V&V Activities within two Brazilian Space Research Institutes

Miriam C. Bergue Alves*
Valdivino Alexandre de Santiago Júnior+
Nandamudi L. Vijaykumar+

NASA IV&V Workshop
Morgantown, WV
September 11-13, 2012

* National Institute for Space Research - INPE
São José dos Campos, SP, Brazil

+ Institute of Aeronautics and Space - IAE
São José dos Campos, SP, Brazil
Objective

This presentation relates some of the research initiatives of the Institute of Aeronautics and Space (IAE) and the National Institute for Space Research (INPE) with respect to the use of formal methods for the improvement of their V&V activities within the software development life cycle.
Outline

- Brazilian Space Program
- Presentation of IAE
- V&V Projects at IAE: Software Engineering Lab
- Presentation of INPE
- V&V Activities (Products/Projects) at INPE: CEA/LAC
- Conclusions
Brazilian Space Program

– **Rocketry**: launching and sounding rockets (IAE)

– **Space exploration**: satellites (INPE)

– **Launch sites**: Alcantara Launch Center and Barreirada do Inferno Launch Center (DCTA)
V&V Projects at IAE

• Use of topology for verification of deadlocks in concurrent systems

  – This project proposes a method that maps scenario-based specifications of concurrent systems, represented formally by MSCs (Message Sequence Charts), to a topological space. This mapping allows to formally verify these specifications for deadlock scenarios.

  – A simple “proof-of-concepts” prototype was constructed.
V&V Projects at IAE

• Use of topology for verification

- Requirements Specification
- Message Sequence Charts
- Progress graph elaboration
- Deadlock detection algorithm
- Identification of forbidden scenarios
- Generation of the deadlock scenarios
- Scenario analysis

Deadlock

V&V Projects at IAE

• Use of statechart-assertions for requirements specification, validation and verification

  – Formal computer-aided validation and verification of critical time-constrained requirements of the Brazilian Satellite Launcher flight software. It included the entire specification, validation, and verification process based on UML statechart-assertions and log files.
V&V Projects at IAE

• The SV&V process

1. System requirements analysis/Reqs Specification using natural language (NL)
   - Create UML statechart-assertions
   - Validate the statechart-assertions using JUnit-based testing
   - Automatically instrument source code

2. Build the instrumented version of the system on the VxWorks-based target thereby creating log files
   - Execute the resulting program on the VxWorks-based target

3. Import the log files into the SV&V environment
   - Create a namespace mapping
   - Run the JUnit verification tests against the assertions.

V&V Projects at IAE

- **SV&V – Some results**

<table>
<thead>
<tr>
<th>Validation Tests</th>
<th>Verification Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>220 tests (around 5 tests per assertion)</td>
<td>4 log files (4 tests per assertion)</td>
</tr>
<tr>
<td>220 JUnit classes - 1 JUnit class per test</td>
<td>4 JUnit class- 1 JUnit class per log file</td>
</tr>
<tr>
<td>132 success scenarios (around 60% of the scenarios)</td>
<td>31 assertions passed in all tests (around 70% of the assertions)</td>
</tr>
<tr>
<td>88 scenarios expect an assertion failure (around 40% of the scenarios)</td>
<td>13 assertions failed at least in one test (around 30% of the assertions)</td>
</tr>
</tbody>
</table>

V&V Projects at IAE

• Use of timed automata for model verification

  – A case study of a legacy space flight software system is being conducted, where the flight control and the flight events sequence chain of a satellite launcher are under study.

  – Use of model checking and a timed automata (TA) network to model the original requirements specification, incorporating new mission requirements and modifications.

  – Improve reliability in legacy systems evolution.
V&V Projects at IAE

- Use of timed automata for model verification

![Diagram showing the process of using timed automata for model verification.](image)
V&V Projects at IAE

• Use of Event-B and Rodin Platform

  – The UML-B and Event-B language are being used for the models elaboration of a case study that involves the control of the first stage of a launch vehicle, with the support of the computer-aided tool Rodin Platform (Rigorous Open Development Environment for Complex Systems).

  – The work is at its initial phases of creating and refining the models, with emphasis to the improvement of the system dependability.
V&V Projects at IAE

Satellite Launcher's concurrent system engineering process

- Flight Dynamics
- Flight control
- Electrical network

Software Development

- Software System Requirements specification
  - Modeling
  - Refinement
  - Proving

Analysis

- RAMS Analysis
- Safety reqts
- can generate new scenarios

Event-B and Rodin Platform: the process

can identify missing and incomplete reqts

v&v

can generate improvements and corrections

Analysis – Design – Implementation ➔ Acceptance

V&V Projects at IAE

- Use of Event-B and Rodin Platform: example

INPE’s Organization Chart

INPE ⇒ 7 Postgraduate programs.
V&V Activities at INPE: Products

• **Automated Test Case Generation based on Statecharts (GTSC):**
  
  – Model-based test case generation based on Statecharts ⇒ four test criteria (*all-transitions, all-simple-paths, all-paths-k-C0-configuration, all-paths-k-configurations*) from the Statechart Coverage Criteria Family (SCCF);

  – Model-based test case generation based on FSM ⇒ three test criteria (*DS, UIO, H-switch cover*) where one (H-switch cover) is a new test criterion.
GTSC 2.0: Main Interface
### V&V Activities at INPE: Products

**WEB - PerformCharts**

<table>
<thead>
<tr>
<th>Step</th>
<th>Event</th>
<th>State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EB9</td>
<td>CountingTimeWaitingExpid</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>WaitingTimeExpired</td>
<td>IdleWaitingSync</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>EB9</td>
<td>CountingTimeWaitingExpid</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ExpiryRec</td>
<td>CountingTimeWaitingType</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>WaitingTimeExpired</td>
<td>IdleWaitingSync</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>EB9</td>
<td>CountingTimeWaitingExpid</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ExpiryRec</td>
<td>CountingTimeWaitingType</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Type=Rec</td>
<td>CountingTimeWaitingSize</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>WaitingTimeExpired</td>
<td>IdleWaitingSync</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>EB9</td>
<td>CountingTimeWaitingExpid</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ExpiryRec</td>
<td>CountingTimeWaitingType</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Type=Rec</td>
<td>CountingTimeWaitingSize</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Size=Rec</td>
<td>CountingTimeWaitingData</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>WaitingTimeExpired</td>
<td>IdleWaitingSync</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>EB9</td>
<td>CountingTimeWaitingExpid</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>ExpiryRec</td>
<td>CountingTimeWaitingType</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Type=Rec</td>
<td>CountingTimeWaitingSize</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Size=Rec</td>
<td>CountingTimeWaitingData</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Data=Rec</td>
<td>CountingTimeWaitingChecksum</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>WaitingTimeExpired</td>
<td>IdleWaitingSync</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>NotEB9</td>
<td>IdleWaitingSync</td>
<td></td>
</tr>
</tbody>
</table>
V&V Activities at INPE: Products

• SOLIMVA ⇒ A methodology aiming at:

  – the generation of model-based system and acceptance test cases considering Natural Language (NL) requirements deliverables (artifacts) ⇒ Version 1.0 (software testing);

  – the detection of incompleteness in software specifications ⇒ Version 2.0 (software inspection with the aid of formal verification);

  – Formal Verification (Model Checking) of UML-based software ⇒ Version 3.0 (Formal Verification in the traditional approach).
The SOLIMVA methodology 1.0: Workflow

- Define and Input Dictionary
- Define Scenarios
- Select and Input NL Requirements
  - [dictionary update]
  - [else]
  - Generate Model
    - [manual refinement]
    - [else]
    - Generate Abstract Test Cases
    - Generate Executable Test Cases
      - [more scenarios]
      - [end of scenarios]
The SOLIMVA methodology 1.0: Workflow

Software Testing:
- Model-Based Testing (Statechart-based testing; GTSC);
- Combinatorial Designs;
- Part Of Speech Tagging (SOLIMVA tool 1.0);
- Word Sense Disambiguation (SOLIMVA tool 1.0).

Define and Input Dictionary
Define Scenarios
Select and Input NL Requirements
[dictionary update] Update Dictionary
[else]
Generate Model
[manual refinement]
[else]
Generate Abstract Test Cases
Generate Executable Test Cases
[more scenarios]
[end of scenarios]
Clear Requirements and Model
The SOLIMVA methodology 1.0: Tool (1.0)

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Dictionary</th>
<th>Model Generation</th>
<th>Test Case Generation</th>
<th>Analysis of Defects</th>
<th>Help</th>
</tr>
</thead>
<tbody>
<tr>
<td>SR5001: The PDC shall be powered on by the Power Conditioning Unit.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SR5002: The PDC shall be in the Initiation Operation Mode after being powered on. The SWPDC shall then accomplish a P...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SR5003: If PDC does not present any irrecoverable problem, after the initiation process, the PDC shall automatically ent...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POCP001: The PDC can only respond to requests (commands) from OBDH after the PDC has been energized for at least 1 ...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB001: The OBDH shall send VER-OP-MODE to PDC.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB002: The PDC shall switch each Event Pre-Processor (EPP Hx, x = 1 or 2) on or off independently, when the OBDH s...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FECP001: Each EPP Hx can only respond to requests (commands) from PDC after each EPP Hx has been energized for at l...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SR5004: The OBDH should wait 600 seconds before asking for a Housekeeping Data frame.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SR5005: Housekeeping data transmission shall start with prep-Hk. After that, the OBDH can send several b-data-Hk to P...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB003: The OBDH shall send CH-OP-MODE-Nominal to PDC.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB004: The OBDH shall send CH-OP-MODE-Safety to PDC. After that, the PDC shall be in the Safety Operation Mode.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB005: The PDC shall switch each Event Pre-Processor (EPP Hx, x = 1 or 2) on or off independently, when the OBDH s...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB008: After switching both EPPHxs off via PDC, the OBDH shall switch the PDC off via the Power Conditioning Unit.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Project Opened.
Dictionary Done.
Tuples generated.
Model generated.

The SOLIMVA methodology 2.0: Workflow

Software Inspection with the aid of Formal Verification:
- Model Checking;
- Specification Patterns;
- k-Permutations of n Values of Variables (Characteristics).

The SOLIMVA methodology 3.0: Workflow

Formal Verification (UML)

- Define and Input Dictionary
  - Analyze Incompleteness
    - [else] [incomp detected] Improve Specifications
  - Define Scenarios
    - Select and Input NL Requirements
      - [dictionary update] Update Dictionary
      - [else] Generate Model
    - Clear Requirements and Model
      - [manual refinement] [else] Generate Abstract Test Cases
        - Generate Executable Test Cases
          - [more scenarios] [end of scenarios]
    - [end of scenarios]

- Select Use Case
  - Identify Scenarios
    - Start Formal Verification
      - Select Diagrams
        - Select Requirements
          - Generate NuSMV Model
            - Simulate Model
              - [mode's defects] [else]
                - Apply Model Checking
                  - Generate Report of System Defects based on Counter Examples
                    - [more scenarios] [end of scenarios]
V&V Activities at INPE: Products

- **Quality of Space Application Embedded Software – Automated Software Testing (QSEE-TAS):** Automated test case execution, Automated test process documentation generation.

V&V Activities at INPE: Application to Projects

• Alpha, Proton and Electron Monitoring Experiment in the Magnetosphere (APEX).
  – Products ⇒ GTSC, WEB-PerformCharts, QSEE-TAS.

• Quality of Space Application Embedded Software (QSEE) – Software for the Payload Data Handling Computer (SWPDC).
  – Products ⇒ GTSC, WEB-PerformCharts, SOLIMVA, QSEE-TAS.

• protoMIRAX Scientific Experiment (Balloon application).
  – Products ⇒ GTSC, SOLIMVA.

APEX

IUT:
- Command Recognition Component of the APEX embedded software;
- Simulated version (Java).

QSEE/SWPDC: Physical Architecture

- **OBDH (Simulation Software)**
  - DAQ
  - USB
  - Converter

- **PDC (SWPDC)**
  - ADC
  - RS-232 connection

- **CEU**
  - EPP H1 (Data Simulation)
  - EPP H2 (Data Simulation)

- **IONEX**
  - Temperature Simulation
  - RS-232 connection

- **Scientific Instruments**

- **Temperature Simulation**

*NASA IV&V – Sept. 2012*
QSEE/SWPDC: Example of Statechart model
\neg \exists[(\neg (prim = valprim_i \land sec_j = valsec_{t1}) \cup ((prim = valprim_i \land \neg (prim = valprim_i \land sec_j = valsec_{t1}))]]

\[
\begin{align*}
\text{<nominal, sci, t:Sci, sciDataInc, initAccq>} & \rightarrow \text{<nominal, sci, relAns, sciData, initAccq>}
\rightarrow \text{<nominal, dmp, txDmp, dmpDataInc, stopDAccq>}
\rightarrow \text{<nominal, dmp, propDmp, cmdRec, stopDAccq>}
\rightarrow \text{<nominal, sci, txSciEBuf, noData, initAccq>}
\rightarrow \text{<nominal, sci, txSciEBuf, noData, initAccq>}
\rightarrow \text{<nominal, hK, propHK, cmdRec, stopDAccq>}
\end{align*}
\]
QSEE/SWPDC: Remarks

• GTSC $\Rightarrow$ test suites with more than 300 test cases.

• SOLIMVA 1.0 $\Rightarrow$ better strategy with test objectives clearly separated according to the directives of Combinatorial Designs.

• SOLIMVA 1.0 $\Rightarrow$ Executable Test Cases predicted behaviors that did not exist (Expert's strategy).

• SOLIMVA 2.0 $\Rightarrow$ 362 CTL properties formalized, 21 incompleteness defects detected.
QSEE/SWPDC: IV&V

• Test Case Generation ⇒ Model-Based Testing (FSM).

• Test Case Execution ⇒ QSEE-TAS tool.

• Test Results Evaluation ⇒ Four-step process:
  – Observation of test results (QSEE-TAS interface);
  – Assignment of a preliminary verdict;
  – Meeting (every week) ⇒ IV&V team and customer representatives at INPE to evaluate the test reports;
  – Final verdict ⇒ Non-Conformance Record (NCR).

protoMIRAX Scientific Experiment
Conclusions

- Main V&V activities, products and projects in the area of formal V&V of safety-critical space software systems within IAE-LES and INPE (CEA/LAC).

- More confidence in the right choice of techniques to be used in each phase of development and in each part or component of the space software.

- Importance of computer-aided tools to support the formal V&V process.

- Efforts to bridge the gap between the state of the art and the state of the practice (application of research results to space projects development).
THANK YOU!

Miriam C. Bergue Alves:
miriammcba@iae.cta.br

Valdivino Alexandre de Santiago Júnior:
http://www.cea.inpe.br/~valdivino/

Nandamudi L. Vijaykumar:
http://www.lac.inpe.br/~vijay/Welcome.html