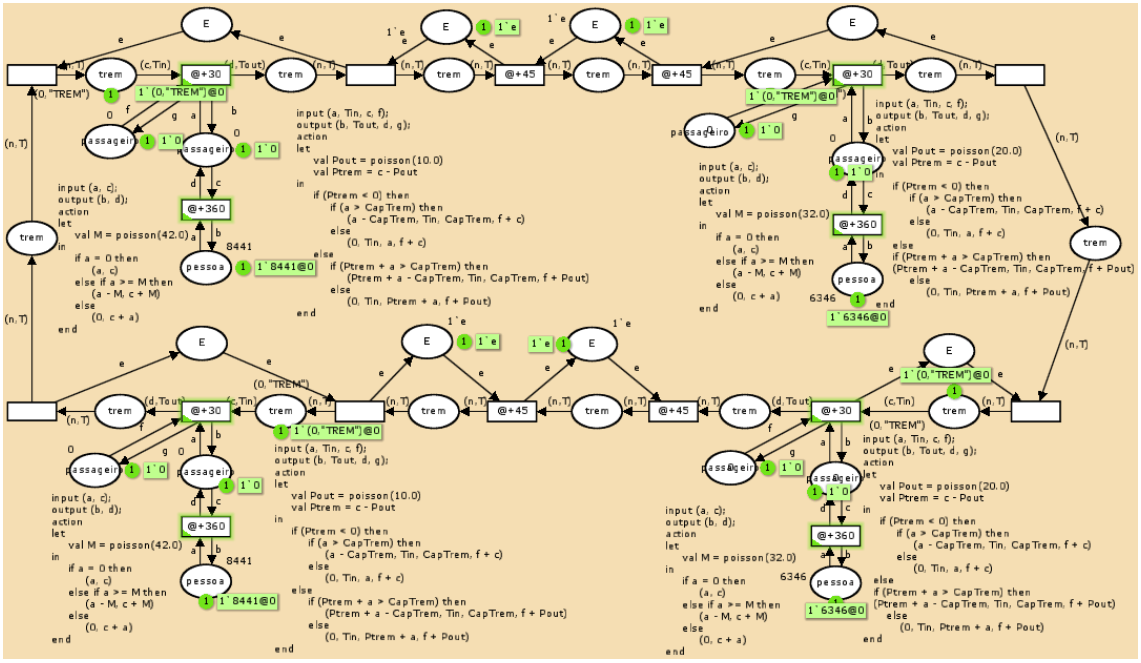

Figura 13 – Modelagem com 2 estações

Como pode ser visto, inicialmente, cada estação começa com 1 trem. Na estrutura real, ao iniciar as atividades, para uniformizar a distribuição de trens ao longo da linha, alguns trens não partem da estação inicial, e sim de estações intermediárias para que todas as estações iniciem as atividades no mesmo horário. Como cada trecho de linha deve conter apenar um trem e em cada plataforma também, e a Rede de Petri Colorida não permite a limitação de capacidade, foi preciso adicionar um lugar complementar que serve para limitar a capacidade de cada lugar.

Na figura 14, tem-se o momento em que um trem chega à estação com 0 passageiros e tempo 0. Na estação há 45 passageiros aguardando embarque.

Na figura 15, pode-se ver o momento em que o trem passou da estação. Dos 30 passageiros iniciais, 11 desembarcaram e todos os 45 que aguardavam embarcaram. Isto ocorreu porque a soma dos passageiros que não desembarcaram e os que aguardavam embarque não excedia a variável “Captrem”, que no exemplo está definida como 1900. Ao realizar o processo de embarque e desembarque, o trem levou 30 segundos, representados por seu atributo @30.

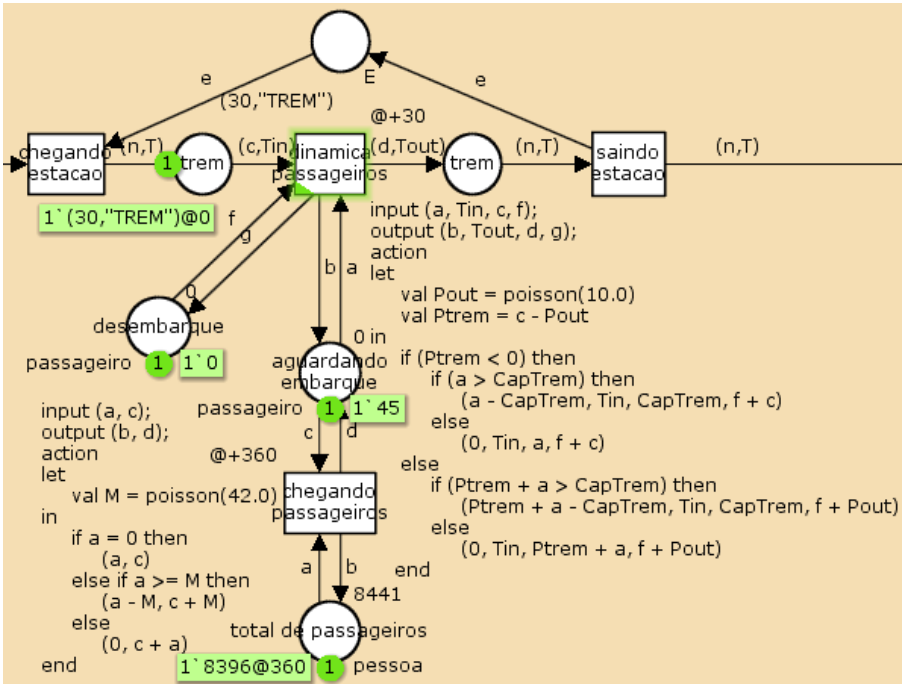

Figura 14 – Trem chegando à plataforma

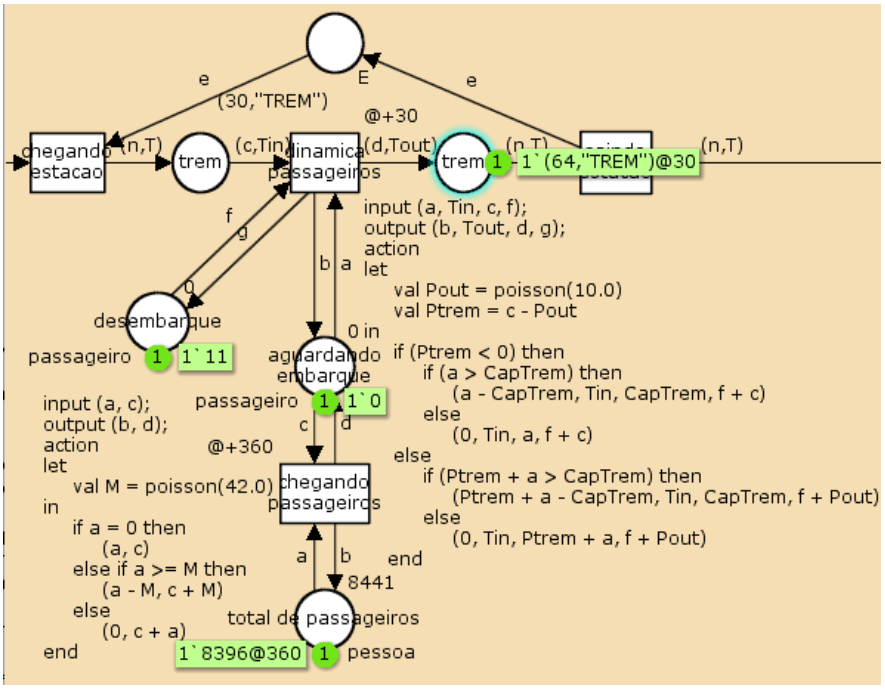

Figura 15 – Trem ao sair da plataforma

3.2. MODELAGEM EM UML

3.2.1. CASOS DE USO

No modelo os atores são: Usuário e Simulador. Como ilustrado na Figura 16, os casos de uso relacionados ao Usuário são: Inserir Elemento, Remover Elemento, Exportar para o

Simulado e Verificar Distâncias. Enquanto o Simulador possui os casos de uso: Abrir Rede de Petri, que depende do caso de uso Exportar para Simulação, Gerar Resultados e Simular. A documentação detalhada dos Casos de Usos pode ser vista no Apêndice A.

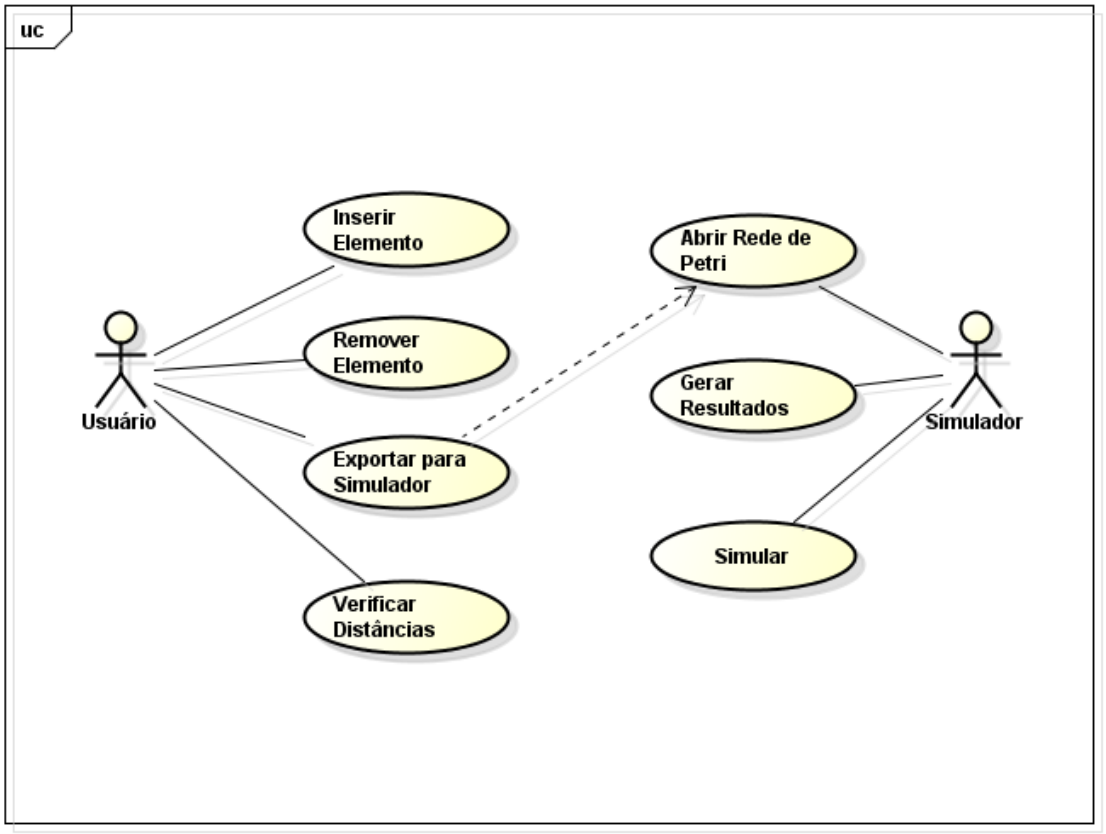

Figura 16 – Diagrama de casos de uso

3.2.2. DIAGRAMA DE CLASSES

As classes do projeto são: Passageiro, Trem, Estação, Linha, Plataforma e Segmento e Plataforma. O diagrama de classe especifica os atributos, métodos e os relacionamentos entre as entidades, assim este diagrama produz uma descrição próxima da estrutura de código do programa. Classes e relacionamento constituem os elementos sintáticos básicos deste diagrama. As classes e seus respectivos relacionamentos são ilustrados na Figura 17.

3.2.3. DIAGRAMA DE SEQUÊNCIA

A. Inserir Elemento

Este diagrama representa a inclusão de um elemento, no caso este elemento é uma estação. A Figura 18 mostra a sequencia da inclusão de uma nova estação e como os objetos se relacionam.

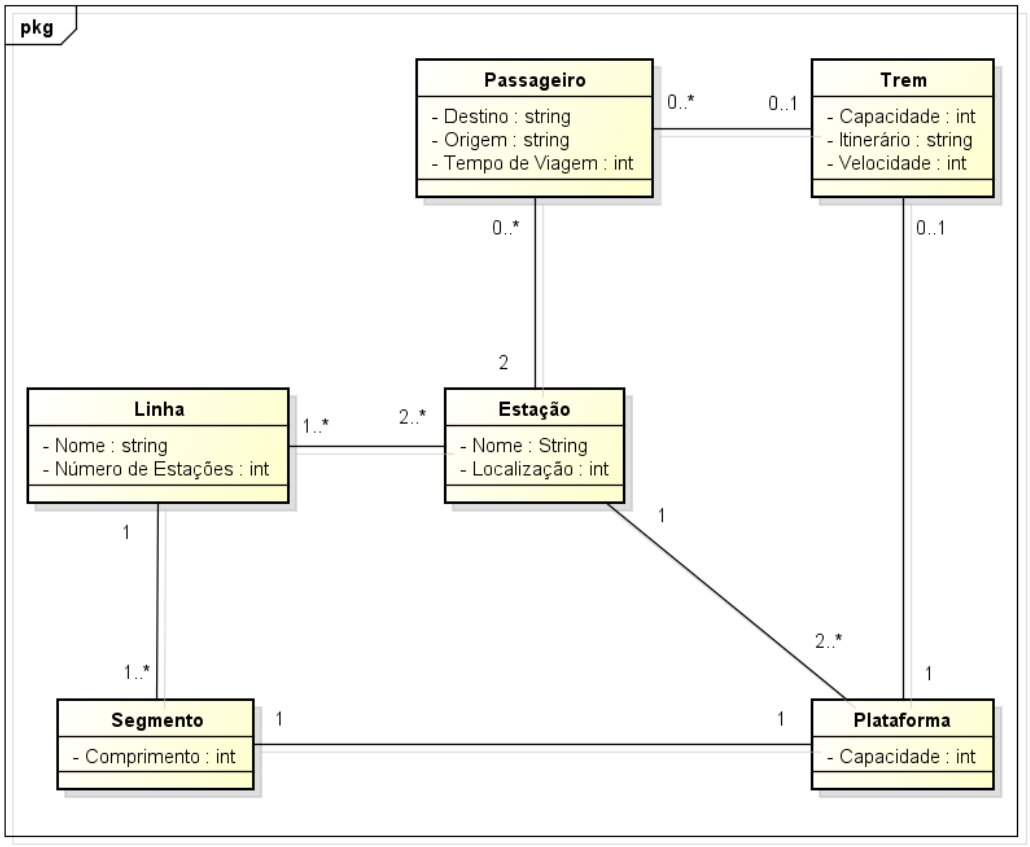

Figura 17 – Diagrama de Classes

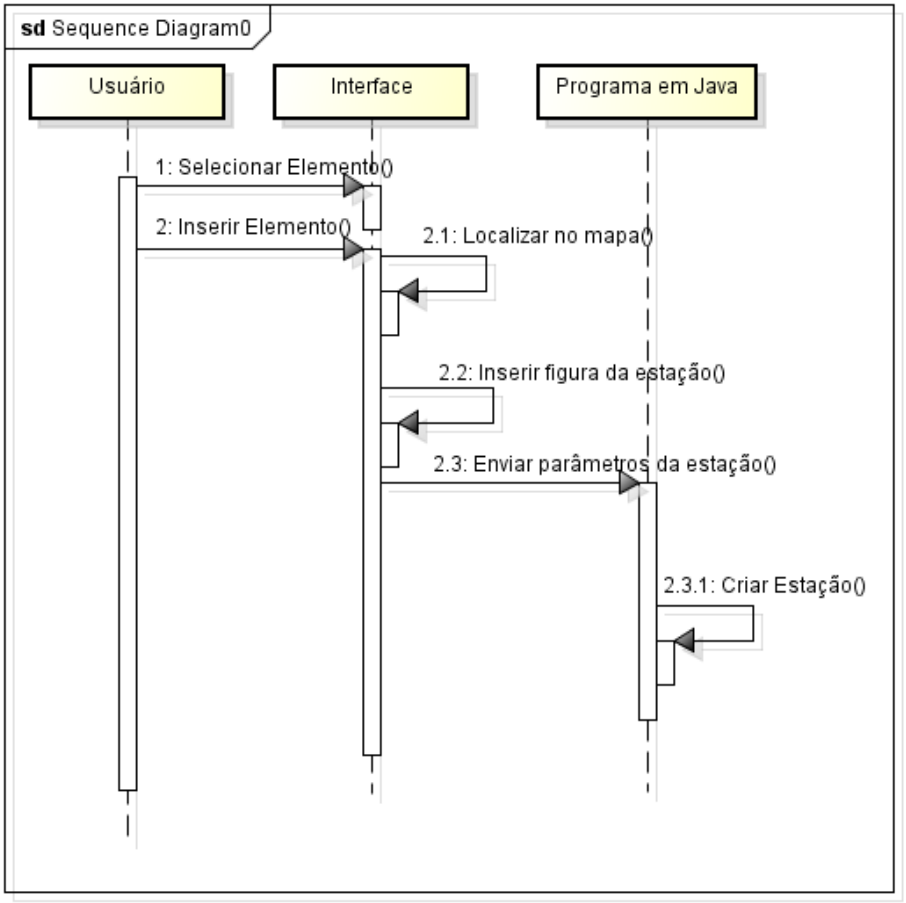

Figura 18 – Diagrama de Sequência - Inserir estação

B. Excluir Elemento

O diagrama representa a inclusão de um elemento, no caso este elemento é uma estação. A Figura 19 mostra a sequencia da exclusão de uma nova estação e como os objetos se relacionam.

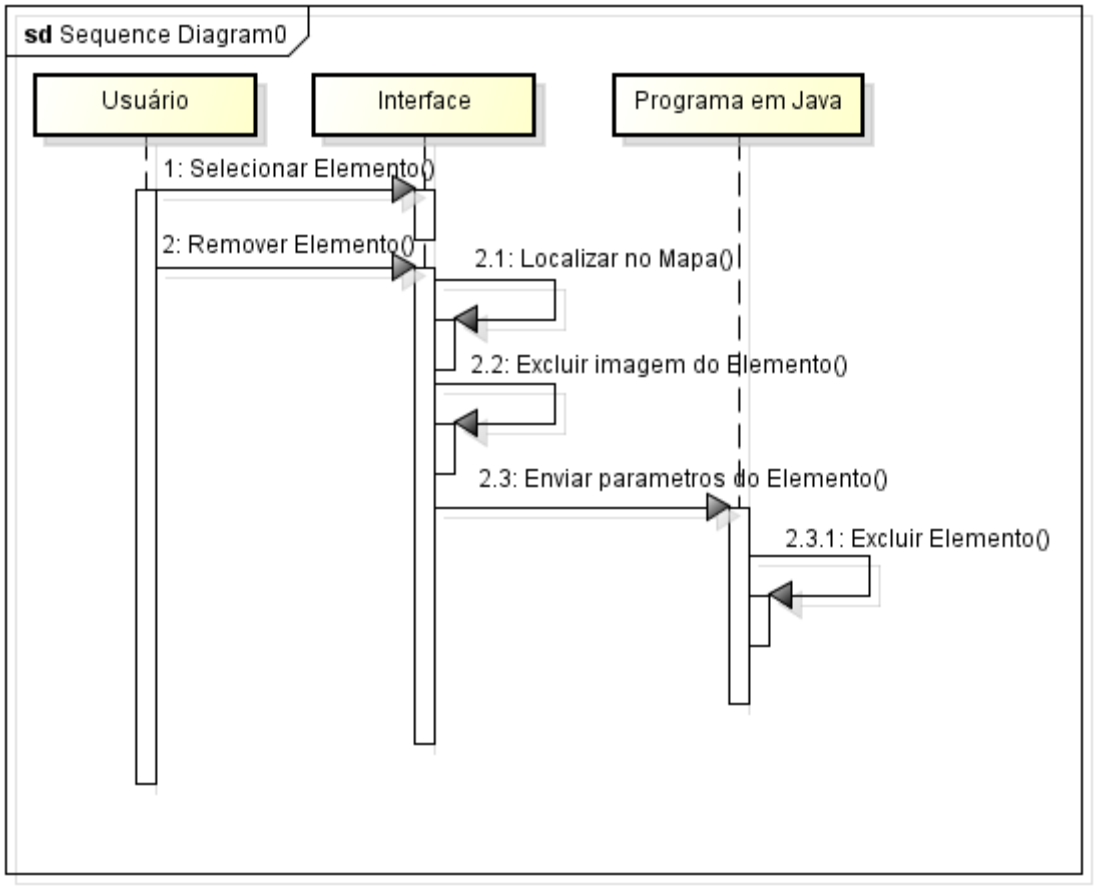

Figura 19 – Diagrama de Sequência - Excluir elemento

C. Exportar para o Simulador

Após a construção da malha ferroviária que se deseja simular, deve-se exportar a rede ao simulador com uma linguagem que o *software* de simulação consiga compilar e construir a rede. (Figura 20).

D. Verificar Distâncias

Durante a construção da rede de trens o programa calcula a distância entre duas estações adjacentes, como exemplificado na Figura 21.

E. Simular

Depois de exportar o modelo ao simulador, o usuário pode inicializar a simulação, (Figura 22).

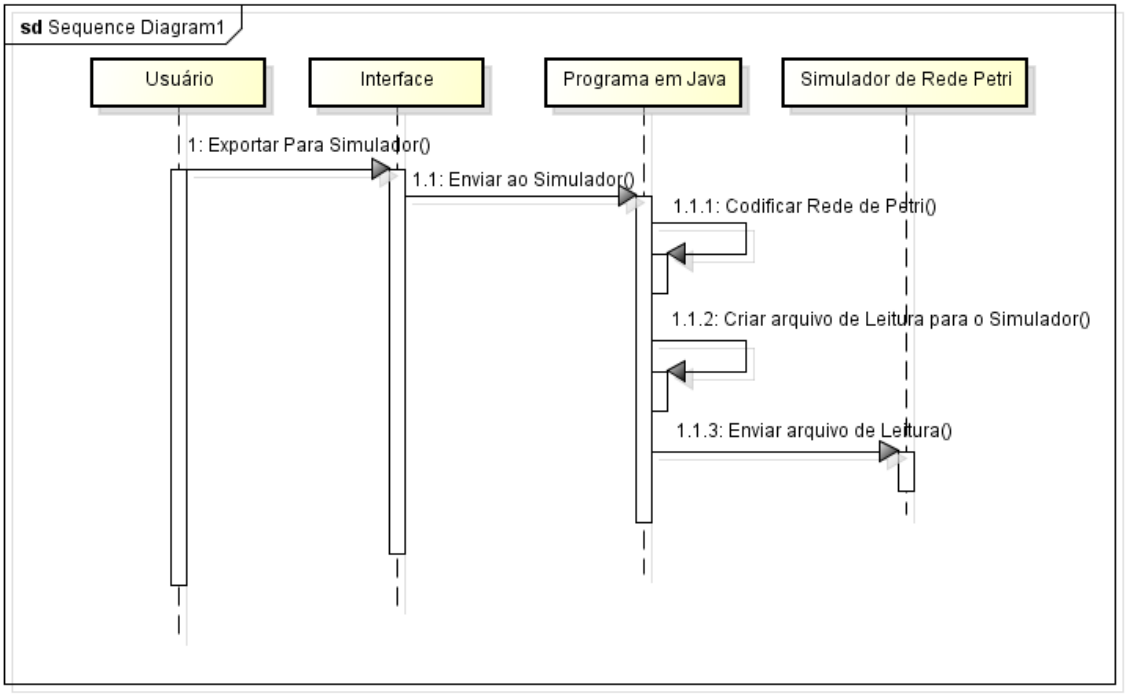

Figura 20 – Diagrama de Sequência - Exportar para o simulador

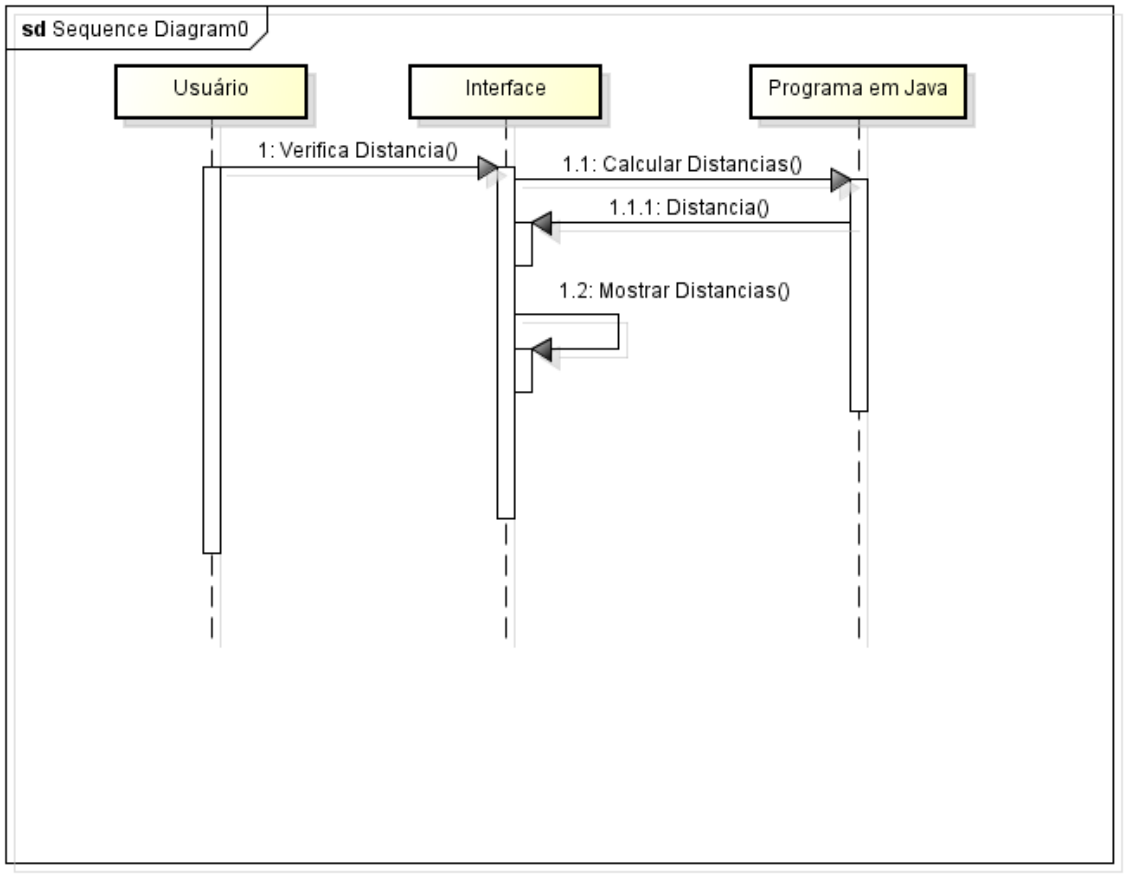

Figura 21 – Diagrama de Sequência - Verificar distância

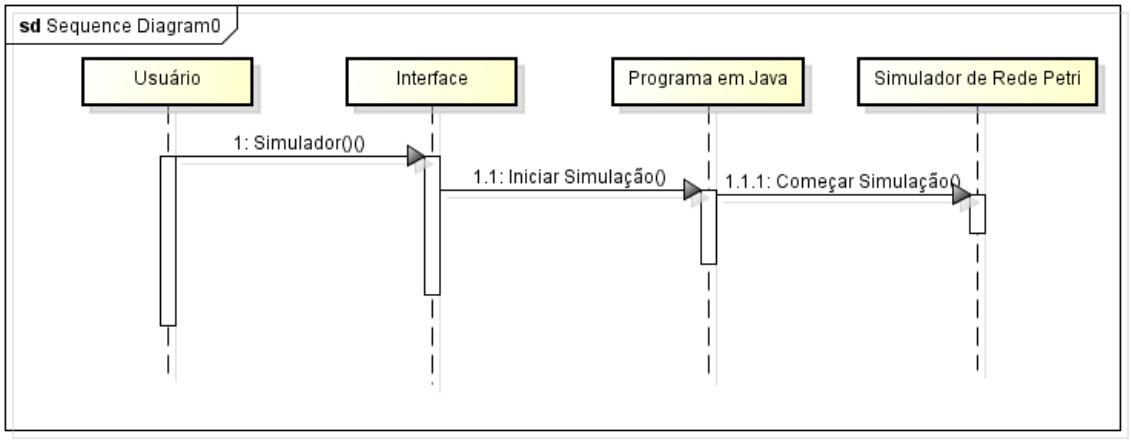

Figura 22 – Diagrama de Sequência - Simular

4. IMPLEMENTAÇÃO

A implementação do *software* pode ser dividida em duas etapas. A primeira consiste no desenvolvimento da interface com auxílio do Google Maps API e a segunda etapa consiste na implementação do construtor de XML que cria um arquivo de leitura compatível com o *software* CPN Tools.

4.1. INTERFACE

A interface é o componente intermediário do projeto. Ela conecta o usuário ao simulador. Ao executar o aplicativo, o navegador inicializa o mapa sobre o qual o usuário definirá a malha ferroviária que deseja construir e simular. O mapa é carregado diretamente de um servidor do Google. Portanto é necessário estar conectado à *Internet*, caso contrário o mapa não irá aparecer.

Na interface são inseridos os parâmetros de entrada, como a quantidade de passageiros que embarcam e desembarcam além da localização de cada estação que compõe a rede, esses parâmetros definirão a rede que será simulada.

Uma página em JSP foi desenvolvida para o projeto. Este formato permite a integração de duas linguagens voltadas ao desenvolvimento web, JavaScript e HTML, e outra, a linguagem *Java* que processa e envia os dados de entrada ao aplicativo, também em *Java*, que escreve o XML lido pelo simulador.

Segue o detalhamento da implementação do aplicativo.

4.1.1. CORPO HTML

O corpo ou *body* inicial do HTML ficou sucinto, pois muitos elementos são criados dinamicamente.

```
    <body onload="inicializar()">
      <input type="submit" onclick="AddPassageiros()"
value="AddPassageiros">
      <input type="submit" onclick="Distancia()" value="Distancia">
      <input type="submit" onclick="Remove()" value="Remover">
      <form method="post" action=index.jsp>
        <input type="submit" name="enviar" value="Enviar ao Simulador" />
        <input type="hidden" id="nEstacoes" name="nEstacoes"/>
        <input type="hidden" name="campo_controle1" />
```

```
      <div class="css_passageiros">
        <table id="Passageiros"></table>
      </div>
      <div class="css_distancia">
        <table id="Distancia"></table>
      </div>
    </form>
    <div class="css_mapa" id="map_canvas"></div>
</body>
```

Ao carregar o programa, o atributo *onload* chama a função “inicializar” que configura as os parâmetros iniciais, como o centro do mapa, o zoom, especificação da polilinha e outros detalhados posteriormente.

Os três *inputs* que seguem,“Addpassageiro”, “Distancia” e “Remover”, representam os botões que carregam as respectivas funções. Estes botões estão distribuídos na região superior, à esquerda da tela.

Na quarta linha do HTML é criado o formulário onde ficam armazenados os dados que serão exportados ao simulador. No formulário, há três elementos *input* e duas tabelas. O primeiro *input* cria o botão “Enviar ao Simulador”, que carrega a respectiva função. Em seguida, é criada uma variável onde é atualizado o número de estações. O último *input*, campo_controle1, é o parâmetro que define quando serão lidos os dados de entrada e enviados ao construtor XML. Em relação às tabelas, elas são apenas declaradas com seus respectivos atributos, mas sem as linhas e colunas que são adicionadas por meio do *JavaScript*, manipulando o DOM (*Document Object Model*). Note que as tabelas são criadas dentro de um elemento *div* que permite a utilização de CSS (*Cascading Style Sheets* - folha de estilo composta por camadas). O CSS auxilia na configuração das características visuais do HTML, criando blocos padrões de formatação. Segue, como exemplo, a especificação da tabela “*passageiros*”:

```
.css_passageiros {
    background-color: #9CC;
    left: 0%;
    position: absolute;
    top: 8%;
    visibility: hidden;
}
```

Finalmente, o último elemento refere-se ao carregamento do mapa na tela conforme configurado na inicialização. A Figura 23 ilustra a tela inicial da interface.

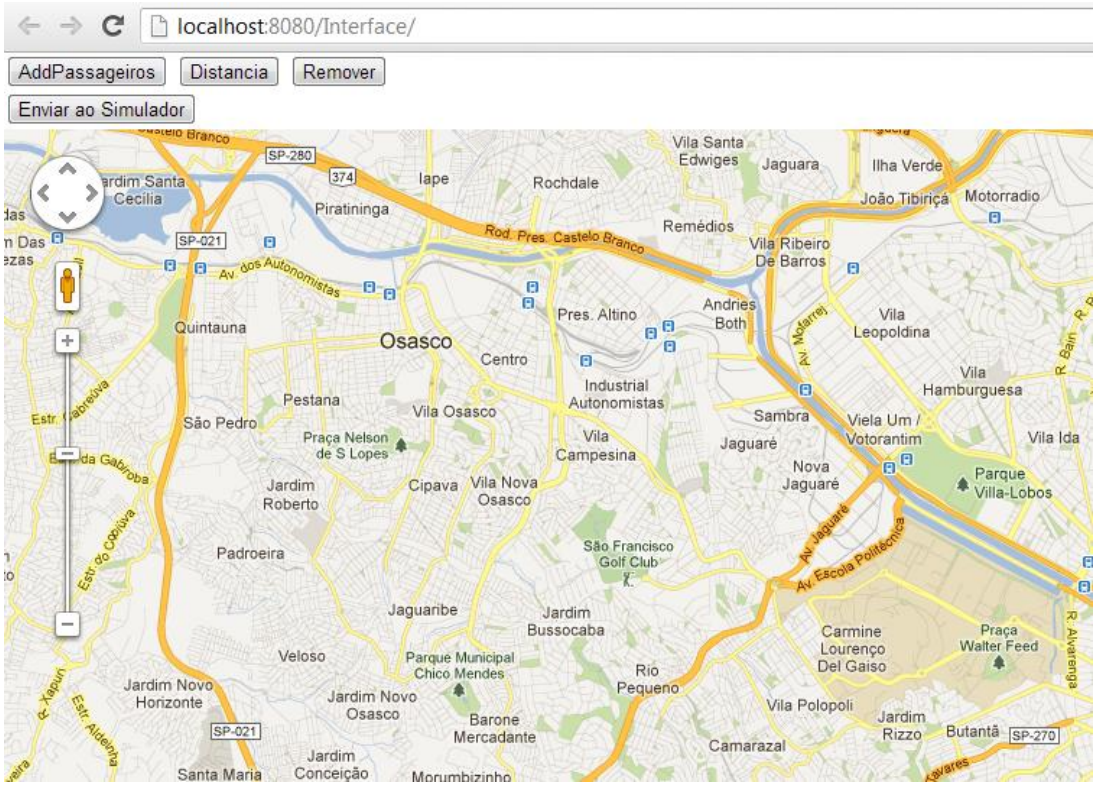

Figura 23 – Tela inicial

4.1.2. INICIALIZAR

"Inicializar()" é a função *JavaScript* que configura as inicializações da interface sendo carregada assim que o programa é executado.

No projeto, o mapa é carregado próximo à cidade universitária, ao inserir nas configurações do mapa a latitude e longitude e *zoom*. Nesta função inicial, a polilinha, que representa o conjunto de nós conectados por seguimento de reta, é especificada, definindo-se sua espessura e cor.

Por fim, é adicionado um *listener* que responde às interações do usuário no mapa. Assim que o usuário pressionar o *mouse*, em qualquer região do mapa, será chamado um método que insere uma estação e realiza o cálculo da distância entre as estações. Segue o trecho do código que inclui o *listener*.

```
google.maps.event.addListener(map, 'click', addEstacao);
```

O primeiro elemento é o mapa, que se refere à região sensível ao evento. O segundo é o evento a ser “escutado” que no caso é o *click* do *mouse* na região sensível, último o atributo é a função que é disparada assim que o evento ocorre.

4.1.3. ADICIONAR ESTAÇÃO

Na interface, esta é a função que efetivamente desenha a malha ferroviária e realiza os cálculos necessários.

Primeiramente, são obtidas a latitude e a longitude do local escolhido pelo usuário e inserido uma superposição no mapa, ou seja, um ícone que representa a própria estação. Ao passar o *mouse* sobre esse marcador, um evento é disparado informando o número da estação. Abaixo segue a codificação, no vetor *x* é armazenada a posição e a variável *marker* representa objeto que é o marcador.

```
x[i]=event.latLng;
var marker = new google.maps.Marker({
  position: x[i],
  title: 'Estação ' + path.getLength()
});
```

Após a inserção da segunda e demais estações, um seguimento de reta é incluído conectando os elementos adjacentes formando a linha ferroviária. Este conjunto de nós ligados por seguimentos de reta dá-se o nome de polilinhas, cujos atributos, latitude e longitude, são armazenados em um vetor.

Baseado nesses parâmetros é calculada a geodésica, que representa a menor distância entre dois pontos em um plano deformado como o terrestre, utilizando uma função da biblioteca *Geometry* do Google Maps API. Na parte do código ilustrado abaixo, é exemplificado o cálculo e armazenamento da distância entre dois nós adjacentes. O objeto *path* é uma polilinha, e o método *getAt* retorna a Latitude e Longitude que é utilizada como parâmetro de entrada da função que calcula a geodésica.

```
distancia =
google.maps.geometry.spherical.computeDistanceBetween(path.getAt(j),
path.getAt(j-1));
```

Em seguida, é inserido no formulário, criado no corpo do HTML, os elementos *inputs*, que são campos de entrada, onde são armazenados dados referentes ao embarque, desembarque e a distância entre duas estações. Esses elementos são criados por meio da manipulação do DOM (*Document Object Model* - Modelo de Objetos de Documentos) que permite alterar e editar dinamicamente a estrutura do HTML. Dessa forma, a quantidade de elementos varia conforme a rede ferroviária criada pelo usuário: quanto mais estações mais linhas.

Os *inputs* são inseridos em duas tabelas, uma que representa a entrada e saída de passageiros e, na outra, o comprimento do seguimento que liga as estações, em metros. O preenchimento desses campos é realizado alterando seus atributos. Assim, o nome, o valor e a visibilidade são os atributos manipulados. O primeiro representa a identificação do campo, que permite o acesso a qualquer momento para realização de leitura e escrita. O segundo é o próprio valor, sendo a distância calculada pelo programa, como descrito anteriormente, e o embarque e desembarque que são inseridos pelo modelista da rede de trens. Quanto à visibilidade, ela é inicialmente atribuída como invisível para facilitar a construção da malha ferroviária no mapa. As tabelas estão ocultas no mapa, e se tornam visíveis ao selecionar a opção “AddPassageiros” e/ou “Distancia” no canto superior esquerdo da tela (Figura 24).

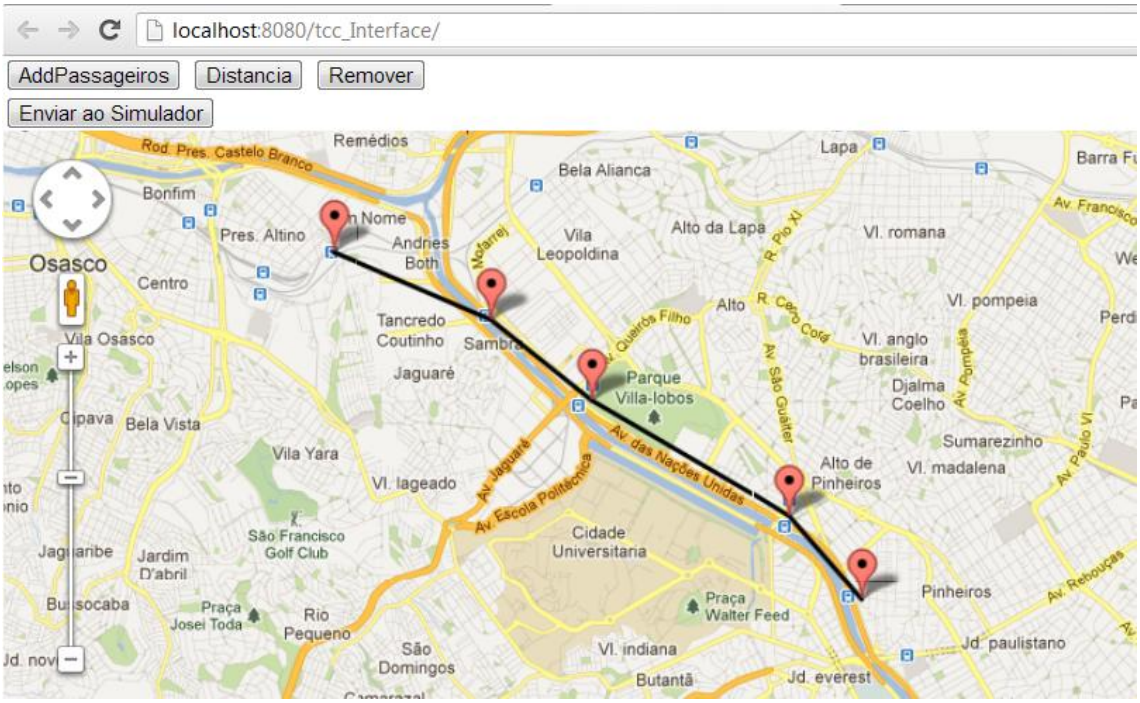

Figura 24 – Linha construída

4.1.4. ADICIONAR PASSAGEIROS

Esta função é responsável por mostrar a tabela de passageiros, que inicia oculta. Ao pressionar este botão, o atributo da tabela que define a visibilidade é alterado para visível e o mapa é deslocado para esquerda, como segue o código:

```
tabela[0].setAttribute("style","visibility:visible");
pos_mapa[2].setAttribute("style","position:absolute;    top:8%;left:0%;
width:100%; height:100%;");
```

Na lista que surge, cada linha possui dois campos de entrada, um de embarque e o outro de desembarque, onde é necessário inserir a quantidade de passageiros que entram e saem em cada estação por hora, conforme a Figura 25. Quando uma nova estação é criada, mais uma linha é inserida dinamicamente na página, possibilitando a inclusão ilimitada de estações no modelo.

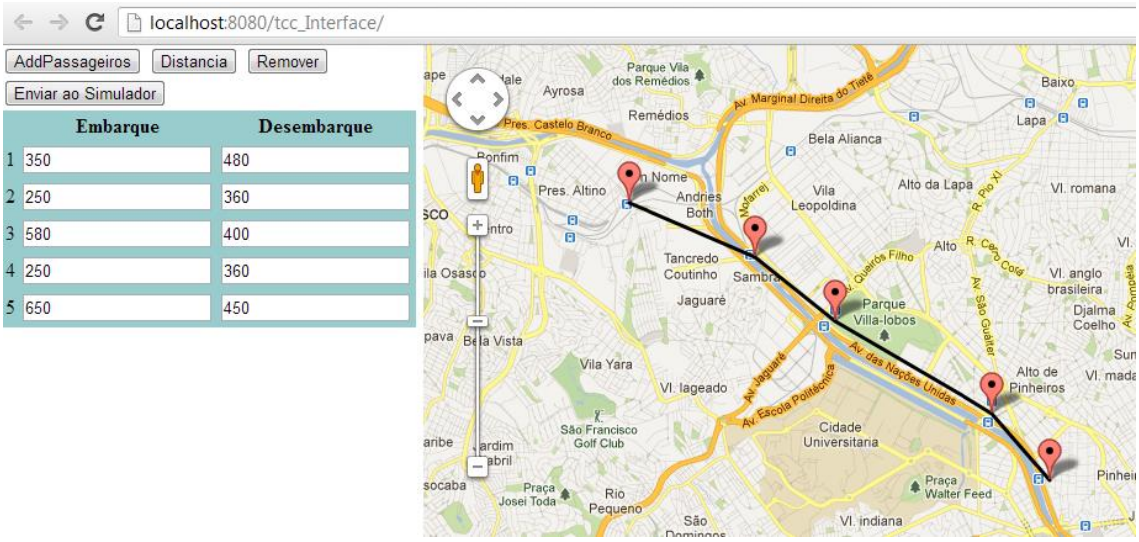

Figura 25 – Dados de Embarque e Desembarque

4.1.5. DISTÂNCIA

Como a função anterior, ao ser pressionado o botão “Distancia” o atributo visibilidade é alterado, então a tabela que estava oculta se torna visível. A Figura 26 ilustra como é apresentada as distâncias entre as estações, calculadas anteriormente. A primeira coluna identifica as estações que estão conectadas e a segunda o comprimento, em metros, do seguimento gerado.

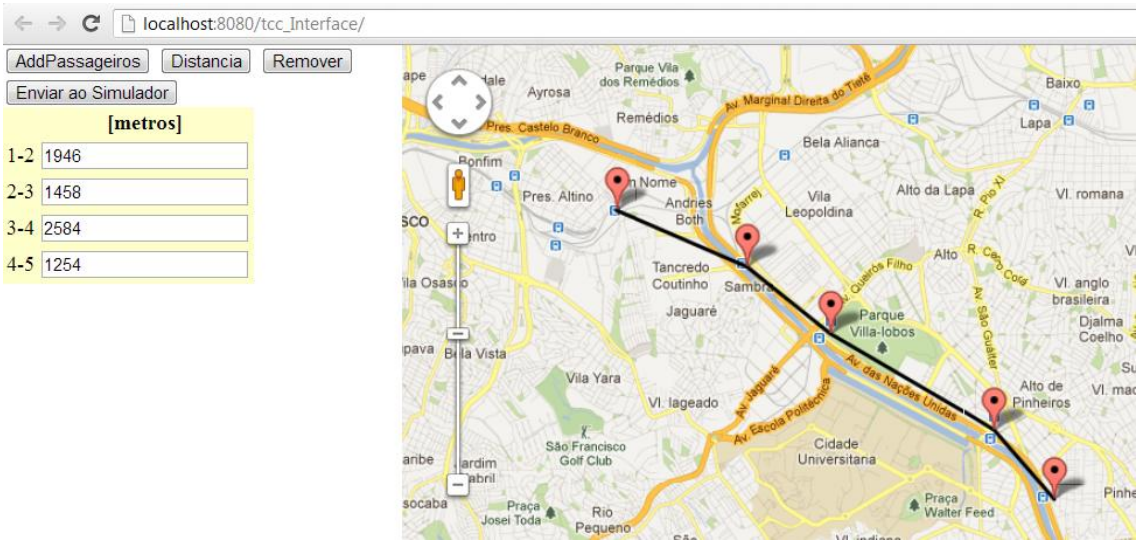

Figura 26 – Distâncias

4.1.6. REMOVER ELEMENTO

Esta função elimina estação, e consequentemente os passageiros daquela estação, escolhida pelo usuário. Ao selecionar a opção “Remover” surge uma janela, Figura 27, que permite digitar qual estação será eliminada.

O comando JavaScript *prompt* cria uma nova janela que permite inserir dinamicamente algum parâmetro sem a necessidade de carregar a página novamente. Desta forma, são mantidas as estações anteriormente criadas retirando apenas aquela que foi digitada.

No trecho do código transcrito à baixo é criado uma variável auxiliar, *aux*, que armazena o número digitado e em seguida retira o nó da polilinha.

```
aux=prompt("Digite a estacao a ser removida","1");
path.removeAt(aux);
```

Assim a rede ferroviária é redesenhada excluindo apenas o nó selecionado.

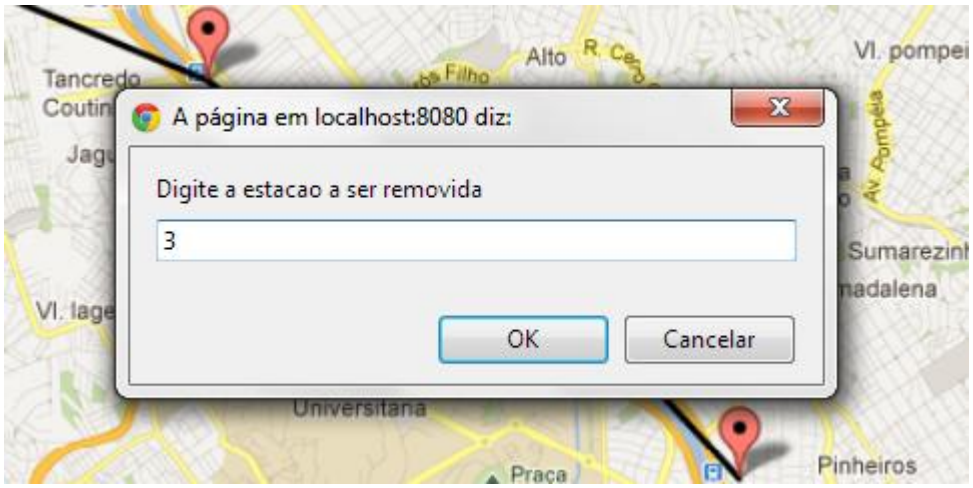

Figura 27 – Excluir estação

4.1.7. ENVIAR AO SIMULADOR

O botão “Enviar ao Simulador” transfere as informações inseridas no modelo ao construtor do arquivo de texto lido pelo simulador.

Primeiramente, os parâmetros da malha ferroviária, que foram criados pelo usuário e armazenados em formulários HTML, são lidos e inseridos em vetores. Após processar os dados de entrada, são enviados os seguintes vetores para o construtor XML: o vetor distância, que contem os comprimentos de todos os seguimentos presente na rede; o vetor embarque e o vetor desembarque, ambos referentes, respectivamente, a entrada e saída de passageiros.

4.2. CONSTRUTOR XML

Neste capítulo, é descrito como é realizada a construção do arquivo XML que será lido pelo simulador CPN Tools.

O Construtor é uma classe chamada pela interface assim que a malha ferroviária está pronta para ser simulada. Os parâmetros de entrada são o número de estações e os vetores de embarque, desembarque e distâncias.

Esta classe cria um vetor de *Strings* onde são inseridas as linhas do XML. Após concluir a criação do vetor, ele é impresso em um arquivo que pode ser lido pelo simulador de Rede de Petri Colorida.

O arquivo XML é criado com base nos modelos padrões de estações apresentados na parte de modelagem em Rede de Petri. Assim, para cada estação inserida no mapa, o código correspondente à uma estação é inserido ao XML e relacionado à quantidade de trechos existentes entre as estações. O mesmo ocorre para os trechos, cuja quantidade é calculada pela distância entre estações.

Cada elemento da rede deve possuir um identificador único, logo foi criada uma regra que os padronizam. O identificador padrão possui três números, a Figura 28 exemplifica a regra de definição dos *ids*, a figura representa a primeira estação com apenas um seguimento. Observe que não foi inserido o *id* dos arcos para não poluir a imagem

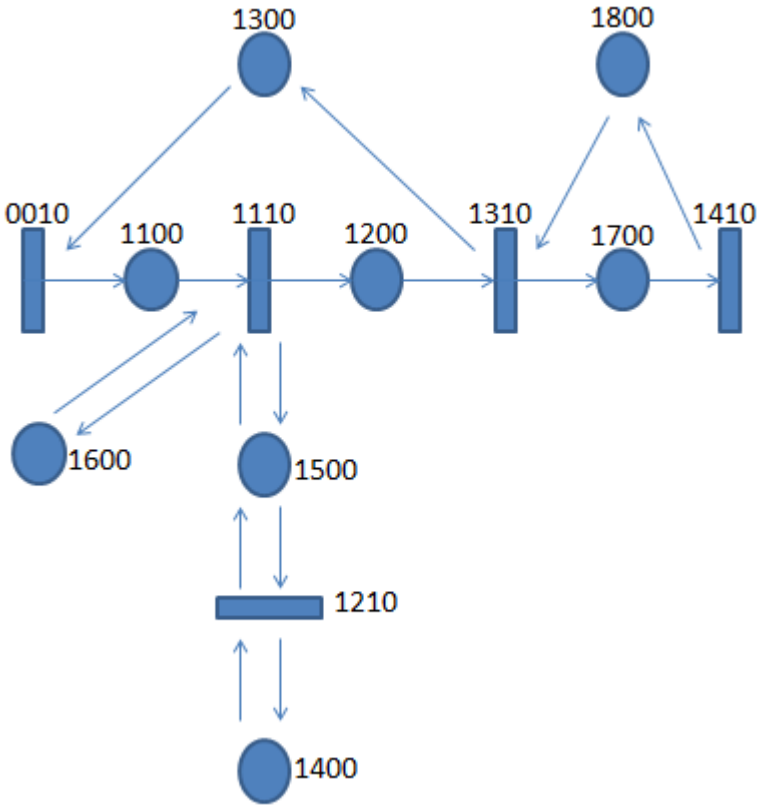

Figura 28 – Lógica para construção dos modelos

O último dígito representa a direção que o trem está seguindo. Atribui-se o índice zero ao elemento relacionado à ida do trem, e um a volta.

Note que o terceiro dígito se refere ao tipo de elemento: o índice 0 representa o lugar, 1 a transição e 2 o arco.

O dígito intermediário refere-se à numeração padronizada de cada elemento ilustrada na Figura 29. A numeração dos elementos no interior da estação é sempre a mesma, no entanto a numeração dos elementos externos varia proporcionalmente à quantidade de seguimentos que une uma estação a outra.

O primeiro dígito representa a estação em que o elemento está incluído. Foi definido que o seguimento pertence à estação de partida do trem, assim os lugares, as transições e os arcos dos seguimentos que unem as estações 3 e 4, por exemplo, pertencem a estação 3. Outra definição: atribui-se o índice zero ao primeiro dígito do elemento garagem.

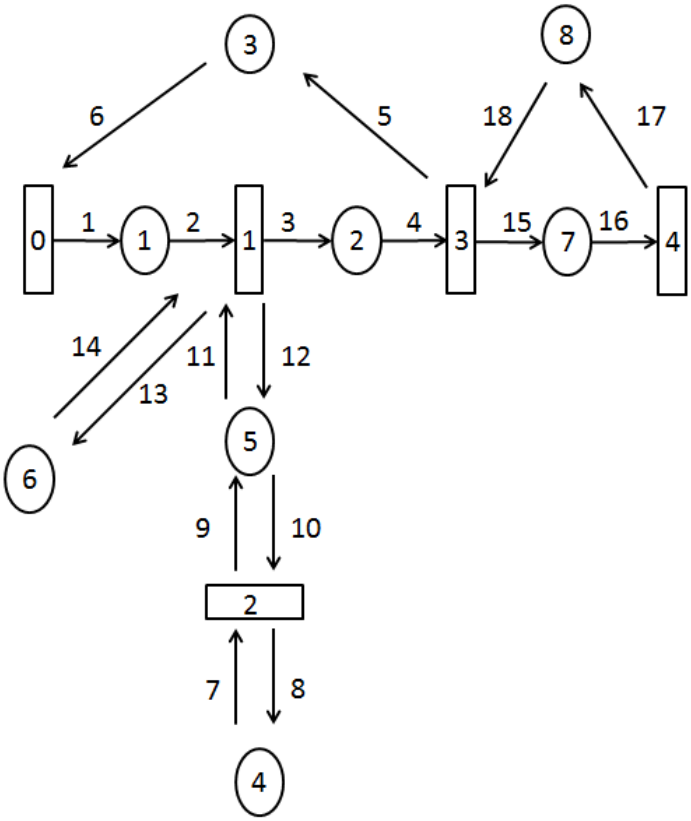

Figura 29 – Numeração dos elementos

Além dos identificadores a posição espacial é outro atributo que está presente nos lugares e nas transições. A posição espacial auxilia apenas na visualização da rede no *software* de simulação. O modelo é desenhado em um plano que possui o eixo da abscissa x e o da coordenada y. A Figura 30 mostra a separação de cada elemento seguindo a escala do próprio simulador. Dessa forma, os lugares e as transições inseridas no eixo x são distanciados à 150 unidades de comprimentos um do outro, enquanto no eixo y são distanciados à 100 unidades de comprimentos.

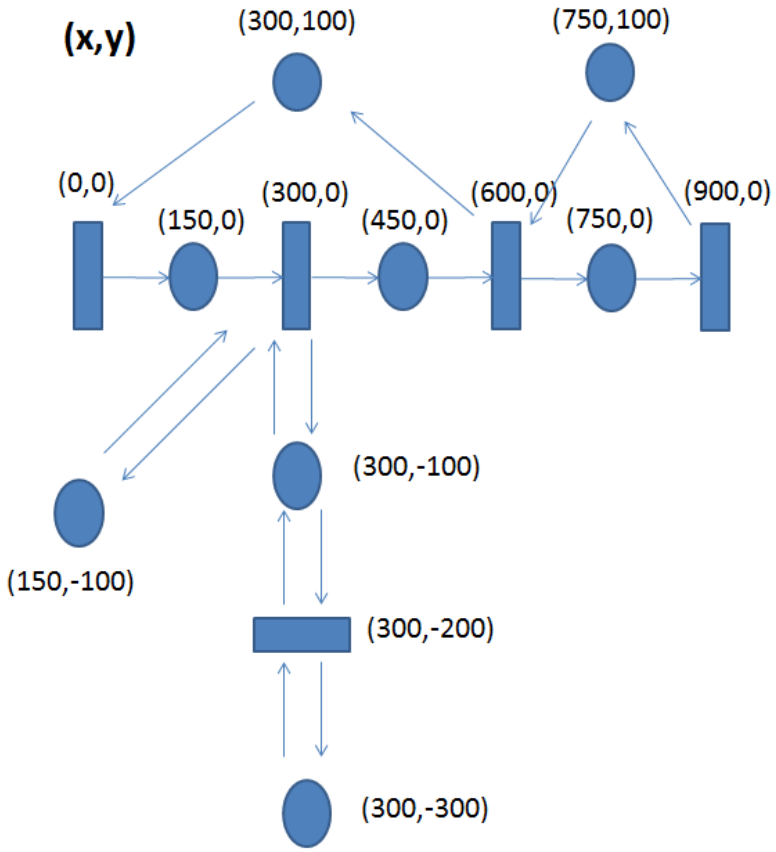

Figura 30 – Posição espacial dos elementos

5. RESULTADOS

Neste item, foi inserido um exemplo que representa a modelagem de linha de trem que conecta as, já existentes, estações de Pinheiros e Butantã.

Inicialmente é carregada a interface que exibe a tela inicial. Assim que carregada, foram inseridas duas estações como ilustrada na Figura 31.

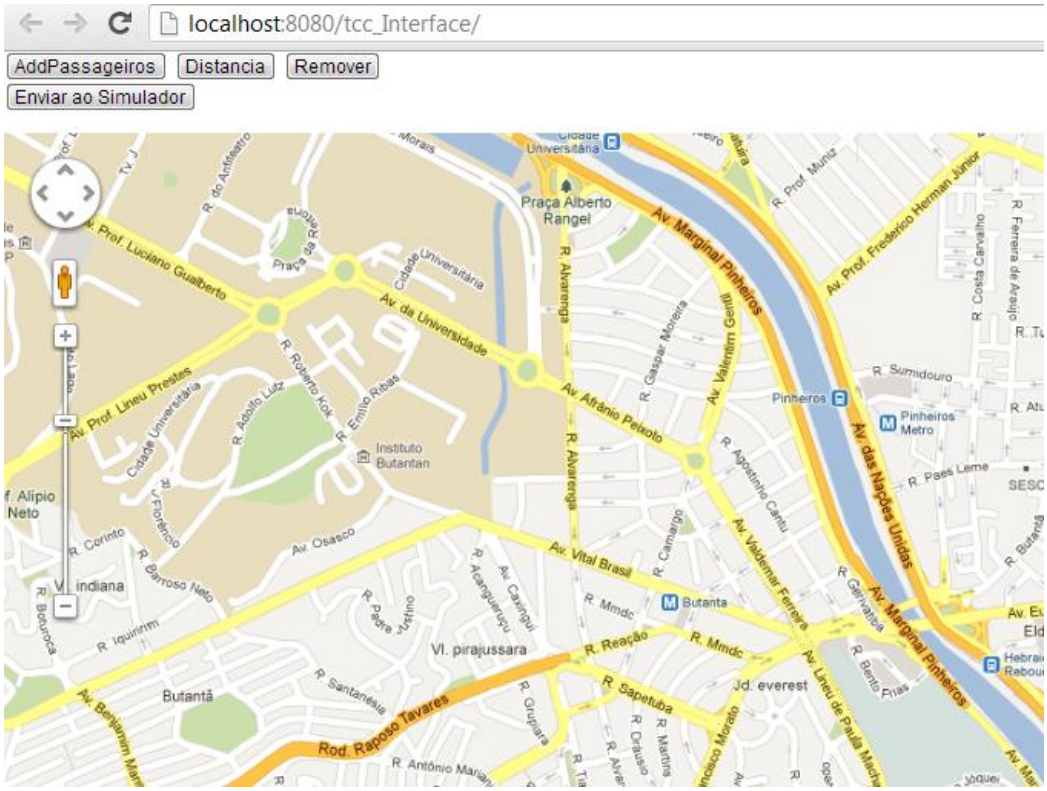

Figura 31 – Interface inicial

A estação à esquerda foi criada primeiramente, e recebe o identificador Estação 1. Em seguida foi criada a Estação 2 e automaticamente uma linha une as duas estações.

Para obter o comprimento do seguimento, o botão “Distancia” é selecionado. Então, o mapa é deslocado à direita e a tabela com o comprimento é exibida. O programa calcula a distância entre as estações, em metros, e insere na segunda coluna na tabela, Figura 32.

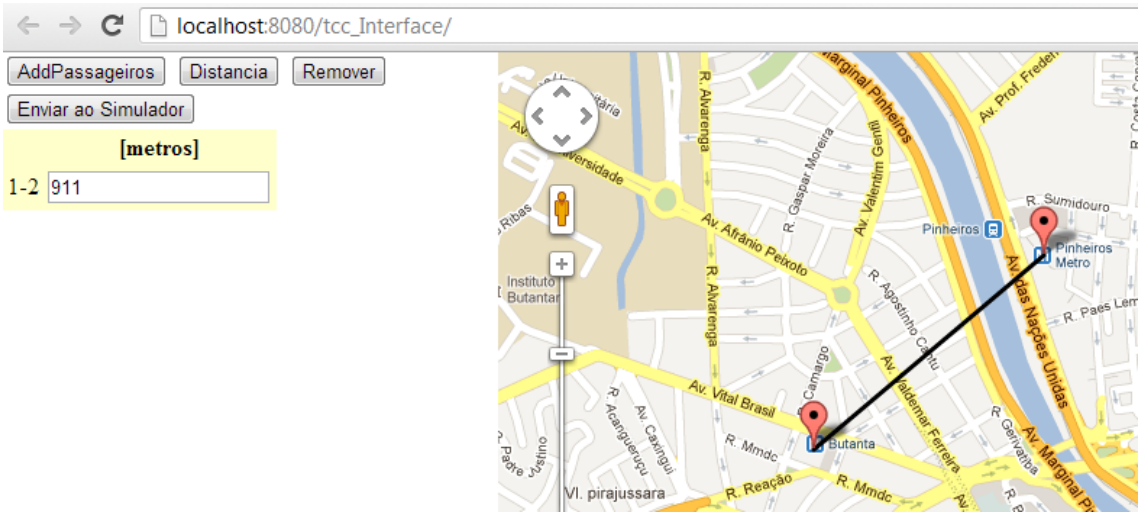

Figura 32 – Distância

Em seguida, são inseridos os passageiros. Ao selecionar a opção “AddPassageiros”, uma tabela com fundo azul é exibida, Figura 33. A tabela possui duas linhas referentes às duas estações, nelas foi inserida a quantidade de passageiros que embarca de desembarca na respectiva estação.

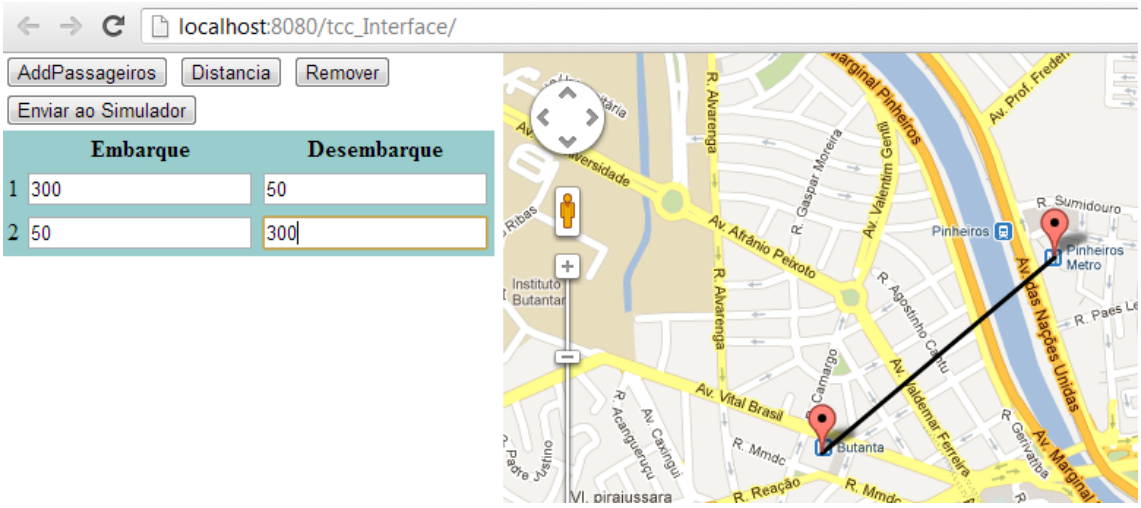

Figura 33 – Inserir passageiros

Realizada a construção da rede ferroviária e inseridos os dados de entrada, é selecionado a opção “Enviar ao Simulador”. Assim, a malha ferroviária é transformada em um arquivo XML que pode ser lido pelo *software* de simulação.

Abrindo o arquivo XML gerado, tem-se o modelo da rede de trens em Rede de Petri Colorida, conforme pode ser visto na Figura 34.

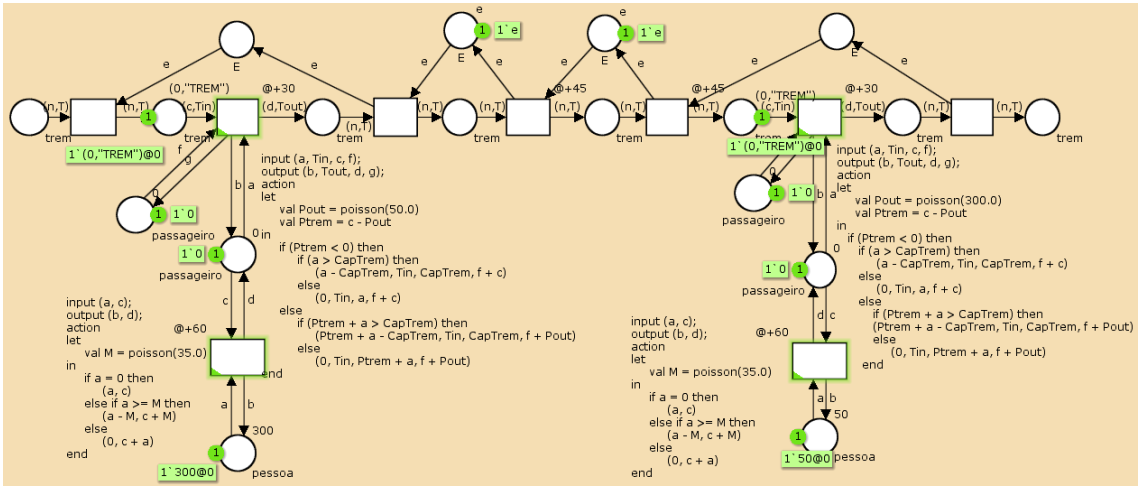

Figura 34 – Modelo gerado

Simulando o modelo gerado, obtém-se a configuração da Figura 35.

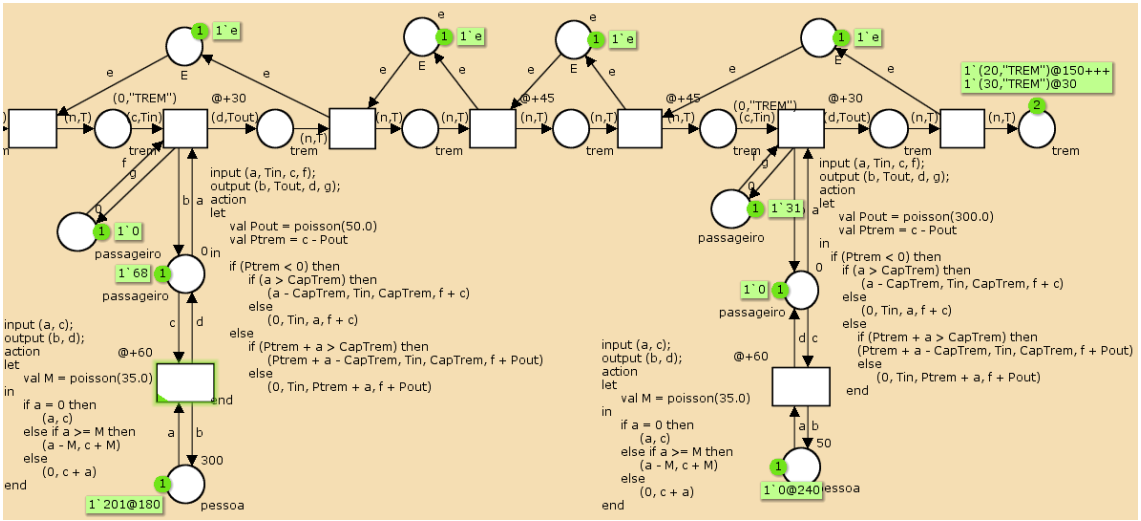

Figura 35 – Configuração após simulação

6. CONCLUSÃO

O principal objetivo deste projeto foi desenvolver uma ferramenta amigável que permitisse, a um usuário sem conhecimento prévio em modelagem, criar um modelo, em Rede de Petri, da malha ferroviária urbana. Operando por meio da interface no *Google Maps*, o usuário pode criar seu modelo da malha ferroviária. Para tanto, ele deve escolher os locais das estações e definir os parâmetros de entrada: embarque e desembarque de passageiros, sem necessitar dos conhecimentos teóricos sobre Rede de Petri. O usuário também pode simular seu modelo, contanto que tenha o programa CPN Tools instalado em seu computador. Para executar a simulação, o usuário necessita aprender a abrir os arquivos e executá-los, podendo extrair muitos resultados sobre o comportamento do sistema visualmente durante o tempo da simulação.

A Rede de Petri Colorida foi escolhida para a modelagem, pois possui um alto nível de abstração permitindo a criação de um modelo mais completo. Este tipo de rede permite a associação de atributos as marcas e a diferenciação das mesmas através de cores. Esta característica permitiu a representação do fluxo de passageiros no modelo, permitindo a modelagem de filas. Dessa forma também é possível verificar o impacto nos usuários quando da modificação de alguma característica nas linhas, bem como quando de alterações nas velocidades e frequência dos trens.

Quanto à modelagem em UML do programa implementado para a transcrição do modelo gerado no *Google Maps* para rede de Petri colorida, ela está estruturada de forma concisa e possibilitou uma visualização global necessária para o início da codificação do programa, assim as definições dos objetos e seus respectivos atributos foram listados facilitando a implementação do programa.

A escolha do *Google Maps* API como base da interface de desenvolvimento foi muito eficiente por três motivos. Primeiro, devido à sua versatilidade. A rede ferroviária pode ser criada tanto em Recife, como no Rio Grande do Sul ou em qualquer região do mundo. Assim, não possui restrição quanto à localização e nem mesmo quanto ao comprimento da linha analisada, que pode ser a ligação de estações em um mesmo município, como é o caso do Metrô de São Paulo, ou mesmo uma rede interestadual, como será o futuro trem que unirá São Paulo ao Rio de Janeiro. Segundo, insere um caráter genérico que permite extensão a um projeto que integre outros sistemas, como a rede de ônibus urbanos. Esta integração permitiria ao desenvolvedor do projeto ter uma visão macro do transporte público da cidade, auxiliando a visualização dos gargalos existentes na rede. Terceiro, é uma ferramenta muito difundida

entre os usuários de *Internet*, logo não seria necessário um treinamento complexo quanto ao funcionamento do aplicativo, além de ter seu mapa constantemente atualizado.

Como qualquer modelo, há espaços para melhorias no sentido de se aproximar mais à realidade. Uma das simplificações foi quanto à linha de trem. A linha, no projeto é uma semirreta que conecta uma estação à outra, porém o correto seria acrescentar a possibilidade da linha realizar curvas. No projeto, uma estação interage apenas com as duas adjacentes, de uma mesma linha. No entanto, não é incomum uma estação ser compartilhada com diferentes linhas e, desta forma, ter que interagir com mais do que duas estações adjacentes. No modelo, a distribuição de passageiros e a frequência de trens é a mesma, tanto nos horários de pico, quanto nos de vale. Para refinar o modelo, esta distribuição poderia ser diferenciada.

Dessa forma, pôde-se verificar que o projeto proposto consegue facilitar a modelagem da rede de trens e pode ser utilizada sem a necessidade de grandes conhecimentos teóricos sobre a Rede de Petri. A ferramenta também é útil para os usuários familiarizados com o tipo de modelo proposto, pois permite criar modelos de forma mais rápida que manualmente.

REFERÊNCIAS BIBLIOGRÁFICAS

AALST, W. M. P. van der; ODIJK, M. A. *Analysis of Railway Stations by Means of Interval Timed Coloured Petri Nets*. **Real-Time Systems**, Vol. 9, No. 3, pp. 241-263, 1995.

CARDOSO, J.; VALETTE, R. **Redes de Petri**. Editora da UFSC, 1997.

CASTELAIN, E.; MESGHOUNI, K. *Regulation of a Public Transport Network with Consideration of the Passenger Flow: Modeling of the System with High-Level Petri Nets*. In. IEEE International Conference on Systems, Man and Cybernetics, Vol. 6, 2002.

CPTM. **Sitio institucional da CPTM**, 2011. Disponível em: <http://www.cptm.sp.gov.br/e_companhia/cptm.asp> Acesso em: 20/04/2012.

FOLHA. **Após pane, linha 7-rubi da CPTM tem circulação restabelecida**, 2012. Disponível em: <http://www1.folha.uol.com.br/cotidiano/1069028-apos-pane-linha-7-rubi-da-cptm-tem-circulacao-restabelecida.shtml> Acesso em: 20/04/2012.

GOOGLE. ***Google Maps Javascript API V3***, 2012. Disponível em: <http://code.google.com/intl/en/apis/maps/documentation/javascript/basics.html>. Acesso em: 20/04/2012.

JENSEN, K.; KRISTENSEN, L. M.; WELLS, L. **Coloured Petri Nets and CPN Tools for modeling and validation of concurrent systems.** Springer-Verlag, 2007.

MARRANGHELLO, N. **Redes de Petri: Conceitos e Aplicações**. UNESP, 2005. Disponível em: <http://www.dcce.ibilce.unesp.br/~norian/cursos/mds/ApostilaRdP-CA.pdf>

METRÔ. **Sitio Institucional do Metrô**. Disponível em: <http://www.metro.sp.gov.br/metro/institucional/quem-somos/index.aspx> Acesso em: 20/04/2012.

MIYAGI, P. E. **Controle Programável**. Editora Edgard Blücher LTDA, 1996.

OMG. ***Unified Modeling Language - Superstructure***, 2005. Disponível em: <http://www.omg.org/spec/UML/2.0/Superstructure/PDF/> Acesso em: 15/04/2012.

PEREIRA, P. ***Usuários de trens e metrô/SP crescem mais de uma Campinas***, 2012. Disponível em: <http://www.estadao.com.br/noticias/geral,usuarios-de-trens-e-metrosp-crescem-mais-de-uma-campinas,861371,0.htm> Acesso em: 15/04/2012.

SILVA, J. B. **Alckmin promete mais 126 km de metrô em São Paulo até 2018**, 2012. Disponível em: <http://www1.folha.uol.com.br/cotidiano/1078302-alckmin-promete-mais-126-km-de-metro-em-sao-paulo-ate-2018.shtml> Acesso em: 19/04/2012.

WIKIPEDIA. **Figura do METRÔ de São Paulo**. 2012. Disponível em: <http://pt.wikipedia.org/wiki/Ficheiro:Metr%C3%B4_SP.png> Acesso em: 20/04/2012.

ANEXO A – DADOS TÉCNICOS E ESTATÍSTICOS DA CPTM

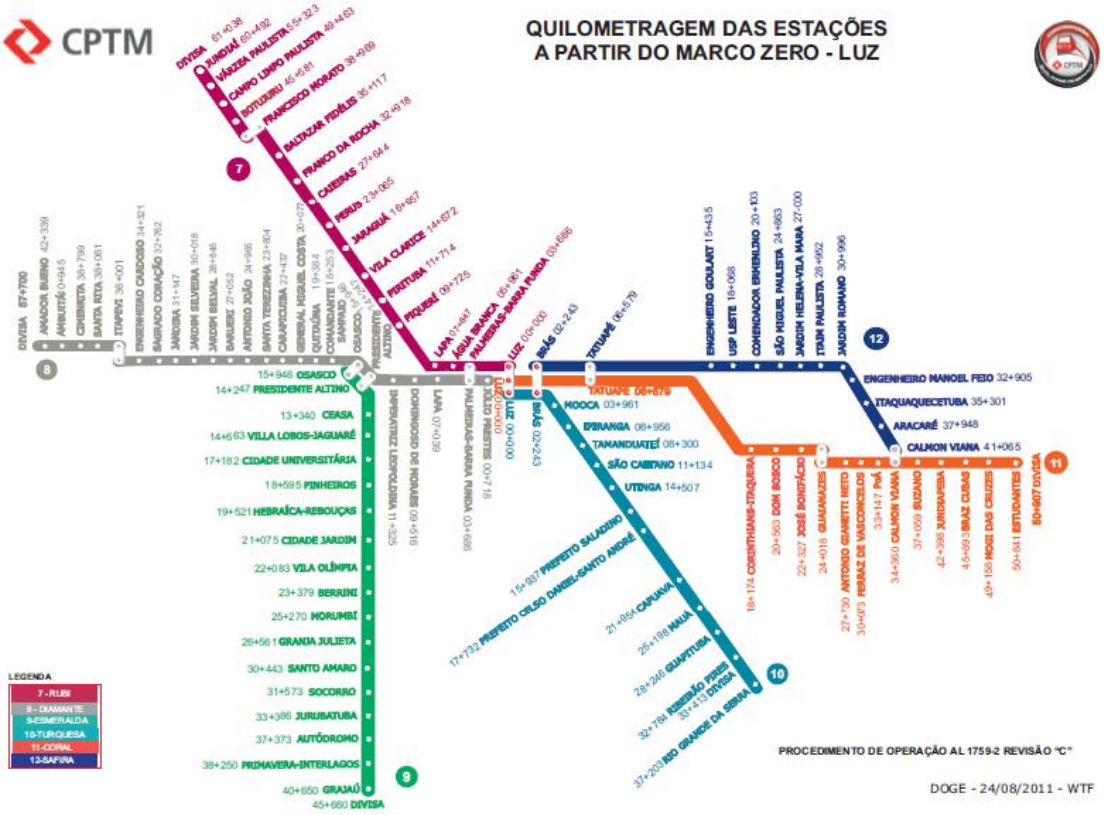

Figura 36 – Distância entre estações em Relação a estação da Luz (CPTM, 2011)

Tabela 3 – Embarque de passageiros 2011 Linha 7 (CPTM, 2011)

ESTAÇÕES	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ	TOTAL
Luz	1.083.366	1.125.989	1.210.974	1.173.641	1.252.048	1.125.583	1.170.788	1.237.095	1.176.213	1.316.621	1.321.979	1.328.131	14.522.428
Barra Funda	1.338.184	1.433.701	1.581.189	1.493.020	1.570.595	1.332.296	1.376.656	1.525.563	1.417.294	1.353.709	1.349.768	1.474.425	17.246.400
Água Branca	156.234	185.396	206.531	198.446	220.253	182.053	181.071	220.513	213.886	212.382	209.617	190.780	2.377.162
Lapa	733.824	786.002	833.878	820.308	902.943	807.525	843.774	885.853	829.711	833.310	833.334	893.695	10.004.157
Piqueri	82.410	88.811	96.483	95.084	104.496	90.886	94.394	102.896	100.007	100.102	99.149	98.881	1.153.599
Pirituba	265.454	284.854	315.393	303.365	329.648	286.552	298.964	329.245	317.657	325.214	322.076	325.887	3.704.309
Vila Clarice	80.051	85.207	94.780	91.457	98.027	85.579	87.844	92.436	87.766	87.124	86.017	92.278	1.068.566
Jaraguá	380.586	399.765	440.387	434.697	468.181	420.029	438.222	465.742	465.568	469.463	450.828	468.091	5.301.559
Perus	447.566	473.923	518.095	499.510	537.918	481.874	507.148	542.002	530.140	529.932	511.490	532.207	6.111.805
Caieiras	249.852	274.238	298.833	293.720	314.369	275.805	286.892	318.280	309.324	304.768	295.449	298.523	3.520.053
Franco da Rocha	506.426	578.112	627.066	605.062	649.338	580.017	606.591	650.825	630.392	629.317	611.010	636.104	7.310.260
Baltazar Fidélis	185.549	213.806	227.932	217.393	234.368	210.930	222.277	237.905	227.014	223.027	215.843	220.408	2.636.452
Francisco Morato	759.621	864.268	930.311	888.946	954.048	853.028	899.879	955.639	921.568	916.333	891.001	917.660	10.752.302
Botujuru	39.999	42.679	45.945	44.683	47.453	42.594	44.654	47.095	45.711	45.945	46.171	47.310	540.239
Campo Limpo Pta.	70.620	80.177	91.836	92.503	97.956	81.112	83.269	96.889	93.248	92.565	91.822	91.276	1.063.273
Várzea Paulista	41.641	43.162	45.960	45.954	48.462	45.165	48.851	49.725	48.721	49.504	49.562	53.812	570.519
Jundiaí	190.387	183.281	201.609	198.017	204.819	181.576	198.884	202.323	204.277	210.479	208.412	223.316	2.407.380
TOTAL	6.611.770	7.143.371	7.767.202	7.495.806	8.034.922	7.082.604	7.390.158	7.960.026	7.618.497	7.699.795	7.593.528	7.892.784	90.290.463

Tabela 4 – Embarque de passageiros 2011 Linha 8 (CPTM, 2011)

ESTAÇÕES	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ	TOTAL
Júlio Prestes	271.875	271.973	299.549	289.482	299.296	262.983	282.971	296.001	273.329	258.030	254.072	193.112	3.252.673
Barra Funda	2.032.556	2.177.634	2.401.652	2.267.737	2.385.563	2.023.615	2.090.992	2.317.166	2.152.713	2.056.136	2.050.149	2.239.490	26.195.403
Lapa	522.526	559.118	580.682	550.770	574.870	506.453	537.892	593.550	572.933	565.707	558.504	562.324	6.685.329
Dom. de Moraes	264.048	283.019	305.152	287.852	317.734	273.522	293.162	329.844	317.003	314.287	313.903	303.780	3.603.306
Imp. Leopoldina	354.610	409.504	447.853	421.311	460.614	379.946	378.684	456.189	424.727	412.223	406.318	373.518	4.925.497
Presidente Altino	160.495	176.780	188.445	175.477	195.333	167.093	179.155	215.337	207.463	206.698	206.775	204.721	2.283.772
Osasco	821.695	842.972	915.222	881.688	948.566	835.777	889.626	974.206	930.356	946.335	962.354	1.033.115	10.981.912
Cmte. Sampaio	270.110	288.550	316.193	305.903	332.350	295.084	307.227	337.282	325.243	323.220	319.384	306.708	3.727.254
Quitaúna	77.966	83.323	89.011	86.161	94.356	81.331	88.994	101.401	96.116	91.676	90.580	87.736	1.068.651
Gal. Miguel Costa	350.122	366.343	397.087	378.483	410.558	362.747	370.931	419.482	405.782	402.711	399.853	403.939	4.668.038
Carapicuíba	685.706	692.723	755.842	733.970	783.814	694.513	781.503	867.751	843.263	855.173	843.653	826.534	9.364.445
Santa Terezinha	29.579	31.734	34.316	32.044	35.744	31.323	33.168	37.377	36.144	35.475	35.404	35.171	407.479
Antonio João	141.125	150.707	172.022	169.139	187.608	164.067	170.540	188.677	181.673	184.365	194.882	217.659	2.122.464
Barueri	460.284	487.465	531.489	510.855	548.192	479.724	502.731	558.799	528.421	533.406	534.588	552.351	6.228.305
Jardim Belval	62.123	68.886	75.761	73.453	76.071	65.140	64.716	74.811	70.704	72.843	69.337	63.660	837.505
Jardim Silveira	151.911	153.944	168.546	161.436	171.453	151.781	161.128	171.410	162.989	160.712	160.233	166.507	1.942.050
Jandira	253.699	263.846	286.503	271.386	292.214	261.600	276.268	297.830	290.351	292.513	290.750	297.387	3.374.347
Sagrado Coração	54.244	54.518	60.170	57.787	62.794	54.515	56.355	61.681	59.661	58.635	58.291	60.462	699.113
Engº Cardoso	130.758	133.721	144.388	138.890	149.133	133.526	142.096	150.377	144.206	145.174	141.041	143.656	1.696.966
Itapevi	499.294	568.170	557.158	592.741	664.090	645.013	655.003	670.751	644.720	642.609	565.517	662.543	7.367.609
TOTAL	7.594.726	8.064.930	8.727.041	8.386.565	8.990.353	7.869.753	8.263.142	9.119.922	8.667.797	8.557.928	8.455.588	8.734.373	101.432.118

Tabela 5 – Embarque de passageiros 2011 Linha 9 (CPTM, 2011)

ESTAÇÕES	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ	TOTAL
Presidente Altino	45.268	49.861	53.151	49.493	55.094	47.130	50.531	60.736	58.514	58.300	58.320	57.742	644.140
Osasco	209.290	214.709	233.113	224.570	241.605	212.877	226.592	248.136	236.966	241.037	245.117	263.140	2.797.152
Ceasa	108.808	113.013	123.692	118.382	128.746	114.598	122.442	130.715	127.482	130.392	129.583	130.883	1.478.736
Vila Lobos - Jaguaré	157.248	171.230	179.788	168.278	182.965	163.789	181.042	200.911	203.349	210.783	211.150	203.326	2.233.859
Cidade Universitária	262.880	315.814	356.456	348.824	399.262	299.551	186.832	222.953	199.318	197.398	196.299	179.006	3.164.593
Pinheiros	157.872	185.448	199.549	191.610	234.591	385.112	909.937	1.224.012	1.444.124	2.063.879	2.250.216	2.371.996	11.618.346
Rebouças Hebraica	458.587	513.853	573.930	580.860	635.725	526.956	479.970	515.769	475.695	453.172	440.228	422.937	6.077.682
Cidade Jardim	298.269	344.791	375.504	367.004	408.199	393.612	372.424	426.528	400.942	403.199	399.189	379.854	4.569.515
Vila Olímpia	379.057	432.218	466.333	451.443	519.839	497.713	516.138	596.324	575.021	621.984	621.517	558.511	6.236.098
Berrini	283.102	320.308	343.414	334.420	386.767	348.687	396.067	459.467	443.746	459.307	471.775	444.282	4.691.342
Morumbi	313.868	353.005	394.157	384.836	426.748	387.711	416.093	484.320	481.311	505.789	513.665	517.030	5.178.533
Granja Julieta	198.374	229.080	246.242	239.350	270.603	244.887	271.399	315.186	309.276	324.957	333.490	322.283	3.305.127
Santo Amaro	1.150.829	1.316.369	1.499.673	1.478.044	1.617.326	1.468.440	1.590.778	1.822.158	1.818.821	1.908.641	1.942.625	2.011.432	19.625.136
Socorro	176.556	196.321	213.454	207.243	227.726	200.576	228.826	258.427	256.064	273.367	285.079	284.032	2.807.671
Jurubatuba	226.328	247.628	277.609	269.889	299.233	255.333	289.064	310.106	302.288	318.761	319.865	339.536	3.455.640
Autódromo	111.460	128.571	148.788	149.095	163.831	147.419	157.648	180.379	173.704	181.863	203.179	186.158	1.932.095
Primavera-Interlagos	212.743	232.765	261.040	256.045	287.705	245.874	272.819	308.300	302.122	313.347	316.325	320.478	3.329.563
Grajaú	868.078	928.647	1.040.382	1.031.912	1.146.257	1.034.715	1.115.353	1.235.831	1.225.507	1.283.017	1.318.080	1.379.606	13.607.385
TOTAL	5.618.617	6.293.631	6.986.275	6.851.298	7.632.222	6.974.980	7.783.955	9.000.258	9.034.250	9.949.193	10.255.702	10.372.232	96.752.613

Tabela 6 – Embarque de passageiros 2011 Linha 10 (CPTM, 2011)

ESTAÇÕES	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ	TOTAL
Luz	1.012.931	1.052.783	1.132.243	1.097.336	1.170.645	1.052.402	1.094.668	1.156.665	1.099.743	1.231.021	1.236.029	1.241.781	13.578.247
Brás	1.166.103	1.199.032	1.293.433	1.214.378	1.385.317	1.254.422	1.329.518	1.437.078	1.398.409	1.359.736	1.432.525	1.698.778	16.168.729
Moóca	154.719	172.171	187.494	176.413	199.817	171.998	178.329	198.814	186.623	183.213	179.728	169.424	2.158.743
Ipiranga	224.825	243.191	262.338	244.824	271.187	234.779	251.381	272.451	256.111	254.188	267.083	266.116	3.048.474
Tamanduateí	481.040	512.298	657.457	787.550	910.917	845.446	946.757	1.101.517	1.138.690	1.189.463	1.189.250	1.176.238	10.936.623
São Caetano	576.509	611.536	666.413	644.105	704.975	623.046	655.344	728.055	706.877	710.676	707.399	693.154	8.028.089
Utinga	182.262	217.295	249.214	231.325	258.030	223.160	214.736	261.730	250.539	244.570	242.647	219.301	2.794.809
Prefeito Saladino	162.371	168.890	186.732	178.009	193.177	171.956	182.612	198.103	189.115	190.432	184.860	187.650	2.193.907
Santo André	1.100.106	1.168.830	1.261.286	1.221.808	1.347.479	1.197.194	1.280.069	1.379.326	1.332.524	1.336.879	1.319.733	1.319.797	15.265.031
Capuava	147.467	160.560	181.000	169.107	184.704	167.545	170.658	184.682	173.804	174.320	168.874	160.371	2.043.092
Mauá	1.011.122	1.042.475	1.145.077	1.100.294	1.206.341	1.089.511	1.156.789	1.247.144	1.173.926	1.177.807	1.160.812	1.171.811	13.683.109
Guapituba	213.417	224.077	239.461	230.244	248.972	226.423	230.795	252.920	244.745	242.477	236.822	235.160	2.825.513
Ribeirão Pires	375.196	395.550	428.036	409.937	445.037	397.002	415.535	454.217	433.546	431.961	427.060	436.045	5.049.122
Rio Gde da Serra	216.518	220.538	237.373	231.285	250.270	226.637	246.623	250.536	242.492	241.173	238.249	248.423	2.850.117
TOTAL	7.024.586	7.389.226	8.127.557	7.936.615	8.776.868	7.881.521	8.353.814	9.123.238	8.827.144	8.967.916	8.991.071	9.224.049	100.623.605

Tabela 7 – Embarque de passageiros 2011 Linha 11 (CPTM, 2011)

ESTAÇÕES	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ	TOTAL
Luz	1.257.779	1.307.263	1.405.930	1.362.586	1.453.616	1.306.791	1.359.275	1.436.256	1.365.573	1.528.584	1.534.804	1.541.947	16.860.404
Brás	1.309.116	1.346.082	1.452.062	1.363.310	1.555.215	1.438.263	1.492.573	1.613.323	1.569.913	1.526.495	1.608.212	1.907.116	18.181.680
Tatuapé	533.971	568.476	632.490	606.314	663.975	587.697	606.555	692.723	667.485	667.520	679.216	707.689	7.614.111
Corinthians Itaquera	488.316	515.681	562.047	539.108	597.022	522.738	544.875	629.639	644.484	661.704	674.218	729.477	7.109.309
Dom Bosco	291.568	313.838	343.368	335.045	373.062	325.253	337.779	385.140	386.186	392.333	391.537	402.587	4.277.696
José Bonifácio	300.656	325.891	355.870	345.350	382.744	336.996	350.017	388.700	380.245	379.760	379.701	378.719	4.304.649
Guaianazes	1.506.715	1.577.217	1.726.979	1.642.061	1.805.077	1.624.300	1.695.355	1.863.514	1.816.280	1.848.820	1.841.927	1.909.576	20.857.821
EXPRESSO	5.688.121	5.954.448	6.478.746	6.193.774	6.830.711	6.142.038	6.386.429	7.009.295	6.830.166	7.005.216	7.109.615	7.577.111	79.205.670
Antonio G. Neto	287.182	297.627	319.601	312.614	338.343	291.036	315.664	340.993	336.980	335.387	335.688	344.740	3.855.855
F. de Vasconcelos	526.822	547.781	587.552	573.896	619.870	525.523	574.403	626.475	559.435	549.031	545.406	548.773	6.784.967
Poá	273.037	290.488	315.150	301.713	329.339	275.893	294.627	332.388	329.557	321.323	322.823	322.001	3.708.339
Calmon Viana	126.274	128.408	140.110	132.623	147.965	127.371	136.228	145.533	134.105	129.601	119.823	118.587	1.586.628
Suzano	721.615	752.545	816.530	788.947	859.510	728.340	785.836	867.696	835.152	832.144	818.666	833.528	9.640.509
Jundiapeba	122.830	125.871	135.428	134.338	142.746	110.244	133.601	143.854	139.346	139.026	136.886	143.842	1.608.012
Brás Cubas	128.288	131.510	143.249	138.194	145.061	126.864	145.766	153.220	150.090	150.613	147.731	155.182	1.715.768
Mogi das Cruzes	302.527	300.805	320.256	309.440	333.105	282.755	313.089	333.491	317.945	320.269	316.184	321.646	3.771.512
Estudantes	189.020	245.896	275.459	263.536	293.040	225.785	196.396	274.773	266.268	279.347	271.501	227.201	3.008.222
GUA/EST	2.677.595	2.820.931	3.053.335	2.955.301	3.208.979	2.693.811	2.895.610	3.218.423	3.068.878	3.056.741	3.014.708	3.015.500	35.679.812
TOTAL	8.365.716	8.775.379	9.532.081	9.149.075	10.039.690	8.835.849	9.282.039	10.227.718	9.899.044	10.061.957	10.124.323	10.592.611	114.885.482

Tabela 8 – Embarque de passageiros 2011 Linha 12 (CPTM, 2011)

ESTAÇÕES	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ	TOTAL
Brás	1.191.772	1.225.427	1.321.905	1.241.109	1.415.811	1.252.037	1.358.785	1.468.710	1.429.191	1.389.668	1.464.059	1.736.171	16.494.645
Tatuapé	354.500	377.408	419.906	402.528	440.809	378.286	402.688	459.895	443.138	443.162	450.927	469.831	5.043.078
Engº Goulart	91.809	92.362	94.168	98.194	101.874	87.034	95.143	101.783	99.884	102.770	100.060	102.710	1.167.791
USP Leste	56.205	63.535	95.650	89.676	105.530	84.211	67.440	101.742	89.923	92.496	91.755	75.886	1.014.049
Com. Ermelino	141.258	140.930	153.751	152.240	167.097	140.631	157.063	164.381	157.952	158.600	158.295	166.088	1.858.286
São Miguel Pta.	297.963	302.878	327.840	319.525	353.712	299.379	328.012	350.652	333.266	333.337	334.899	376.061	3.957.524
Jd Helena - Vila Mara	283.792	289.218	313.875	306.646	332.617	283.961	314.619	329.697	319.901	315.188	310.809	325.714	3.726.037
Itaim Paulista	634.181	658.365	709.810	685.869	747.953	640.173	698.317	754.926	710.250	712.847	705.100	744.008	8.401.799
Jardim Romano	381.899	393.125	426.069	411.487	449.866	383.951	422.764	458.296	433.873	439.049	432.192	449.958	5.082.529
Engº Manoel Feio	253.209	263.078	282.821	274.321	294.523	250.400	275.258	297.274	280.995	280.766	276.819	284.518	3.313.982
Itaquaquecetuba	355.504	357.823	386.740	375.100	408.173	343.798	380.054	407.753	389.916	387.413	384.014	389.453	4.565.741
Aracaré	175.223	177.761	190.314	186.219	199.766	171.195	190.724	200.434	192.262	193.592	190.624	195.388	2.263.502
Calmon Viana	67.694	68.839	75.113	71.100	79.324	68.283	73.031	78.020	71.894	69.479	64.237	63.573	850.587
TOTAL	4.285.009	4.410.749	4.797.962	4.614.014	5.097.055	4.383.339	4.763.898	5.173.563	4.952.445	4.918.367	4.963.790	5.379.359	57.739.550

APÊNDICE A – ESPECIFICAÇÃO DOS CASOS DE USO

UC01 – Inserir Elemento

Breve Descrição

Objetivo deste caso de Uso é a inserção de elementos na interface. É possível inserir estações, ao selecionar alguma região do mapa, além de incluir os passageiros que embarcam e desembarcam.

Atores

Usuário.

Pré-Condições

Nenhuma.

Fluxo Básico

1. Incluir as estações que compõe a malha.
2. Selecionar na interface o botão “AddPassageiros”.
3. Inserir os passageiros que embarcam de desembarcam a cada estação.

Fluxos Alternativos

Nenhum

UC02 – Remover Elemento

Breve Descrição

Objetivo deste caso de Uso é a remoção de elementos na interface. Assim, é possível remover estações incluindo os passageiros que embarcam de desembarcam nela.

Atores

Usuário.

Pré-Condições

Nenhuma.

Fluxo Básico

1. Selecionar na interface o botão “Remover”.
2. Inserir o número da estação que deseja remover.
3. A estação é excluída da interface, sendo reconstruída a malha ferroviária.

Fluxos Alternativos

1. Selecionar na interface o botão “Remover”.

2.a Selecionar a opção cancelar.

3.a A malha não se altera.

UC03 – Exportar para o Simulador

Breve Descrição

Objetivo deste caso de Uso é enviar os parâmetros da malha ferroviária, que foram criados pelo usuário, ao Simulador.

Atores

Usuário.

Pré-Condições

Ao menos duas estações devem ser criadas

Fluxo Básico

1. Selecionar na interface o botão “Enviar ao Simulador”.
2. Os parâmetros são enviados ao Construtor XML.
3. É gerado um arquivo XML no formato lido pelo CPN Tolls.

Fluxos Alternativos

1. Selecionar na interface botão “Enviar ao Simulador”.

2.a Se não houver ao menos duas estações a interface exibe uma mensagem de erro informando o usuário que deve inserir mais estações.

UC04 – Verificar Distâncias

Breve Descrição

Objetivo deste caso de Uso é informar ao usuário a distância entre duas estações.

Atores

Usuário.

Pré-Condições

Ao menos duas estações devem ser criadas.

Fluxo Básico

1. Selecionar na interface botão “Distancias”.
2. Uma tabela com as distâncias entre as estações é exibida e o mapa se desloca para esquerda.

Fluxos Alternativos

1. Selecionar na interface botão “Distancias”.

2.a Nada acontece, caso não exista mais que duas estações.

UC05 – Abrir rede de Petri

Breve Descrição

Objetivo deste caso de Uso é abrir a rede criada pelo usuário.

Atores

Simulador.

Pré-Condições

Existência do arquivo em XML no formato especificado pelo CPN Tolls.

Fluxo Básico

1. O simulador lê o arquivo XML.
2. O simulador exibe a malha criada na interface, conforme o modelo pré-estabelecido.

Fluxos Alternativos

1. O simulador lê o arquivo XML

2.a Caso algum exista algum erro sintáxico no arquivo XML o simulador não cria o a Rede de Petri.

UC06 – Simular

Breve Descrição

Objetivo deste caso de Uso é simular a malha criada e exportada ao simulador.

Atores

Simulador.

Pré-Condições

Rede criada pelo simulador.

Fluxo Básico

1. Início da Simulação.
2. Exibição visual das marcas em movimento.
3. Fim da simulação.

Fluxos Alternativos

Nenhum

UC07 – Gerar Resultados

Breve Descrição

Objetivo deste caso de Uso é gerar os resultados da simulação da malha.

Atores

Simulador.

Pré-Condições

Rede simulada.

Fluxo Básico

1. Criação de um arquivo que será inserido o resultado da simulação.

Fluxos Alternativos

Nenhum