Independent Test Capability

NASA IV&V Software Emulator Technology Portfolio

http://www.nasa.gov/centers/ivv/jstar/ITC.html

Matthew Grubb

Matthew.Grubb@TMCTechnologies.com

NASA IV&V Facility
100 University Drive
Fairmont, WV 26554

Independent Test Capability

Agenda

- **Introduction to Software-Only-Simulation**
  - Process and approach for simulation and hardware modeling
- **Independent Test Capability (ITC)**
  - Jon McBride Software Testing & Research Lab (JSTAR)
  - Infrastructure, Deployment, and Users
  - Technologies Developed
- **Development Evolution of Spacecraft Simulators**
- **Closing Remarks**
  - Lessons Learned

Software-Only-Simulation Introduction

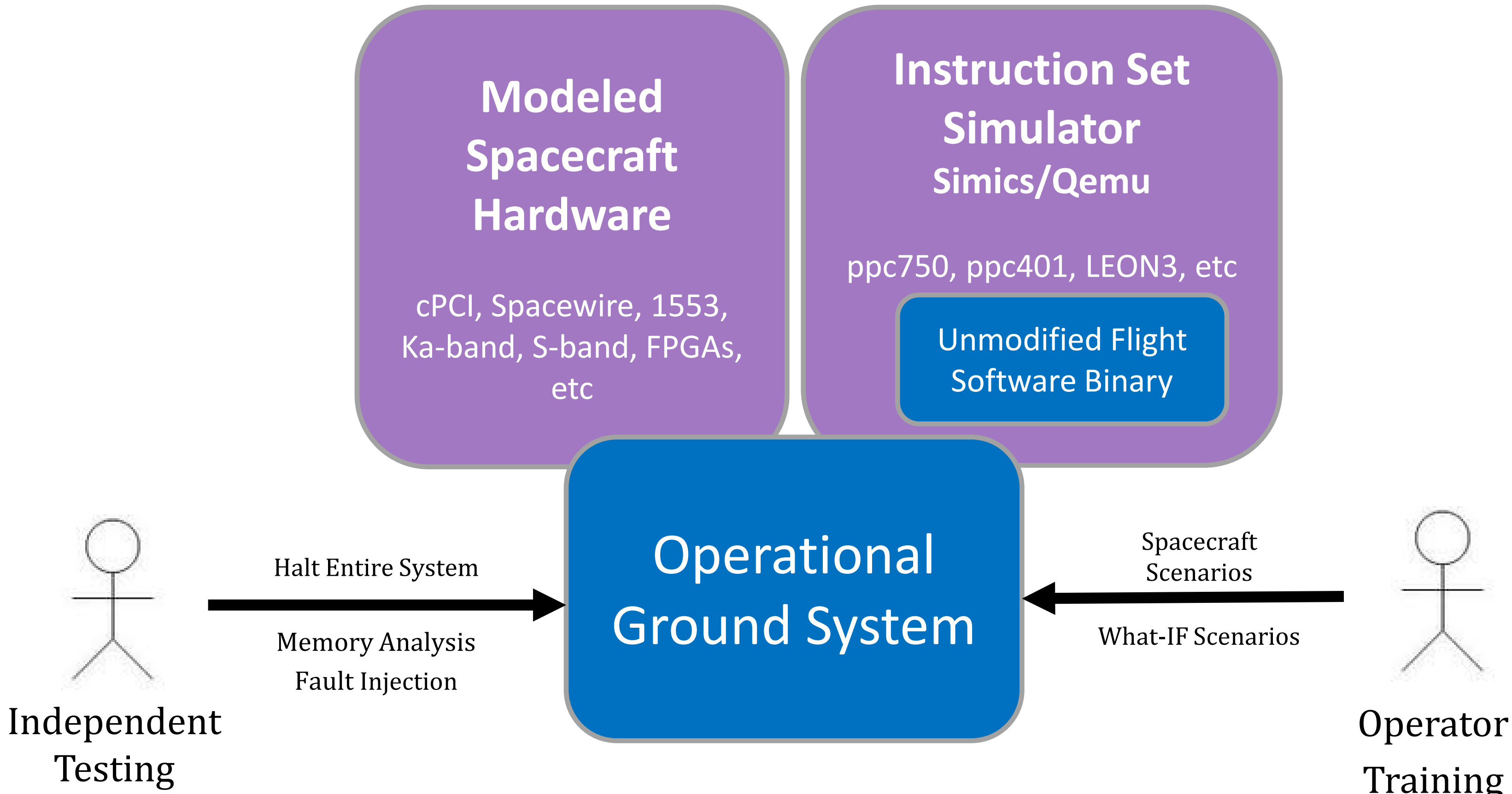
Software-Only-Simulation Introduction

- Software-Only-Simulation is a complete software representation of modeled hardware components and software emulators
- Together, the components form a complete spacecraft simulator
- Software-Only-Simulator provides complete control of CPU, Time, and Memory
  - Can stop all execution for debugging.
  - Can peek/poke memory, perform fault injection
- Spacecraft simulator used for:
  - Independent Testing (IVV)
  - Operator Training
  - Augment Project Hardware Testing

Software-Only-Simulation Introduction

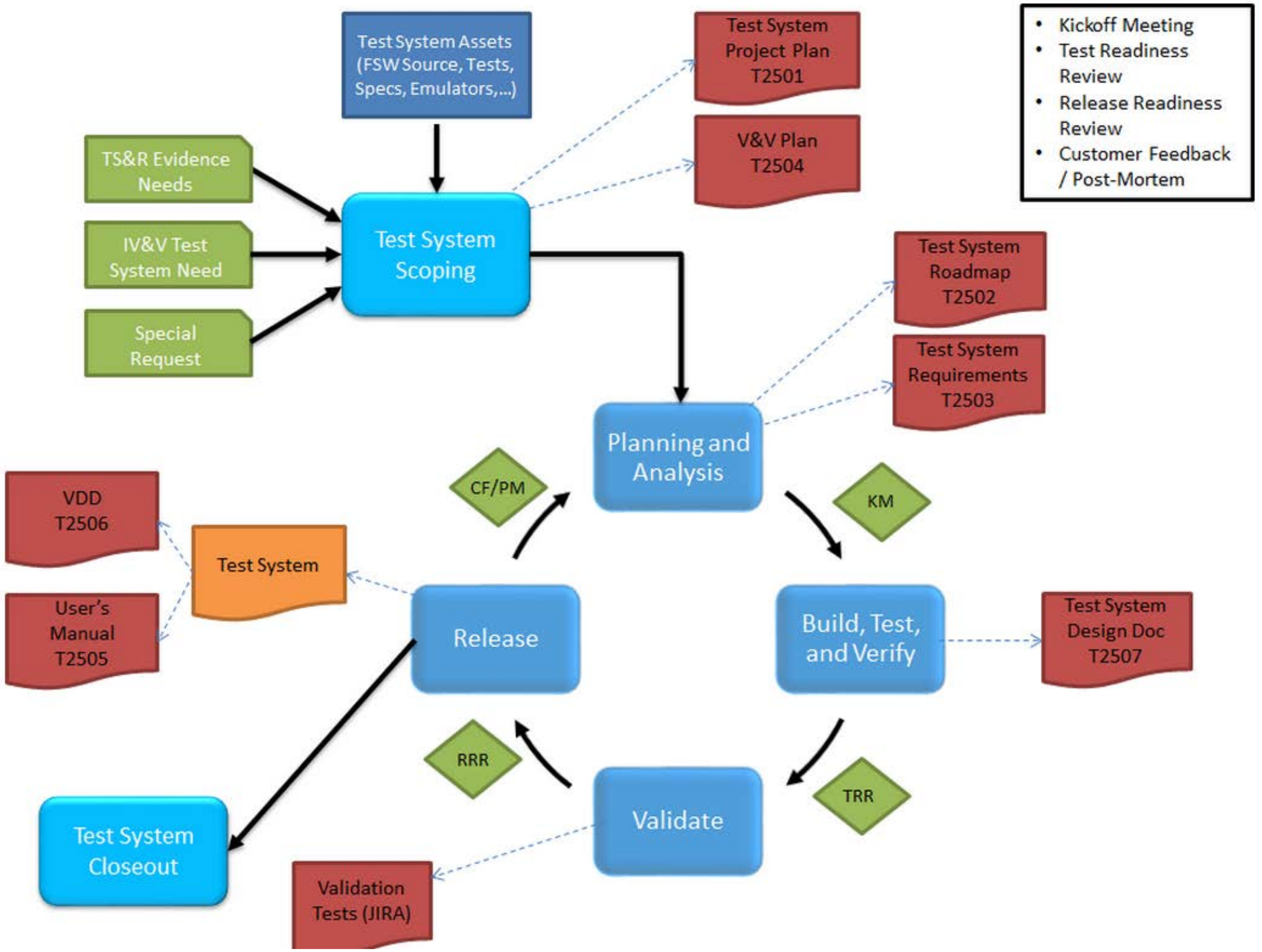
Independent Test Capability

Simulator Components



Software-Only-Simulation Introduction

Simulator Development Process



NASA IV&V Independent Test Capability (ITC) Introduction

Independent Test Capability (ITC) Introduction

Charter

Acquire, develop, and manage adaptable test environments that enable the dynamic analysis of software behaviors for multiple NASA missions

Dynamic Analysis is performed on flight software to verify software behavior

Independent Test Capability (ITC) Introduction

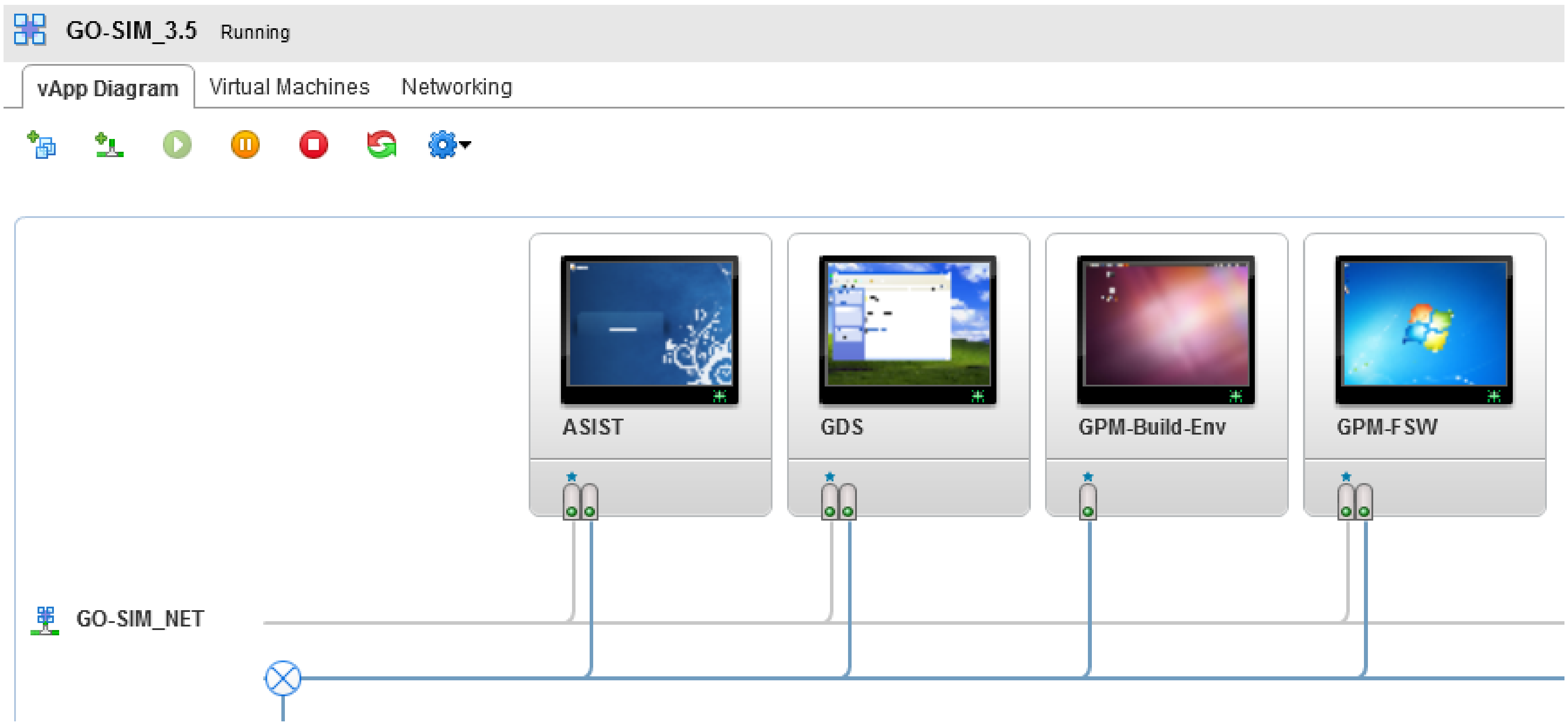
- ITC Develops System Simulators
  - Experts in **<u>Hardware Modeling</u>** and Distributed Simulation
  - Experts in Simulator & Software Integration

Independent Test Capability

Jon McBride Software Testing & Research (JSTAR) Laboratory

- Cloud-based infrastructure using server and desktop virtualization
- Large scale simulator deployments
- Hardware-in-the-loop and software-only test environments
- Integration of COTS and GOTS software tools to support V&V activities

Virtualized Deployment

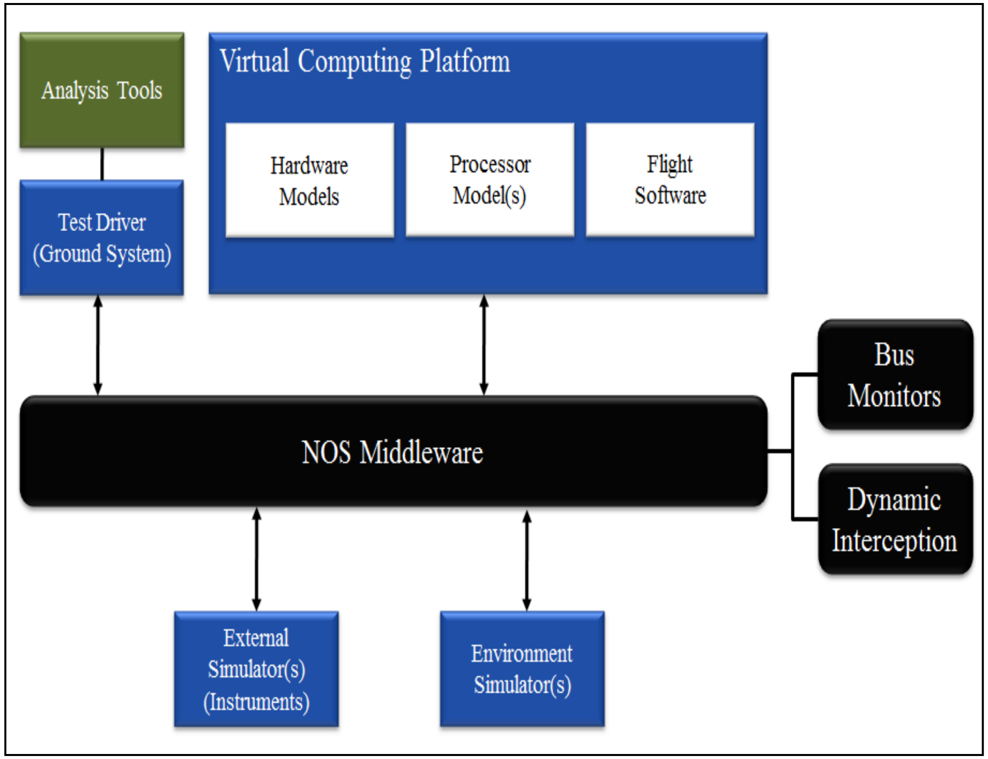


ITC Technologies

ITC Technologies

NASA Operational Simulator (NOS)

- Software-only simulation architecture
- Capable of executing unmodified flight software
- Custom layered-architecture middleware
- Dynamic interception capability
- Reusable software modules and scripts
- Virtual machine deployment

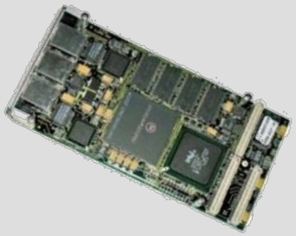


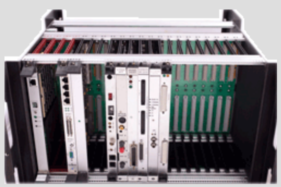
Typical NOS Architecture
(Space Domain)

NOS Feature Set

Independent Test Capability

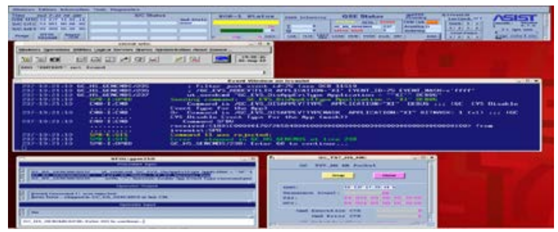
Plug-and-Play Hardware Models





Processors, Boards, Racks

Use of Operational Ground Systems Software

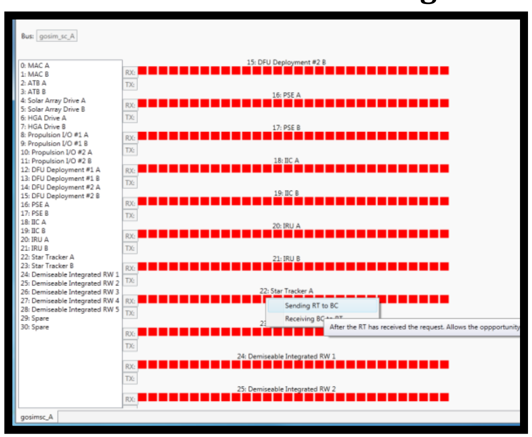


Instrument Model Framework

```
Instrument1
Subaddress HandlerA → FunctionA
Subaddress HandlerB → FunctionB
…
Subaddress HandlerN → FunctionN

InstrumentX
Subaddress HandlerA → FunctionA
Subaddress HandlerB → FunctionB
…
Subaddress HandlerN → FunctionN
```

Internal Bus Monitoring



NOS Middleware

Specialized Layers

↕

Base Layer Communications

Deployment & Maintenance

Virtualization

Independent Test Capability

NOS Middleware

Overview

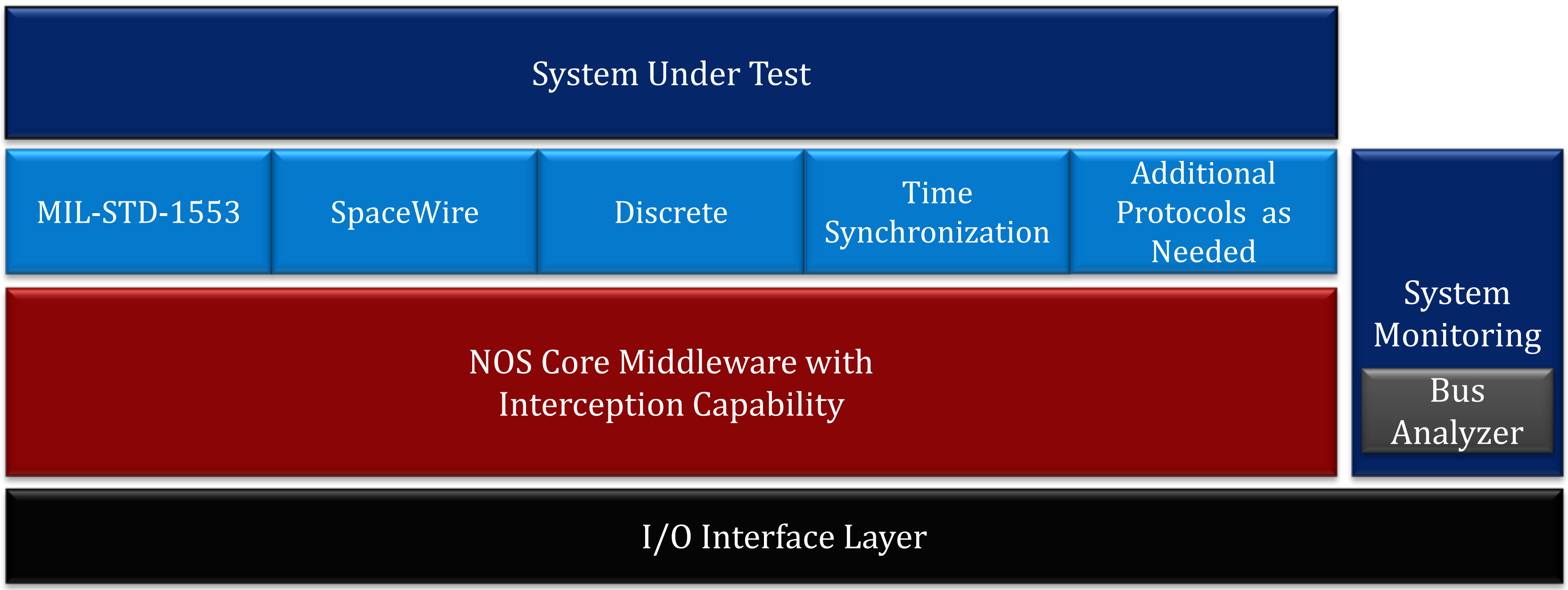
- ✓ Offers re-usable communication mechanism
  - Ensures consistent and correct data passing
- ✓ Provides synchronization between distributed applications
- ✓ Flexible and extensible design
  - Can be extended to incorporate any communication protocol

Features

- ✓ Transport agnostic
- ✓ Cross platform C++ implementation
- ✓ Robust User API
- ✓ Specialized User API Layers
  - MIL-STD-1553B
  - ESA SpaceWire
  - Discrete Signals
  - Time Synchronization
- ✓ Interception allows for V&V analysis
  - No modification to software-under-test

Independent Test Capability

NOS Middleware Architecture



NOS Software Utilities

- Virtual Oscilloscope
  - Virtual CompactPCI (cPCI) Analysis
  - Board-Level Signal Analysis

```
0x16d28 49732611220} output signal lowered
0x16d04 49734659404} output signal raised
0x16d28 49736707599} output signal lowered
0x16d04 49738755772} output signal raised
0x16d28 49740803956} output signal lowered
0x16d04 49742849199} output signal raised
0x16d28 49744897380} output signal lowered
0x16d04 49746945570} output signal raised
0x16d28 49748993748} output signal lowered
0x16d04 49751041977} output signal raised
0x16d28 49753090140} output signal lowered
```

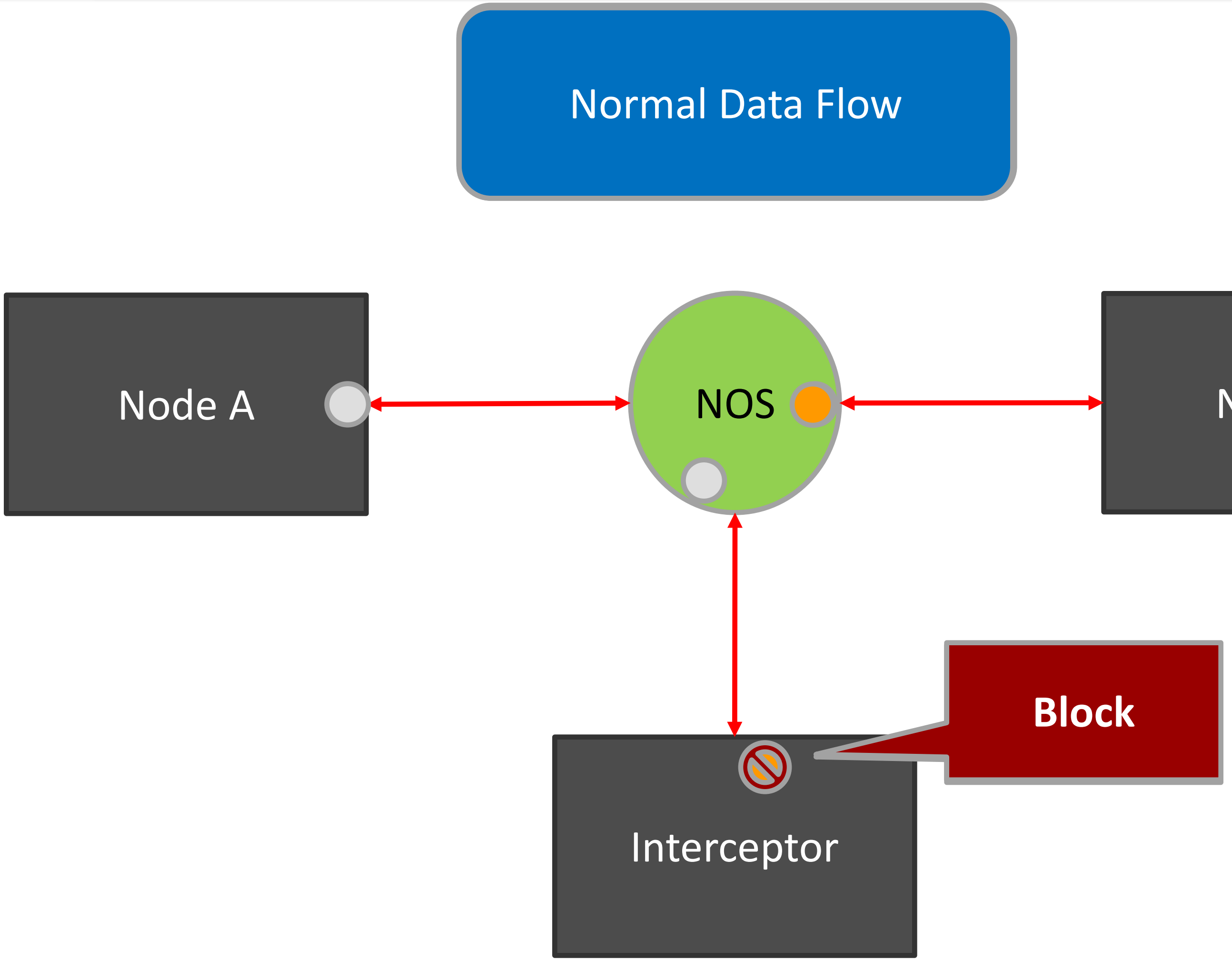
- Virtual MIL-STD-1553 Bus
  - Bus Controller with XML Defined Schedules
  - Remote Terminal
  - Bus Monitor/Logger
  - PASS3200 Software Emulator


```xml
<?xml version="1.0" encoding="utf-8"?>
<BusController_Schedule
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://itc.ivv.nasa.gov/1.0/bc_schedule"
        xsi:schemaLocation="http://itc.ivv.nasa.gov/1.0/bc_sched

<!-- This XML file serves as a basic example of a BC schedule fo
    <BCSchedule Name="bc_schedule_example" Repeat="true" DoubleBu
        <Frame FirstMsgDelayMicroSec="10">
            <Message MessageNum="1" RT="30" SA="14" WordCount="1
            <Message RT="29" SA="1" WordCount="4" GapTimeMicroSe
            <Message RT="30" SA="14" WordCount="1" GapTimeMicroS
            <Message RT="29" SA="14" WordCount="1" GapTimeMicroS
        </Frame>
```


- Virtual SpaceWire Router

NOS Dynamic Interception

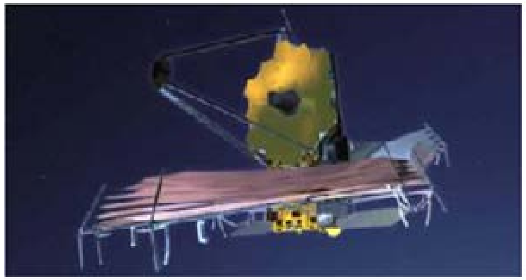


Evolution of ITC Spacecraft Simulators

Evolution of ITC Spacecraft Simulators

Global Precipitation Measurement (GPM) Operational Simulator (GO-SIM)

Closed-loop simulator including **<u>unmodified</u>** operational ground system, **<u>unmodified</u>** flight software, environmental simulator, and science instrument simulators



James Webb Space Telescope (JWST) Integrated Simulation and Test (JIST)

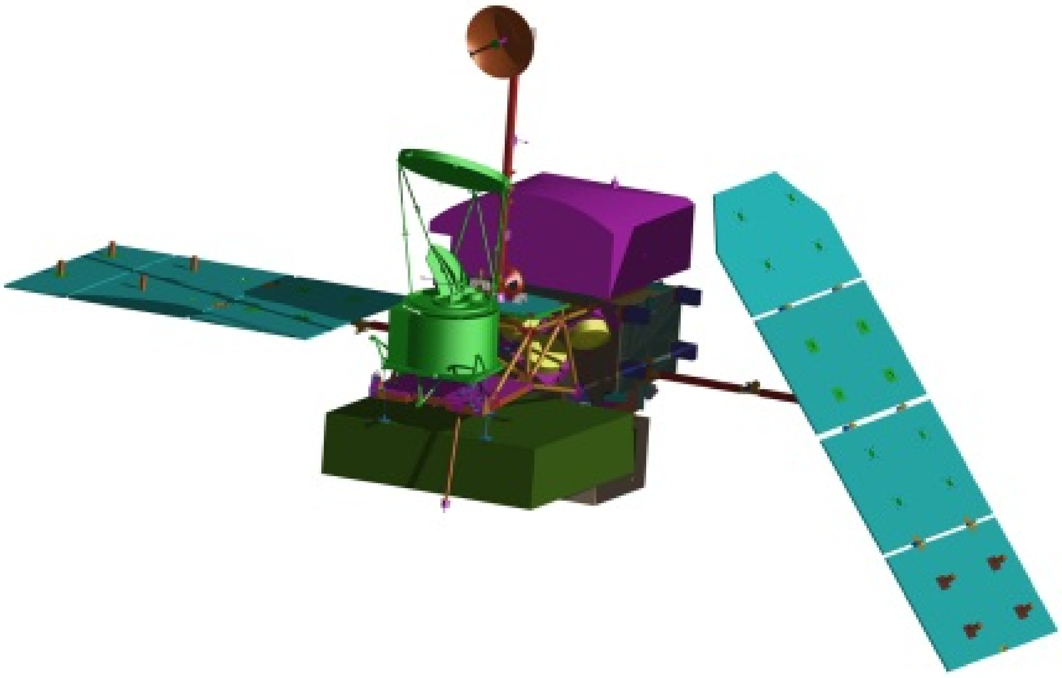
Simulator that demonstrates reusable NOS technologies can be applied to other NASA missions

Deep Space Climate Observatory (DSCOVR)

Turn-key modeling effort for spacecraft C&DH

Evolution of ITC Spacecraft Simulators

GPM Operational Simulator (GO-SIM)



GPM Operational Simulator GO-SIM

Independent Test Capability

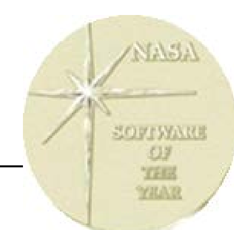
Components

- COTS Emulator
- Primary Instrument Simulations (GMI/DPR)
- GPM Ground System
- GSFC Goddard Dynamic Simulator (GDS)
- NOS Middleware
- GPM Hardware Models

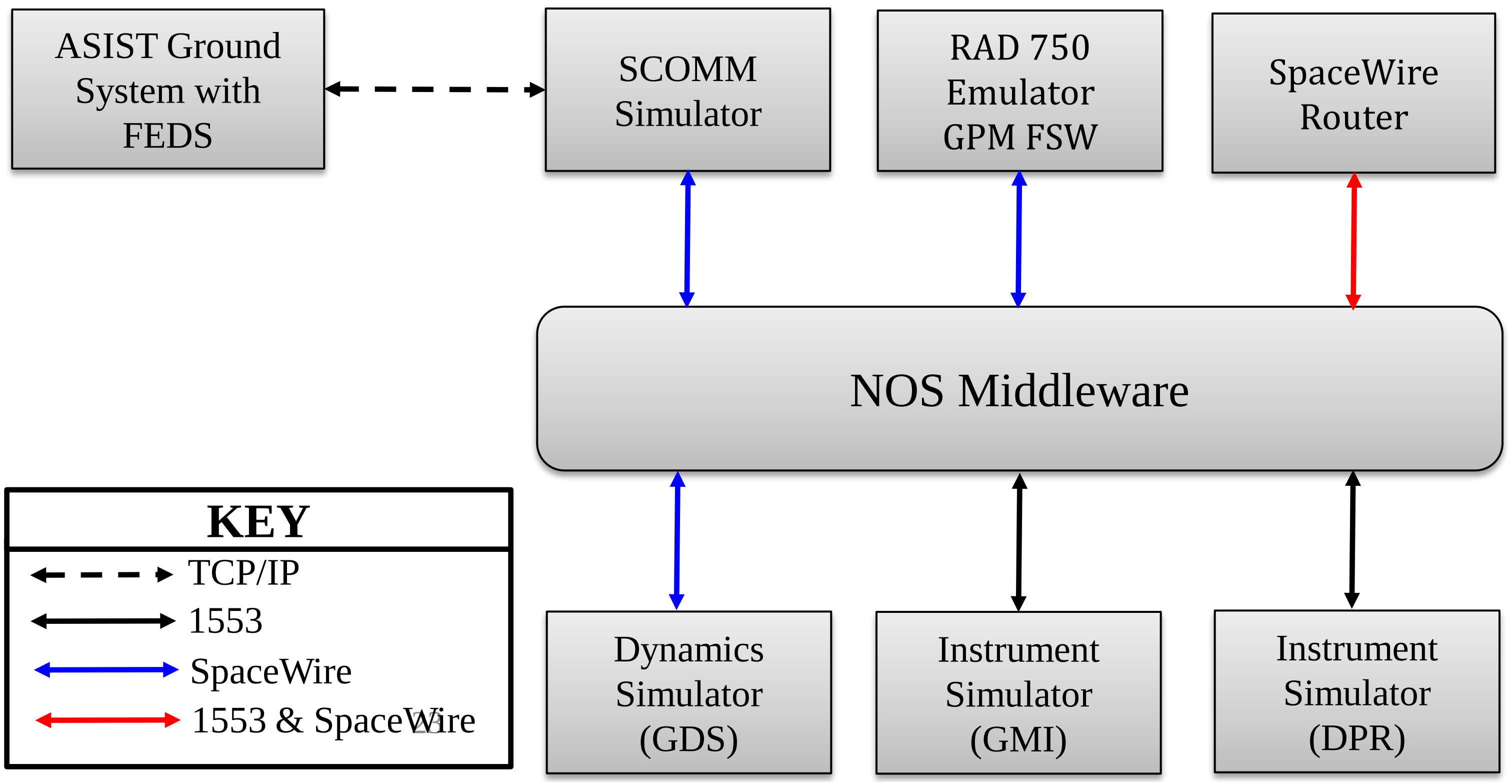
Capabilities

- Load and run unmodified flight software binaries
- Execute test flight scripts
- Single-step debugging
- Inject errors via ground system and NOS middleware
- Stress system under test

NASA Software of the Year Honorable Mention 2012

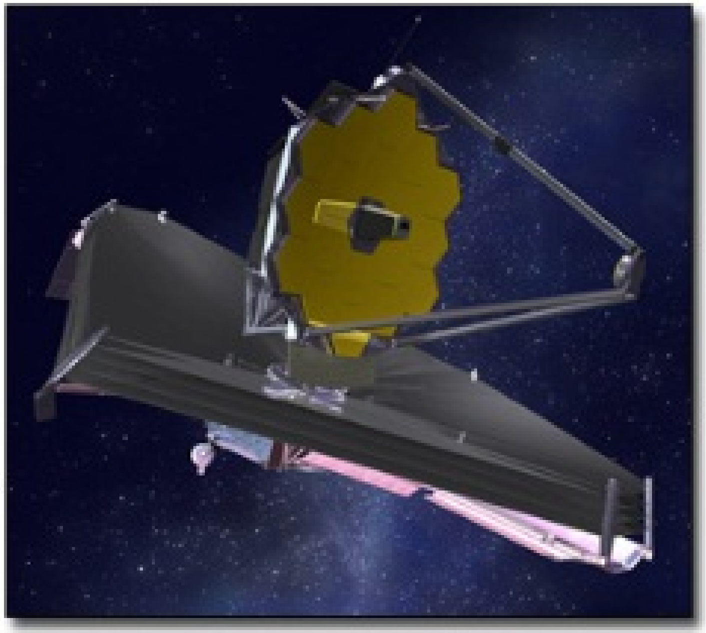


GO-SIM Architecture



Evolution of ITC Spacecraft Simulators

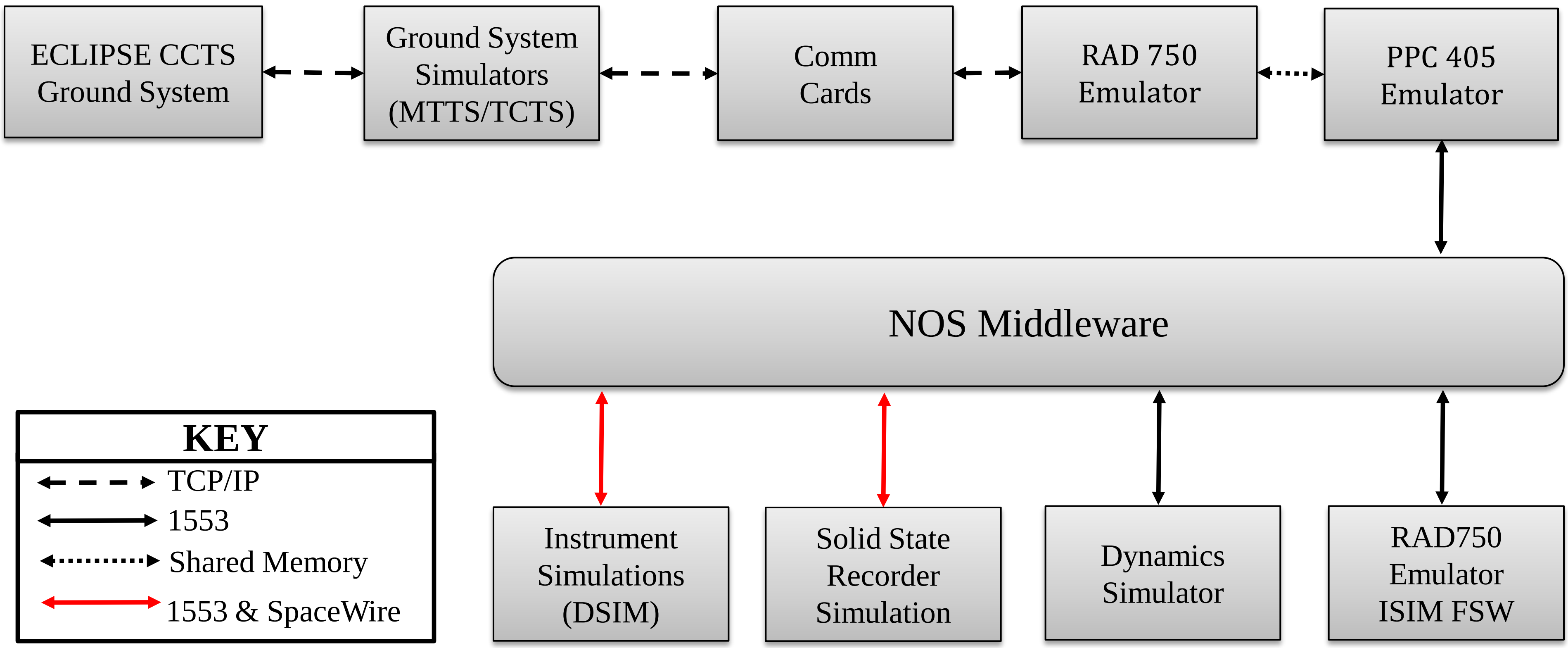
James Webb Space Telescope (JWST) Integrated Simulation and Test (JIST)



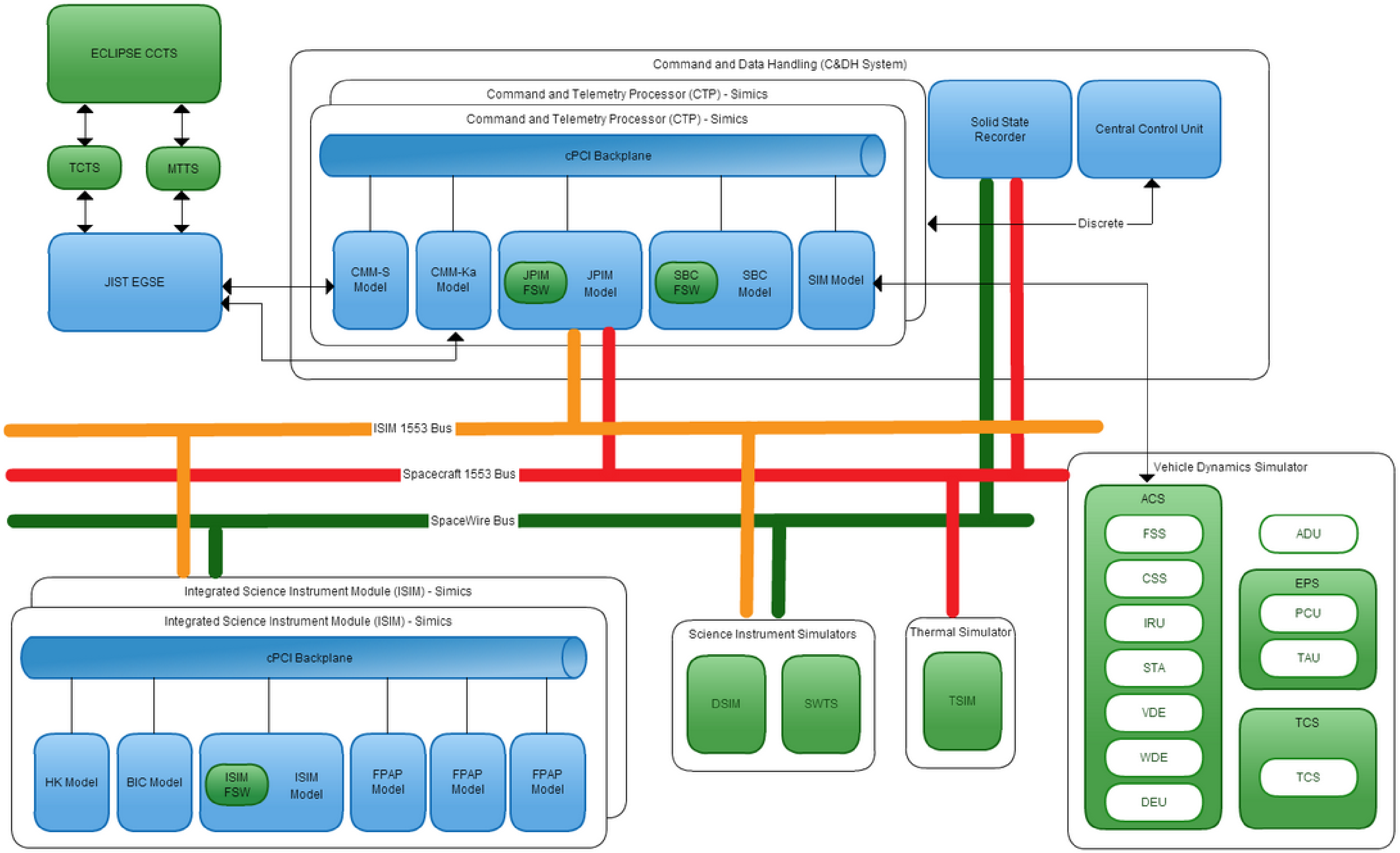
JWST Integrated Simulation and Test (JIST)

- Software-only spacecraft simulator
- Flexible environment to support V&V activities
- Unmodified ground system and scripts
- Unmodified software-under-test binaries
- Integration of COTS, GOTS and in-house developed components
- Custom hardware models
- Automated Testing Framework
- Fault Based Testing

JIST Architecture



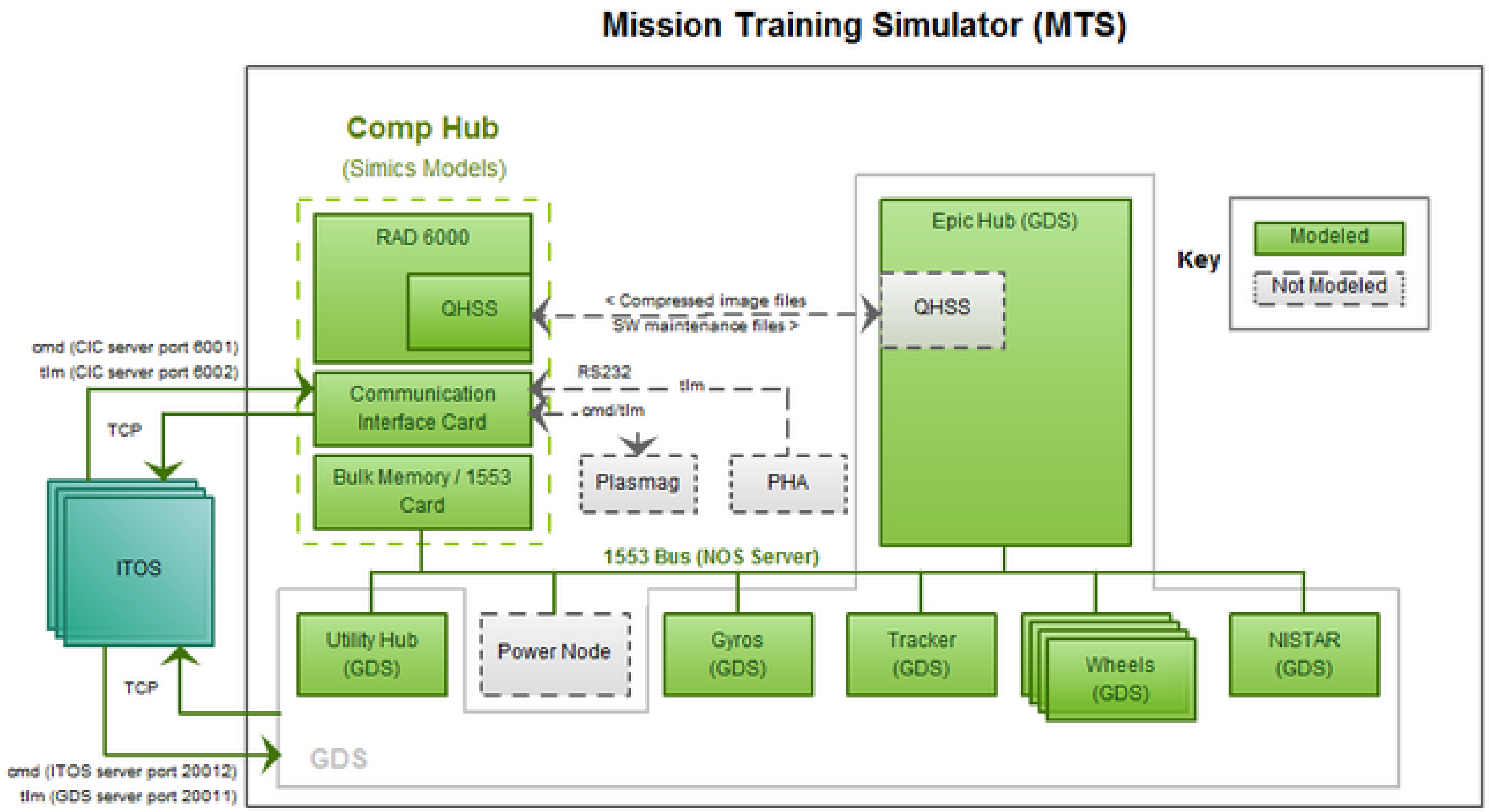
JIST Architecture



Evolution of ITC Spacecraft Simulators

Deep Space Climate Observatory (DSCOVR)

DSCOVR Architecture



Simulator Level-of-Effort Comparison

Year Usage	Simulator	Effort	Prototype (Basic C&DH)	Complexity	Users
2011-2014	GO-SIM	2 FTEs	6 Months	Medium	IV&V, GPM Project Testers Launch Support
2012 - Ongoing	JIST	2 FTEs	4 Months	Very High	IV&V, JWST Test Labs, JWST Operations
2013 - Ongoing	DSCOVR	1 FTE	2 Months	Low	DSCOVR Testers DSCOVR Operations

Evolution Lessons Learned

- Establishment of a reusable simulation architecture has proven to save costs and reduce future effort
- Automate tests and deployments as much as possible as it allows for engineers to focus on more challenging tasks
- Hardware modeling should focus on the minimum needed in order for the flight software to execute. Establish this baseline then augment to support full V&V dynamic testing using an iterative process.
- Spend considerable time writing unit tests for the hardware models. When things go wrong, debugging is very difficult.
- Integration of simulators to form a system will require significant development labor, cost, and time.

Contact Information

- Web Page
  - http://www.nasa.gov/centers/ivv/jstar/JSTAR.html
- E-Mail
  - Scott.Zemerick@TMCTechnologies.com
  - Justin.R.Morris@nasa.gov
- Contact us for…
  - Demonstrations of test beds
  - Middleware usage agreements
  - Simulator development
  - Hardware modeling
  - V&V Services, HWIL Testing, Performance Testing