Verificação e Validação de Softwares de Missão Crítica

Camila Fernandez Achutti e Ana Cristina Vieira de Melo
Instituto de Matemática e Estatística - Universidade de São Paulo
Processo FAPESP: 2012/16913-2

Introdução

"Os testes podem apenas mostrar a presença de erros, não sua ausência" *Dijkstra*

Apenas testar um **software de missão crítica** é insuficiente para garantir sua qualidade.
Solução: **Verificação + Validação**

Verificação : *Estamos construindo o software corretamente? O software está correto em relação à sua especificação (para qualquer entrada)?*

Validação : *Estamos construindo o software correto? Estamos resolvendo o problema?*

Motivação

Queremos viabilizar o uso dessas técnicas de verificação e validação sob o ponto de vista prático:

- Contribuir com a comunidade científica com um **estudo sobre verificação formal** de programas de sistemas críticos reais, já que grande parte ainda está restrita a problemas artificiais;
- **Facilitar a modelagens** de verificação formal;
- **Automatizar as tarefas** sobre modelos abstratos e sobre programas;
- Aumento do uso de Java para sistemas críticos e embarcados. Desenvolvimento da biblioteca **Javolution**.

O que foi feito

1. Modelagem e implementação de um sistema crítico de gerenciamento de dados espaciais SWPDC através de sua especificação formal feita pelo INPE [3];
2. Avaliação de estratégias e ferramentas de verificação e validação. Ao final desta etapa, selecionamos o verificador de modelos[1] **Java Pathfinder (JPF)** desenvolvido pela **NASA**, além de desenvolver **testes automatizados** baseado em cenários e um simulador de eventos espaciais chamado **PDCSimulador** para a validação;
3. Especificação e acompanhamento do desenvolvimento do PDCSimulador;
4. Aplicação das técnicas de verificação e validação selecionadas sobre o sistema desenvolvido;
5. Ao longo das etapas, acompanhamos a **cobertura de código** como métrica do aumento da qualidade do software;

Metodologia

Utilizamos a metodologia **ESPIRAL**

- Processo de desenvolvimento de software iterativo e incremental, que combina elementos da prototipação em etapas com a análise de riscos);
- Interação com pesquisadores do INPE para validação do sis ;

Ferramentas

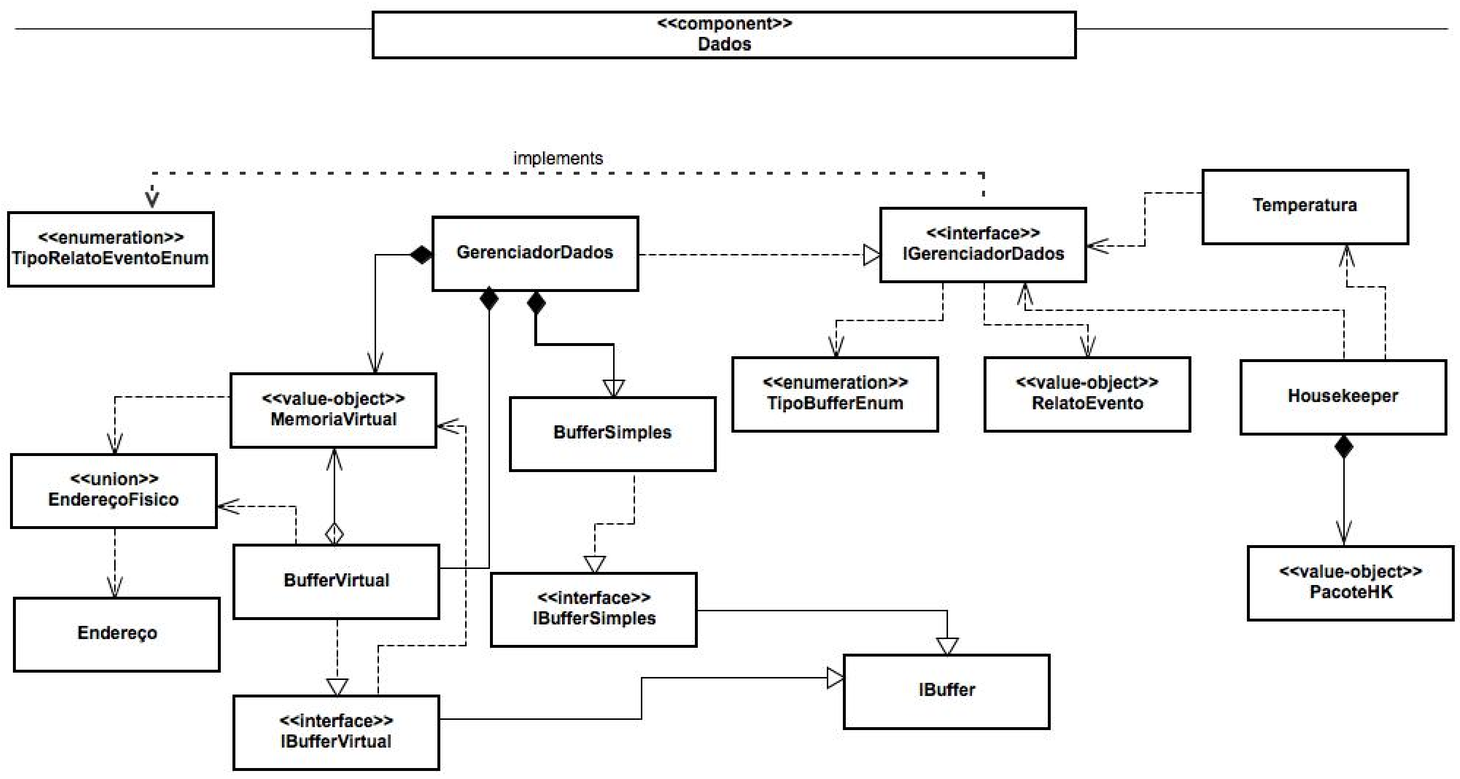
Acesso à ampla especificação do **Software Piloto Embarcado** sob um **Payload Data Handling Computer** produzido pelo **INPE (Instituto Nacional de Pesquisas Espaciais)**.

Principais ferramentas utilizadas:

- *Ambiente de desenvolvimento:* Eclipse;
- *Verificação:* Java PathFinder (JPF);
- *Validação:* JUnit e EclEmma.

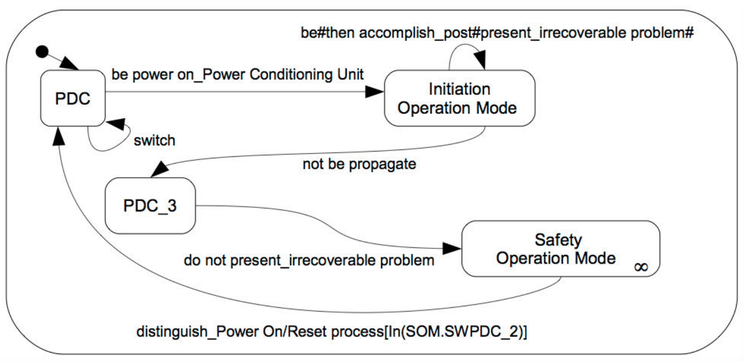
Resultados e Discussão

DIAGRAMA DE CLASSES DO COMPONENTE CENTRAL DO SWPDC



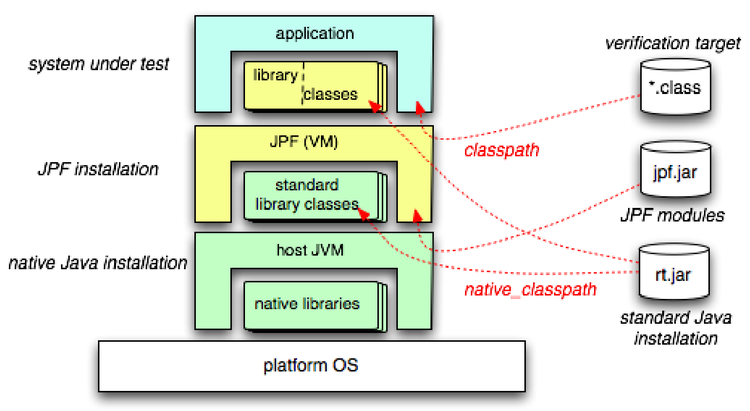
O SWPDC [3]:

- **contexto:** Qualidade de Software Embarcado em Aplicações Espaciais (QSEE).
- **requisitos:** em linguagem natural e Statecharts [3], como o ilustrado abaixo:

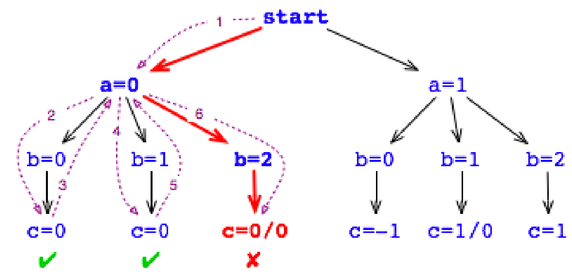


VERIFICAÇÃO:

O JPF - desenvolvido pela NASA - tem sua estrutura representada abaixo:

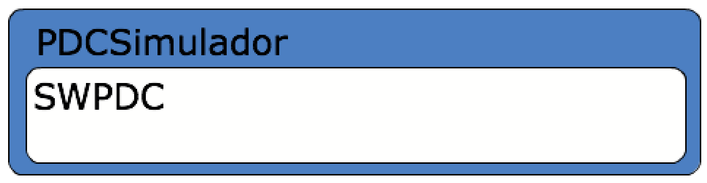


O seu funcionamento é ilustrado a seguir:



VALIDAÇÃO:

Foram desenvolvidos dois produtos, a seguir:



Cenário real para geração dos casos de teste:

SRS005: A transmissão de 12 amostras deve ser realizada a cada requisição.

Teste	Aquisição de temperatura bem sucedida.
Func.	AQUISIÇÃO
Pré-condição	SWPDC em NOMINAL; Emissão de comando de aquisição; PDC em estado consistênte
Proced.	Limpeza do buffer; Adquirir temperaturas; Verificar se cada amostra pertence ao intervalo; Armazenar as corretas no buffer
Res. Esperado	Armazenamento das amostrar corretas no buffer

Considerações finais

- Desenvolvimento formal de sistemas reais é, ainda, incipiente na literatura;
- Avaliação do JPF dentro da literatura de verificação formal de programas se fazia necessária;
- É necessário deixar evidente que a combinação de técnicas de verificação e teste proveem ao software mais qualidade se comparada ao uso de somente uma delas;
- As ferramentas e técnicas existentes de verificação ou validação ainda não são adequadas para o uso da grande comunidade de desenvolvedores.

Agradecimentos

Os devidos agradecimentos pela ajuda constante da Doutoranda Simone Hanazumi e do Mestrando Alexandre Locci Martins, o suporte de ambos durante o desenvolvimento desse projeto foi essencial para o seu andamento bem sucedido.

Referências

[1]Java Pathfinder. http://babelfish.arc.nasa.gov/trac/jpf/.
[2]Oracle: Java Technology. http://www.oracle.com/br/technologies/java/index.html. Acesso em: Jul. 2012
[3]Santiago Júnior, Valdivino Alexandre de: SOLIMVA: A Methodology for Generating Model-Based Test Cases from Natural Language Requirements and Detecting Incompleteness in Software Specications, 2011.
[4]Eclipse. http://www.eclipse.org/. Acesso em: Jul. 2012.