Cost-Effective IV&V of Flight Software via a Compiled Shared Library and Independent Flight Simulation

D.S. Stauffer
Rob Wichtman
J. Benton Derrick
Independent Flight Software Testing Challenge

I need you to test this software and make sure it works

I can’t be specific but it needs to control a flight vehicle in the real-world

Sure software V&V, what are the requirements?

What conditions in the real-world?

All conditions in the real-world

No...

Hmmm... Can I see the Code?

©Scott Adams
What Capability Maturity Model Integration (CMMI) Says About V&V

Verification
- Verification addresses whether the work product properly reflects the specified requirements

“you built it right”

Validation
- Validation demonstrates that the product fulfills its intended use

“you built the right thing”
Flight Software V&V Problem

How does one perform V&V on mission-critical flight software without ever seeing the code?

The Flight Software is provided as a compiled shared library, that means not having access to the flight source code or any of the decision logic. That is, the flight software is a Black Box.

Flight Software can’t be tested in isolation; it can only be evaluated as to how it performs in its intended environment.

Black box testing is required for V&V of compiled flight software
In traditional black box testing, the test cases are built around what the application is supposed to do. i.e. the specifications, requirements and/or mission design concept.

Q: How do we know if the S/W States have been properly sequenced?
Q: How do we know if all the logic paths have been exercised?
Ans: We can’t. We don’t have access to internal states or know what the logic paths are. All we know is if the flight software interacts appropriately with the simulated environment.*

* This situation is similar to the problem of consciousness that Turing tried to address. That is “how do you know if a machine is intelligent?” His answer was that you simply interact act with it and observe if it is acting intelligently.

Test cases must cover all expected variations in the physical models
Flight Software Testing Approaches

Vendor Simulation

Flight S/W Testing with Compiled Simulation

Formal IV&V with Flight S/W Source Code

MegaSim Simulation

Environment  Aerodynamics  Propulsion  Vehicle Dynamics

Navigation  IMU

Flight S/W

Actuators  Staging Disretes

MegaSim Simulation

Environment  Aerodynamics  Propulsion  Vehicle Dynamics

Flight S/W

Actuators  Staging Disretes

Flight S/W Testing with Compiled Shared Library and Independent Simulation
Flight Software Testing Spectrum

Flight S/W Testing with Compiled Simulation
- May be able to change simulation inputs
- Limited access
- Hidden logic
- Fixed structure limits possible tests
- Able to monitor trajectory & evaluate performance
- Protects intellectual property of flight software developer
- Lower cost

Flight S/W Testing with Compiled Shared Library and Independent Simulation
- Flight software is compiled into DLL (MS Windows OS) or DSO (Unix-Like OS)
- Application Programming Interface (API) to flight software required
- Simulation is controlled by independent party
- Protects intellectual property of the flight software developer
- Allows more thorough testing of flight software than compiled flight model testing
- More in depth failure mode evaluation possible
- Provides Monte Carlo & stress testing analysis capability

Formal IV&V with Flight S/W Source Code
- Develop flowcharts and calling trees
- Perform unit & module tests
- Analyze algorithms
- Integrate into independent simulation
- Intellectual property of flight software developer is exposed
- Costly

Lower Cost, Lower Quality
Higher Cost, Higher Quality
Library Linking Methods

**Static Linking**

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Certain that libraries exist</td>
<td>• Updates to the library require rebuilding the application</td>
</tr>
<tr>
<td>• Certain that libraries are the right version</td>
<td>• One application file (larger file size)</td>
</tr>
<tr>
<td>• One application file (simple to distribute)</td>
<td>• Library determination must be done at compile time</td>
</tr>
<tr>
<td>• May be significantly faster than loading libraries at run-time</td>
<td></td>
</tr>
</tbody>
</table>

**Dynamic Linking**

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Updates to the library do not require rebuilding the application</td>
<td>• Application and library version mismatch</td>
</tr>
<tr>
<td>• Smaller application size</td>
<td>• Multiple files to distribute</td>
</tr>
<tr>
<td>• Library determination can be decided at run-time</td>
<td>• Loading libraries at run-time may incur a significant performance hit</td>
</tr>
</tbody>
</table>

**Diagram**

- **Static Linking**
  - Application Object Code
  - Build
  - Static Linking
  - Library1
  - Call Procedure1
  - Call Procedure2
  - Call ProcedureX
  - Application

- **Dynamic Linking**
  - Application Object Code
  - Build
  - Call Procedure1
  - Call Procedure2
  - Call ProcedureX
  - Library1
  - Application
The real world can vary greatly. How much tolerance is enough?
Applications – Stress Testing

Define Stress Parameter
• Set to Nominal Value

Perform Monte-Carlo Trajectory Simulations

MegaSim Simulation

Environment  Aerodynamics  Propulsion  Vehicle Dynamics

Navigation  Flight S/W
IMU  Shared Library  Actuators  Staging Discretes

Monitor Level of Performance
• Miss Distance
• Vehicle Control Maintained
• All Modes Transitioned
• DLL Executes w/o Fault

<table>
<thead>
<tr>
<th>Stress Parameter</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change from Nominal</td>
<td>Degree of Tolerance (DoT)</td>
</tr>
</tbody>
</table>

How much variation can we tolerate?
Applications – Stress Testing

- Individual stressors are varied one by one, while leaving the other parameters randomly varied.
- This makes it possible to pinpoint specific weaknesses and vulnerabilities.
- It is also possible to determine if the system’s performance gracefully degrades or abruptly breaks.

*Degree of Tolerance Is defined as the amount of parameter variation. The system can withstand And still meet acceptable requirements.
Applications – Beyond IV&V

Shared Memory

Share Large Amounts of Data

Sockets

Scalable Distributed Systems
Conclusions

• V&V quality is directly related to how well one replicates the real world
  – IV&V with an independent simulation uses an alternate replication of the real world
• Testing of a compiled shared library of flight software is a cost-effective method for IV&V while protecting intellectual property
• Stress testing is a great way to deal with lack of transparency in black box testing

• Questions?