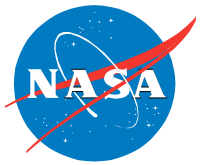
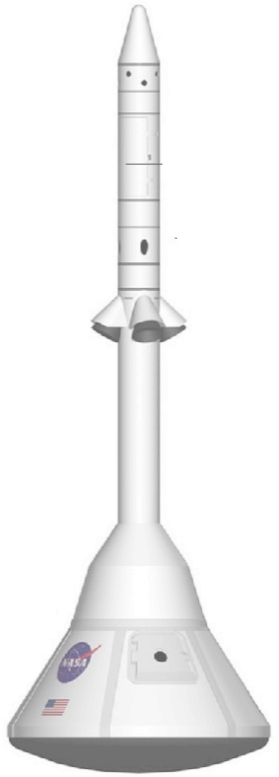


Simulation Environment for Orion Launch Abort System Control Design Studies

J. Dana McMinn
&
E. Bruce Jackson
NASA Langley Research Center
Hampton, Va

David M. Christhilf
Lockheed-Martin Engineering Services Company
Hampton, Va



Outline

- Motivation
- Introduction
- Simulation Tool
- Version Control
- Aero Model Import
- Control System Development
- Team Communication
- Lessons Learned /
  Concluding Remarks

Motivation

- Constellation program
  - Johnson Space Center leads development for the crew module
  - Langley Research Center leads the launch abort system development

- Tasks
  - requirements definition
  - trade studies
  - controller design
  - Insight and oversight responsibilities

Introduction

"Simulation Environment for Orion Launch Abort System Control Design Studies"

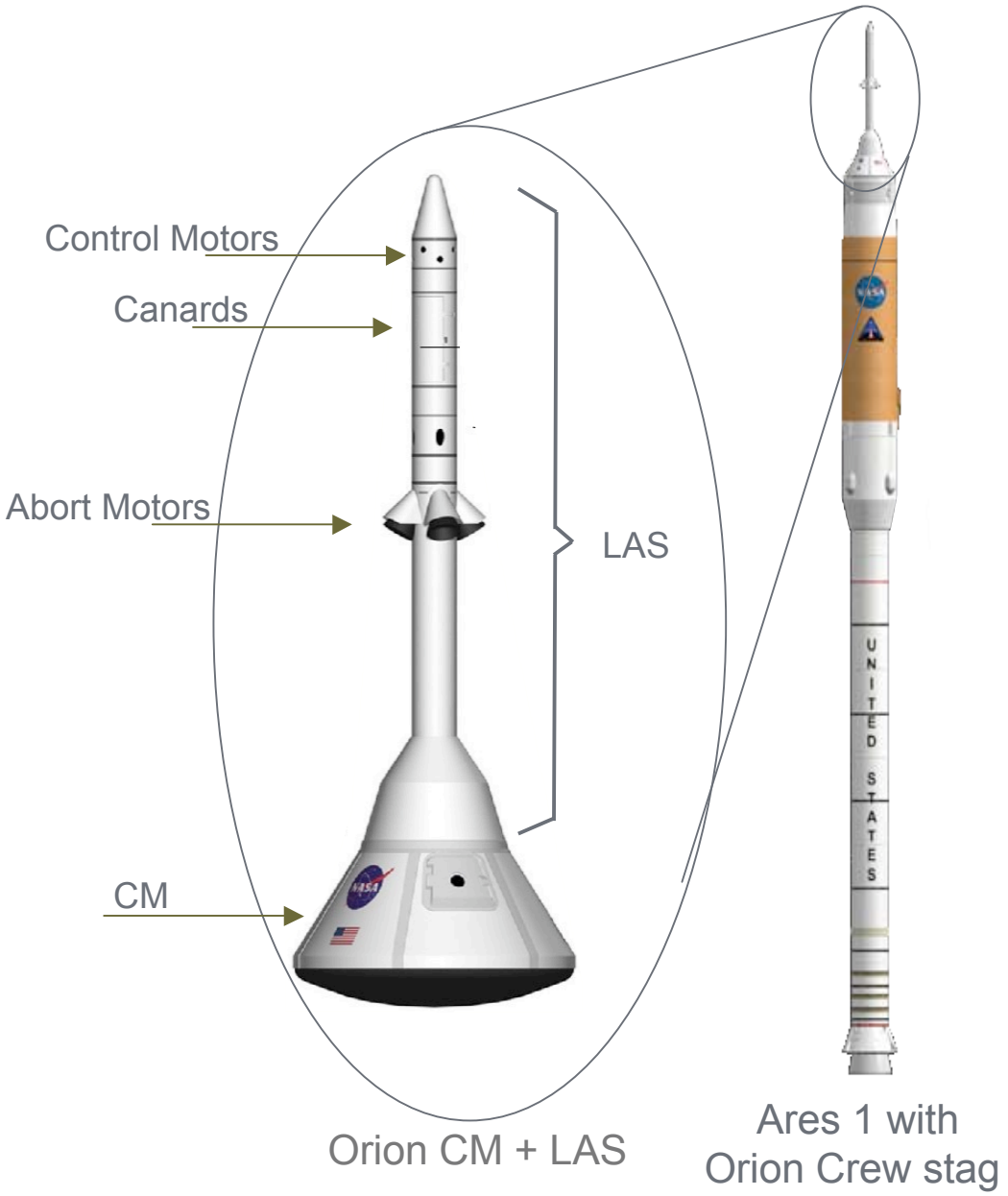
- Simulation environment
  - simulation tool
  - supporting applications
  - team processes
- Orion and Ares
- Control Design Studies
  - Sparks & Raney, "Crew Exploration Vehicle Launch Abort System Controller Performance Analysis"
  - GN&C Session 29

Ares 1 with
Orion Crew stage

Introduction

- Orion has an Apollo-like launch abort tower for first stage aborts
- Uses active rather than passive stability
- NASA & industry team responsible for control design and analysis of the launch abort system

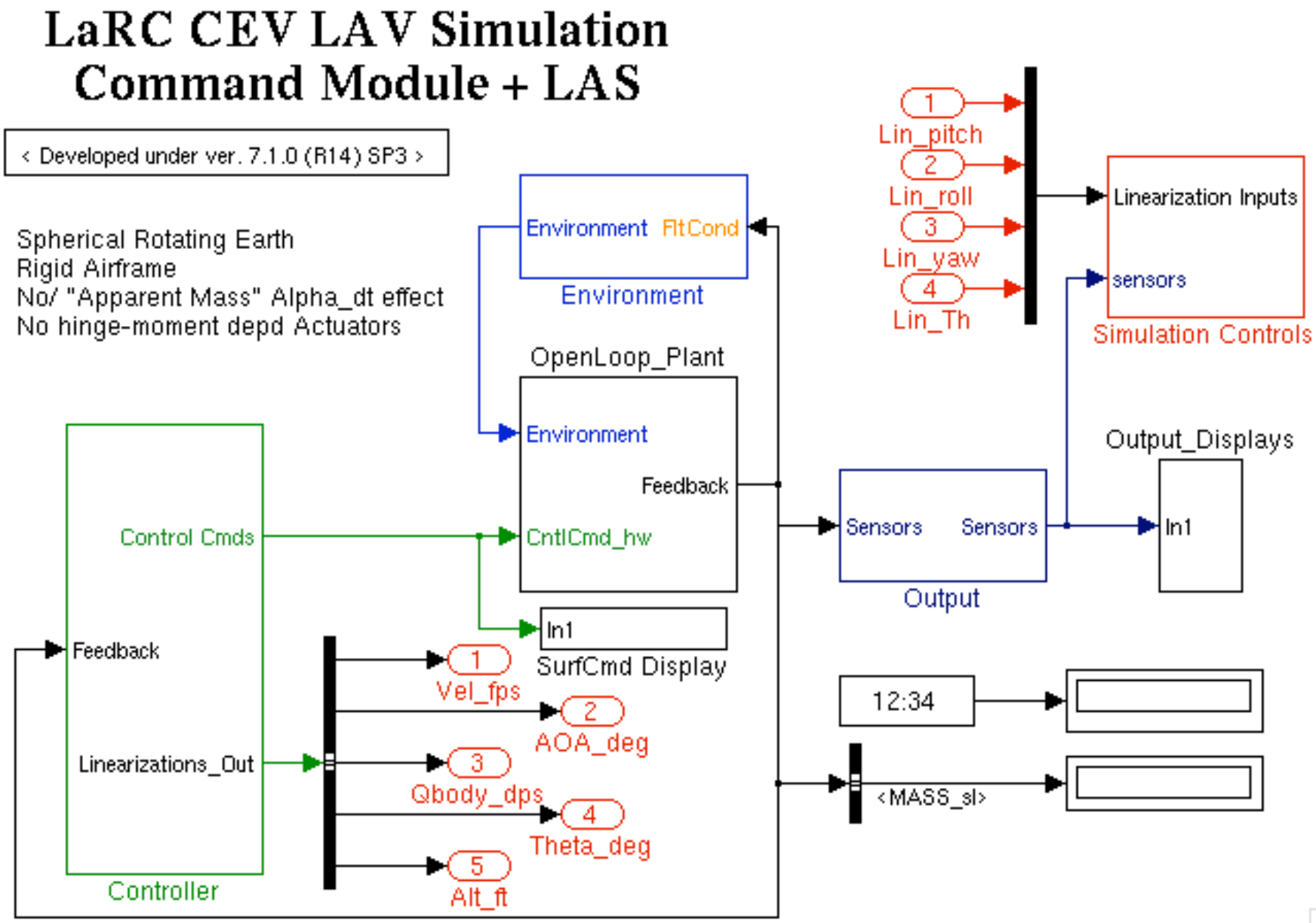
Approach: Design & analysis centered on the development and use of a flexible, rapidly changeable simulation environment.



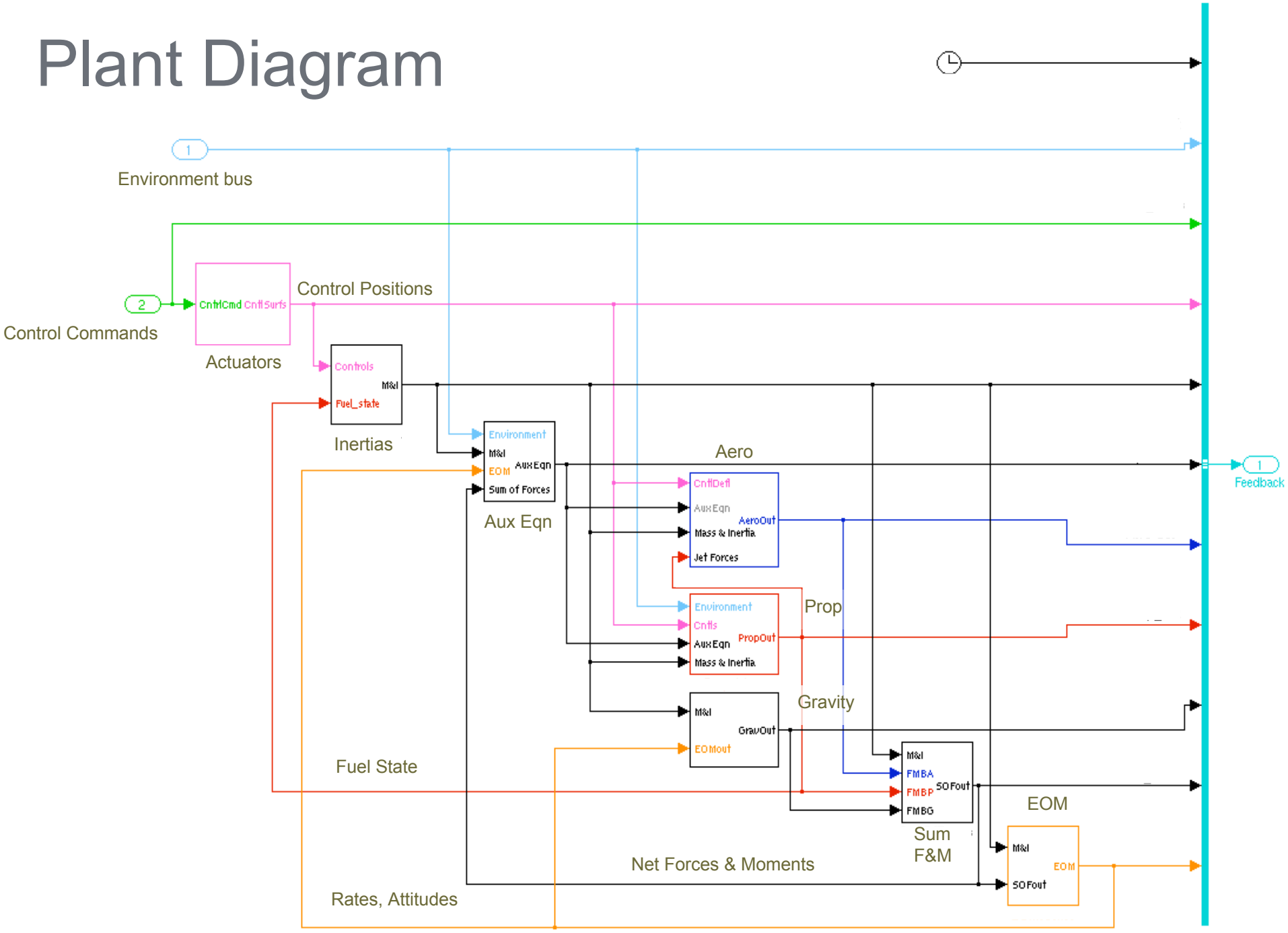
SAREC Simulation Tool

- Simulation ARchitecture for Evaluating Controls
- Six degree-of-freedom, non-linear, spherical rotating Earth, rigid-body simulation.
- Built within MathWorks Simulink® dynamic system modeling tool.
- SAREC has been developed and used over the past 7 years
- Set-up scripts load the data (block diagram is parameterized)
- Graphical user interface sets the initial conditions for each run
- Additional features
  - Run results stored as a data structure
  - Linear model generation along the flight path
  - Animated playback tool

SAREC Top-Level Diagram

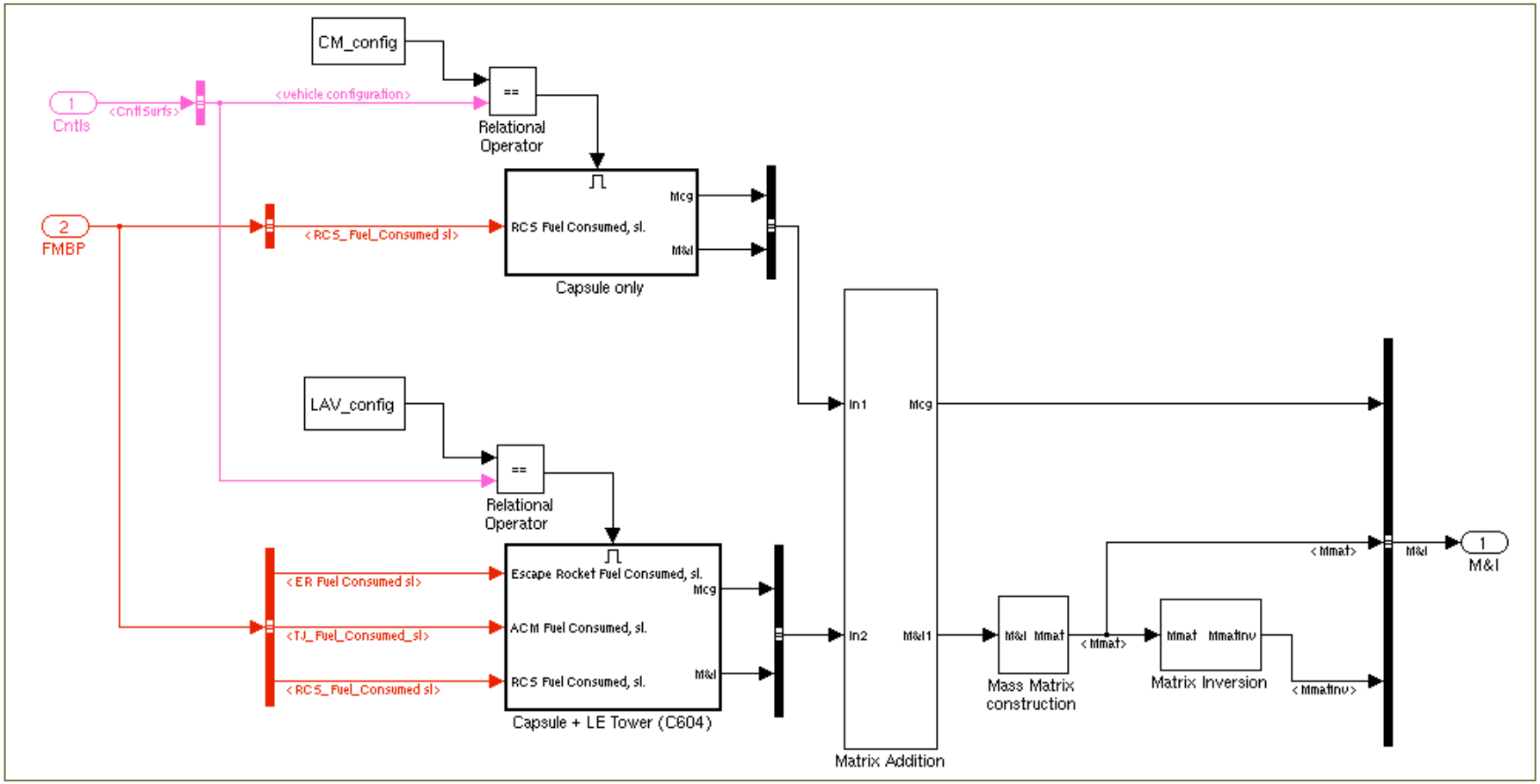


Plant Diagram



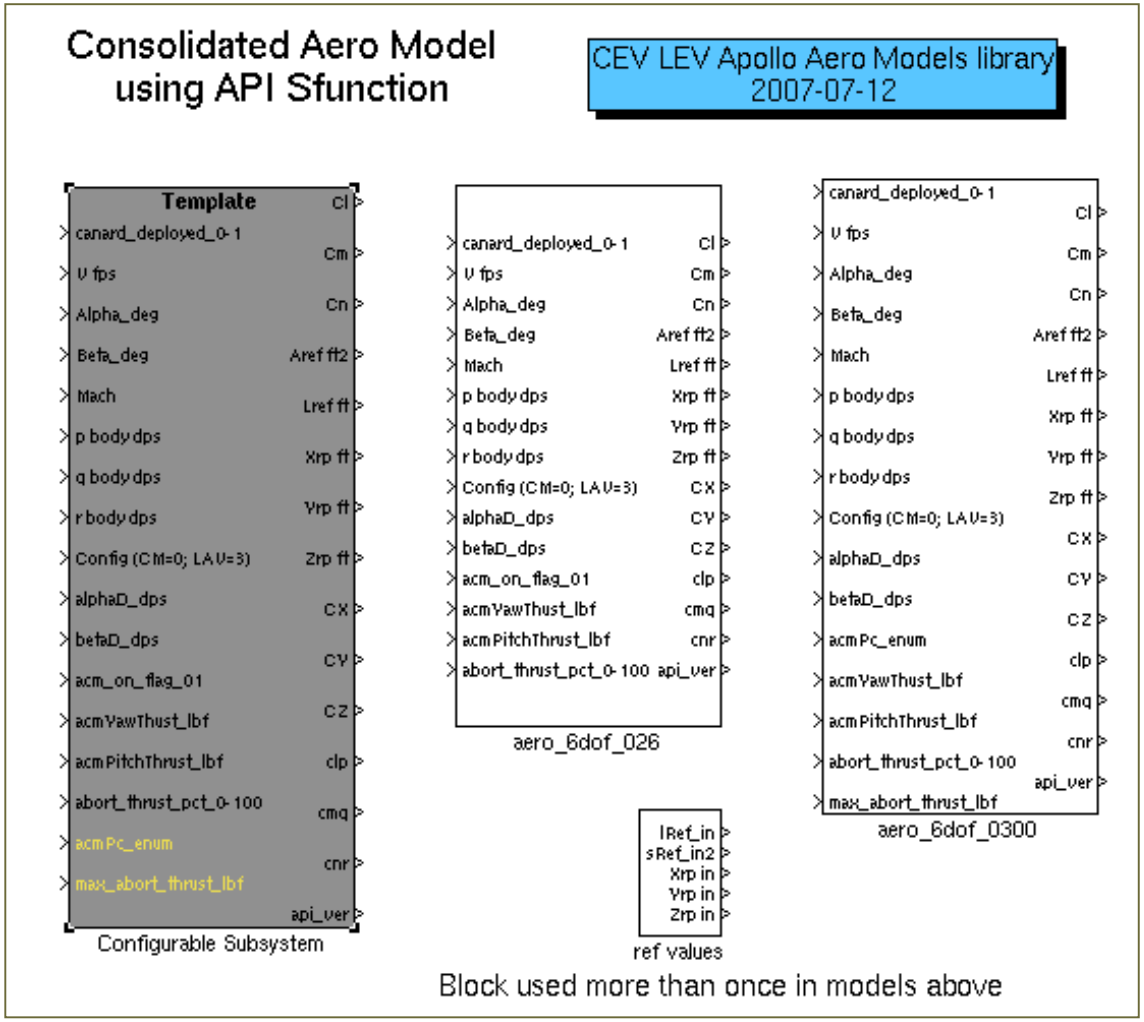
Mass & Inertia Socket

- Enabled subsystems used for stage separation
- Parameterized implementation (no hard-coded numbers)
- Full evaluation of mass properties based at each time step



Aerodynamics Socket

- Multiple aero datasets (releases) being used
- Configurable subsystem
  - allows run-time selection of aero release version
  - scripted data loading
  - no re-wiring diagram
- Library files reduce errors
  - locked by default
  - restore capability



Temporary Edit Pallet

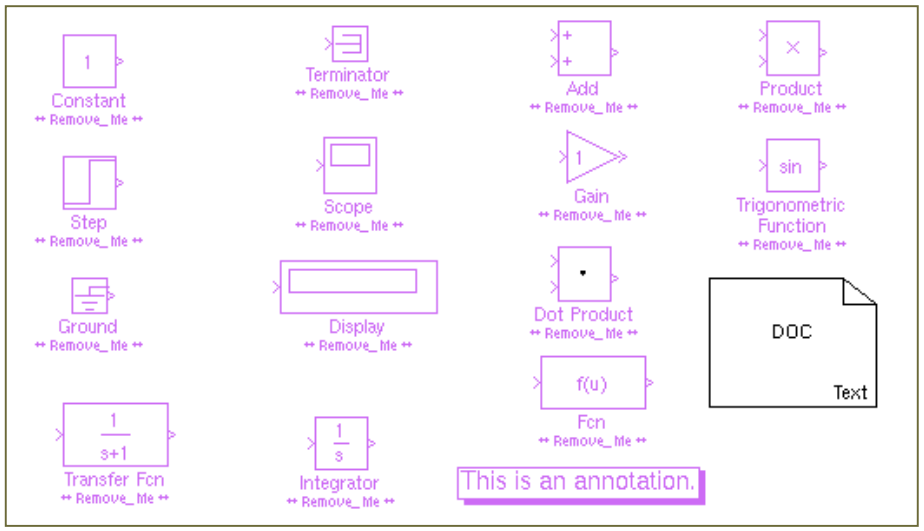
Problem:

Temporary block diagram changes often required but edits using standard blocks are easily forgotten or overlooked

Remedy:

Use a custom pallet of blocks that have features that are not easily overlooked

- Color: non-standard block color
- Tag field: “Remove Me”



Extra Precaution:

Embed code within the block diagram object that scans at run-time for “Remove Me”

Simulation Tool Wrap-up

- Controller implemented as configurable subsystem to select among versions
- Environment block also a configurable subsystem
  - US 1976 standard atmosphere
  - Parameterized atmosphere
- Four different LAV designs have been modeled
- Six aero revisions and eight mass property updates
- Two alternate configurations implemented
- Effectively utilized by aero team members for aero update assessment
- Meets goal of being flexible and rapidly changeable

Version Control

- Effective development across multiple organizations requires configuration control of information to manage and facilitate work.
- Subversion (svn) provides the required control
  - Open-source
  - Multi-platform
  - GUI and/or command line interfaces for most computer platforms
- Trunk and branch analogies
  - Trunk contains main development path
  - Branches allow short-lived development areas or long-lived alternative simulations

svn Version Control Highlights

- Users develop and modify locally then upload to ‘check-in’ changes
- At check-in the svn server increments the revision number for ALL files
- No revisions are lost (infinite undo) - past revisions readily available
- Revisions are logged with text describing the changes
- Most Orion simulation information is tracked within the svn repository
  - Simulation code and data
  - Reference material
  - Received code
  - Project documentation

Aero Model Import

- Aerodynamics data given in a variety of formats over the life of the task
- Data arriving in formats not readily usable within the simulation are converted to DAVE-ML format
  - human and machine readable XML
  - a proposed AIAA standard for aerodynamic models
  - can be automatically converted to Simulink diagram form
  - Unix *makefile* scripts automate the conversion to DAVE-ML
- The project's current use of C code for aero data deliveries allows the use of alternative import methods (s-function)

Control System Development

- Orion LAS control laws were initially designed by the government and are evaluated within the SAREC environment
- MathWorks Real-Time Workshop® utilized to convert block diagrams to C-code to share with others
- Unix *makefile* scripts automate the process
- Control laws developed by others using C-code are converted by hand to block diagram form and verified with unit comparisons

Team Communication

- Communication is critical to success
- Email is not always the most effective way
- An internal, web-based wiki provides an alternative
  - Editable library of project documentation
  - Change tracking
  - Web pages can be automatically generated and posted

Automatically Generated Wiki Page

LaRC CEV simulation revision log

This log shows, in reverse chronological order, the changes associated with each revision to the CEV simulation since revision 600. This is generated using a Perl script each time a revision is committed.

The previous set of 200 revisions is here.

The next set of 200 revisions is here.

The latest changes are available here.

Changes key:

- M Modified
- R Renamed
- A Added
- D Deleted

Rev	By	Date
600	jdm	Tue, 13 Mar 2007

Changed default controller to Hybrid and Mux's to BusCreators in Main_simulation to eliminate warning messages. Cleaned up GUI files sequence. The Trim_IC_description will appear in the Trim_GUI list.

```
M /trunk/Actions/GUI_files/GUI_main.fig
M /trunk/Actions/GUI_files/GUI_trim2.fig
M /trunk/Actions/GUI_files/trimSet_1.m
M /trunk/Model/Main_simulation.mdl
M /trunk/main_setup.m
```

599	dlr	Tue, 13 Mar 2007

Added conversion factors to hybrid_claw_setup.m

```
M /trunk/Model/GNC/Hybrid_abort_controller/hybrid_claw_setup.m
```

598	bjax	Tue, 13 Mar 2007

Incremented version number of hybrid controller autocode template from 3.0 to 4.0 in preparation for autocoding new hybrid controller.

```
M /trunk/Model/GNC/Hybrid_abort_controller/hybrid_claw_ert_rtw/hybrid_claw_code_template.cgt
```

Automatically Generated Wiki Page

Changes key:

- M Modified
- R Renamed
- A Added
- D Deleted

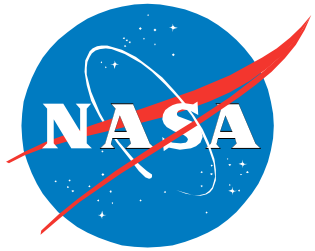
Rev	By	Date
600	jdm	Tue, 13 Mar 2007

Changed default controller to Hybrid and Mux's to BusCreators in Main_simulation to eliminate warning messages.
sequence. The Trim_IC_description will appear in the Trim_GUI list.

```
M /trunk/Actions/GUI_files/GUI_main.fig
M /trunk/Actions/GUI_files/GUI_trim2.fig
M /trunk/Actions/GUI_files/trimSet_1.m
M /trunk/Model/Main_simulation.mdl
M /trunk/main_setup.m
```

Lessons Learned / Concluding Remarks

- Version control simplifies development and coordination
- Branches and tags benefit parallel development
- Wiki pages enhance communication
- Automate repetitive tasks as much as possible
- Importance of documentation and unit testing
- “Remove Before Flight” tags prevent inadvertent mutations


NASA