Extensões da linguagem de script do Mercury

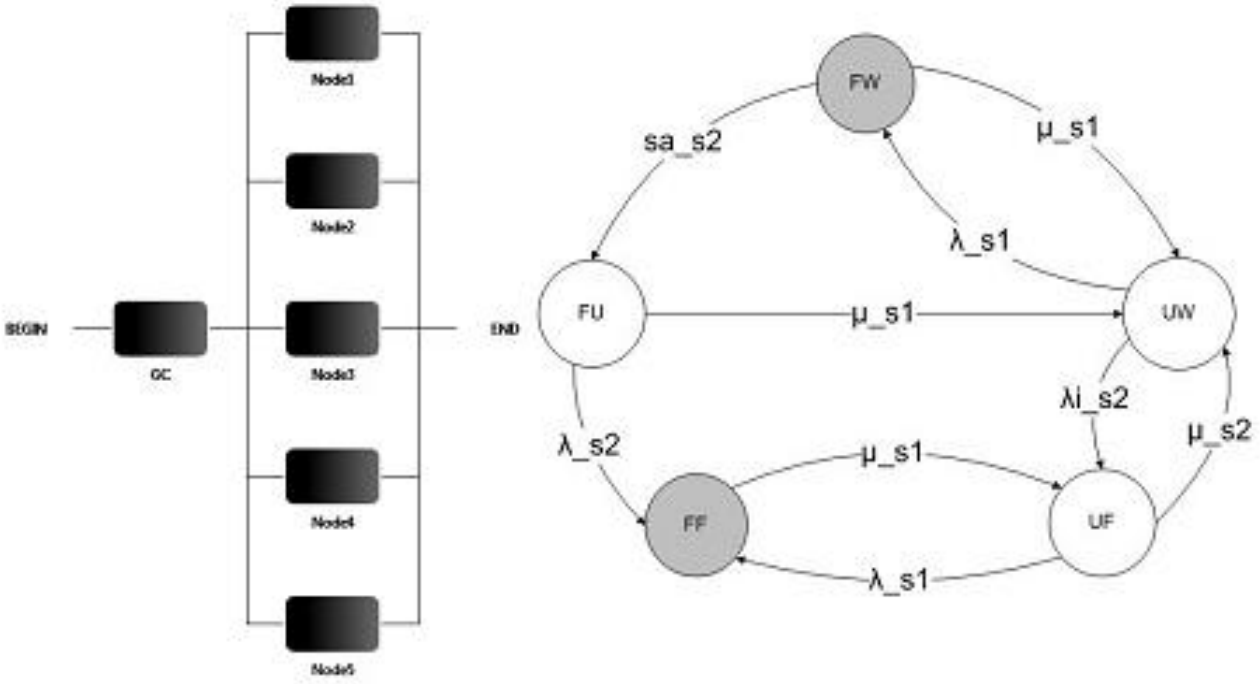
Danilo Oliveira

Paulo Maciel

Linguagem de script do Mercury

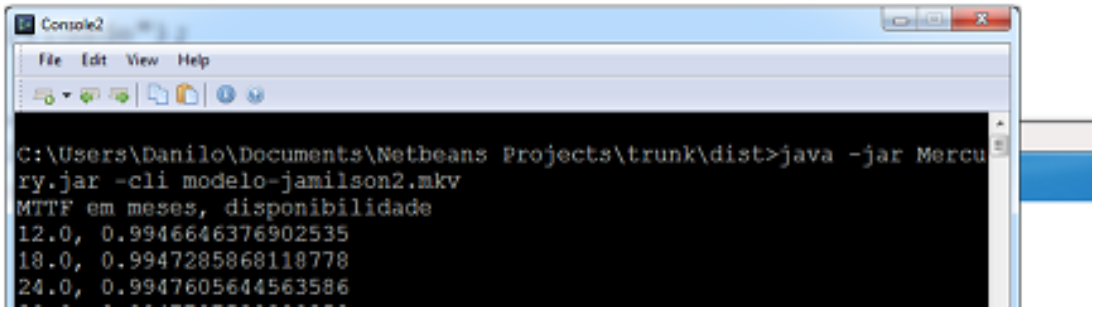
- Essa linguagem foi desenvolvida para introduzir no Mercury um melhor suporte à:
  - Modelagem **hierárquica**
  - Avaliação **simbólica**
  - Flexibilidade na avaliação de modelos

Linguagem de script do Mercury



```
RBD RedundantCloud{
    hierarchy gc(availability=solve(model = RedundantGC, metric = aval) );

    hierarchy node1(availability=solve(model = Node, metric = aval));
    hierarchy node2(availability=solve(model = Node, metric = aval));
    hierarchy node3(availability=solve(model = Node, metric = aval));
    hierarchy node4(availability=solve(model = Node, metric = aval));
    hierarchy node5(availability=solve(model = Node, metric = aval));

    parallel b1(node1, node2, node3, node4, node5);

    series b2(gc, b1);

    top b2;

    metric aval = availability;
}
```

```
println("MTTF em meses, disponibilidade");

for mttfmeses in [12, 18, 24, 30, 36, 43, 48, 54, 60]{
    mttfhw = 30 * 24 * mttfmeses;
    a = solve( model = NonRedundantCloud, metric = aval )
    println(mttfmeses .. ", " .. a );
}
```

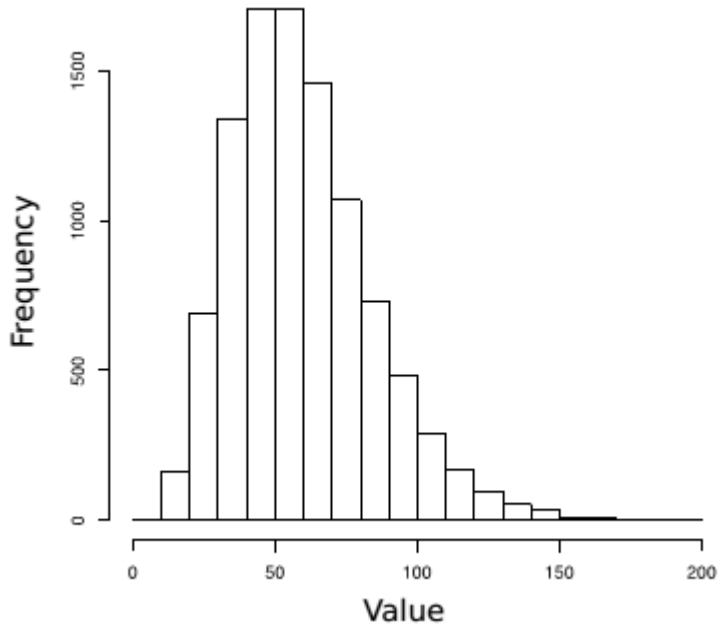


Extensões da linguagem

- A linguagem recebeu duas extensões na sintaxe de redes de Petri
  - Suporte a transições hierárquicas (de substituição)
  - Suporte à distribuições poli-exponenciais (phase type)

Suporte à transições poli-exponenciais

- Suponha que, ao parametrizar seu modelo com dados obtidos do mundo real, uma certa taxa apresente a seguinte distribuição

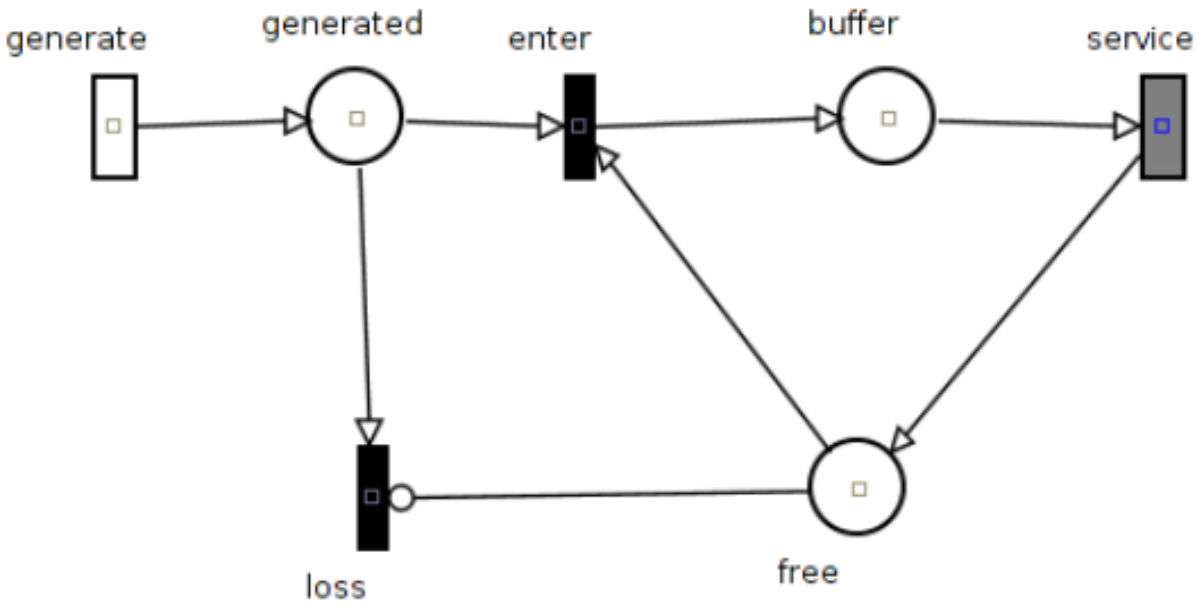


Suporte à transições poli-exponenciais

- Modelos estocásticos como SPNs e CTMCs assumem que os tempos de transição entre os estados são exponencialmente distribuídos
- Violar essa suposição faz seu modelo ficar menos preciso e mais distante da realidade
- Uma solução é usar distribuições poli-exponenciais, que podem ser usadas para aproximar uma distribuição genérica qualquer

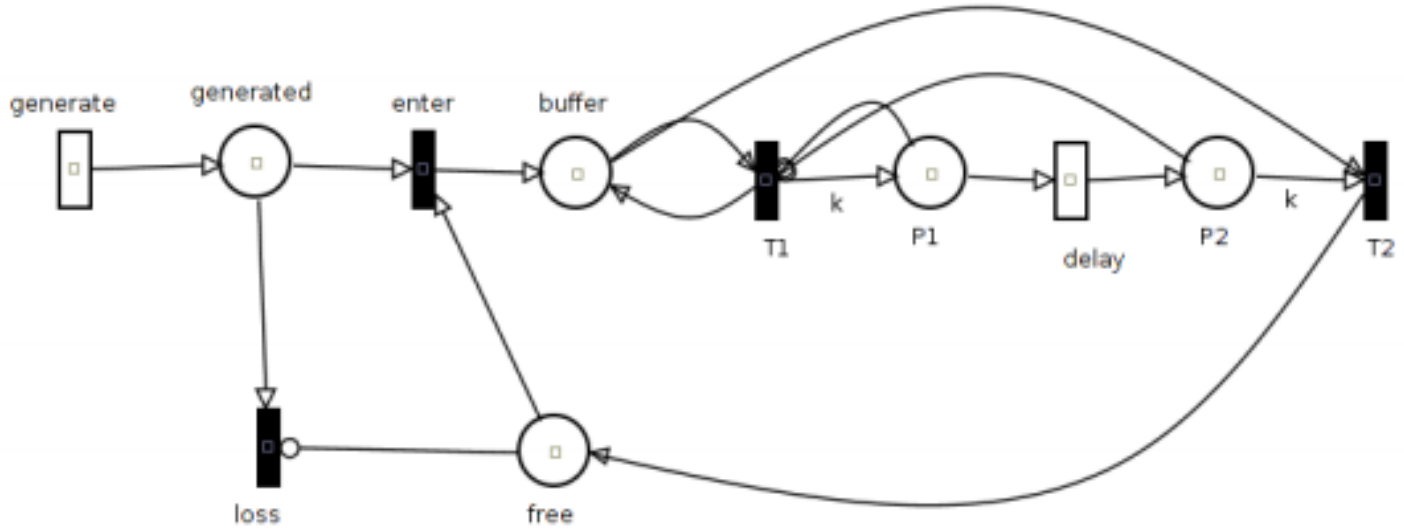
Suporte à transições poli-exponenciais

- Considere o modelo em rede de Petri abaixo (uma fila M/M/1/k)
- Suponha que a taxa da transição *service* segue a distribuição apresentada anteriormente



Suporte à transições poli-exponenciais

- Usando técnicas de **moment-matching**, verificamos que os dados capturados seguem uma distribuição **Erlang**, que pode ser expressa em SPN como:



Suporte à transições poli-exponenciais

- **Problema:** estrutura do modelo fica “poluída” com as transições e lugares extras
- **Solução:** criar uma sintaxe especial para representar transições poli-exponenciais e gerar os lugares e transições extras “on the fly”

Suporte à transições poli-exponenciais

- Sintaxe:

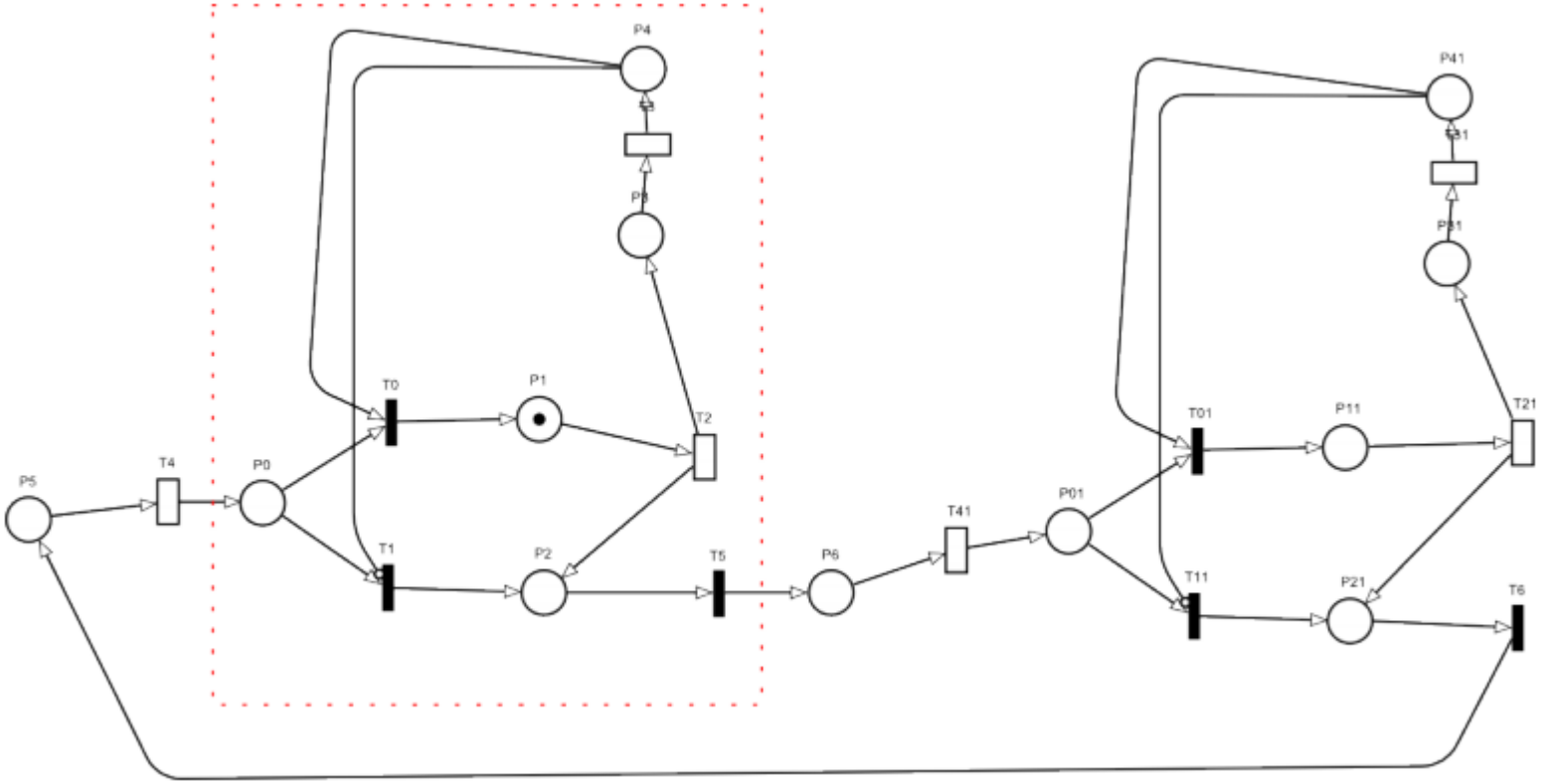
```
timedTransition T0(
    inputs = [A],
    outputs = [B],
    delay = (type="Erlang",
        parameters = (delay=10,
          shape=6
        )
    )
);
```

Suporte à transições hierárquicas

- Igual a linguagens de programação, redes de Petri podem se beneficiar de técnicas de modularização
  - Deixar um modelo complexo mais simples usando uma estratégia de “dividir para conquistar”
  - Encontrar estruturas que se repetem e reutilizá-las

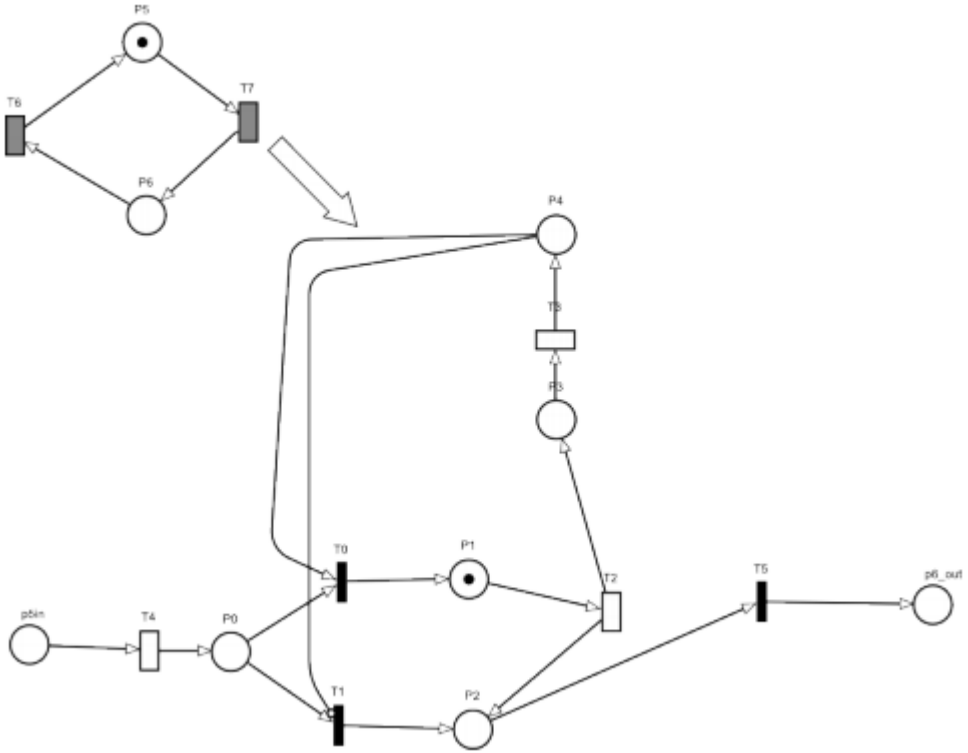
Suporte à transições hierárquicas

- Considere o seguinte modelo, que representa uma rede *token ring*



Suporte à transições hierárquicas

- Podemos deixar o modelo mais simples com uma transição de substituição no lugar da estrutura destacada



Suporte à transições hierárquicas

Declaração da estrutura da transição:

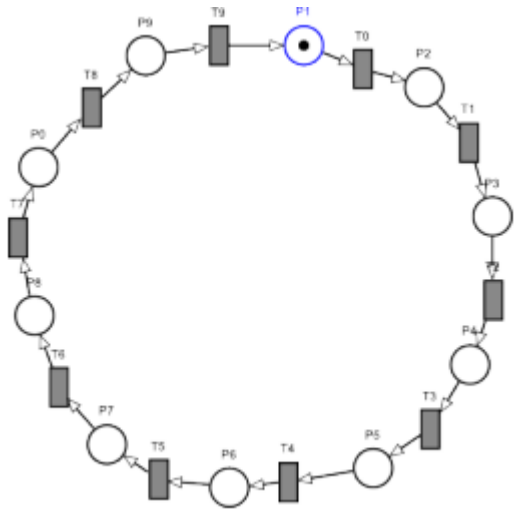
```
SPN Transition{

    in P5in;
    out P6out;

    ...

}
```

Uso da transição na rede de mais alto nível

```
SPN PetriNet{

    place P5;
    place P6;

    ...

    substitutionTransition Tsub(
        subnet = Transition,
        inputs = ( P5in = P5 ),
        outputs = ( P6out = P6 )
    );

    substitutionTransition Tsub(
        subnet = Transition,
        inputs = ( P5in = P6 ),
        outputs = ( P6out = P7 )
    );
```

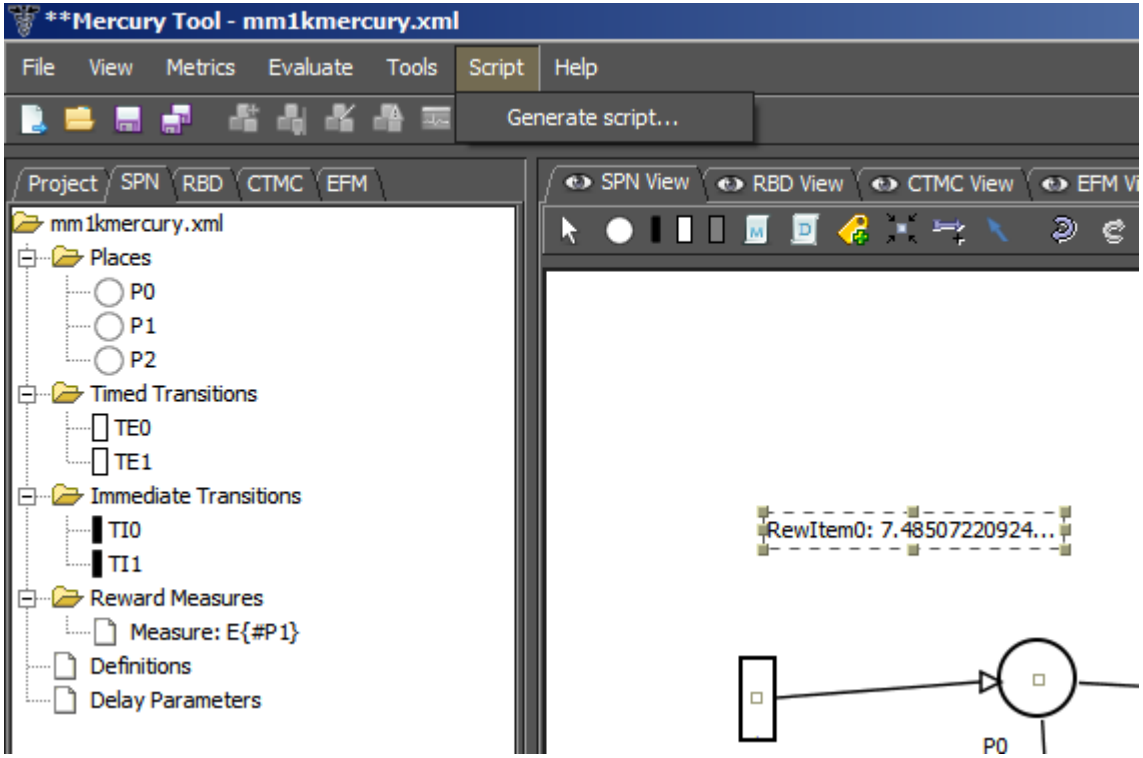
Suporte à transições hierárquicas

- Rede token ring de 10 nós

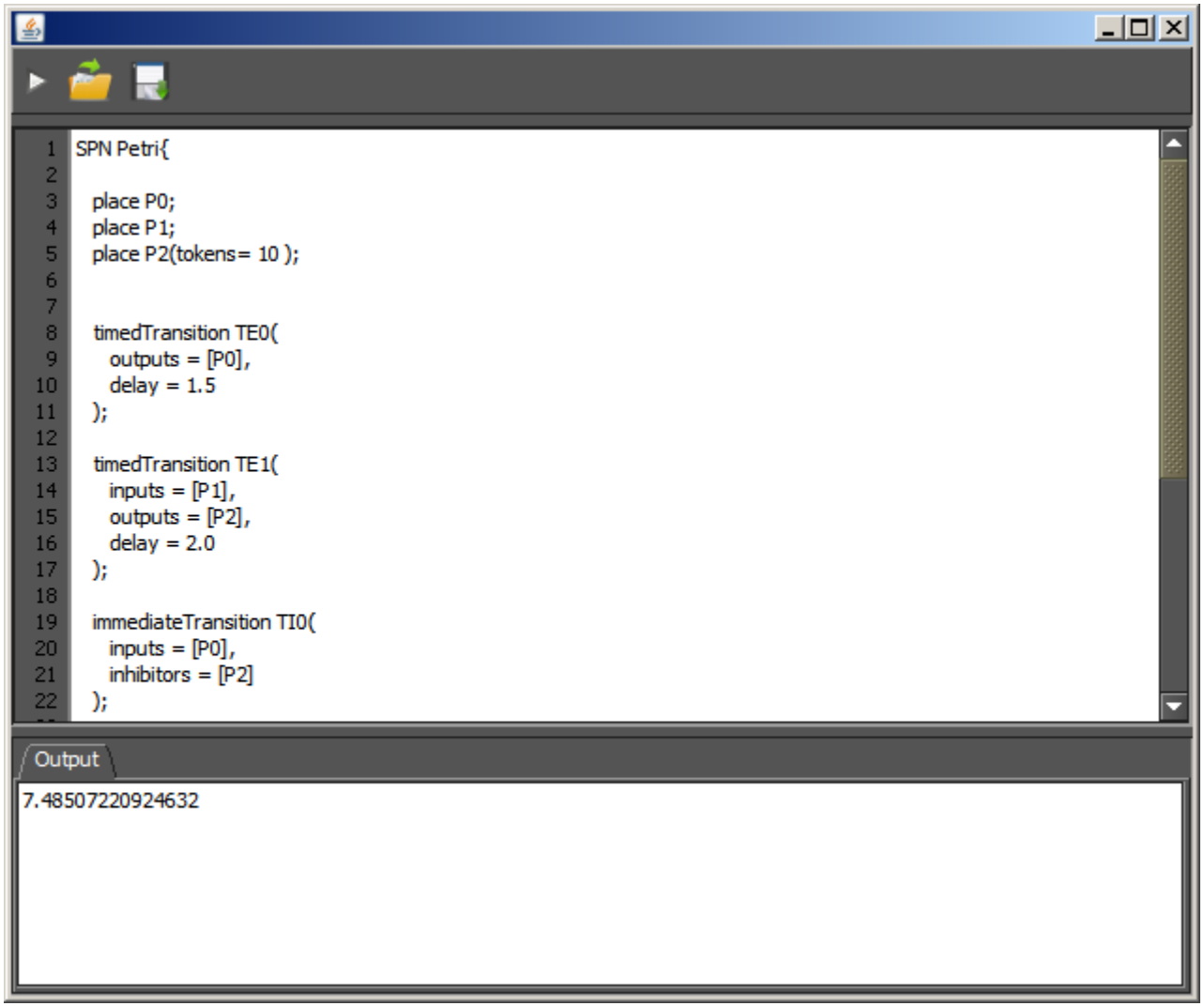



```
SPN TokenRing10{
    place p1;
    place p2;
    ...

    substitutionTransition T1(
        subnet = Transition,
        inputs = ( P5in = P1 ),
        outputs = ( P6out = P2 )
    );

    substitutionTransition T2(
        subnet = Transition,
        inputs = ( P5in = P2 ),
        outputs = ( P6out = P3 )
    );

    ...

    substitutionTransition T10(
        subnet = Transition,
        inputs = ( P5in = P10 ),
        outputs = ( P6out = P1 )
    );
}
```

Integração da linguagem de Script com a GUI



Integração da linguagem de Script com a GUI



Próximos passos...

- Comando “include”
- Mecanismo de “herança" para criar novos modelos baseados em modelos existentes
- Avaliação de RdPs por simulação
- Estruturas de repetição para componentes de um modelo