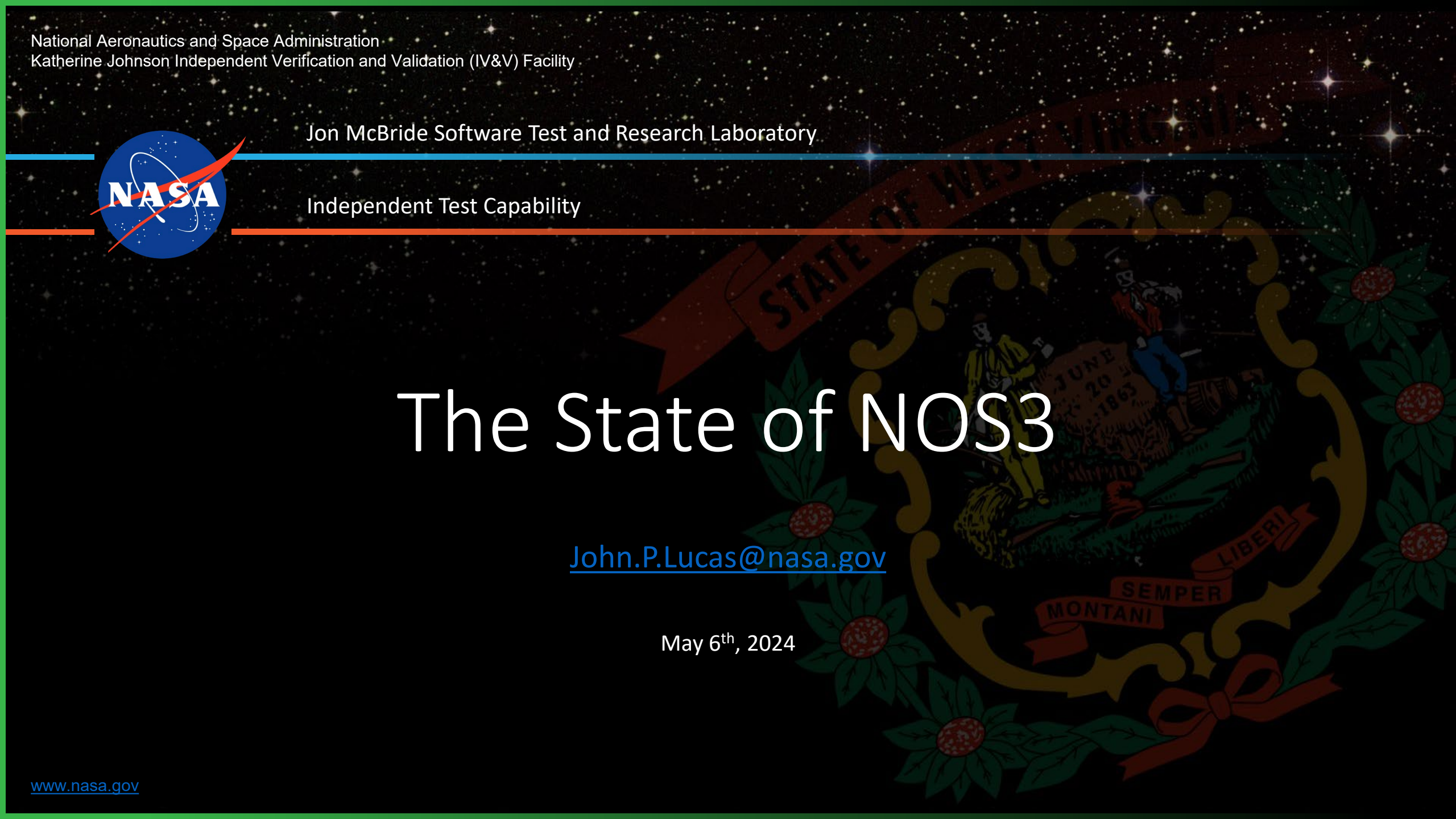
National Aeronautics and Space Administration
Katherine Johnson Independent Verification and Validation (IV&V) Facility

Jon McBride Software Test and Research Laboratory

Independent Test Capability



The State of NOS3

John.P.Lucas@nasa.gov

May 6th, 2024

Agenda

- Background
  - NASA IV&V
  - JSTAR Laboratory
  - STF-1
- NOS3
  - Overview
  - Architecture
  - Recent efforts
  - Ongoing efforts
- Benefits
- Path forward

Katherine Johnson IV&V Facility

- Independent Verification and Validation (IV&V)
- Enabling NASA Mission Success since 1993
  - Established as a direct result of the NRC recommendations after the Challenger accident
- Contributes to the safety and success of NASA's highest-profile missions by assuring the software on those missions performs correctly
- Involved in the full development lifecycle
  - Software and System Requirements, Design, Code, Test, AI&T, etc.

Katherine G. Johnson

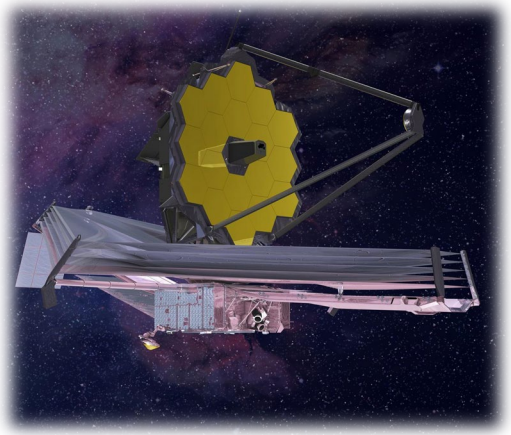
Jon McBride Software Test and Research

- Digital Twin Development Team
  - Team focused on the acquisition, development, deployment, and maintenance of test environments to support IV&V independent testing
- Risk Reduction Testing
  - Directly support and work alongside IV&V team to identify, execute, and record the results of independent tests
- Simulation & Software Services
  - Provide flight software and simulation services directly to NASA missions and other government organizations

Captain Jon McBride is West Virginia's First Astronaut

Jon McBride Software Test and Research

Condense the entire flight system to run on a laptop or cloud-based solution



1. Flight computer hardware is **emulated**
2. Sensors / actuators are simulated
3. Flight software binaries executed as delivered
4. Ground operations software integrated as-is

JSTAR is a Digital Twin Factory

Mission	Platform
Integrated Tri-Program Simulation	ARRISTOTLE
Mars Sample Return, CCRS	CHASE
Europa Clipper	ECHOES
Gateway	Gateway-in-a-Box
Exploration Ground System (EGS)	GS2
HLS	HOOTLE
James Webb Space Telescope (JWST)	JIST
GSFC SmallSat Product Line	NOS3
Psyche	PHASE
Space Launch System	S3
Multi-Purpose Crew Vehicle (MPCV)	SOCRRATES
LM Product Line	Soft Sim
Roman Space Telescope (RST)	WISP

ARRISTOTLE
ADVANCED RISK REDUCTION INTEGRATED SOFTWARE TEST & OPERATIONS TRI-PROGRAM LIGHTWEIGHT ENVIRONMENT

CHASE
CCRS Hardware Abstraction Simulation Environment

ECHOES
Europa Clipper Hardware Abstraction Operational Environment System

JIST

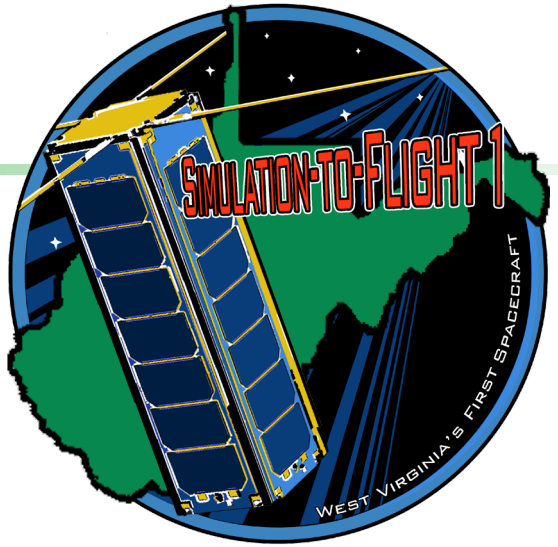
nos3

PHASE

WISP
RST Digital Twin

S3
Space Launch System
Software Only Simulation

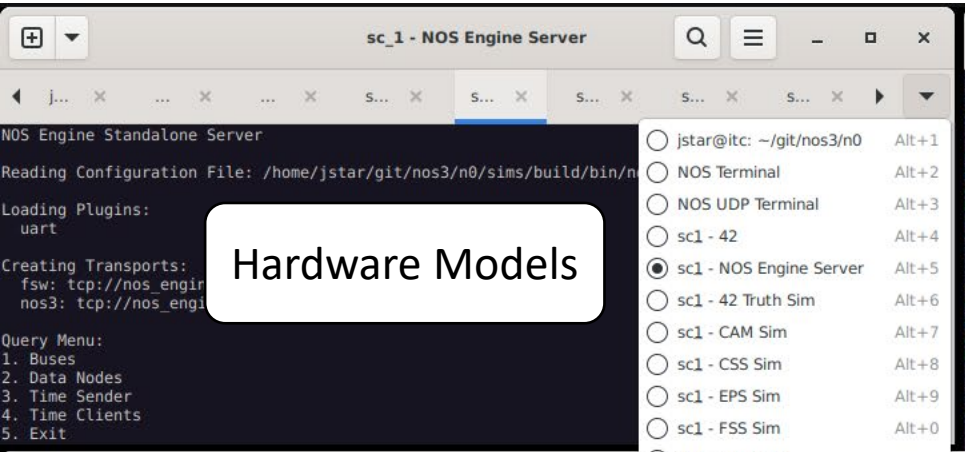
Simulation To Flight 1



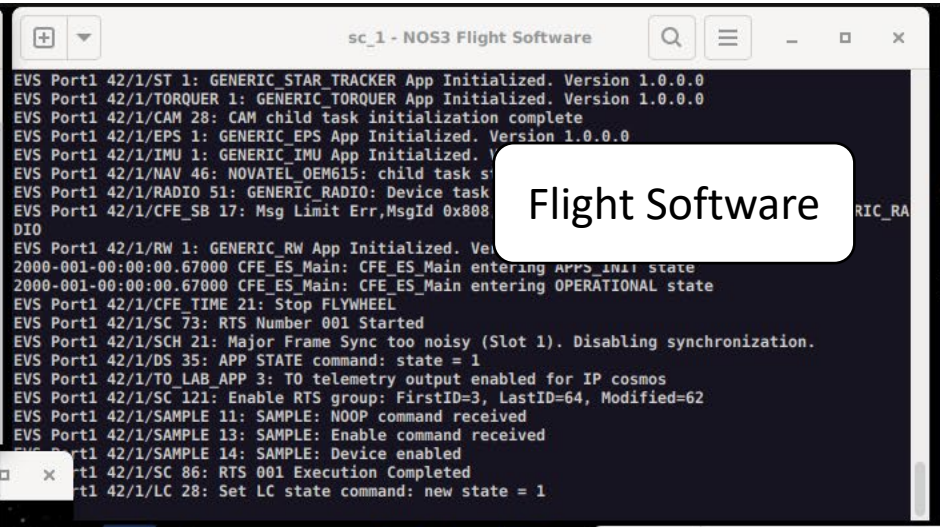
- West Virginia's first spacecraft
- Partnered with WVU and leveraged existing NASA GSFC efforts with Dellingr
- Demonstrated digital twin capabilities to improve mission processes
- Provided an open-source solution for researchers and technology development
- Resulted in the NOS3 platform
  - Initially a framework
  - Now a design reference mission

NOS3 Simulation Example

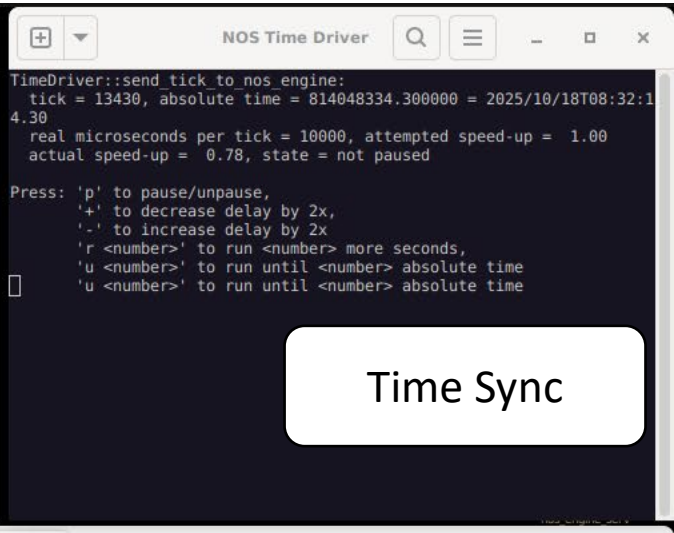
IV&V



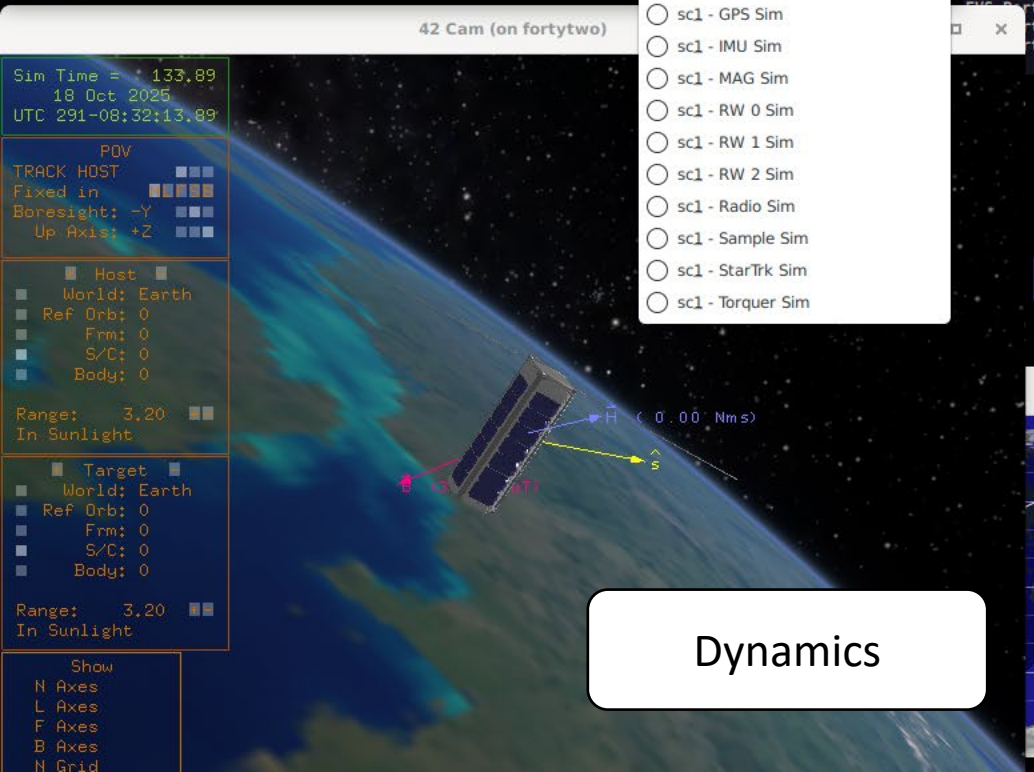
Hardware Models



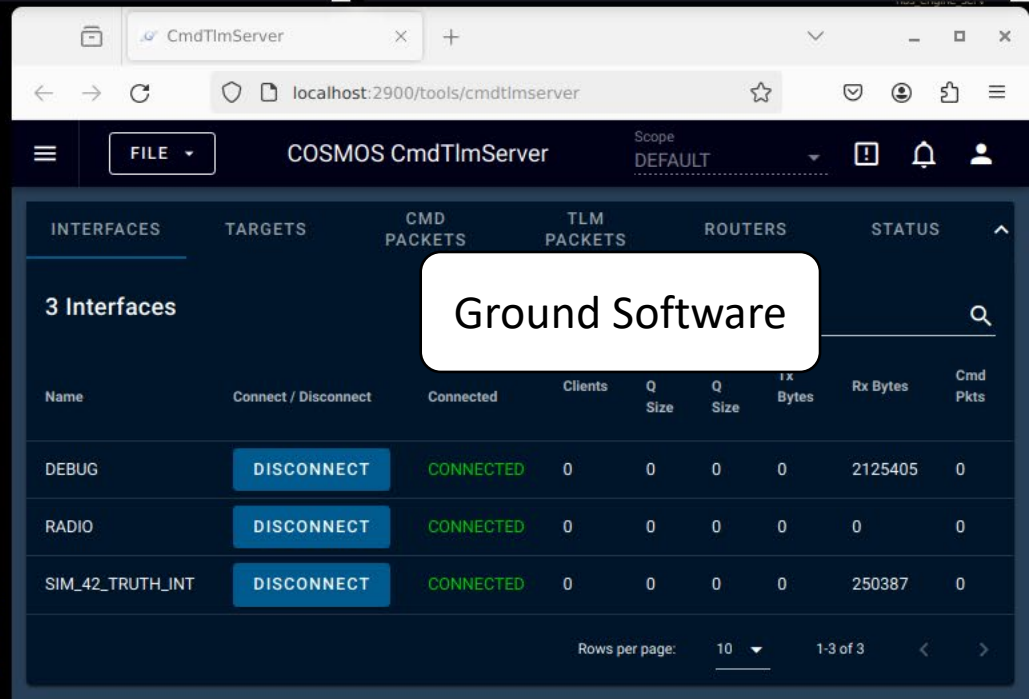
Flight Software



Time Sync



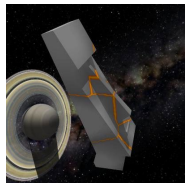
Dynamics

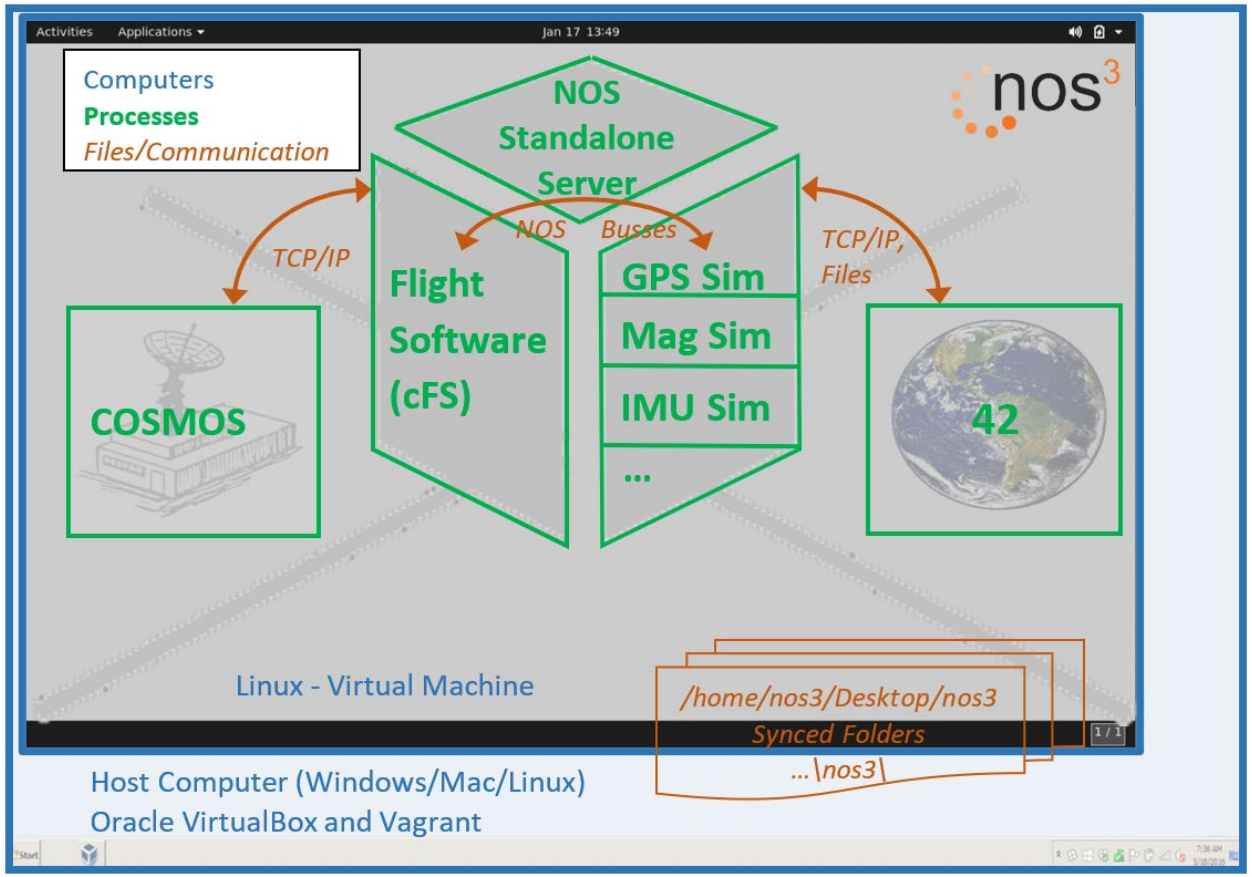


Ground Software

NOS3 Architecture

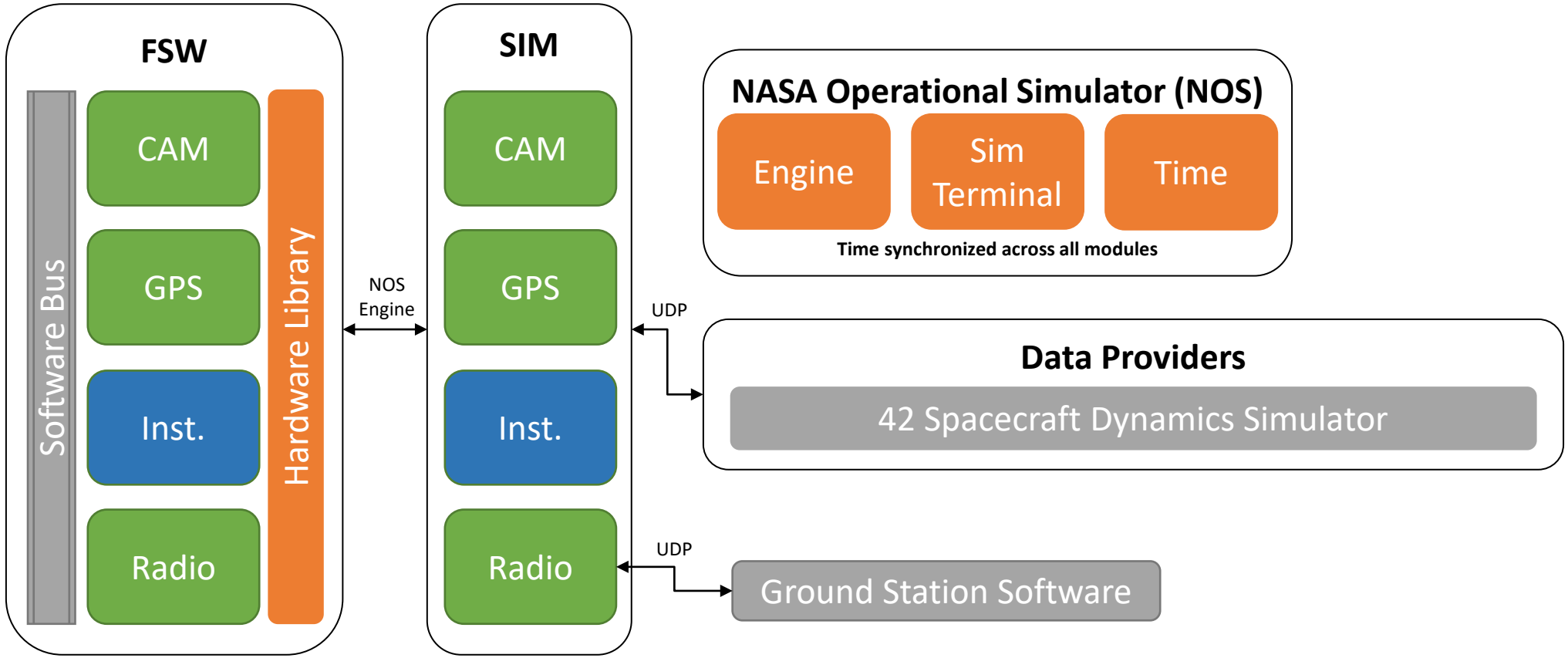
- Combination of multiple pieces of open-source software
  - A virtual spacecraft!
  - Learn on, do research, baseline as your own
- Flight Software
  - Core Flight System (cFS)
- Ground Stations
  - Ball Aerospace COSMOS
  - JPL AMMOS Instrument Toolkit
  - OpenC3 COSMOS
- Dynamics / Visualization
  - 42
- NOS Engine
  - Communication protocols
  - Hardware abstraction layer
  - Component simulators





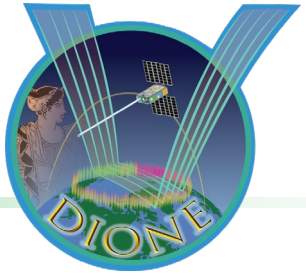
NOS3 Architecture

- NOS3 enables cross compiling for your flight and simulation targets
  - HWLIB enables selection of build type via flags at compile time



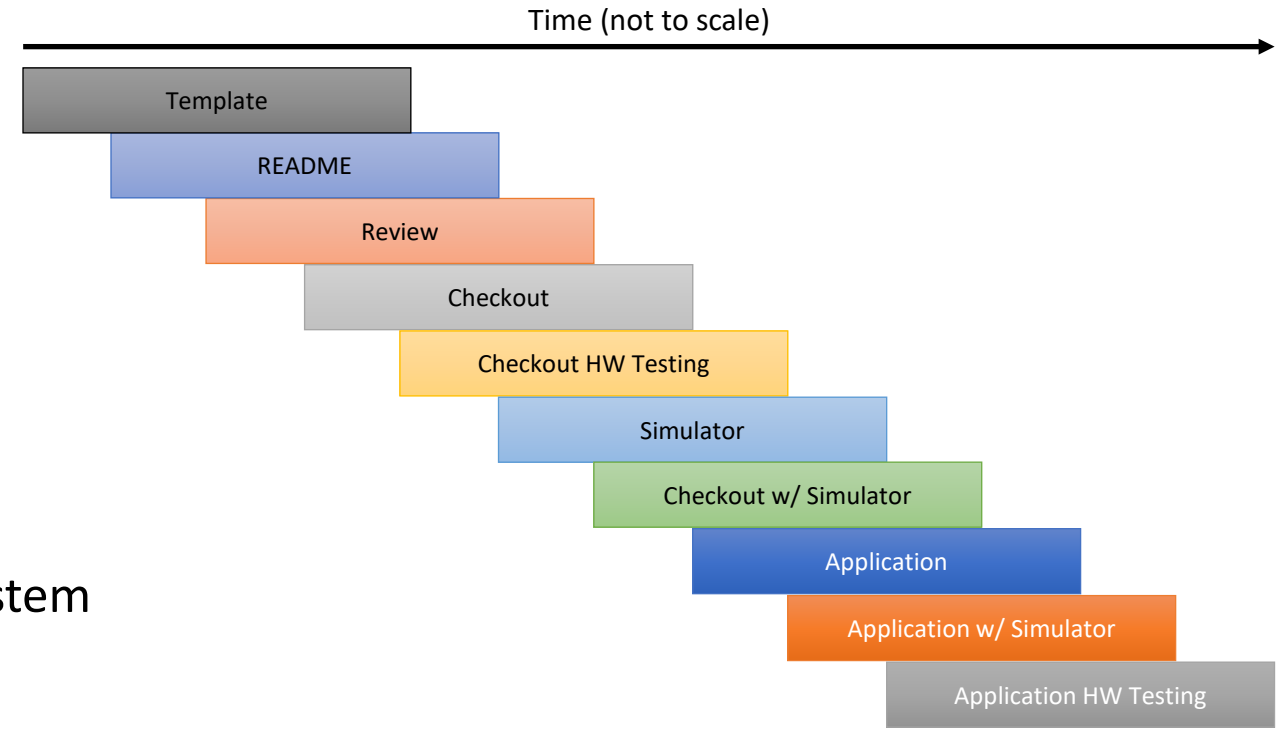
Design Reference Mission

- NOS3 benefits have been proven across multiple missions
- Each mission develops several hardware specific applications
  - Due to licensing concerns these are not available outside of NASA for re-use
- Missions often become so specific that implementations require a complete understanding
  - No easy method of training or baselining for new efforts
- To provide a baseline for advanced development in NOS3 a complete mission is required
  - Generic components and interfacing was the first step to resolve this



Completed Efforts – Generic Components

- As a GSFC baseline was established, the component flow was formed
  - Iterative process to develop and improve FSW application, simulation, and tests
- All code for a component stored in a single repository
  - Maintain sync between various pieces
  - Simplifies updates for multiple efforts
- Template enables readability
  - Specifics to component in known files
- Development flow enables compatibility
  - Able to start development immediately
  - Simulators are no replacement for hardware
- Generic components currently available
  - Coarse Sun Sensor, Fine Sun Sensor
  - Electrical Power System, Global Positioning System
  - Inertial Measurement Unit, Magnetometer
  - Radio, Reaction Wheel
  - Star Tracker, Torquers

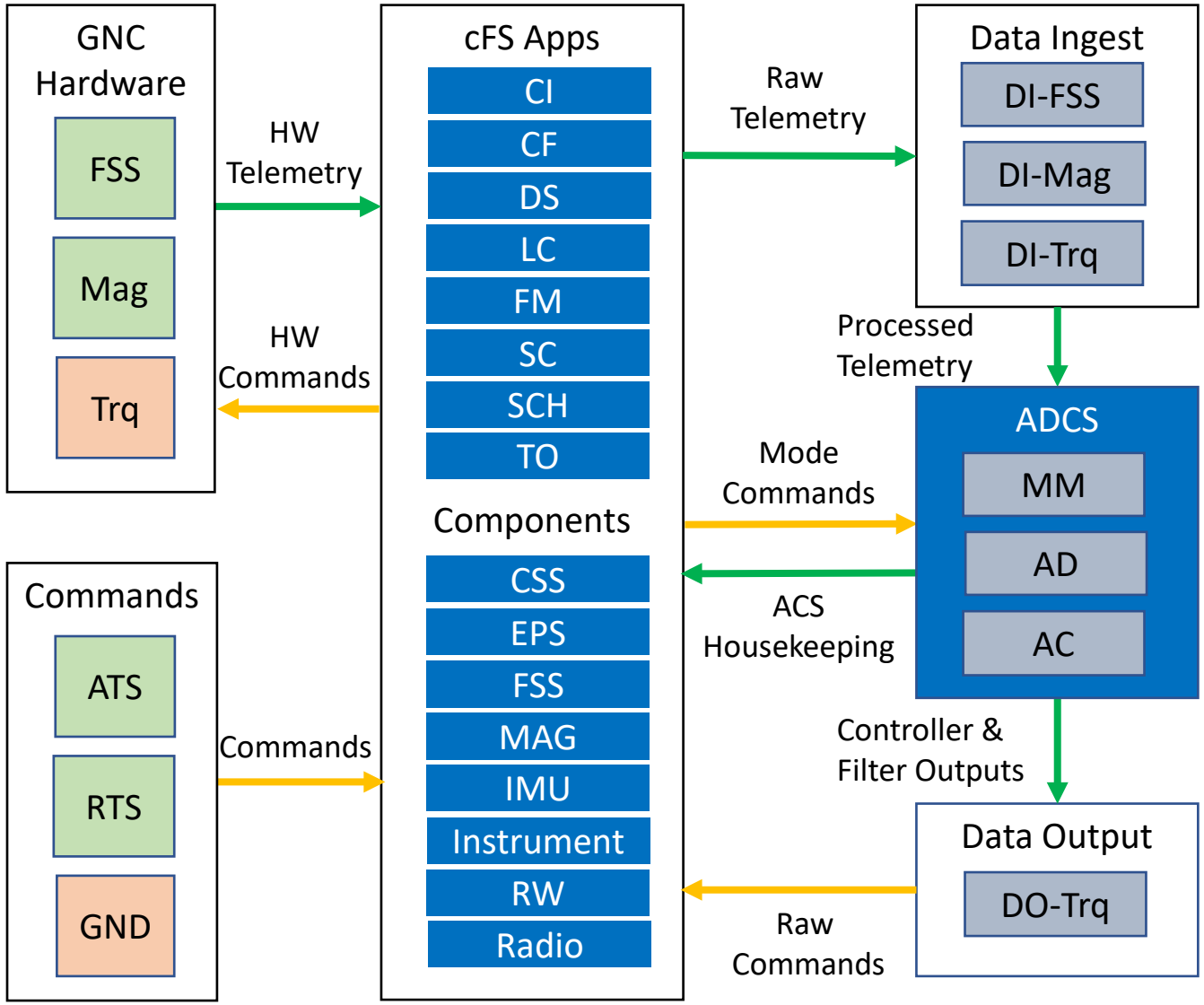


Completed Efforts – Containerization

- In and effort to ease deployment, docker containerization was done
  - Podman implementation planned for future
- Open-source container available for use and review
  - https://github.com/nasa-itc/deployment
  - https://hub.docker.com/r/ivvitc/nos3-64
- Vagrant base box still available if desired
  - Leverages docker internally
  - Allows shared folders from Host > VM > Docker
  - Still able to run faster than real time
- Created helper scripts
  - `make debug` to launch container image
- Upgraded modules to latest versions
  - 42, cFS, NOS Engine, OpenC3 COSMOS

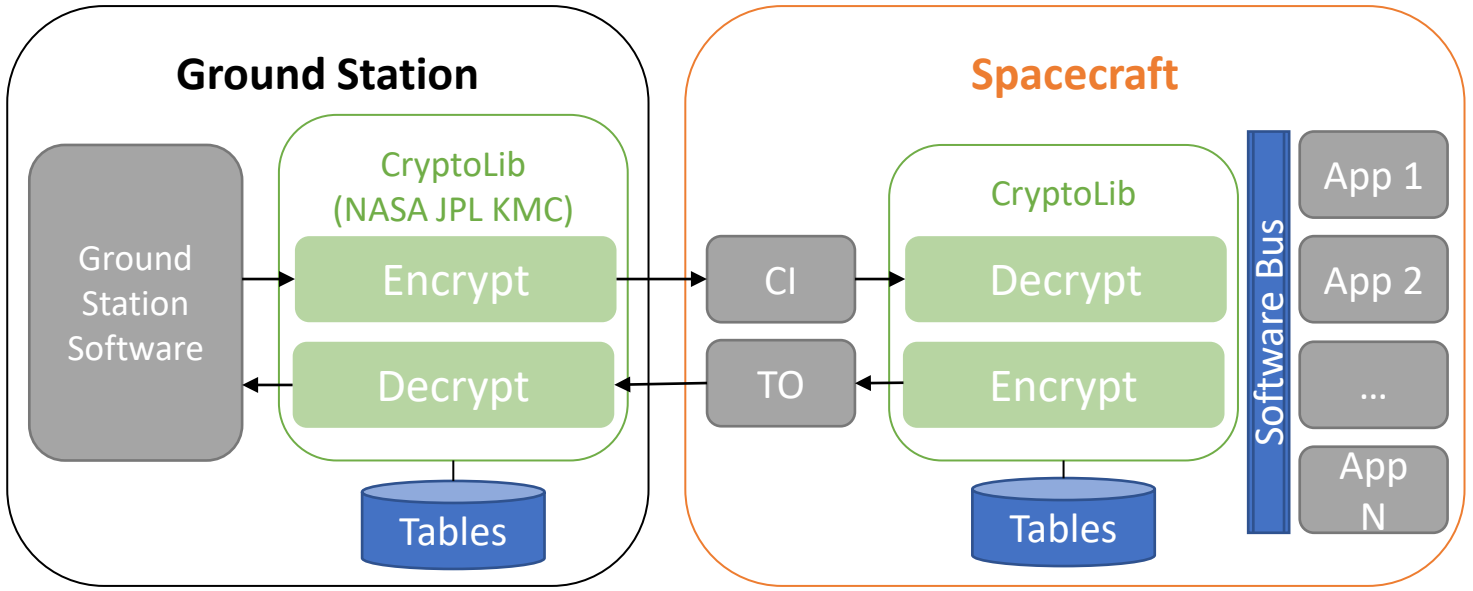
Completed Efforts – Generic ADCS

- With generic versions of actuators and sensors, higher level generic libraries can be established
  - Generic ADCS first to be realized
- Currently supports:
  - Passive, BDOT, and Sun Safe
  - 10Hz closed loop interfacing with components and dynamics
- Plan to implement:
  - Three axis pointing
  - Sun pointed spinning



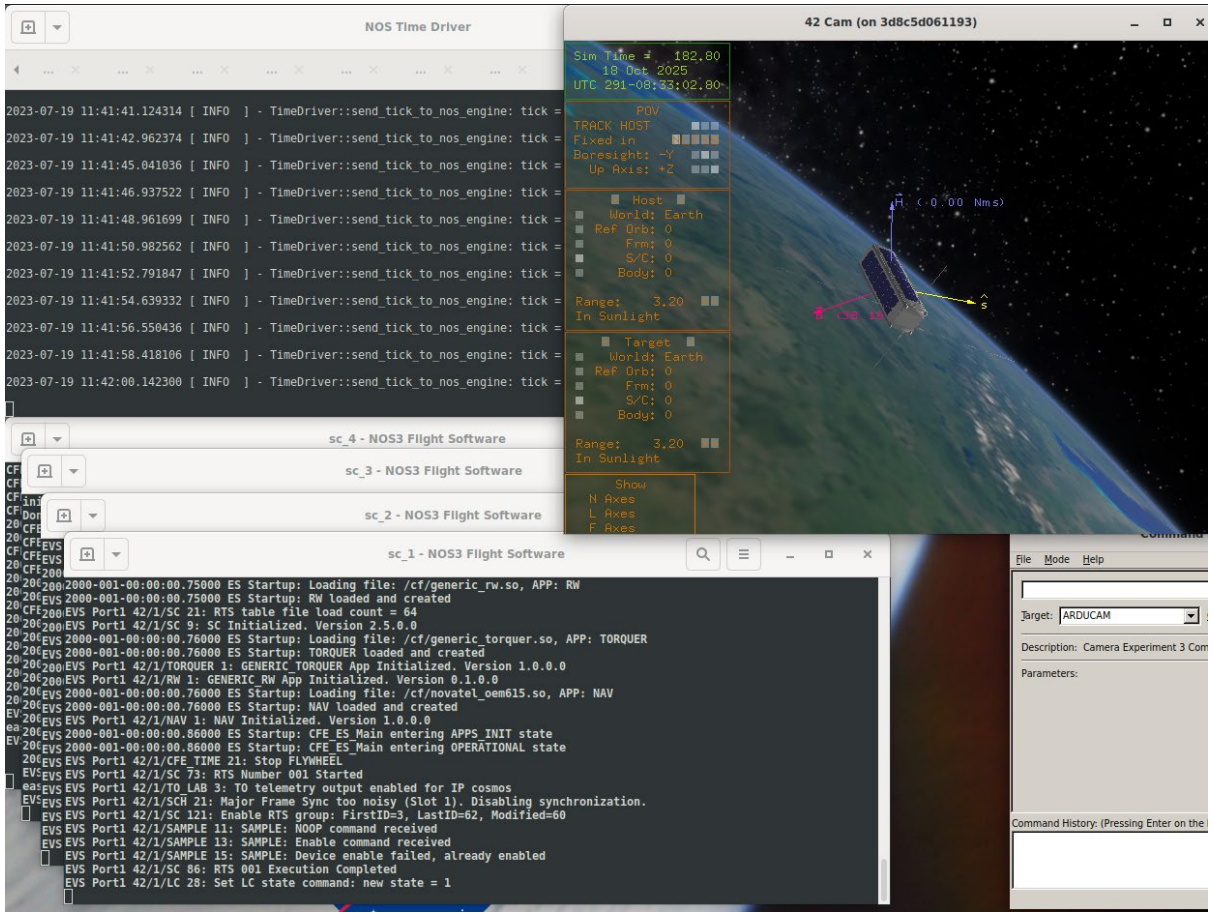
Completed Efforts – CryptoLib

- Goals
  - Provide a software only solution
  - Support both flight and ground
- Standards supported
  - Space Data Link Security (SDLS)
    - TC / TM / AOS
- Standards to be supported
  - SDLS – Extended Procedures (EP)
    - TC / TM / AOS
- Interchangeable Modules
  - Cryptography
  - Key Management
  - Monitoring and Control
  - Security Association



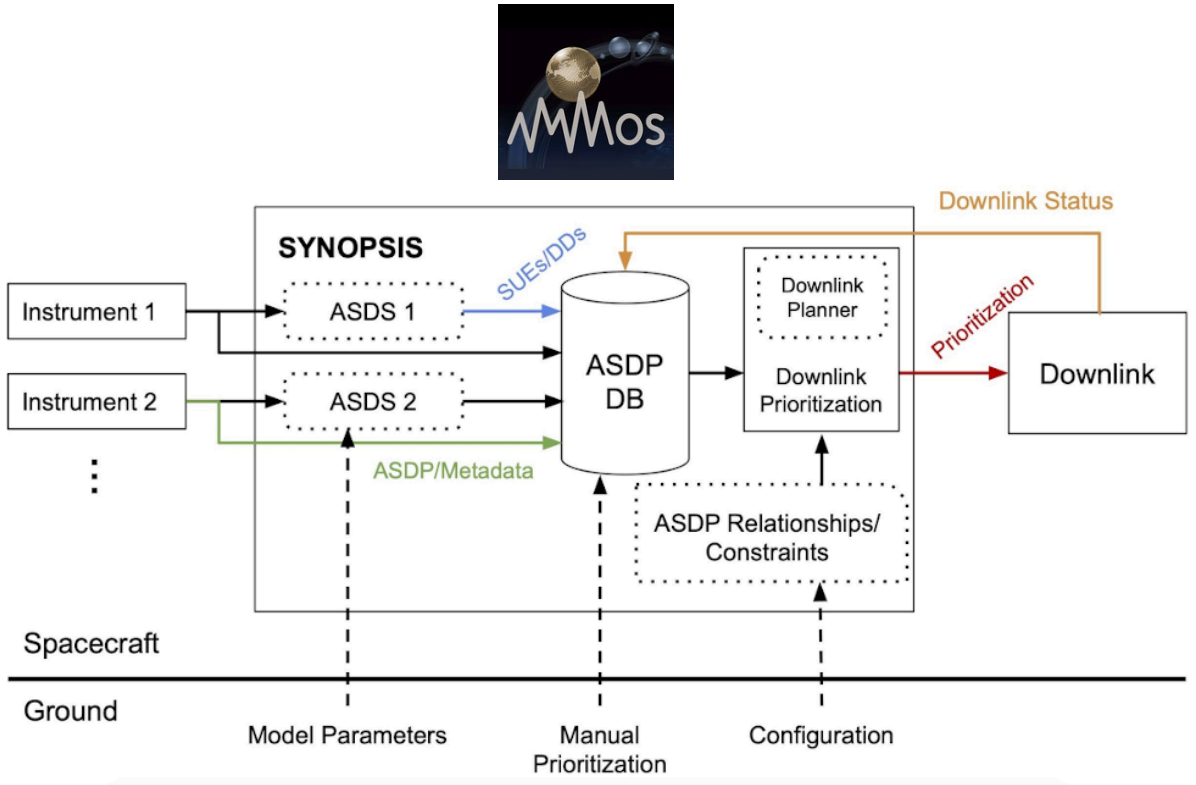
Completed Efforts – Multiple Spacecraft

- Developed launcher scripts to change the number of spacecraft desired
- Demonstrated included string of pearls constellation each sending its status to the next in the chain
  - SC0 identifies cool science
  - SC0 sends message to SC1 to get into correct mode to collect data
- Single time driver maintaining sync across all pieces of architecture
  - All spacecraft running similar FSW
  - Docker networking playing large role



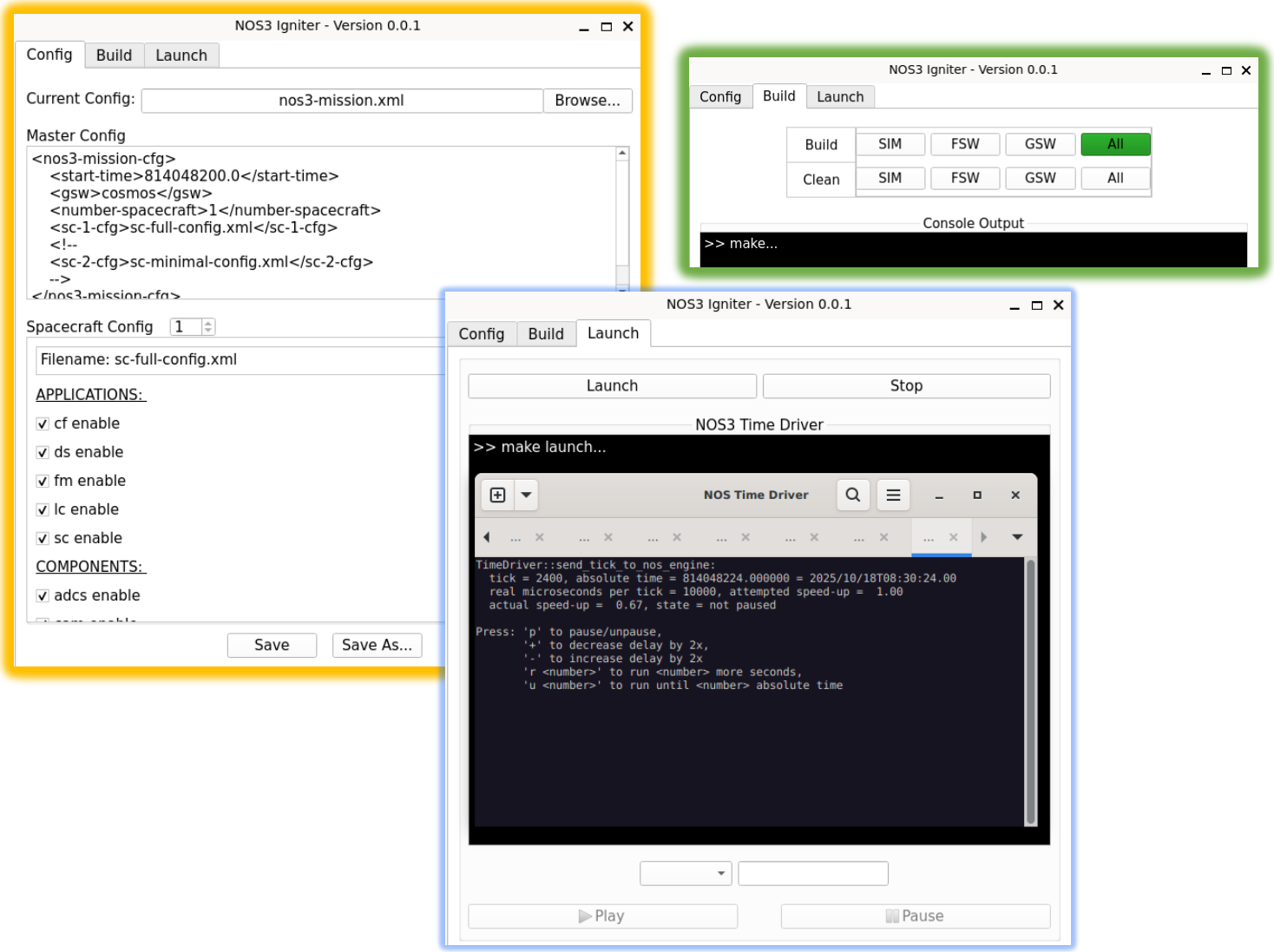
Completed Efforts – SYNOPSIS

- Science Yield Improvement via Onboard Prioritization and Summary of Information System
  - https://github.com/NASA-AMMOS/synopsis
- Designed to significantly enhance scientific return from interplanetary spacecraft missions
- Transition data strategies from “collect some, return all” to “collect much, return best”



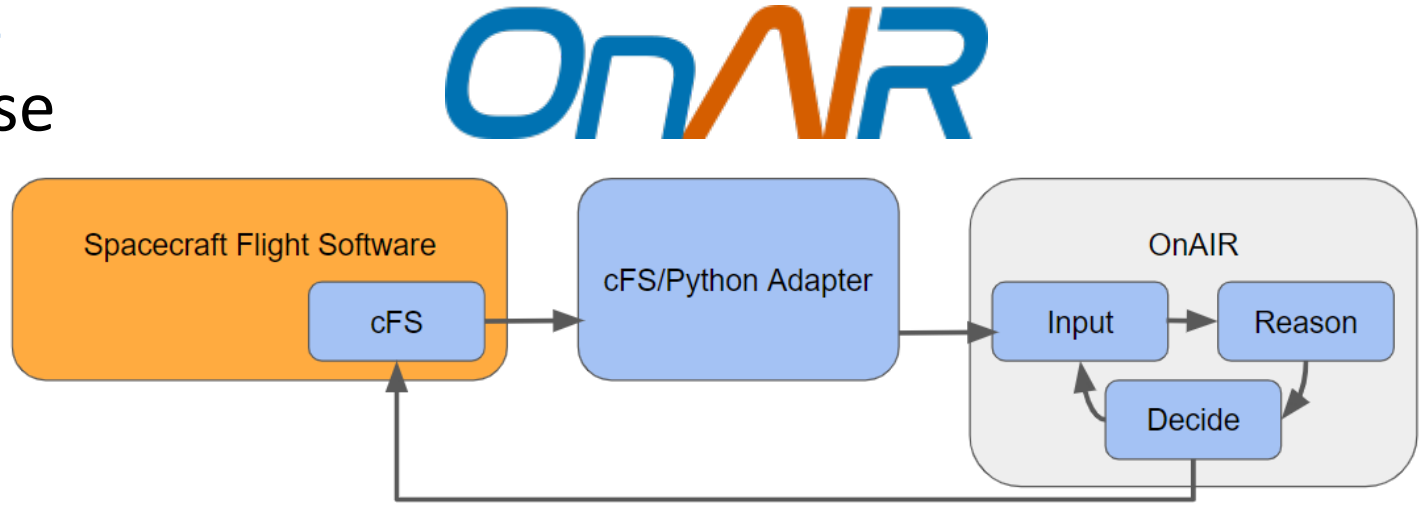
Current Efforts – Igniter

- GUI tool to enable easy reconfiguration
  - Python based
- Mission configuration
  - Start time
  - Ground software to use
  - Number of spacecraft and configurations
- Spacecraft configuration
  - cFS application enable / disable
  - Component enable / disable
  - 42 GUI and initial tipoff rates
- System test framework
  - Allow staging and running of tests directly from the GUI
  - Reduce knowledge needed to repeat tests and perform verification



Current Efforts – OnAIR

- On-Board Artificial Intelligence Research
  - https://github.com/nasa/OnAIR/
- Spacecraft need tools to increase their autonomy for both fault recovery and science data processing
- OnAIR provides:
  - An open-source architecture to enable continued research and development
  - Isolation to enable high level languages on a flight project



Current Efforts – F’

- Flight Software & Embedded Systems Framework
  - https://nasa.github.io/fprime/
- Component architecture with well-defined interfaces
- C++ framework providing core capabilities like queues, threads, and operating-system abstraction
- Tools for designing systems and automatically generating code from systems design
- A standard library of flight-worthy components
- Testing tools for unit and system-level testing

NASA’s Ingenuity Mars Helicopter

Which disciplines can benefit?

Developers
Software development environment
Early development opportunities before hardware arrives

I&T
Enables end-to-end testing early
Manipulate hardware models

Operations
Dry-run scenarios
Day in the life testing

V&V
Increases available test resources
Code coverage
Exploratory testing

New Personnel
Training platform that you cannot break
Easily run lab tests to demonstrate faults

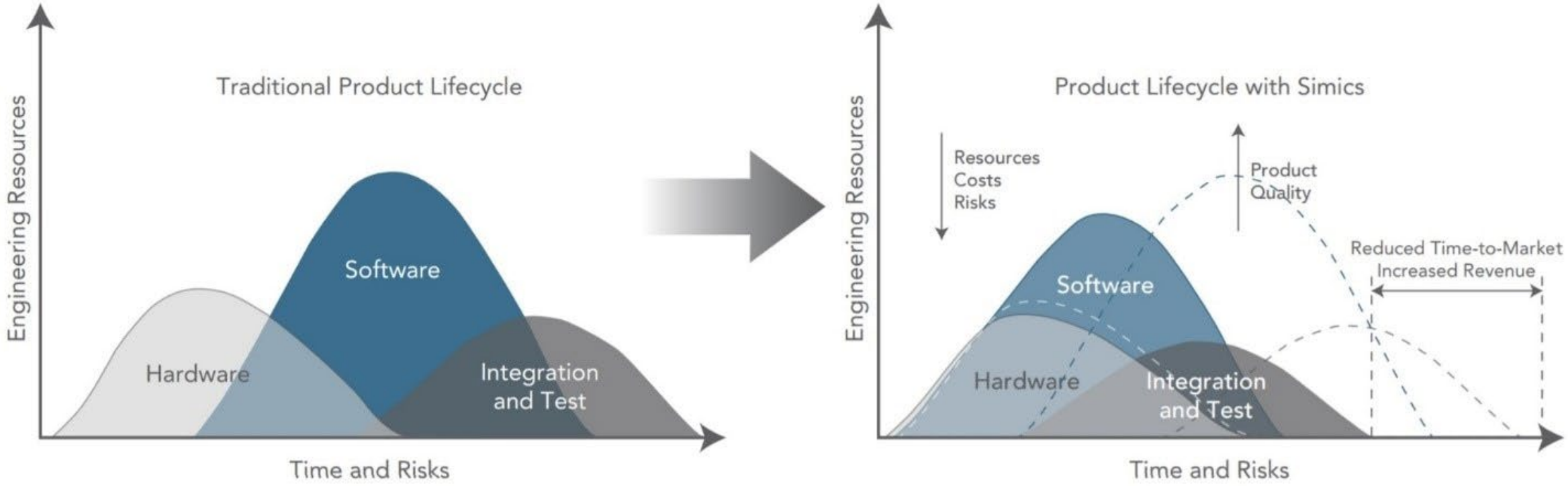
IV&V
Risk reduction testing and mission assurance
Hardware modeling and fault injection

Researchers
Integrate and demonstrate new technologies

New uses and users reported regularly!

What's the driver for project utilization?

- Digital twins shift the traditional software development cycle
- Digital twins greatly increase test resources and opportunities



Shifting Left—Building Systems & Software Before Hardware Lands (intel.com)

Path Forward

- Mission Cloud Platform deployment
  - Use the new containerized interfaces to spawn and run system tests
- Proof of concept deployment - distributed systems mission
  - Constellation of spacecraft around the moon in a variety of configurations
- Documentation, maintenance, tutorials, and trainings
  - Produce video series and training package
- CI/CD and unit testing across the board
- Intermix different types of systems (landers, drones, satellites, relays, etc.)
  - Integrate a lander with its own FSW and simulation platform into the constellation
- Partnerships with university and industry to develop COTS models
- Processor emulation through QEMU
  - Running RTEMS
- **Open to community requests!**

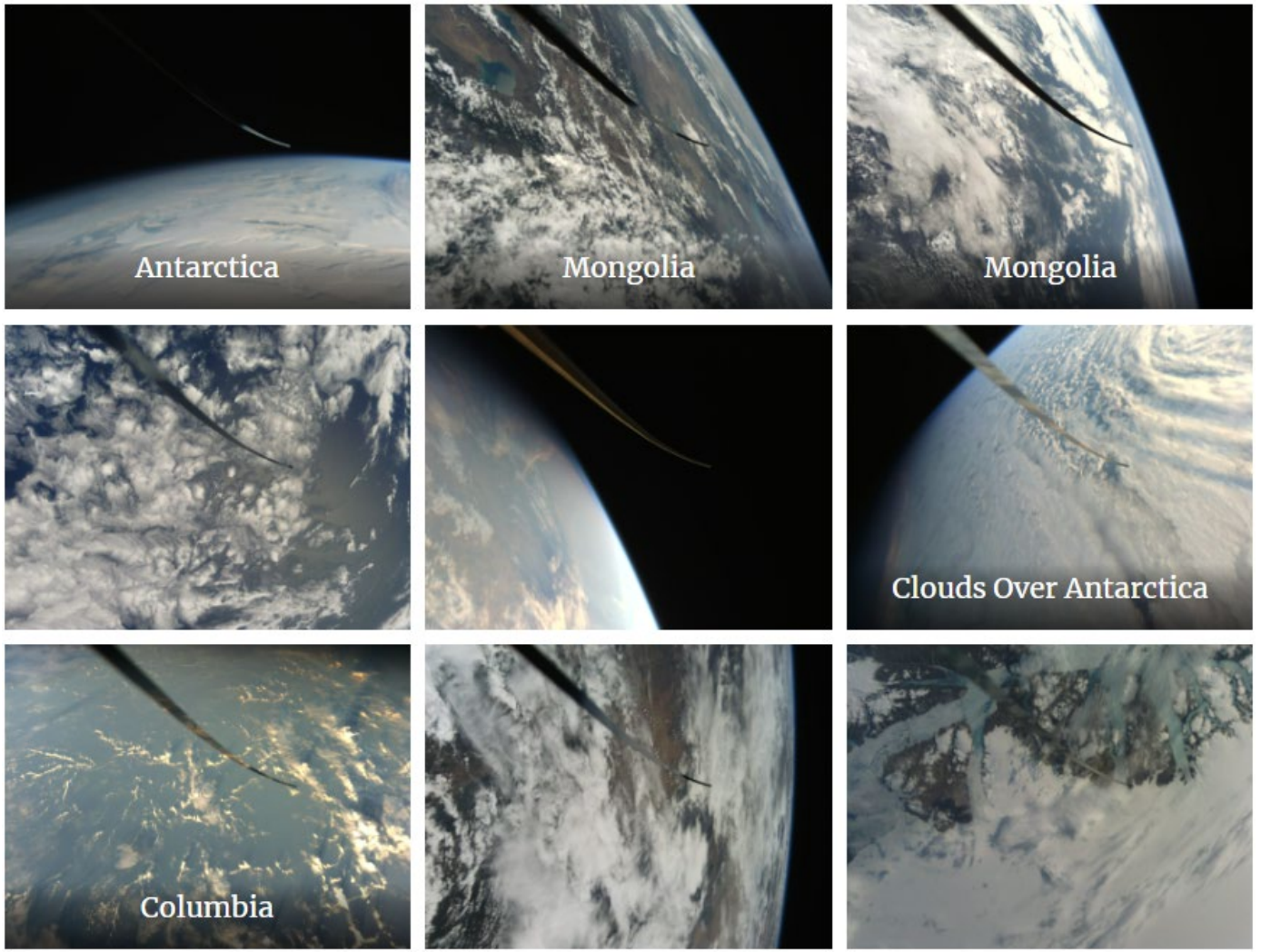
Thanks to everyone who supports this work!

IV&V

Goddard SPACE FLIGHT CENTER

JPL

Jet Propulsion Laboratory
California Institute of Technology



http://stf1.com/news/?m=202212