MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

PADRONIZAÇÃO XML DE INTERFACE PARA WEB-PERFORMCHARTS, BASEADA EM W3C E INCORPORAÇÃO DE NOVOS MÉTODOS DE GERAÇÃO DE TESTES PARA SOFTWARE CRÍTICO E APLICAÇÕES ESPACIAIS

Ana Paula de Oliveira Garcia

Relatório final de Iniciação Científica, orientada pelo Dr. Nandamudi Lankalapalli Vijaykumar e Dr. Gian Ricardo Berkenbrock, aprovada em 03 de agosto de 2019.

URL do documento original:
<http://urlib.net/>

INPE
São José dos Campos
2021

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3945-6923/6921
Fax: (012) 3945-6919
E-mail: pubtc@sid.inpe.br

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

PADRONIZAÇÃO XML DE INTERFACE PARA WEB-PERFORMCHARTS, BASEADA EM W3C E INCORPORAÇÃO DE NOVOS MÉTODOS DE GERAÇÃO DE TESTES PARA SOFTWARE CRÍTICO E APLICAÇÕES ESPACIAIS

Ana Paula de Oliveira Garcia

Relatório final de Iniciação Científica, orientada pelo Dr. Nandamudi Lankalapalli Vijaykumar e Dr. Gian Ricardo Berkenbrock, aprovada em 03 de agosto de 2019.

URL do documento original:
<http://urlib.net/>

INPE
São José dos Campos
2021

Dados Internacionais de Catalogação na Publicação (CIP)

Sobrenome, Nomes.

Cutter Padronização XML de interface para WEB-PerformCharts, baseada em W3C e incorporação de novos métodos de Geração de testes para software crítico e aplicações espaciais / Nome Completo do Autor1; Nome Completo do Autor2. – São José dos Campos : INPE, 2021.

xvii + 18 p. ; ()

Dissertação ou Tese (Mestrado ou Doutorado em Nome do Curso) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, AAAA.

Orientador : José da Silva.

1. Palavra chave. 2. Palavra chave 3. Palavra chave. 4. Palavra chave. 5. Palavra chave I. Título.

CDU 000.000

ATENÇÃO! A FOLHA DE APROVAÇÃO SERÁ INCLUIDA POSTERIORMENTE.

Mestrado ou Doutorado em Nome do Curso

A meus pais ***Arnaldo*** *e* ***Erika****, à minha irmã* ***Isabela***

AGRADECIMENTOS

Agradeço ao INPE, a CNPq e a Universidade Federal de São Paulo (UNIFESP).

Agradeço principalmente aos meus orientadores Dr. Nandamudi Lankalapalli Vijaykumar e Dr. Gian Ricardo Berkenbrock.

RESUMO

O presente relatório descreve as atividades desenvolvidas durante o Programa Institucional de Bolsas de Iniciação Científica (PIBIC), que conta com o objetivo de implementação de uma interface textual baseada na linguagem de marcação SCXML e também a incorporação de outros métodos de geração de testes já desenvolvidos, como H-Switch Cover, Breadth First Search e Depth First Search, na ferramenta WEB-PerformCharts. O projeto em questão facilita o uso da ferramenta através de uma interface XML padrão W3C para usuários interessados que já utilizam UML e Statecharts. Dessa forma, foi feita uma pesquisa bibliográfica das principais áreas relacionadas ao trabalho a fim de ter um amplo conhecimento de como funciona o WEB-PerfomCharts para, então, desenvolver testes para certificar o correto funcionamento da ferramenta.

Palavras-chave: Software crítico. Testes automáticos. WEB-PerformCharts.

XML STANDARDIZATION OF INTERFACE FOR WEB-PERFORMCHARTS, BASED ON W3C AND INCORPORATION OF NEW TEST GENERATION METHODS FOR CRITICAL SOFTWARE AND SPATIAL APPLICATIONS

ABSTRACT

This report describes the activities developed during the *Programa Institucional de Bolsas de Iniciação Científica* (PIBIC), which has the objective of implementing a textual interface based on the SCXML markup language and also the incorporation of other methods for generating tests already developed. , such as H-Switch Cover, Breadth First Search and Depth First Search, in the WEB-PerformCharts tool. The project in question facilitates the use of the tool through a W3C standard XML interface for interested users who already use UML and Statecharts. Thus, a bibliographic research of the main areas related to the work was carried out in order to have a broad knowledge of how WEB-PerfomCharts works, and then to develop tests to certify the correct functioning of the tool.

Keywords: Critical software. Automatic tests. WEB-PerformCharts.

LISTA DE FIGURAS

Pág.

LISTA DE ABREVIATURAS E SIGLAS

INPE	–	Instituto Nacional de Pesquisas Espaciais
UNIFESP	–	Universidade Federal de São Paulo
CNPq	–	Conselho Nacional de Desenvolvimento Científico e Tecnológico
LABAC	–	Laboratório Associado de Computação e Matemática
MEF	–	Máquinas de Estados Finitos
PcML	–	*PerformCharts Markup Language*
UML	–	Linguagem de Modelagem Unificada
W3C	–	*World Wide Web Web Consortium*
XML	–	*Extensible Markup Language*
SCXML	–	Linguagem de Marcação - *State Chart XML*
HTML	–	*HyperText Markup Language*
PHP	–	*HyperText PreProcessor*
COPDT	–	Coordenação de Pesquisa e Desenvolviment Tecnológico

SUMÁRIO

Pág.

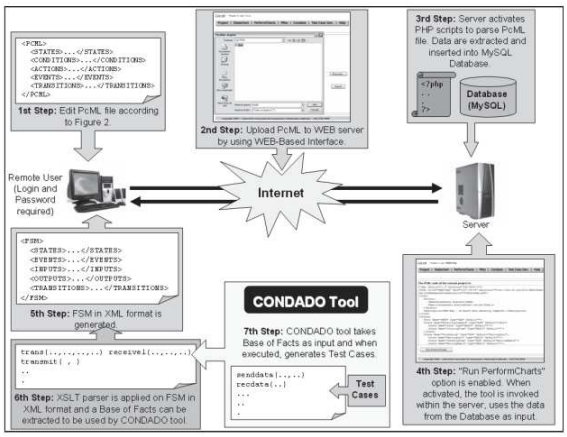
1 INTRODUÇÃO

Aplicações espaciais dependem progressivamente mais de sistemas computacionais para melhor desempenho na operação de satélites, foguetes e balões estratosféricos. E os sistemas necessitam de softwares inteiramente dedicados que vão a bordo dos equipamentos. Devido à particularidade dos softwares, é preciso que sejam extremamente bem qualificados para evitar problemas eventuais.

Sistemas críticos são aqueles que as falhas existentes dentro de um software podem causar um grande impacto financeiro, pessoal ou físico, por conta do alto custo do projeto e a dificuldade de sua manutenção. Por exemplo, uma vez que os satélites estejam no espaço, se houver algum problema, sua correção é inviabilizada. Dessa forma, uma aplicação espacial é considerada um sistema crítico.

Uma prática que qualifica o software é a Validação por meio de testes (DELAMARO et al., 2013) e (AMMANN; OFFUTT, 2017). Desse modo, o COPDT (Coordenação de Pesquisa e Desenvolvimento Tecnológico) do INPE desenvolveu a ferramenta WEB-PerfomCharts, para geração automática de testes de software na qual sua especificação é modelada por meio de Máquinas de Estados Finitos (MEF) (LEE; YANNAKAKIS, 1996) ou Statecharts (HAREL, 1987b). No caso de uma especificação seja modelada em Statecharts, a ferramenta fará uma conversão para uma MEF plana, a partir da qual se obtêm os testes aplicando alguns critérios já implementados.

Figura 1.1 - Arquitetura do WEB-PerfomCharts



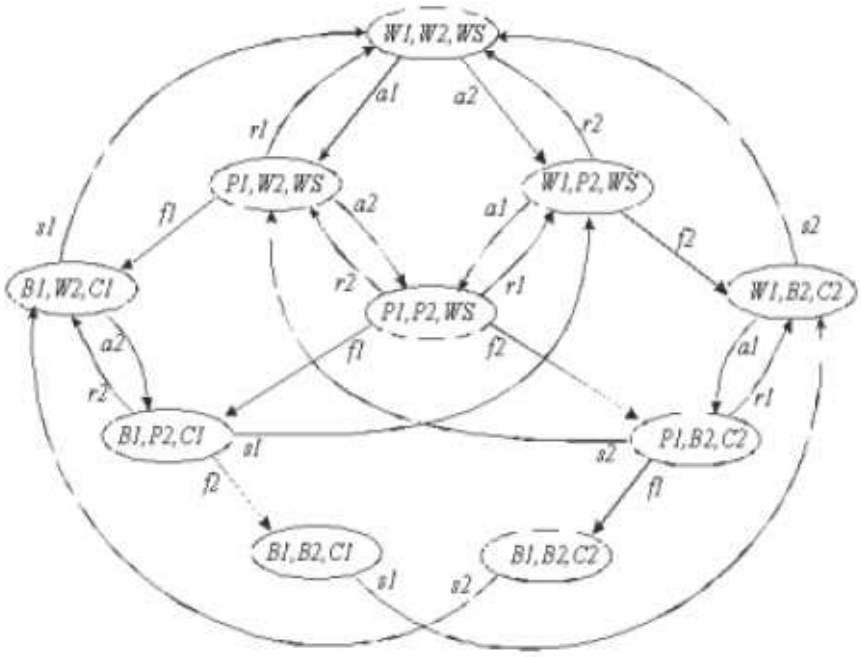
Fonte: (ARANTES et al., 2008c)

Uma MEF é um modelo formal usado para representar sistemas reativos que podem ser programas de computadores ou circuitos lógicos. Ela consiste em estados, eventos e transições. Cada estado descreve um certo aspecto do sistema que se encontra à espera de um determinado evento para que uma transição seja executada. Os Statecharts estendem máquinas de estado com a adição de mecanismos para representar atividades paralelas, condições de guardas e até eventos considerados internos onde a reação ocorre sem uma intervenção externa. Ela é formalmente definida por

$$M = (\sum, S, s_0, \delta, F) \tag{1.1}$$

Onde o primeiro elemento, $\sum$, representa a entrada de um sinal no sistema, o elemento S representa o conjunto de estados que o sistema pode assumir, o elemento s_0 é o estado inicial em que a máquina começa, o elemento δ é a função de transição de estados e, por fim, o elemento F é um subconjunto de S que especifica os possíveis estados finais da máquina.

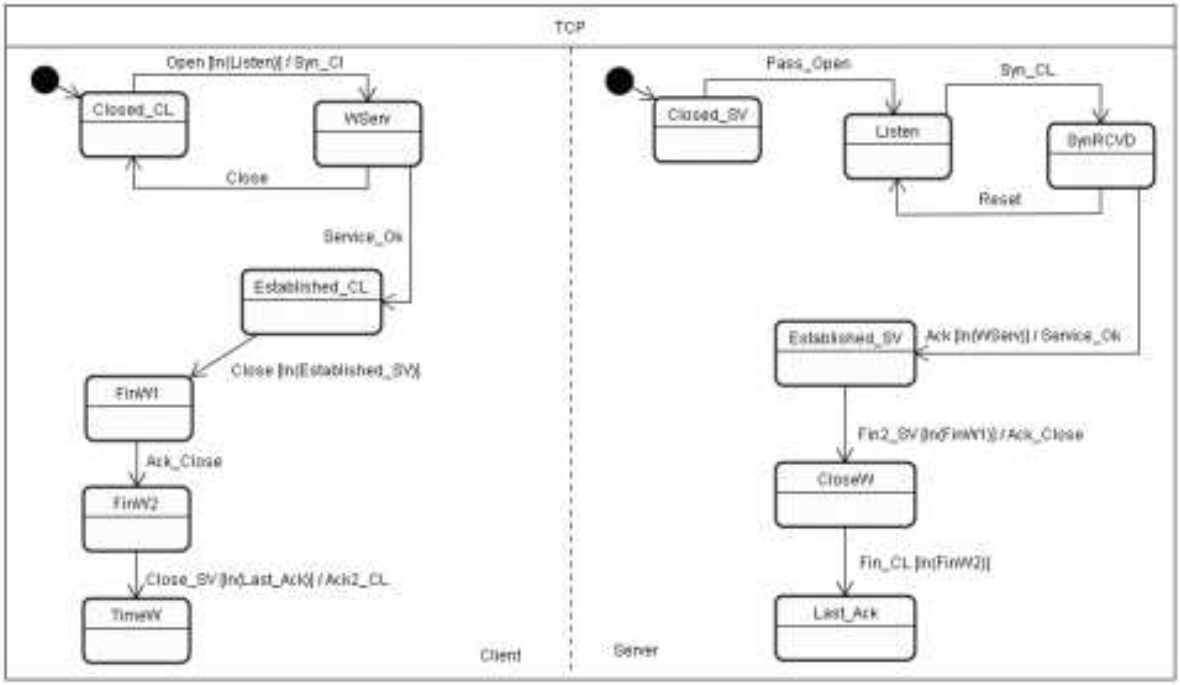
Figura 1.2 - Exemplo de uma MEF



Fonte: (AMARAL, 2005)

O acionamento da ferramenta é por meio de uma interface textual baseada em XML, a qual consiste de *tags* para referir-se aos estados, eventos, transições e condições. A linguagem em questão foi denominada de PcML (*PerformCharts Markup Language*) (AMARAL, 2005). Desse modo, assim que o usuário inicia a ferramenta e faz seu acesso, é necessário que o seja feito o upload de um arquivo PcML, o qual possui o modelo de uma especificação de software. Então, a WEB-PerfomCharts chama um interpretador para gerar o programa principal em C++, e este é executado (se a especificação estiver em Statecharts) e gera uma MEF plana eliminando a hierarquia e atividades concorrentes. Com a MEF é escolhido um critério (SIDHU; LEUNG, 1989) para se obter as sequências de testes.

Figura 1.3 - Representação de um statechart

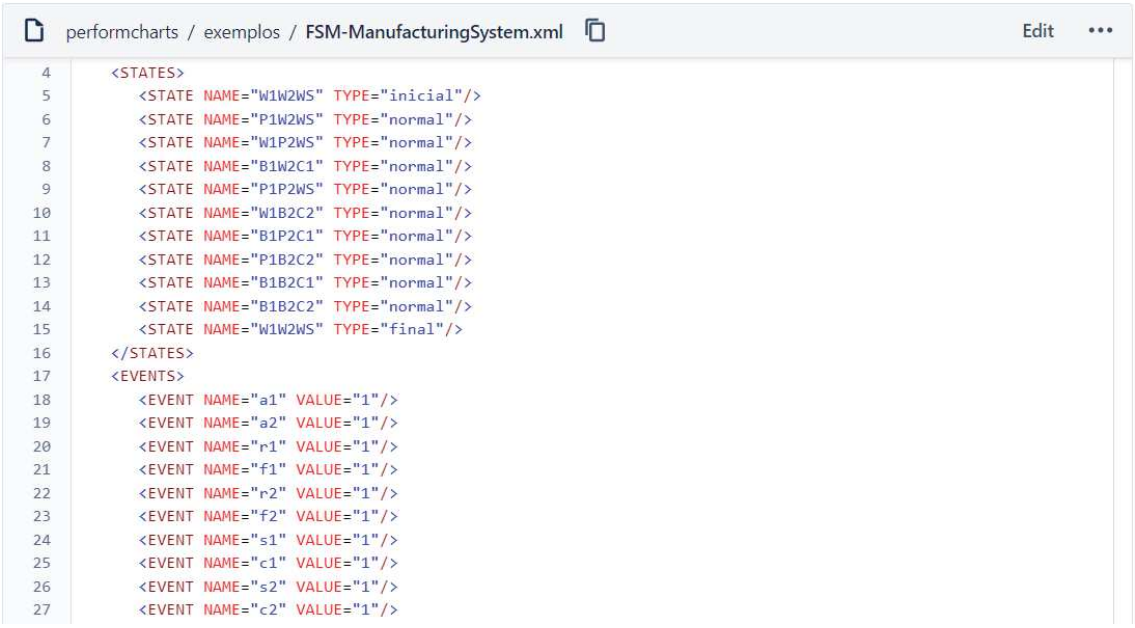


Fonte: (AMARAL, 2005)

Unified Modeling Language, ou Linguagem Unificada de Modelagem (UML) é uma linguagem utilizada para modelar e documentar as diversas fases do desenvolvimento de um sistema orientado a objetos.

Quando a linguagem, PcML, foi proposta, não havia nenhum padrão. Com a popularidade dos Statecharts e, em particular, quando seu uso foi incorporado na linguagem UML para modelar comportamento de uma classe, a W3C propôs uma linguagem de marcação – State Chart XML (SCXML). Em futuro próximo, a ideia do grupo é incorporar SCXML também para especificar um Modelo de Sistemas Reativos em Statecharts.

Figura 1.4 - Exemplo do uso de SCXML



Fonte: Autor

2 DESENVOLVIMENTO

2.1 METODOLOGIAS

O projeto tem como objetivo principal a implementação de um algoritmo para a automação de testes como uma tentativa de reduzir custos associados a eles. Isso possibilita a detecção de falhas e, então, diminuindo os ciclos de testes, de modo que facilita o uso da ferramenta WEB-PerformCharts através de uma interface XML padrão W3C (W3C, 1994) para usuários que já trabalham com UML e Statecharts. Portanto, a fim de atingir o objetivo proposto, o desenvolvimento foi dividido em três etapas:

Figura 2.1 - Tabela descrição da metodologia utilizada

	Meses					
	2	4	6	8	10	12
Etapa 1: Realização de pesquisas						
Pesquisa bibliográfica	x					
Estudo do PcML e SCXML	x					
Etapa 2: Implementação do algoritmo						
Estudo de bibliotecas e arquiteturas		x				
Identificação e modelagem			x			
Início do desenvolvimento					x	
Estender a ferramenta para interpretar o modelo			x			
Testes e validação				x		
Etapa 3: Divulgação científica						
Participação SICINPE 2021	x					
Resumo do projeto	x					
Relatório final	x					
Publicação de artigo	x					

Fonte: Autor

2.2 STATECHARTS

Statecharts (HAREL, 1987a) são modelos baseados em *Communicating Finite State Machines* apropriados para modelar sistemas reativos complexos. São uma técnica formal de especificação do comportamento de sistemas reativos estendendo diagramas de estado com conceitos de decomposição hierárquica de estados fornecendo noções de profundidade (abstração); ortogonalidade que permite a representação de atividades paralelas; e interdependência e sincronismo através de comunicação do tipo broadcasting (HAREL, 1987b). Além disso, Statecharts são fundamentados nos seguintes elementos básicos: Estados, Eventos, Condições, Ações, Expressões, Variáveis, Rótulos e Transições.

Estados são usados para descrever componentes (e suas possíveis situações) de um determinado sistema. Os estados de um Statechart (que representam os valores das variáveis do sistema em um determinado instante) podem ser classificados em dois grupos: básicos e não-básicos. Os estados básicos são aqueles que não possuem subestados. Já os não-básicos são decompostos em subestados. Essa decomposição pode ser de dois tipos: XOR ou AND. Se a decomposição é do tipo XOR, então o sistema sempre estará em um único subestado em um certo instante. Entretanto, se a decomposição é do tipo AND, o estado estará em mais de um subestado, simultaneamente.

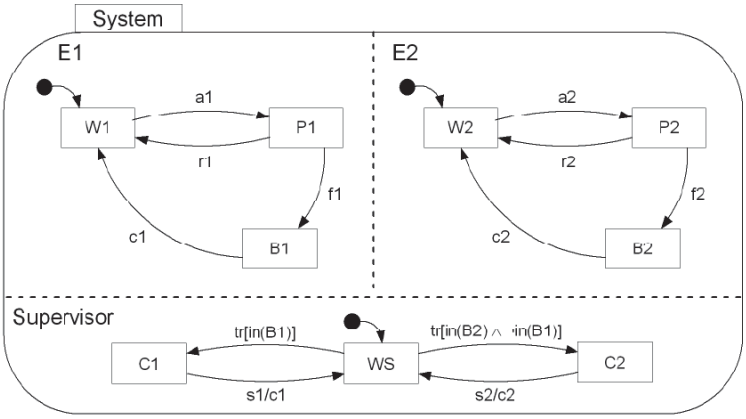
Eventos representam uma interferência no comportamento atual do sistema, levando esse sistema a outro comportamento ou seja o estado atual é modificado para um outro. Eventos podem ser externos ou internos. Os externos são aqueles que devem ser estimulados explicitamente, já os internos são automaticamente acionados pelo próprio sistema. Opcionalmente, a um evento pode ser anexada uma condição, também chamada de condição-guarda, de maneira que o evento só ocorrerá se satisfeita aquela determinada condição. Os Statecharts proporcionam alguns eventos especiais internos como true (condição) e false (condição), abreviados na notação Statecharts para tr (condição), fs (condição), respectivamente.

O elemento ação representa os efeitos do paralelismo em Statecharts (a influência de um estado paralelo em outro, também ortogonal). Ações podem ser uma mudança de uma expressão, uma mudança de uma variável ou eventos que são disparados em outros componentes paralelos. Eventos como ação são considerados internos, ou seja, não é necessário o seu estímulo explícito.

As transições, representadas graficamente por setas dirigidas, denotam uma mu-

dança de estado para um outro dentro do sistema. Rótulos nas setas indicam eventos e opcionalmente condições de guarda e ações para prover algum significado adicional.

Figura 2.2 - Especificação em Statecharts de um sistema com duas máquinas e um reparador



Fonte: (AMARAL, 2005)

A Figura 2.2 exemplifica um Sistema de Manufatura com duas máquinas e um reparador para o conserto das máquinas, caso quebrem. Os componentes paralelos (sub-estados AND da raiz System) são E1, E2, Supervisor e são estados XOR (Ou-Exclusivos). Os componentes E1 e E2 têm suas lógicas representadas por uma MEF e seus subestados. O componente Supervisor é acionado quando há quebra das máquinas com prioridade para conserto da E1 quando as duas estão quebradas. Os estados W1, W2 e WS (dos componentes E1, E2 e Supervisor respectivamente) são os estados iniciais do System. No caso de geração de testes, como os métodos implementados dependem de uma MEF, esta representação Statecharts é convertida para uma MEF plana já mostrada na Figura 1.2. Detalhes e algoritmo desta conversão podem ser encontrados em (AMARAL et al., 2003) e (ARANTES et al., 2014).

Para o desenvolvimento do projeto, foi preciso um estudo mais aprofundado da técnica Statecharts, a qual é utilizada para a especificação de um software. Dessa forma, pode-se observar que a técnica gráfica de Statecharts é uma extensão à MEF, permitindo a representação da composição hierárquica de estados, atividades paralelas, sincronismo e interdependência através de comunicação entre componentes broadcast.

Quando uma especificação é modelada através de Statecharts, a ferramenta (WEB-PerformCharts) converte para uma MEF e, a partir daí, obtêm-se os testes aplicando alguns critérios já implementados. Portanto, atualmente, é utilizada a interface textual, baseada em XML, a qual consiste em tags para se referir aos estados, eventos, transições e condições de guarda. A linguagem em questão é conhecida como PcML.

Quando esta linguagem, PcML (PerformCharts Markup Language) foi proposta, não havia nenhum padrão. Com a popularidade dos Statecharts e, em particular, quando o seu uso foi incorporado na linguagem UML para modelar comportamento de uma classe, a W3C propôs uma linguagem de marcação - State Chart XML (SCXML).

2.3 SCXML

SCXML significa Statechart XML, que é uma notação de máquina de estado para abstração de controle. É uma linguagem de marcação baseada em XML que fornece um ambiente de execução baseado em máquinas de estados genéricas que se baseiam em gráficos de estados de (HAREL, 1987b), também é capaz de descrever MEF complexas.

Figura 2.3 - Exemplo da linguagem de marcação utilizada no projeto

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="mfeeX.xsl"?>
<MFEE>
   <STATES>
      <STATE NAME="IdleWaitingSync" TYPE="inicial"/>
      <STATE NAME="CountingTimeWaitingExpid" TYPE="normal"/>
      <STATE NAME="CountingTimeWaitingType" TYPE="normal"/>
      <STATE NAME="CountingTimeWaitingSize" TYPE="normal"/>
      <STATE NAME="CountingTimeWaitingData" TYPE="normal"/>
      <STATE NAME="CountingTimeWaitingChecksum" TYPE="normal"/>
      <STATE NAME="IdleWaitingSync" TYPE="final"/>
   </STATES>
   <EVENTS>
      <EVENT NAME="EB9" VALUE="1"/>
      <EVENT NAME="StartTimingCount" VALUE="1"/>
      <EVENT NAME="NotEB9" VALUE="1"/>
      <EVENT NAME="WaitingTimeExpired" VALUE="1"/>
      <EVENT NAME="Timeout" VALUE="1"/>
      <EVENT NAME="ExpidRec" VALUE="1"/>
      <EVENT NAME="TypeRec" VALUE="1"/>
      <EVENT NAME="SizeRec" VALUE="1"/>
      <EVENT NAME="DataRec" VALUE="1"/>
      <EVENT NAME="ChecksumRec" VALUE="1"/>
      <EVENT NAME="CommandReceived" VALUE="1"/>
   </EVENTS>
```

Fonte: Autor

A notação SCXML é uma candidata para a linguagem de controle dentro de várias linguagens de marcação provenientes do W3C. SXCML define uma especificação completa para representar os diagramas de estados por um arquivo XML. O SCXML define várias classes e interfaces Java para controlar um aplicativo.

Todas as tags devem ter início e fim, como pode ser observado no exemplo, <STATES> indica o início e </STATES> indica o final dessa informação.

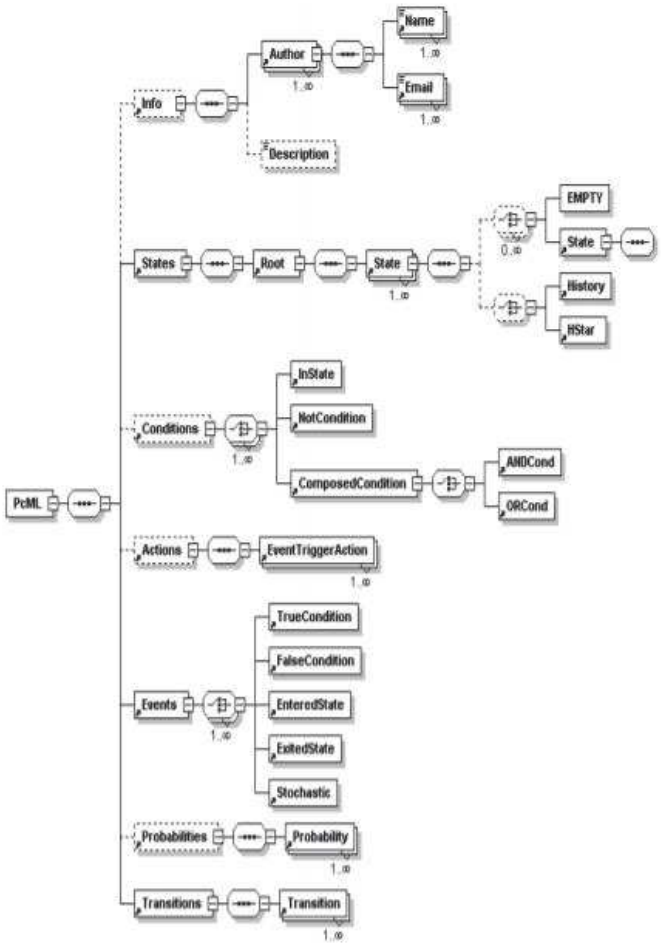
Para o desenvolvimento do projeto, foi necessário um estudo da interface padrão SCXML, pois o XML define todos os estados, transições, eventos e variáveis através do uso na máquina de estados. Além dos elementos estruturais dela, a linguagem SCXML também define a semântica de execução dessa máquina de estados.

Para se fazer uso do software desenvolvido (WEB-PerformCharts), é necessária a criação de um módulo principal na linguagem de programação C++ a fim de especificar o modelo para a obtenção de avaliação de desempenho por meio de Cadeias de Markov ou testes caixa preta. A solução inicial encontrada para tal problema seria através de uma interface gráfica, ainda não disponível. Dessa forma, teve-se uma solução intermediária a qual seria através de uma interface textual. Assim, uma linguagem de marcação baseada em XML foi criada com o objetivo de especificar e tratar modelos de desempenho.

2.4 PCML E INTERPRETADOR

A linguagem de marcação, PcML (AMARAL et al., 2004a) PerformCharts Markup Language, foi criada para especificação do sistema reativo em Statecharts. É uma linguagem de marcação baseada em XML, cujas tags, atributos e outras características representam os elementos usados em Statecharts na e especificação de sistemas reativos com o propósito de avaliação de desempenho (MARUYAMA et al., 2002).

Figura 2.4 - Diagrama PcML



Fonte: (AMARAL et al., 2004a)

Para o desenvolvimento do projeto, foi necessária a transformação da especificação em PcML do sistema em um programa C++ para poder gerar as suas medidas de desempenho. Para atingir tal objetivo, foi preciso criar algumas aplicações na linguagem Java. O programa em C++ consiste, basicamente, de chamar as funções para que:

a) criem estruturasde dados que correspondem à representação do sistema em Statecharts;

b) simulem a dinâmica do sistema através de estímulos dos eventos criando a cadeia de Markov;

c) resolvam a cadeia de Markov a fim de obter as probabilidades limites que são a base das medidas de desempenho.

O código na linguagem Java cria uma árvore hierárquica de nós/objetos, em que a busca aos nós e realizada através do uso de expressões XPath. Em seguida, são mostradas algumas linhas da descrição do sistema em PcML e o código correspondente em C++ gerado (AMARAL et al., 2004b).

2.5 WEB-PERFORMCHARTS

Inicialmente desenvolvida para avaliar o desempenho de sistemas reativos, representados em gráficos de estados (ARANTES et al., 2008a), a ferramenta WEB-PerformCharts (AMARAL et al., 2003), desenvolvida pelo Instituto Nacional de Pesquisas Espaciais (INPE), tem o objetivo de permitir que diferentes equipes trabalhem em um mesmo projeto através de uma interface baseada na web e na aplicação de banco de dados por meio da internet. Idealizada para ajudar software testers que trabalham em locais diferentes para cooperarem em projetos em comum e, assim, aproximar suas experiências a fim de beneficiar a qualidade do software.

Ainda em desenvolvimento, porém conta com a utilização de tecnologias como HTML, PHP e MySQL. Dessa forma, ela pode ser hospedada em um servidor Apache com um servidor operacional Linux de modo a ficar totalmente livre de custos com pacotes de softwares.

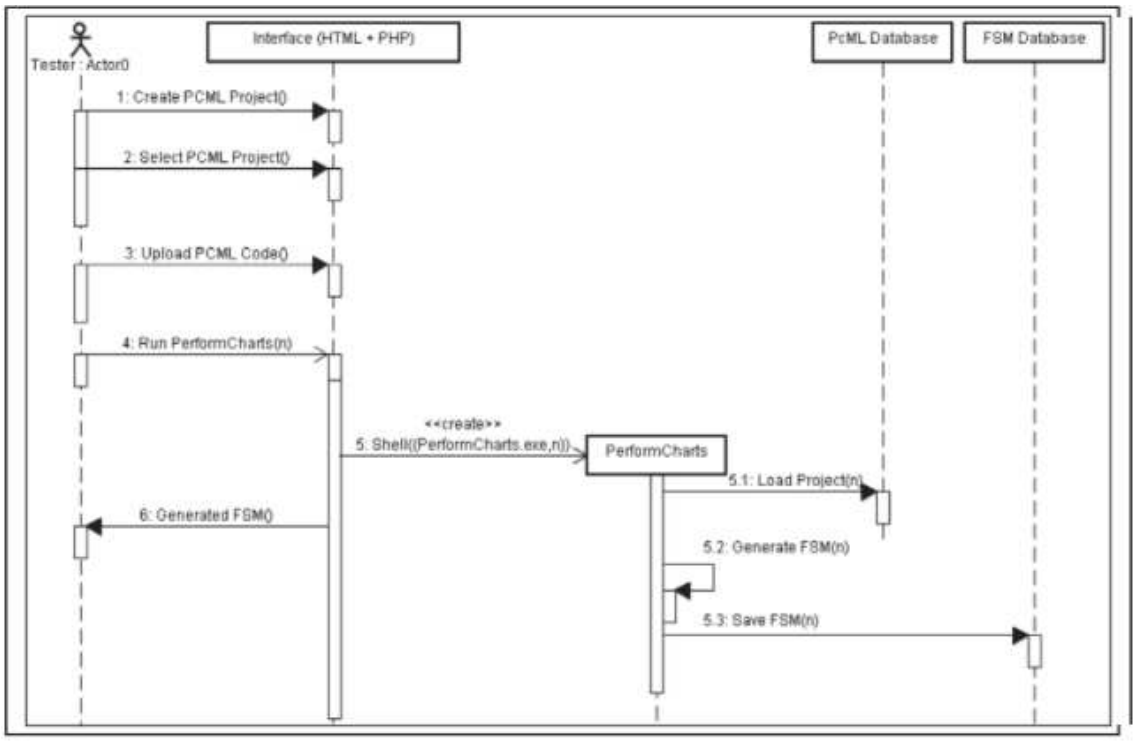
O WEB-PerformCharts ainda permite a geração de casos de teste baseados em modelos por meio de métodos formais, ou seja, gráficos de estados de linguagem formal. A ferramenta em questão apresenta quatro critérios de testes diferentes para gerar casos de teste baseados em modelo, fornecendo maior flexibilidade ao designer de teste. Ela converte uma especificação em Statecharts que é uma representação sintética de sistemas críticos e complexos, em uma Máquina de Estados Finitos.

No INPE, temos vários setores responsáveis pela construção de equipamentos que vão a bordo de satélites em que o software é embarcado, dessa maneira, utilizando a ferramenta em questão, a realização de testes acaba sendo facilitada. A ferramenta proposta baseia-se na especificação de sistemas reativos utilizando a técnica Statecharts e na geração de casos de teste para ela de acordo com alguns critérios disponíveis.

O principal objetivo do desenvolvimento do WEB-PerformCharts é aproximar as

equipes combinando sua experiência e know-how para beneficiar a qualidade do software. Assim, inicialmente, a ferramenta interpreta uma especificação em PcML, transformando esta em uma cadeia de Markov ou em uma MEF no padrão XML e, a partir da MEF, os critérios de teste são aplicados para, então, gerar casos de teste.

Figura 2.5 - Diagrama de sequência da ferramenta WEB-PerformCharts



Fonte: (AMARAL et al., 2003)

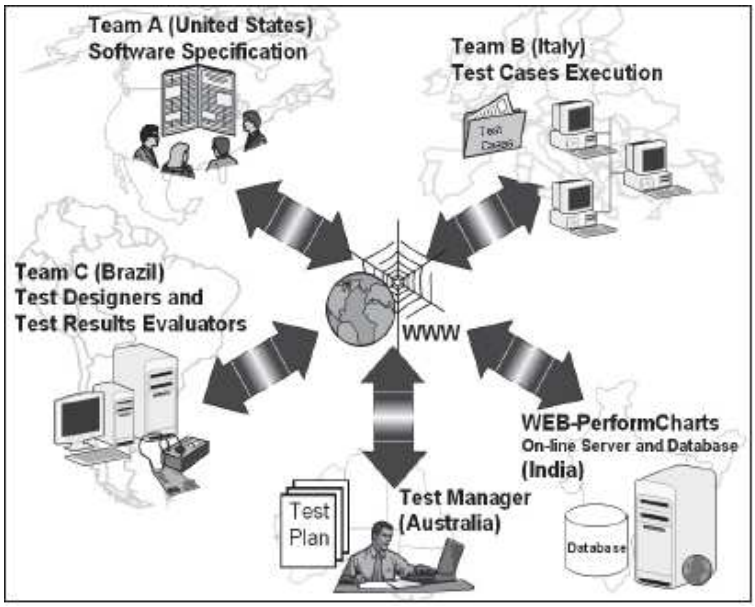
2.6 SISTEMAS COLABORATIVOS

O principal objetivo de sistemas colaborativos é ajudar usuários de diferentes locais envolvidos em um mesmo projeto, de modo a ter um grande apoio à comunicação, coordenação e cooperação. Basicamente, eles são aplicativos e/ou ferramentas que operam em redes que auxiliam o trabalho em equipe, assim, ajudando na troca de informações.

As aplicações neste sentido possuem diversos meios de promover qualquer usuário de internet, permitindo uma grande economia de custos e tempo, aumentando o

trabalho em conjunto e promovendo a eficiência e qualidade. Uma vez que todos os dados manipulados por um indivíduo podem ser imediatamente percebidos por todos os outros usuários em locais remotos.

Figura 2.6 - Exemplo do funcionamento de um Sistema Colaborativo



Fonte: (ARANTES et al., 2008b)

As aplicações WEB-based apresentam grandes vantagens, pois oferecem soluções de baixo custo, uma vez que nesta arquitetura o usuário pode utilizar qualquer sistema operacional e não requer nenhum outro software proprietário. Sem mencionar que, nos dias atuais, a grande maioria das pessoas possuem um fácil acesso à internet.

3 CONCLUSÃO

A importância de testes para a garantia da qualidade de produtos de software é clara nos dias atuais. Porém, é fato conhecido que as atividades associadas ao processo de teste demandam um tempo significativo do esforço de desenvolvimento de um produto de software. A automação de testes surgiu como uma tentativa de reduzir custos associados aos testes, aumentar a detecção de faltas e diminuir os ciclos de teste. Embora automação de testes não seja a "bala de prata" para resolver todos os problemas relativos a testes, se for corretamente planejada e implementada, ela pode ajudar a obter custo baixo e eficiência das atividades do processo de teste durante o ciclo de vida de desenvolvimento do software. Ambientes que possibilitem a geração automática de casos de testes, a partir da modelagem de softwares complexos, são relevantes em um contexto prático de desenvolvimento de software. A automação da geração de casos de testes é de alguma forma facilitada se o sistema em questão for representado por uma técnica formal, como Statecharts, MEFs ou por outras. Atualmente, existe um ambiente denominado WEBPerformCharts que gera casos de testes caixa preta baseados em alguns critérios quando a especificação de um software é modelada ou em uma MEF ou em Statecharts.

Um projeto descentralizado é uma tendência muito comum para empresas amplamente dispersas nos dias modernos, uma vez que pode resultar em uma grande economia de tempo e custos, assim, diminuindo as necessidades de viagens e infraestrutura. Desse modo, a fim de promover tal tendência para geração de casos de teste, a ferramenta WEB-PerformCharts foi planejada para ser usada via internet em uma abordagem de desenvolvimento distribuído.

A decisão de usar um banco de dados online como método de armazenamento permite que os designers de teste compartilhem seus projetos e, então, facilitando o controle de versões, já que seu gerenciamento é mais fácil do que copiar vários arquivos locais de diversos computadores.

A ferramenta WEB-PerformCharts possui dois níveis de acesso:

- Administradores, com acesso total a qualquer projeto;
- Usuários, com acesso apenas aos projetos criados por eles mesmos.

Os resultados apresentados no trabalho sugerem que a transição pode ser reconhecida como um critério rápido para obter um conjunto de casos de teste, mas gera

maior número de casos de teste para aplicações complexas. Dessa forma, a principal contribuição deste projeto é viabilizar uma ferramenta de suporte ao processo de teste em um ambiente distribuído por meio do desenvolvimento de uma ferramenta baseada na web.

REFERÊNCIAS BIBLIOGRÁFICAS

AMARAL, A. Geraçao de casos de testes para sistemas especificados em statecharts. **Master (Master at post graduation course in applied computing)—National Institute for Space Research, Sao José dos Campos, Brazil**, 2005. 2, 3, 7

AMARAL, A.; VELOSO, R.; VIJAYKUMAR, N.; FRANCÊS, C.; OLIVEIRA, E. On proposing a markup language for statecharts to be used in performance evaluation. **International Journal of Computational Intelligence**, v. 1, n. 3, p. 260–265, 2004. 9, 10

AMARAL, A. S. M. S.; VELOSO, R. R.; VIJAYKUMAR, N. L.; FRANCÊS, C. R. L.; OLIVEIRA, E. da L. Uma linguagem de marcação para statecharts para ser usada em avaliação de desempenho baseada em xml. **XXXVI Simpósio Brasileiro de Pesquisa Operacional**, p. 1–10, 2004. 11

AMARAL, A. S. M. S.; VIJAYKUMAR, N. L.; MARTINS, E. Geração automática de casos de teste de conformidade para software de aplicações em protocolos de comunicação. **Workshop em Computação Aplicada**, p. 1–6, 2003. 7, 11, 12

AMMANN, P.; OFFUTT, J. Introduction to software testing edition 2. 2017. 1

ARANTES, A. O.; SANTIAGO, V. A. de; VIJAYKUMAR, N. L.; SOUZA, E. F. D. Tool support for generating model-based test cases via web. **International Journal of Web Engineering and Technology iiWAS**, Inderscience Publishers Ltd, v. 9, n. 1, p. 62–96, 2014. 7

ARANTES, A. O.; VIJAYKUMAR, N. L.; JUNIOR, V. A. de S.; GUIMARÃES, D. Web-performcharts: a collaborative web-based tool for test case generation from statecharts. In: **Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services**. [S.l.: s.n.], 2008. p. 374–381. 11

_____. _____. In: **Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services**. [S.l.: s.n.], 2008. p. 374–381. 13

ARANTES, A. O.; VIJAYKUMAR, N. L.; SANTIAGO, V. A.; CARVALHO, A. R. Automatic test case generation through a collaborative web application. In: **IASTED International Conference on Internet and Multimedia Systems**

and Applications (EuroIMSA 2008), Innsbruck, Austria. [S.l.: s.n.], 2008. p. 17–19. 1

DELAMARO, M.; JINO, M.; MALDONADO, J. **Introdução ao teste de software**. [S.l.]: Elsevier Brasil, 2013. 1

HAREL, D. **Statecharts**. 1987. Disponível em: <https://statecharts.dev/>. 6

_____. Statecharts: A visual formalism for complex systems. **Science of computer programming**, Elsevier, v. 8, n. 3, p. 231–274, 1987. 1, 6, 8

LEE, D.; YANNAKAKIS, M. Principles and methods of testing finite state machines-a survey. **Proceedings of the IEEE**, IEEE, v. 84, n. 8, p. 1090–1123, 1996. 1

MARUYAMA, H.; TAMURA, K.; URAMOTO, N. **XML and Java: developing Web applications**. [S.l.]: Addison-Wesley Professional, 2002. 9

SIDHU, D. P.; LEUNG, T.-K. Formal methods for protocol testing: A detailed study. **IEEE transactions on software engineering**, IEEE, v. 15, n. 4, p. 413–426, 1989. 3

W3C. **XML: extensible markup language**. 1994. Disponível em: <http://www.w3.org/XML/Activity>. 5