ARTEMIS
Ares Real Time Environment for Modeling, Integration, and Simulation

Presented By:
Ryan Hughes & David Walker
Ares Overview

♦ Ares I - Launch Vehicle for Orion Crew Module

♦ Two Stage Rocket
  ◊ First Stage – Reusable Solid Rocket Motor
  ◊ Upper Stage – Liquid Engine (LH2, LOX)

♦ Ares V will carry Lunar Surface Access Module (LSAM) and Earth Departure Stage to join with Orion for lunar missions
ARTEMIS Overview

Ares Real Time Environment for Modeling, Simulation and Integration (ARTEMIS) is the real time simulation supporting Ares I hardware-in-the-loop (HWIL) testing.

ARTEMIS accurately models all Ares / Orion / Ground subsystems which interact with Ares avionics components from pre-launch through orbit insertion.
Why ARTEMIS?

♦ One combined source code tree supporting:
  ◊ Multiple model fidelities
    ♦ Vehicle – flex body, rigid body
    ♦ Selectable winds, atmosphere, TVC nozzle, etc.
  ◊ Non real-time, all-digital
  ◊ Real-time, distributed configurations
    ♦ all-digital, partial HWIL, all HWIL
  ◊ Multiple Ares development and test labs
  ◊ Software portability between Linux distributions

♦ ARTEMIS operates in a HWIL environment such that a user can select between models of avionics components or interfaces to avionics hardware

♦ Simulation will interact with lab configuration and control software to support model selection and fault insertion

♦ Utilizes modern computing technology to achieve real-time performance of high-fidelity models
Ares Test Facilities Overview

**Software Development Facility**
- Ares Flight S/W (GNC, Aborts, etc)
- Ares RT & NRT Simulation

**Ares Element Development & Integration Lab**
- Upper Stage SITF
- First Stage HWIL
- J-2X HWIL

**Ares System Integration Lab**

**Model Cross Verification, Validation, & Accreditation**
- Algorithm Development Environment
- Validation
- Fly Ares Mission

**KSC Stack Integration**

**MAF Upper Stage / J-2X Integration**

**Primary Ares Element Development Environment**
- Early Element-to-Element Integration
- Extensive Fault Injection Capability
- Easily Configurable to Support Element Integration
- Real-time SIL Simulation

**Upper Stage, J-2X, and First Stage Integration**
- High Fidelity Hardware Configuration
- Ares Level III Verification Test Environment
- Vehicle Anomaly Resolution
- Constellation DSIL Integrated
ARTEMIS SIL Architecture

ARTEMIS SCRAMNet

MAESTRO

Core SIM Models
- Flex Body Dynamics
- Rigid Body Dynamics
- Slosh
- Aerodynamics
- Environments

Subsystem Models
- J2X Engine
- SRB
- BDM, BTM, USM
- US and FS TVC
- ReCS and RoCS
- GPS Range Safety
- MPS
- LSC

Component Models
- ACC
- ASA
- CCSE
- CTC
- DACU
- FC
- FS RGA
- US FSS
- ISC
- PDCU
- RCU
- RINU
- RoCSE
- US ECU
- US TVC DCU
- HPUC
- RF System

Specific Configuration of SIL Models and Avionics HW is dependent upon specific test objectives
ARTEMIS SITF Architecture

**Core SIM Models**
- Flex Body Dynamics
- Rigid Body Dynamics
- Slosh
- Aerodynamics
- Environments

**Subsystem Models**
- J2X Engine
- SRB
- BDM, BTM, USM
- US and FS TVC
- ReCS and RoCS
- GPS Range Safety
- MPS
- LSC

**Component Models**
- ACC
- ASA
- CCSE
- CTC
- DACU
- FC
- FS RGA
- US FSS
- ISC
- PDCU
- RCU
- RoCSE
- US RGA
- US FSS
- AGTU
- BCPDU
- CLSS
- DARU
- FCC
- FDAU
- US RGA
- FS FSS

**Real-time Models of Vehicle Dynamics and Avionics Boxes**

**Specific Configuration of SIL Models and Avionics HW is dependent upon specific test objectives**

**Consoles for Automating Simulation, Test, & Real-time Operations**

**MAESTRO**
- MAESTRO Real-Time Display
- Operator/Configuration Manager
- MAESTRO Data Repository

**ARTEMIS SCRAMNet**

**MAESTRO Ethernet**

**TBD (e.g. Ethernet)**

**USA EGSE** (Certified by SITF/SIL)

**Upper Stage Test Articles**

**LCC Emulator**

**Orion Emulator**

**HW Interfaces for each US Box**
ARTEMIS Organization

 ARTEMIS is organized into six functional components

 ◊ Simulation
   ◦ Contains the executive framework

 ◊ Timing
   ◦ Scheduling, synchronization, global timing source, time stamps

 ◊ Models
   ◦ Three major categories: Core Simulation, Components, Subsystems

 ◊ Input / Output
   ◦ SCRAMNet, shared memory, discrete, analog, EIA-422, MIL-STD-1553B, Gigabit Ethernet

 ◊ DataRecording
   ◦ Global, local, meta data definition

 ◊ Hardware
   ◦ Computers, I/O cards, cables, racks
ARTEMIS Organization

♦ ARTEMIS is organized into six functional components

◊ Simulation
  ♦ Contains the executive framework

◊ Timing
  ♦ Scheduling, synchronization, global timing source, time stamps

◊ Models
  ♦ Three major categories: Core Simulation, Components, Subsystems

◊ Input / Output
  ♦ SCRAMNet, shared memory, discrete, analog, EIA-422, MIL-STD-1553B, Gigabit Ethernet

◊ Data Recording
  ♦ Global, local, meta data definition

◊ Hardware
  ♦ Computers, I/O cards, cables, racks
Simulation Overview

♦ Executive Framework Consisting of:

◊ Input data processing of XML input files
◊ Multi-phased initialization
◊ Scheduled (run-time) loop
  ♦ Derivative / Integration
◊ Shutdown
◊ Error handling
◊ Monte Carlo
◊ Fault Insertion
ARTEMIS is organized into six functional components

- **Simulation**
  - Contains the executive framework

- **Timing**
  - Scheduling, synchronization, global timing source, time stamps

- **Models**
  - Three major categories: Core Simulation, Components, Subsystems

- **Input / Output**
  - SCRAMNet, shared memory, discrete, analog, EIA-422, MIL-STD-1553B, Gigabit Ethernet

- **Data Recording**
  - Global, local, meta data definition

- **Hardware**
  - Computers, I/O cards, cables, racks
Sync controls timing and scheduling of frames for each ARTEMIS executable

- ARTEMIS executables may run at different frame rates that are a multiple of Sync minor frame rate

Maintains hard real-time operation using a timing card such as IRIG-B or RCIM

- Can also run non-real-time

Creates and controls access to the shared/reflective memory region for ARTEMIS

Receives and responds to commands from MAESTRO for both Master and Slave Sync

- MAESTRO passes test configuration and startup commands through Master Sync
- MAESTRO issues Sync commands to control ARTEMIS execution
- Sync responds to MAESTRO with status messages
♦ One Master Sync process runs on the Master Node
♦ Each additional simulation node runs the Slave Sync process that is controlled by Master Sync
♦ Master node controls real time synchronization via reflective memory
  ◊ Receives timer interrupt from timing card
♦ Sync Data Coherence
  ◊ Data input at beginning of sim thread’s start cycle
  ◊ Data output at end of cycle prior to sim thread’s next start cycle
ARTEMIS Organization

♦ ARTEMIS is organized into six functional components
  ◦ Simulation
    ♦ Contains the executive framework
  ◦ Timing
    ♦ Scheduling, synchronization, global timing source, time stamps
  ◦ Models
    ♦ Three major categories: Core Simulation, Components, Subsystems
  ◦ Input / Output
    ♦ SCRAMNet, shared memory, discrete, analog, EIA-422, MIL-STD-1553B, Gigabit Ethernet
  ◦ Data Recording
    ♦ Global, local, meta data definition
  ◦ Hardware
    ♦ Computers, I/O cards, cables, racks
Models Overview

♦ Core Simulation
  ◊ Flexible and rigid body equations of motion and environment models

♦ Component
  ◊ Digital models representing the functionality of actual Ares avionics boxes

♦ Subsystem
  ◊ Digital, physics-based models representing the vehicle’s physical subsystems that are not typically tested in the lab.
ARTEMIS Animation
ARTEMIS Organization

ARTEMIS is organized into six functional components

- Simulation
  - Contains the executive framework

- Timing
  - Scheduling, synchronization, global timing source, time stamps

- Models
  - Three major categories: Core Simulation, Components, Subsystems

- Input / Output
  - SCRAMNet, shared memory, discrete, analog, EIA-422, MIL-STD-1553B, Gigabit Ethernet

- Data Recording
  - Global, local, meta data definition

- Hardware
  - Computers, I/O cards, cables, racks
I/O Layer Overview

♦ Provides a transparent, consistent architecture for performing I/O for the ARTEMIS models

♦ Handles simulated device communication between the models via either shared memory or SCRAMNet reflective memory

♦ Transfers to real or simulated devices must be transparent to the models

♦ Handles the following real devices contained in the Ares I avionics architecture:

♦ Handles other real devices needed by the simulation system such as:
  ◊ GPIB, RCIM II / RCIM III, SCRAMNet GT, IRIG
The I/O Layer consists of:

- A set of common library calls that the ARTEMIS models use for communication with the I/O Layer
- The I/O Layer process which performs all the I/O with real or simulated devices
- An XML file describing the configuration of the Ares I avionics rings, simulation computers, and I/O devices used during a simulation
- A python based GUI that allows a user to build the XML configuration file
- An I/O Layer library:
  - Contains the initialization, read, write and close calls for each device the models control
  - Communicates with the I/O Layer process via shared memory semaphores
  - Passes unique device information and data from the models to the I/O Layer process via device structures in shared memory
  - The read and write calls communicate directly with the device driver threads
ARTEMIS Organization

- ARTEMIS is organized into six functional components
  - Simulation
    - Contains the executive framework
  - Timing
    - Scheduling, synchronization, global timing source, time stamps
  - Models
    - Three major categories: Core Simulation, Components, Subsystems
  - Input / Output
    - SCRAMNet, shared memory, discrete, analog, EIA-422, MIL-STD-1553B, Gigabit Ethernet
  - Data Recording
    - Global, local, meta data definition
  - Hardware
    - Computers, I/O cards, cables, racks
Data Recorder Overview

♦ Data Recorder supports generic data recording of multiple types of interfaces:
  ◊ SCRAMNet, MIL-STD-1553, Gigabit Ethernet, EIA-422, Discrete I/O, Cross Channel Data Link (CCDL)

♦ Configured via an XML file

♦ Data is recorded in its raw format
  ◊ Each packet/message is recorded with a timestamp

♦ Each interface is recorded in a separate file
  ◊ Filenames contain the beginning and ending timestamp for its corresponding data

♦ Interfaces with the local MAESTRO daemon

♦ Provides periodic archiving capability for early analysis during long tests
ARTEMIS Organization

- **ARTEMIS is organized into six functional components**
  - Simulation
    - Contains the executive framework
  - Timing
    - Scheduling, synchronization, global timing source, time stamps
  - Models
    - Three major categories: Core Simulation, Components, Subsystems
  - Input / Output
    - SCRAMNet, shared memory, discrete, analog, EIA-422, MIL-STD-1553B, Gigabit Ethernet
  - Data Recording
    - Global, local, meta data definition
  - **Hardware**
    - Computers, I/O cards, cables, racks
Hardware Overview

♦ Concurrent RedHawk real-time operating system
  ◊ Devices verified by vendor to meet real-time requirements

♦ I/O Cards
  ◊ SCRAMNet, MIL-STD-1553, Gigabit Ethernet, EIA-422, Discrete I/O, Analog Sensors, D/A and A/D boards, RCIM, IRIG

♦ The SIL will have flight-like cables
♦ All simulated components will be positioned in computer racks near avionics boxes in each ring
Current Status and Future Work

♦ Current
   ◊ Implementing Ares communications via the I/O Layer
     ♦ 1553B, GbE
   ◊ Integrating preliminary high fidelity vendor models

♦ Future
   ◊ Support flight software and vehicle testing
   ◊ Integrate all high fidelity vendor models
   ◊ Integrate flight hardware avionics components
   ◊ Transition to Ares V
Backup Slides
Fault Insertion

♦ Two Types of Faults
  ◊ Overwrite exposed simulation variables in SCRAMNet
    ◇ Least expensive to implement
    ◇ Limited to exposed simulation variables
    ◇ Won’t cover all fault requirements
  ◊ Execute an embedded fault in the simulation
    ◇ Require additional software development and V&V in models to simulate fault
    ◇ Initiated by tripping fault flag in SCRAMNet
    ◇ Need to streamline number of embedded faults

♦ Fault Insertion Mechanism
  ◊ Peek and Poke via MAESTRO
    ◇ User can trip embedded faults via MAESTRO interface to sync
    ◇ Non-Deterministic
  ◊ Separate fault insertion executable
    ◇ Controlled by sync master
    ◇ Provides logical conditions to determine when fault inserted
    ◇ Input file driven
    ◇ Deterministic
Core Simulation Models

♦ Stack Dynamics
  ◊ Coupled rigid body and flexible body dynamics formulation which properly accounts for variable mass effects and force following terms
  ◊ Supports all nominal and abort configurations
  ◊ Input data developed from EV30 LA2 structural models
  ◊ Multithreaded partitioned equations to achieve real-time performance with a frame time under 2ms

♦ Stage Dynamics
  ◊ 6 DOF rigid body formulation with vehicle states defined with respect to Constellation structural frame (fixed point off nose of LAS)
  ◊ Supports all nominal and abort configurations

♦ Mass Properties
  ◊ Propellant mass computed using mass flow rate defined by engine model
  ◊ Propellant mass properties computed from structural model mass matrices
  ◊ Compute mass properties of each stage from sum of dry structure and propellant
  ◊ Mass properties of stack (or combined stages) computed from sum of stages for current configuration defined by flight phase
Core Simulation Models

♦ Structural Properties
  ◊ Stage mass and stiffness matrices defined by NASTRAN models
  ◊ Family of propellant mass matrices based on stage mass
  ◊ Assemble stack mass and stiffness matrices from stage and propellant matrices based on vehicle configuration
  ◊ Update generalized vehicle mass and stiffness matrices each time step for coupled flex body EOM
  ◊ All vehicle node geometry extracted from integrated NASTRAN model

♦ Nozzle Dynamics
  ◊ Rigid body formulation uses discrete nozzle EOM driven by vehicle dynamics, TVC actuator forces, aerodynamic forces, and flex bearing stiffness
  ◊ Rigid body formulation also includes Tail-Wag-Dog effects
  ◊ Flex body formulation utilizes coupled nozzle dynamics embedded in system Ritz vectors or modes

♦ Slosh Dynamics
  ◊ Rigid body formulation uses discrete slosh masses per tank modeled by spring-mass-damper systems, Lookup tables for slosh parameters
  ◊ Flex body formulation utilizes slosh modes developed from additional effects superimposed on propellant mass and stiffness matrices
Core Simulation Models

♦ Atmosphere and Winds
  ◊ US76 standard atmosphere model
  ◊ 2007 Global Reference Atmospheric Model (GRAM2007)
  ◊ 1800 Measured Day-of-Launch Winds
  ◊ Ground winds to support pre-launch

♦ Lumped Aerodynamics
  ◊ Linear 1-D table lookup and Nonlinear 2-D table lookup for aerodynamic coefficients for stack and stages (SRB, LAS, etc)

♦ Distributed Aerodynamics
  ◊ Aerodynamic data mapped to NASTRAN mesh for loads applied to the stack (primary driver of flex)

♦ Gravity
  ◊ 3 model options:
    ♦ Kepler
    ♦ J-2, J-3, & J-4
    ♦ Gravity Recovery and Climate Experiment (GRACE)
Component Models

♦ Flight Computer (FC)
  ◆ Controller algorithm
    ♦ Exact representation of DAC2 Ares Controller algorithms (Gain-scheduled Flex Mitigation Filters + PID)
  ◆ Navigation algorithm
    ♦ Fundamental Navigation Equations for multiple sensors and rate gyros
  ◆ Guidance algorithm
    ♦ Exact representation of DAC2 Ares Guidance algorithms (Open-Loop Profile for 1st Stage; Closed-Loop Algorithm for US)
  ◆ Mission Manager and Event Controller
    ♦ Event handler to control flight and vehicle phasing based on flight time and mission events

♦ Booster Control & Power Distribution Unit (BCPDU)
  ◆ Passes commands from the flight computer to downstream avionics boxes
  ◆ Prototype MIL-STD-1553 interface from FC with TVC commanded rock and tilt current message

♦ Sensors
  ◆ Medium-fidelity RINU model with gyroscope and accelerometer error terms (bias, noise, scale factor, misalignments, initial condition errors)
Component Models

♦ Recovery Control Unit (RCU)
  ◊ Commands the BTM, aeroshell jettison, and forward skirt extension jettison on the first stage during recovery operations

♦ Ignition & Staging Controller (ISC)
  ◊ Commands the firing of the first stage, BDM, USM, and first stage separation pyros based on commands from the BCPDU

♦ Altitude Sensor Assembly (ASA)
  ◊ Pressure sensor that activates first stage recovery system once the SRB falls below a given altitude

♦ Command and Telemetry Computer (CTC)
  ◊ Currently relays ground commands to the FC during pre-launch and ascent

♦ Rate Gyro Assembly Electronics (RGAE)
  ◊ Buffers the RGA outputs for use in the FC for both the first stage and upper stage RGAs

♦ Redundant Inertial Navigation Unit Electronics (RINUE)
  ◊ Uses $\Delta V$ & $\Delta \theta$ from the RINU to estimate vehicle states & other data needed by the flight software
Component Models

♦ Combined Control System Electronics (CCSE)
  ◊ Partial CCSE model outputs valve commands to support tanking ground ops. Incorporates previous ReCSE model as well.

♦ Roll Control System Electronics (RoCSE)
  ◊ Relays the fire commands for the first stage roll control system from the FC to the RoCS thrusters

♦ Upper Stage Engine Control Unit (US ECU)
  ◊ Controls the J-2X firing, mixture ratio, and throttle

♦ Upper Stage TVC Data & Control Unit (US TVC DCU)
  ◊ Converts a commanded set of gimbal angles from the FC into a current value used by the upper stage TVC
Subsystem Models

♦ Reaction Control System (RCS)
  ◊ Ideal thrust, general valve dynamics developed but not activated
  ◊ Lookup tables for thrust & valve dynamics

♦ Booster Separation Motors (BDM, BTM, Ullage)
  ◊ Uses lookup table for thrust, supports delayed firing

♦ Engines
  ◊ Lookup table driven, supports separate tables for nominal, startup and shutdown operations

♦ Thrust Vector Control (US & FS TVC)
  ◊ High-fidelity simplex algorithm with models of servo valves, power spool, and actuator

♦ Main Propulsion System (MPS)
  ◊ Simple tanking model
  ◊ High fidelity model incorporated using existing ROCETS code
Current Subsystem Models

♦ Hold-Down Post (HDP)
  ◊ Uses stiffness and damping matrix to model flexibility of launch platform
  ◊ Spring can only provide force while in compression

♦ Linear Shaped Charges (LSC)
  ◊ Model does not provide forces, but sets flags indication whether stage separation has occurred

♦ Redundant Inertial Navigation Unit (RINU)
  ◊ Converts sensed vehicle motion signals into ΔV & Δθ values needed by the flight software

♦ Rate Gyro Assembly (RGA)
  ◊ Senses the vehicle motion and converts to Δθ signals used by the FC controller