Status of the AIAA Modeling and Simulation Format Standard

E. Bruce Jackson
NASA Langley Research Center
Hampton, Virginia

Bruce L. Hildreth
SAIC, Inc.
Lexington Park, Maryland

Royal Aeronautical Society
5 June 2008
Overview

• Introduction
  – Background of problem
  – Motivation for standard format
  – Goal of effort

• Proposed solution
  – AIAA/ANSI/ISO standard
  – Applications for standard

• Examples
• Tools
• Status
• Summary

http://daveml.nasa.gov
Background

Because…
• Independent development of R&D sim labs
• Incomplete standards - just a few, locally applied
• Different architectures (data table formats, variable names, measurement axes, moment transfer method)

And with…
• Increased reliance on flight simulation for research & development and procurement
• Increased need for collaboration and teaming

We should explore…
• Possibilities for increasing productivity
• More *portable* simulation models is the goal
Motivation

- **Amazing fact:** Flight dynamic models from different organizations are usually **incompatible** with each other
  - Delivering or updating a flight dynamics model between contractor, government, or training simulation facilities can take **months**
  - There are valid, historical reasons for this
  - Still, this is a major hindrance to cooperative research & updating training fidelity

- Think PC vs. Mac, only less compatible!
- Simulation rehosting is similar to swapping out train trucks (wheel assemblies) when changing rail gauge…
Changing rail gauge in Manzhouli, China
Example for single aircraft type

• A 2002 NASA/SAIC estimate for one fighter type:

<table>
<thead>
<tr>
<th>Pilot Training Devices</th>
<th>Research Simulations</th>
</tr>
</thead>
<tbody>
<tr>
<td>31 simulators</td>
<td>28 simulators</td>
</tr>
<tr>
<td>6+ model types</td>
<td>16 locations</td>
</tr>
</tbody>
</table>

• 59 mostly incompatible simulations (built by different vendors/software architecture)
• Potential savings: $6 M per year for single type
Solution: standard exchange format

Only need to build one set of import/export tools
Project background

- Hildreth (1998) proposed dev of AIAA standard
- Hildreth and Jackson (2002) showed $6 M savings for one aircraft type
- Dynamic Aerospace Vehicle Exchange Markup Language (DAVE-ML) proposed in 2002
- Successful model exchange between NASA Ames and NAVAIR Patuxent River held in 2004
- Initial DAVE-ML grammar in use for aero models (DSTO, NASA Langley, NAVAIR)
Proposed solution

- AIAA/ANSI draft model exchange standard
  - Standard variable names; axis systems per AIAA/ANSI R-004-1994
  - Model implemented in XML extension: DAVE-ML

- Applications
  - Static subsystem models (the major components of a flight model)
    - Aerodynamic models
    - Mass/inertia models
    - Performance models
  - Dynamic models
    - Integration of states are not explicitly included in this standard
    - External integration of states is possible, however
    - Will be a backwards-compatible XML implementation
    - Will not include standard equations of motion (F=ma)

http://daveml.nasa.gov
The proposed standard includes:

• An *exchange* standard for flight dynamic models
  – Does not require internal adoption of format
  – Does not require replacement of legacy code & tools

• Standard variable names for common parameters
  – Includes units of measurement for most parameters
  – Describes how to construct & interpret new names
  – Does not require internal adoption of new names

• Incorporates existing standard for axis systems

• Initial application: encoding of aero & mass models
  – These represent sizeable portions of flight simulation data
DAVE-ML introduction

Dynamic Aerospace Vehicle Markup Language
– Based on Extensible Markup Language (XML)
– Currently includes
  • Function data tables (N-dimensional)
  • Non-linear build-up equations
  • Units, sign convention
  • Provenance of model
  • Uncertainty/statistical bounds
  • Static check case data
– Self-documenting text file

http://daveml.nasa.gov
DAVE-ML examples/test cases

F-16 subsonic aero model (from NASA TM)
- 51 variables, 18 tables, 744 points
- Switches & absolute value nonlinear elements
- 17 verification checkcases included
- 154 KB file with 2,712 lines

Concept development lifting body aero model
- Supersonic and subsonic regimes
- Polynomial equations; tables of coefficients
- 361 variables, 168 tables, 6,240 values
- 24 verification checkcases included
- 1.2 MB file with 22,299 lines

These examples are available on project website

http://daveml.nasa.gov
Other DAVE-ML uses

NASA/Boeing Blended-wing-body (X-48A)
- Complete aero model in 12.5 MB text file
- 22 breakpoint sets, 97 tables (up to 5-D)
- 256 functions using 716,826 data points
- Compresses to 2.6 MB
- Parsed in 5 seconds on average PC

Boeing X-37 air-launched test vehicle
- Boeing data in Excel tables
- Langley simulation in Simulink
- DAVE-ML used as intermediate format
- Generated multiple Simulink models, one per Excel file

http://daveml.nasa.gov
Other DAVE-ML uses

Orion launch abort vehicle
(CEV with the escape tower)
- 46,332 data points, 3 inputs, 8 outputs
- 385 KB text file
- Includes uncertainty & checkcases

http://daveml.nasa.gov
Available DAVE-ML tools

- DAVE-ML 2.0 format specification DTD
- DAVE-ML 2.0 reference manual
- JANUS (C++ library) - Australian DSTO
- LaSRS++ (C++ library) - NASA Langley
- NASA Ames FTP tool (import/export Perl scripts)
- NASA Langley XSLT conversion script
  DAVE-ML => HTML (self-documenting!)
- DAVEtools (Java) - NASA Langley:
  DAVE-ML => Simulink (popular analysis engine)
- SAIC developing Python-based DAVE-ML editor
  (supports Joint Next-Gen Threat System, NGTS)
Status

- Standard format is in use within parts of NASA and Australian DSTO; being applied to Joint Next-Gen Threat System (NGTS) by NAVAIR

- Draft standard submitted to main AIAA Standards organization by AIAA Modeling and Simulation Technical Committee October 2007; currently in editing

- Main application is exchange of aero models, but can be used to specify entire flight dynamics package.

- Wider review (public comment period) after AIAA publishes draft standard
Follow-on opportunities

• Add time history check data format (possibly HDF-5?)
  – Needed for dynamics verification
  – Would like to reuse JSF time-history HDF-5 tools

• Add support for vectors & matrices
  – Current request from DSTO

• Add specification for integration methods
  – Initial condition logic, numerical method selection
  – Add past value / zero-order-holds

• Add support for subsystems/libraries
  – Currently one monolithic model
Summary

• Draft standard for static (aero) models developed; in review by AIAA/ANSI
• Initial set of tools are appearing
• In use by Australian DSTO for threat models
• US Navy building first set of threat models
• Further development for full dynamic models
• Seeking wider (voluntary) participation

http://daveml.nasa.gov
Backup slides
Previous model standards efforts

- MODSIM
  Air Force initiative, mid-80s, standard CPUs/SW

- SIMNET/WARNET - DIS - HLA
  1980s DARPA project to network/fight existing sims

- Project 2851 - SEDRIS
  Standard visual/terrain model database formats

- Internal NASA: NASP project
  Late 80s internal NASA: Fortran-only, but standard table format, axes, variable names (8 characters)
<xml version="1.0" standalone="no"?>
<!DOCTYPE DAVEfunc SYSTEM "DAVEfunc.dtd">
<DAVEfunc>
 <variableDef
    varID="angleOfAttack_d"
    name="Alpha" units="deg"
 />
 <variableDef
    varID="CmAlfa" name="Cma" units=""
 />
 <breakpointDef
    bpID="angleOfAttack_d_bp1">
    <bpVals>
      0, 10, 18, 20, 22, 23, 25, 27, 30
    </bpVals>
 </breakpointDef>
 <griddedTableDef gtID="CmAlfa_Table1">
    <breakpointRefs>
      <bpRef bpID="angleOfAttack_d_bp1"/>
    </breakpointRefs>
    <dataTable>
      -0.3, -0.2, -0.1, -.08, -0.05, -0.05, -0.07, -0.15, -0.6
    </dataTable>
 </griddedTableDef>
 <function name="Cm_alpha_func">
    <independentVarRef varID="angleOfAttack_d"/>
    <dependentVarRef varID="CmAlfa"/>
    <functionDefn>
      <griddedTableRef gtID="CmAlfa_Table1"/>
    </functionDefn>
 </function>
</DAVEfunc>
Previous XML syntax encodes this function:

Possible to encode, but not shown, are

- buildup equations (combinations of functions)
- confidence bounds associated with this function
Janus API library

- Developed by Australia’s Defence Science & Technology Organization (DSTO)
- Janus is a C++ library to read/write and manipulate DAVE-ML files
- Reads DAVE-ML directly at run-time
- AES-256 encryption for classified models
- Associated Matlab code to read/write DAVE-ML
- Available under Open Source license from DSTO

http://daveml.nasa.gov