

# UM LEITOR E CONVERSOR DE ARQUIVOS XMI PARA SISTEMAS DE TRANSIÇÃO DE ESTADOS

Eduardo Rohde Eras<sup>1</sup>, Luciana Brasil Rebelo dos Santos<sup>2</sup>, Reinaldo Gen Ichiro Arakaki<sup>3</sup>

<sup>1,3</sup> FATEC São José dos Campos - Prof<sup>o</sup> Jessen Vidal

<sup>1,2</sup> INPE - Instituto Nacional de Pesquisas Espaciais

eduardorohdeeras@gmail.com, reinaldoarakaki@gmail.com

## 1. Introdução

De um lado, temos os complexos Métodos de verificação formal de software que, segundo Baier [1], “oferecem grande potencial para prover técnicas de verificação mais efetivas, pois utilizam rigor matemático para estabelecer a correção do sistema”. De outro lado, temos os práticos diagramas UML (*Unified Modeling Language*) que, segundo a OMG [2], “permitem trabalhar a um determinado nível de abstração, omitindo ou mascarando detalhes e exibindo uma visão geral do processo”.

O objetivo deste trabalho é desenvolver uma ferramenta que possibilite o uso de verificação formal em diagramas UML, fazendo a conversão dos arquivos XMI (*eXtensible Markup Language Metadata Interchange*) gerados pelos diagramas UML em Sistemas de Transição de Estados, utilizados pela verificação formal.

## 2. Estrutura da ferramenta

A ferramenta trabalha com dois diagramas UML diferentes: Os diagramas de sequência e de atividades. Sua estrutura é dividida em dois módulos: o módulo Leitor e o módulo Conversor. O primeiro módulo possui uma classe *parser* que faz a leitura do arquivo XMI de entrada e separa todas as *tags* deste arquivo em uma lista para ser utilizada pelo módulo Conversor. O módulo Conversor identifica qual diagrama UML gerou essa entrada (*Diagram Selector*) e direciona a lista para uma lógica específica para seu tratamento. As lógicas são compostas por diversas classes responsáveis por iterar a lista de *tags* e chamar uma função específica para o tratamento de cada uma delas. Por fim, toda a informação gerada pelas lógicas é direcionada para a classe *Builder*, responsável por gerar o Sistema de Transição de Estados final. Toda esta estrutura é ilustrada pela Figura 1.

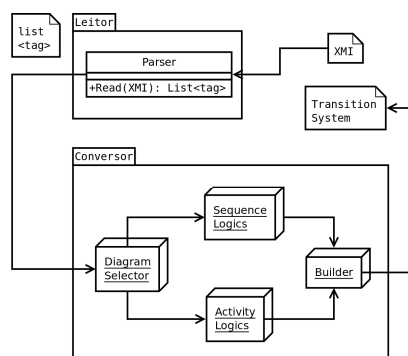


Figura 1 – Estrutura da ferramenta.

## 3. Exemplo de Aplicação

Ao submeter o diagrama UML de atividades mostrado na Figura 2 obtemos como saída o Sistema de Transição de Estados da Figura 3.

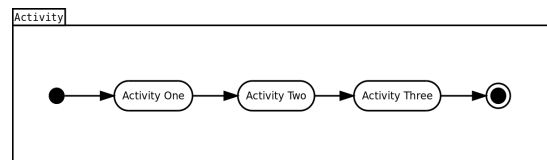


Figura 2 – Diagrama de Atividades

```
Node Children
-----
{1, 1, 0}
-----
Transition System Output
-----
Message: Activity One
-----
Message: Activity Two
-----
Message: Activity Three
-----
NUMBER OF STATES: 3
```

Figura 3 – Sistema de Transição de Estados

Nota-se que as três atividades do diagrama UML de entrada aparecem na mesma ordem como “estados” no Sistema de Transição de Estados de saída. Essa saída está pronta para uso em uma aplicação de Verificação Formal.

## 4. Conclusões

Foram submetidos diversos diagramas, tanto de atividades quanto de sequência. Mesmo em um alto grau de complexidade, contendo decisões, paralelos e loops, em todos os casos, foram obtidos como resultado um Sistema de Transição de Estados dentro do esperado.

## 5. Referências

- [1] Baier, C.; Katoen, J. P. Principles of Model Checking. [S.l.]: MIT Press, 2008.
- [2] OMG – Introduction to OMG's UML. Disponível em [http://www.omg.org/gettingstarted/what\\_is\\_uml.htm](http://www.omg.org/gettingstarted/what_is_uml.htm) Acessado em 03/04/2014.

## Agradecimentos

Os autores agradecem ao INPE e a FATEC pela infraestrutura e suporte prestado.

<sup>1</sup> Aluno de IC do CNPq.