

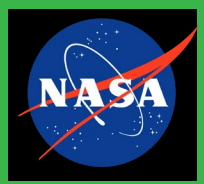
Jon McBride Software Test and Research Laboratory

Independent Test Capability

# The State of NOS3

[John.P.Lucas@nasa.gov](mailto:John.P.Lucas@nasa.gov)

May 6<sup>th</sup>, 2024

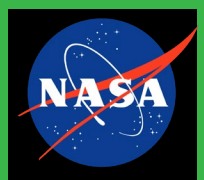


# Agenda



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



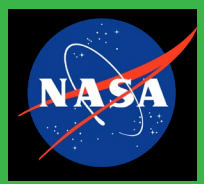


# Katherine Johnson IV&V Facility



- 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.





# Jon McBride Software Test and Research

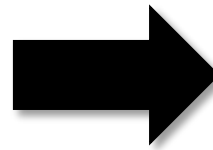
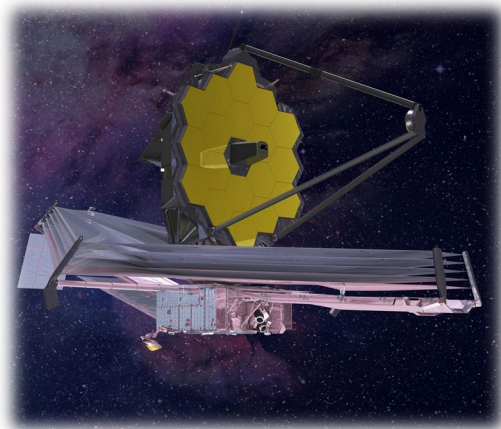
- Digital Twin Development Team
  - Team focused on the acquisition, development, deployment, and maintenance of test environments to support IV&V independent testing
- Risk Reduction Testing
  - Directly support and work alongside IV&V team to identify, execute, and record the results of independent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



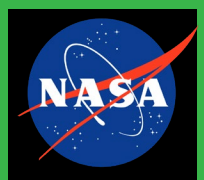
*Captain Jon McBride is West Virginia's First Astronaut*

# 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

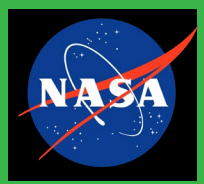


# JSTAR is a Digital Twin Factory



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)	SOCRATES
LM Product Line	Soft Sim
Roman Space Telescope (RST)	WISP

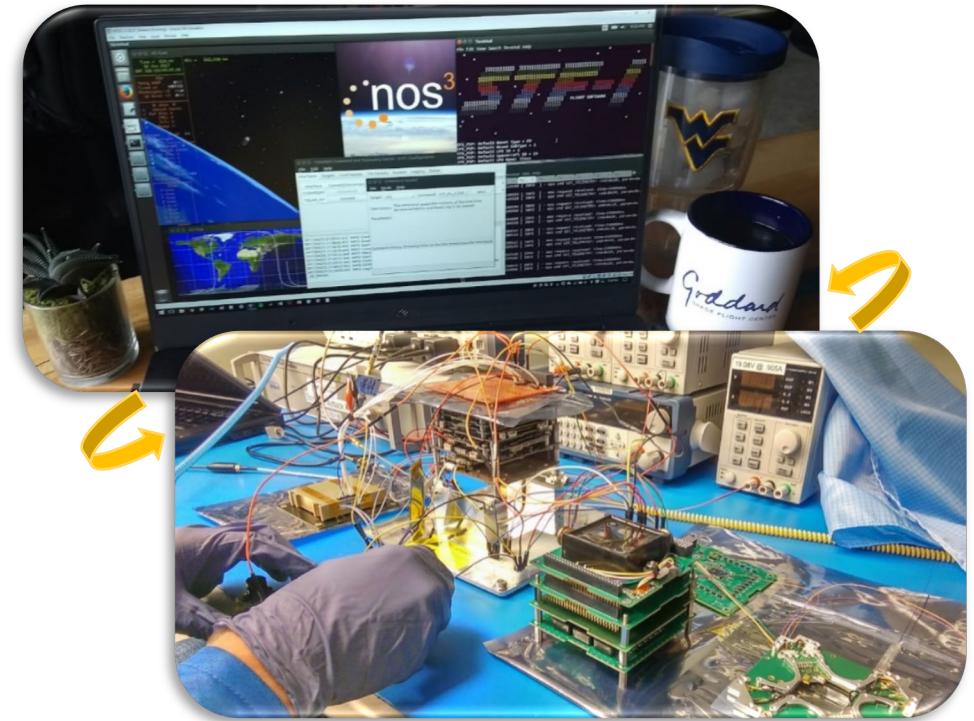
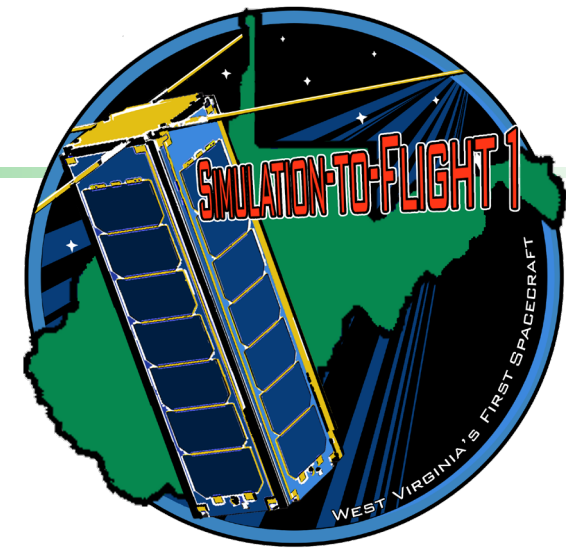


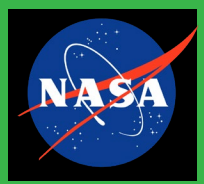


# Simulation To Flight 1



- West Virginia's first spacecraft
- Partnered with WVU and leveraged existing NASA GSFC efforts with Dellinger
- 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



sc\_1 - NOS Engine Server

NOS Engine Standalone Server

Reading Configuration File: /home/jstar/git/nos3/n0/sims/build/bin/n...

Loading Plugins:  
uart

Creating Transports:  
fsw: tcp://nos\_engir  
nos3: tcp://nos\_engi

Query Menu:  
1. Buses  
2. Data Nodes  
3. Time Sender  
4. Time Clients  
5. Exit

42 Cam (on fortytwo)

Hardware Models

- jstar@itc: ~/git/nos3/n0 Alt+1
- NOS Terminal Alt+2
- NOS UDP Terminal Alt+3
- sc1 - 42 Alt+4
- sc1 - NOS Engine Server Alt+5
- sc1 - 42 Truth Sim Alt+6
- sc1 - CAM Sim Alt+7
- sc1 - CSS Sim Alt+8
- sc1 - EPS Sim Alt+9
- sc1 - FSS Sim Alt+0
- sc1 - GPS Sim
- sc1 - IMU Sim
- sc1 - MAG Sim
- sc1 - RW 0 Sim
- sc1 - RW 1 Sim
- sc1 - RW 2 Sim
- sc1 - Radio Sim
- sc1 - Sample Sim
- sc1 - StarTrk Sim
- sc1 - Torquer Sim

sc\_1 - NOS3 Flight Software

```

EVS Port1 42/1/ST 1: GENERIC STAR TRACKER App Initialized. Version 1.0.0.0
EVS Port1 42/1/TORQUER 1: GENERIC TORQUER App Initialized. Version 1.0.0.0
EVS Port1 42/1/CAM 28: CAM child task initialization complete
EVS Port1 42/1/EPS 1: GENERIC EPS App Initialized. Version 1.0.0.0
EVS Port1 42/1/IMU 1: GENERIC IMU App Initialized.
EVS Port1 42/1/NAV 46: NOVATEL OEM615: child task s
EVS Port1 42/1/RADIO 51: GENERIC RADIO: Device task
EVS Port1 42/1/CFE_SB 17: Msg Limit Err,MsgId 0x808
DIO
EVS Port1 42/1/RW 1: GENERIC RW App Initialized. Ver
2000-001-00:00:00.67000 CFE_ES_Main: CFE_ES_Main entering APPS_INIT state
2000-001-00:00:00.67000 CFE_ES_Main: CFE_ES_Main entering OPERATIONAL state
EVS Port1 42/1/CFE TIME 21: Stop FLYWHEEL
EVS Port1 42/1/RW 1: GENERIC RW App Initialized. Ver
EVS Port1 42/1/SC 73: RTS Number 001 Started
EVS Port1 42/1/SCH 21: Major Frame Sync too noisy (Slot 1). Disabling synchronization.
EVS Port1 42/1/DS 35: APP STATE command: state = 1
EVS Port1 42/1/TO LAB_APP 3: TO telemetry output enabled for IP cosmos
EVS Port1 42/1/SC 121: Enable RTS group: FirstID=3, LastID=64, Modified=62
EVS Port1 42/1/SAMPLE 11: SAMPLE: NOOP command received
EVS Port1 42/1/SAMPLE 13: SAMPLE: Enable command received
EVS Port1 42/1/SAMPLE 14: SAMPLE: Device enabled
EVS Port1 42/1/SC 86: RTS 001 Execution Completed
EVS Port1 42/1/LC 28: Set LC state command: new state = 1
  
```

Flight Software

NOS Time Driver

```

TimeDriver::send_tick to nos engine:
tick = 13430, absolute time = 814048334.300000 = 2025/10/18T08:32:14.30
real microseconds per tick = 10000, attempted speed-up = 1.00
actual speed-up = 0.78, state = not paused

Press: 'p' to pause/unpause,
      '+' to decrease delay by 2x,
      '-' to increase delay by 2x
      'r <number>' to run <number> more seconds,
      'u <number>' to run until <number> absolute time
      'U <number>' to run until <number> absolute time
  
```

Time Sync

42 Cam (on fortytwo)

Sim Time = + 133.89  
18 Oct 2025  
UTC 291-08:32:13.89

POV  
TRACK HOST  
Fixed in  
Boresight: -Y  
Up Axis: +Z

Host  
World: Earth  
Ref Orb: 0  
Frm: 0  
S/C: 0  
Body: 0

Range: 3.20  
In Sunlight

Target  
World: Earth  
Ref Orb: 0  
Frm: 0  
S/C: 0  
Body: 0

Range: 3.20  
In Sunlight

Show  
N Axes  
L Axes  
F Axes  
B Axes  
N Grid

Dynamics

JSTAR

42 Map (on fortytwo)

Tlm Time: 2025-291-08:32:13

localhost:2900/tools/cmdtlmserver

COSMOS CmdTlmServer

Scope: DEFAULT

INTERFACES TARGETS CMD PACKETS TLM PACKETS ROUTERS STATUS

3 Interfaces

Name	Connect / Disconnect	Connected	Clients	Q Size	Q Size	Tx Bytes	Rx Bytes	Cmd Pkts
DEBUG	DISCONNECT	CONNECTED	0	0	0	0	2125405	0
RADIO	DISCONNECT	CONNECTED	0	0	0	0	0	0
SIM_42_TRUTH_INT	DISCONNECT	CONNECTED	0	0	0	0	250387	0

Rows per page: 10 1-3 of 3

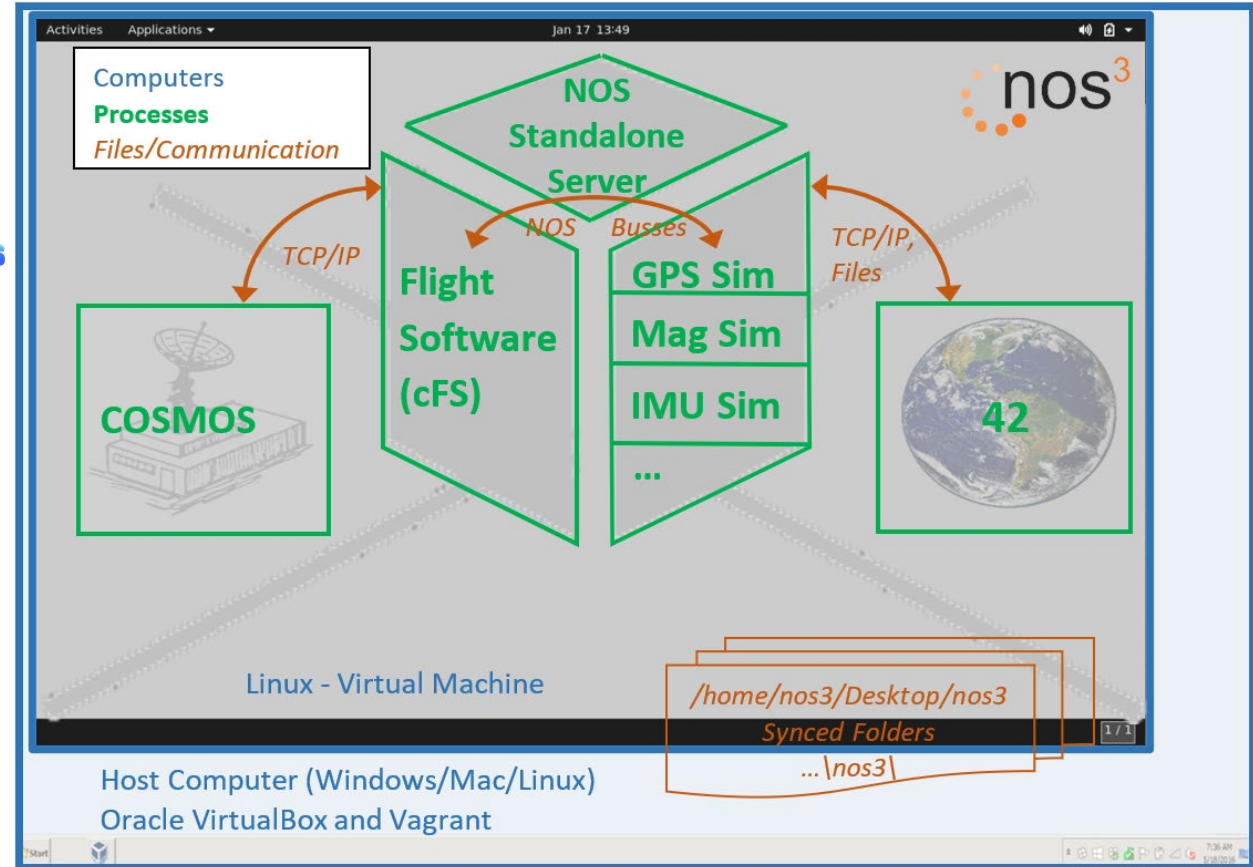
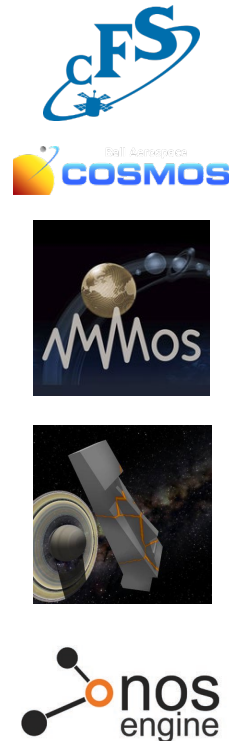
Ground Software

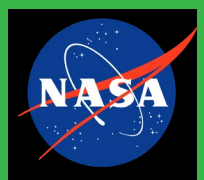


# NOS3 Architecture



- 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

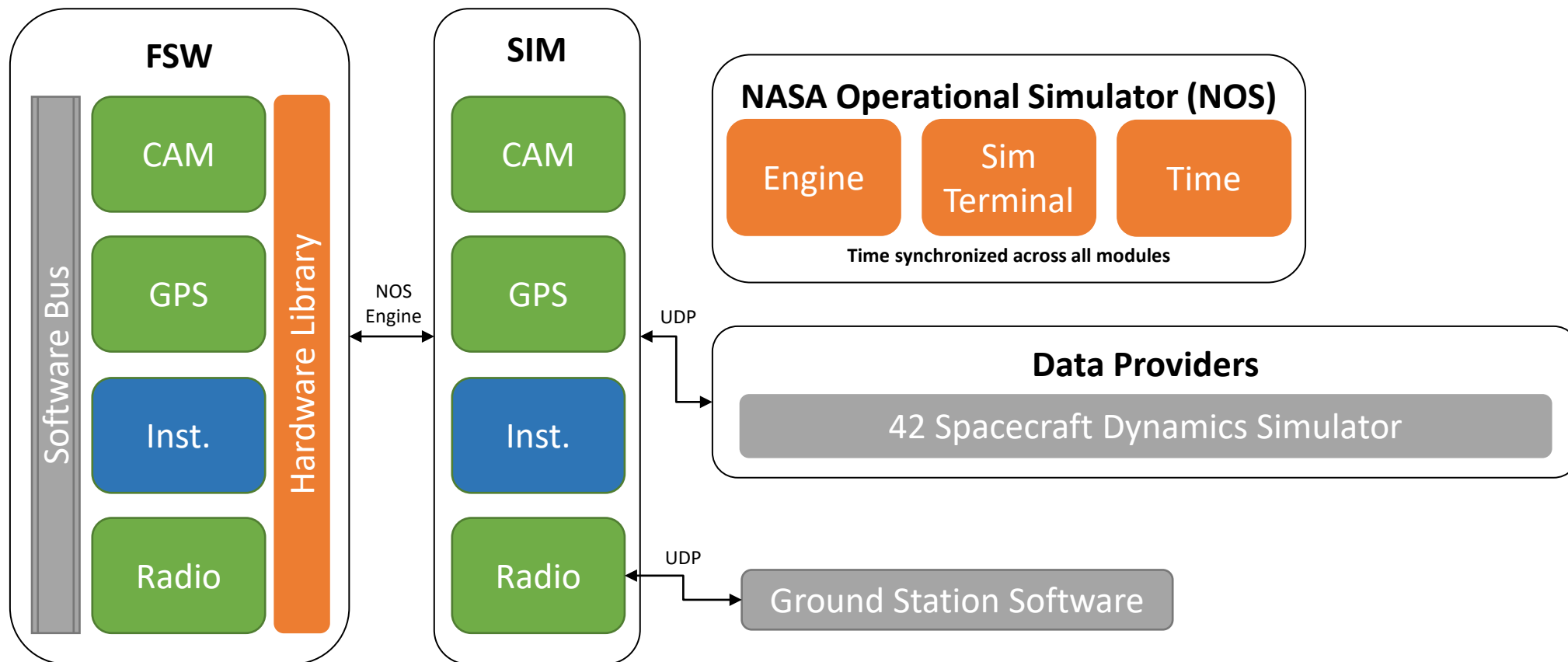


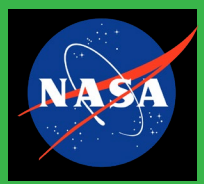


# NOS3 Architecture



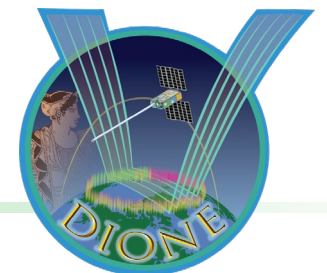
- NOS3 enables cross compiling for your flight and simulation targets
  - HWLIB enables selection of build type via flags at compile time

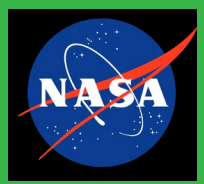




# Design Reference Mission

- 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

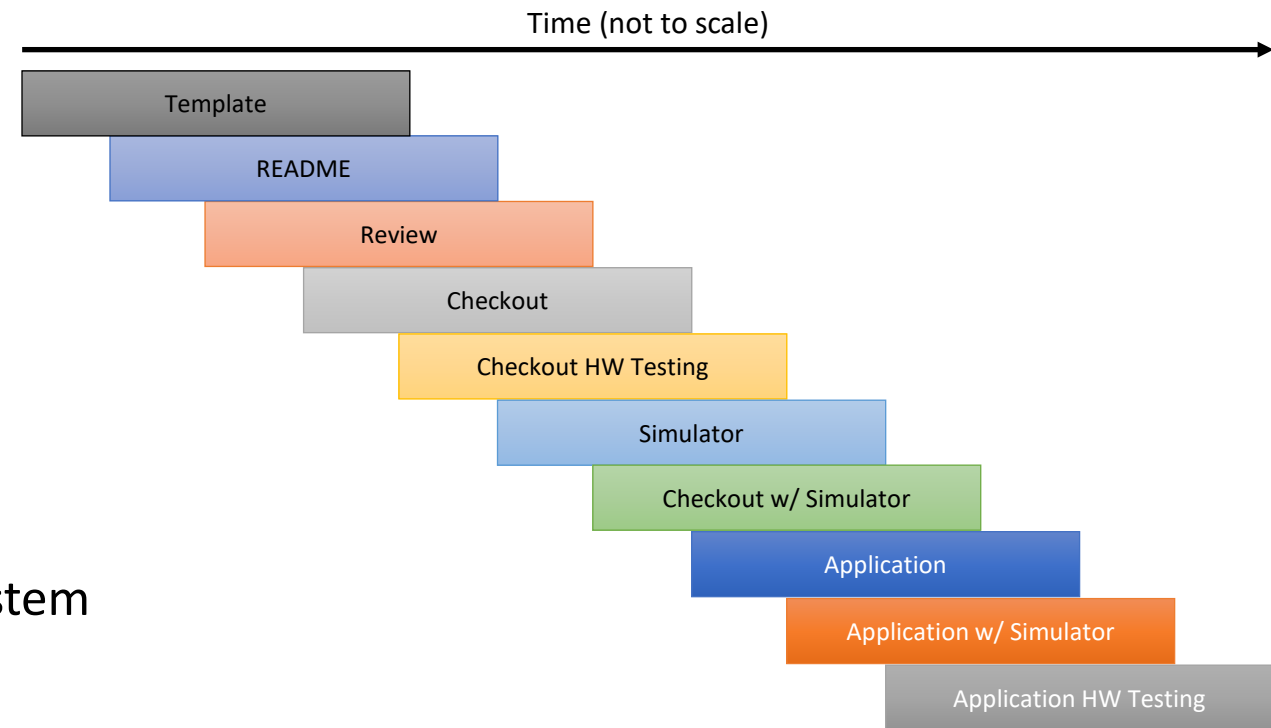


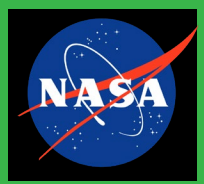


# Completed Efforts – Generic Components



- 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



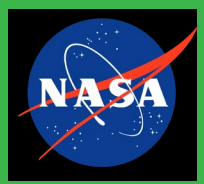


# Completed Efforts – Containerization



- 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

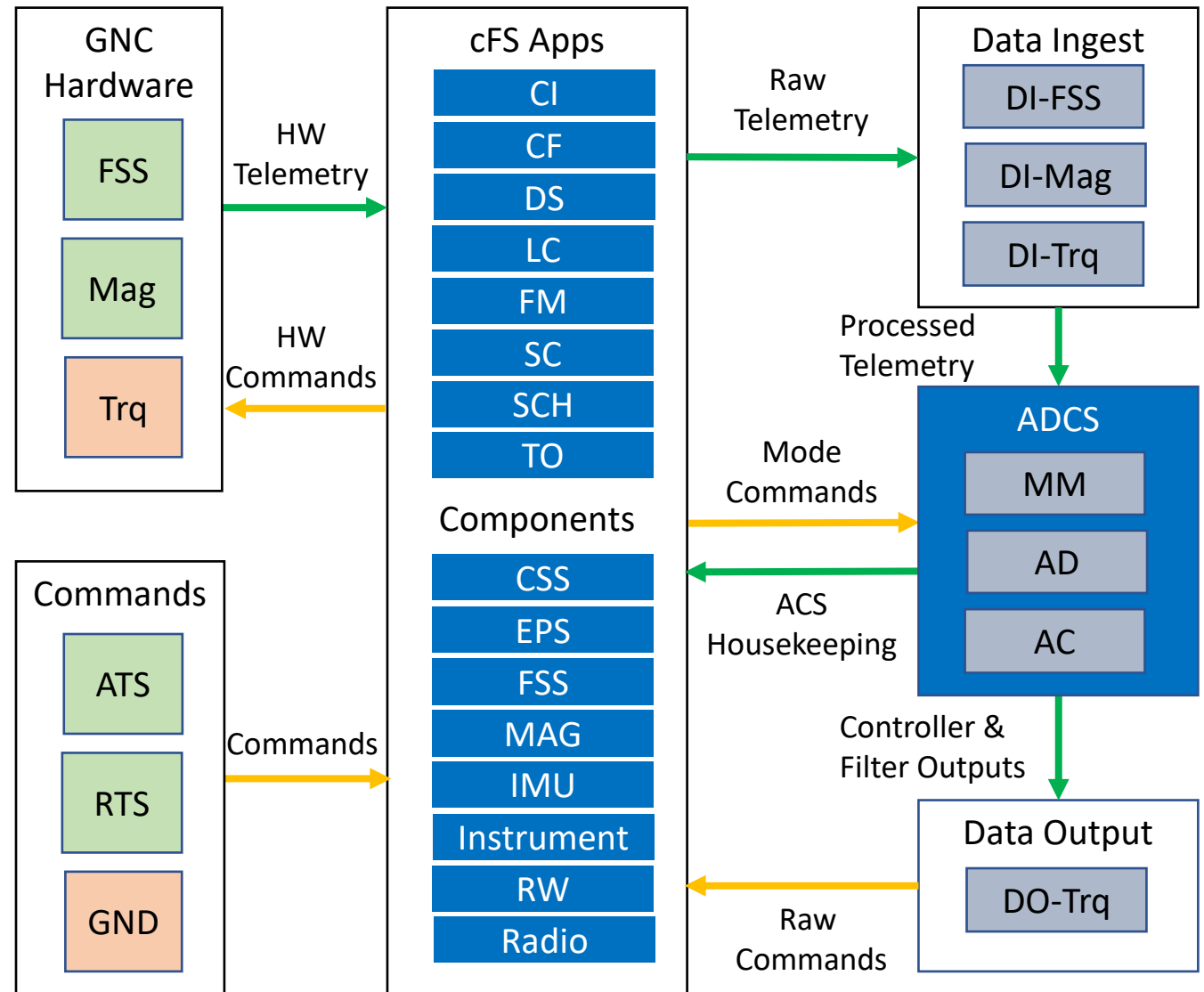
```
jstar@itc: ~  
jstar@itc:~$ docker ps --format "table {{.Image}}\t{{.Names}}\t{{.Command}}"  
IMAGE          NAMES          COMMAND  
ivvitc/nos3-64:dev  nos_time_driver  "/nos3-single-simul..."  
ivvitc/nos3-64:dev  sc_1_radio_sim  "/nos3-single-simul..."  
ivvitc/nos3-64:dev  sc_1_eps_sim    "/nos3-single-simul..."  
ivvitc/nos3-64:dev  sc_1_cam_sim    "/nos3-single-simul..."  
ivvitc/nos3-64:dev  sc_1_truth42sim "/nos3-single-simul..."  
ivvitc/nos3-64:dev  sc_1_nos_engine_server "/usr/bin/nos_engine..."  
ivvitc/nos3-64:dev  sc_1_nos_fsw    "/home/jstar/git/nos..."  
ivvitc/nos3-64:dev  sc_1_cryptolib  "/support/standalone..."  
ivvitc/nos3-64:dev  sc_1_fortytwo   "/home/jstar/.nos3/4..."  
ivvitc/nos3-64:dev  nos_udp_terminal "/nos3-single-simul..."  
ivvitc/nos3-64:dev  nos_terminal    "/nos3-single-simul..."  
ballaerospace/cosmos:4.5.0  cosmos_openc3-operator_1 "/bin/sh -c 'ruby La..."  
jstar@itc:~$
```

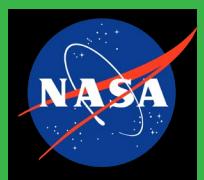


# Completed Efforts – Generic ADCS



- 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

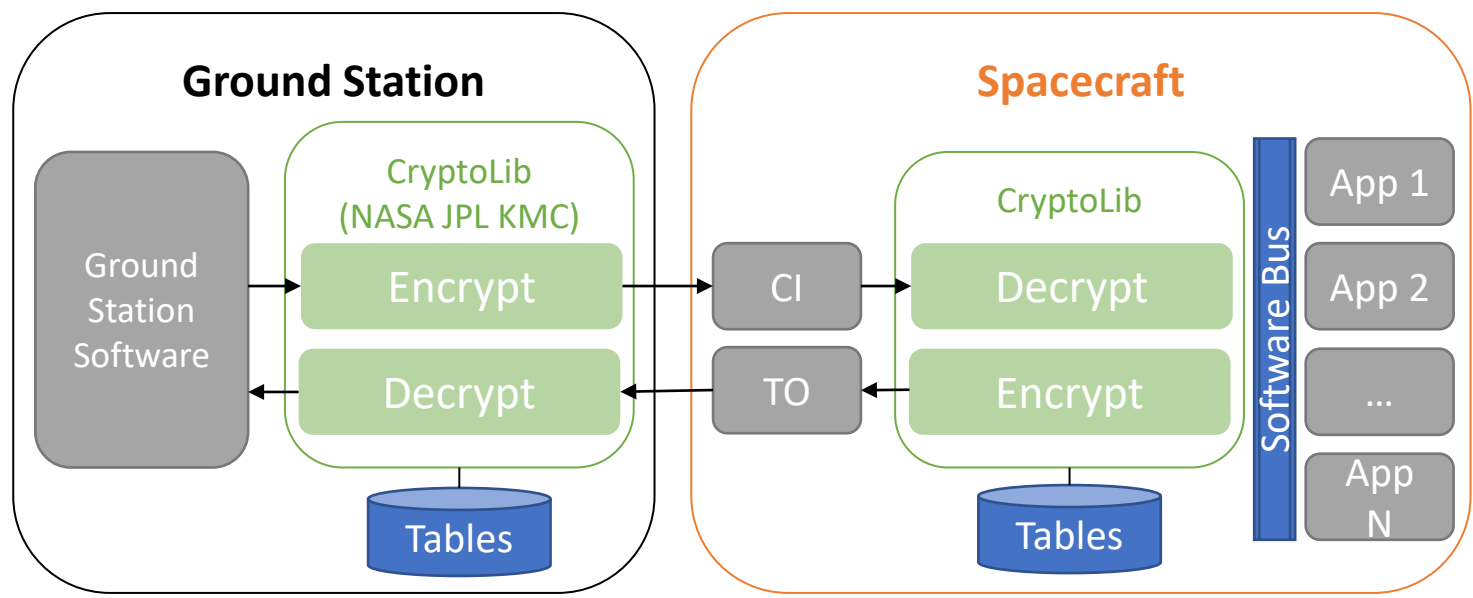


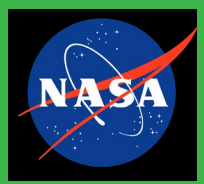


# Completed Efforts – CryptoLib



- 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



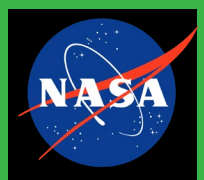


# Completed Efforts – Multiple Spacecraft



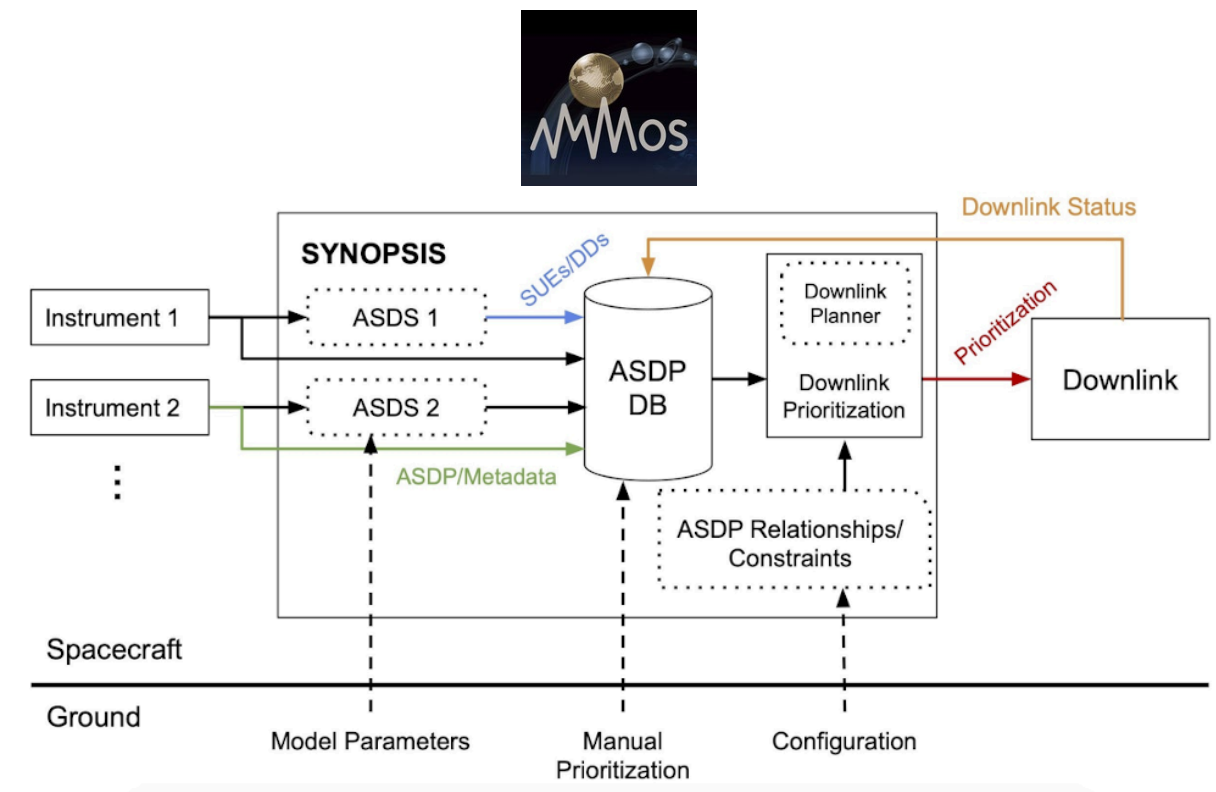
- Developed launcher scripts to change the number of spacecraft desired
- Demonstrated included string of pearls constellation each sending its status to the next in the chain
  - SC0 identifies cool science
  - SC0 sends message to SC1 to get into correct mode to collect data
- Single time driver maintaining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

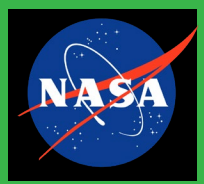
The screenshot displays a simulation environment with several windows. The top window, titled 'NOS Time Driver', shows a log of messages: 'TimeDriver::send\_tick\_to\_nos\_engine: tick ='. The middle window, '42 Cam (on 3d8c5d061193)', shows a 3D view of a satellite in orbit around Earth, with various parameters like 'Sim Time # 182,80', '18 Oct 2025', and 'UTC 201-08:33:02.80'. The bottom window, 'sc\_1 - NOS3 Flight Software', shows a detailed log of system startup and initialization, including messages like 'ES Startup: Loading file: /cf/generic\_rw.so, APP: RW' and 'EV20EVS Port1 42/1/NAV 1: NAV Initialized. Version 1.0.0.0'. A right-hand panel shows camera settings for 'ARUDCAM'.



# 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”

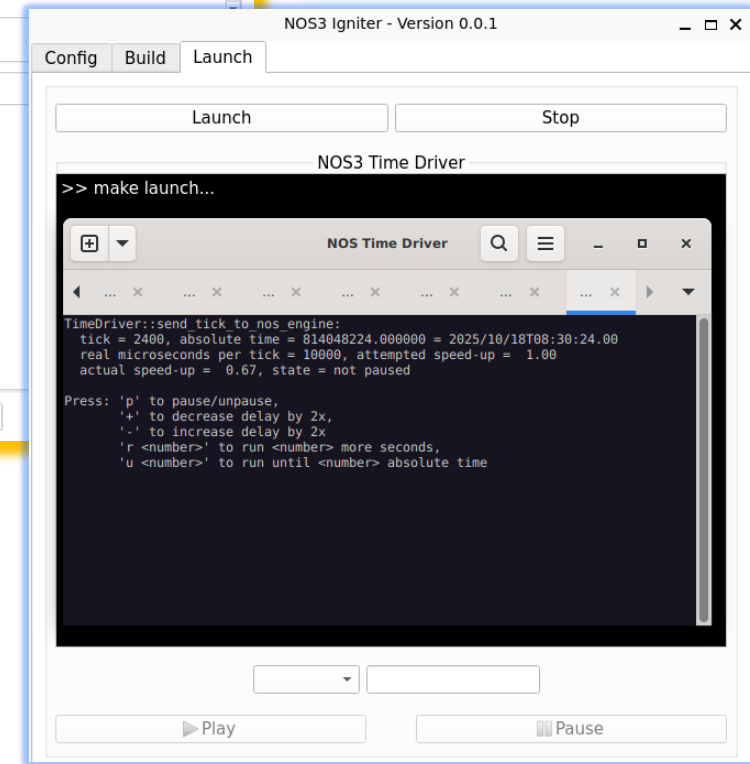
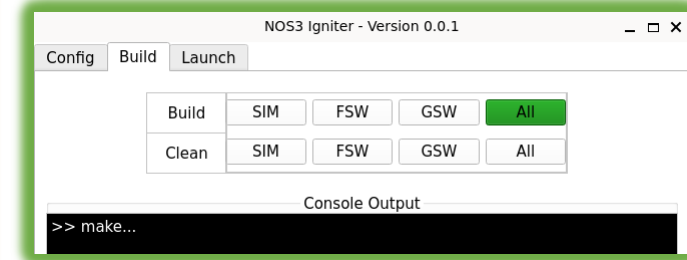
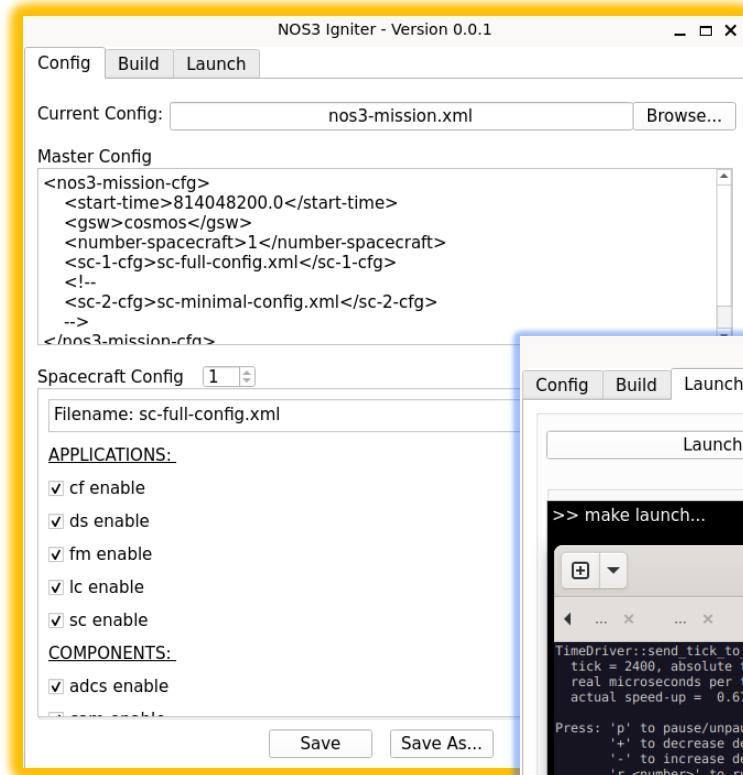




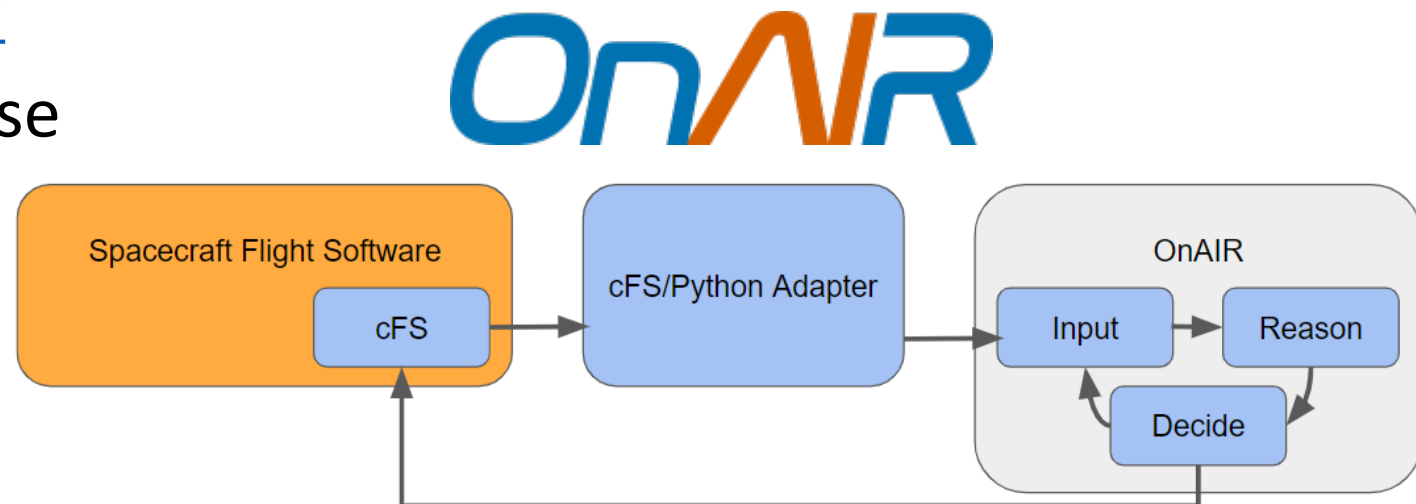
# Current Efforts – Igniter

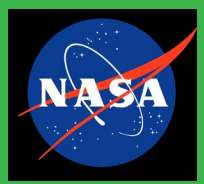


- GUI tool to enable easy reconfiguration
  - Python based
- Mission configuration
  - Start time
  - Ground software to use
  - Number of spacecraft and configurations
- Spacecraft configuration
  - cFS application enable / disable
  - Component enable / disable
  - 42 GUI and initial tipoff rates
-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



- 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



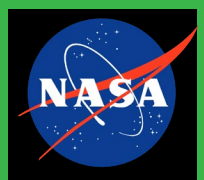


# Current Efforts – F'

- 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

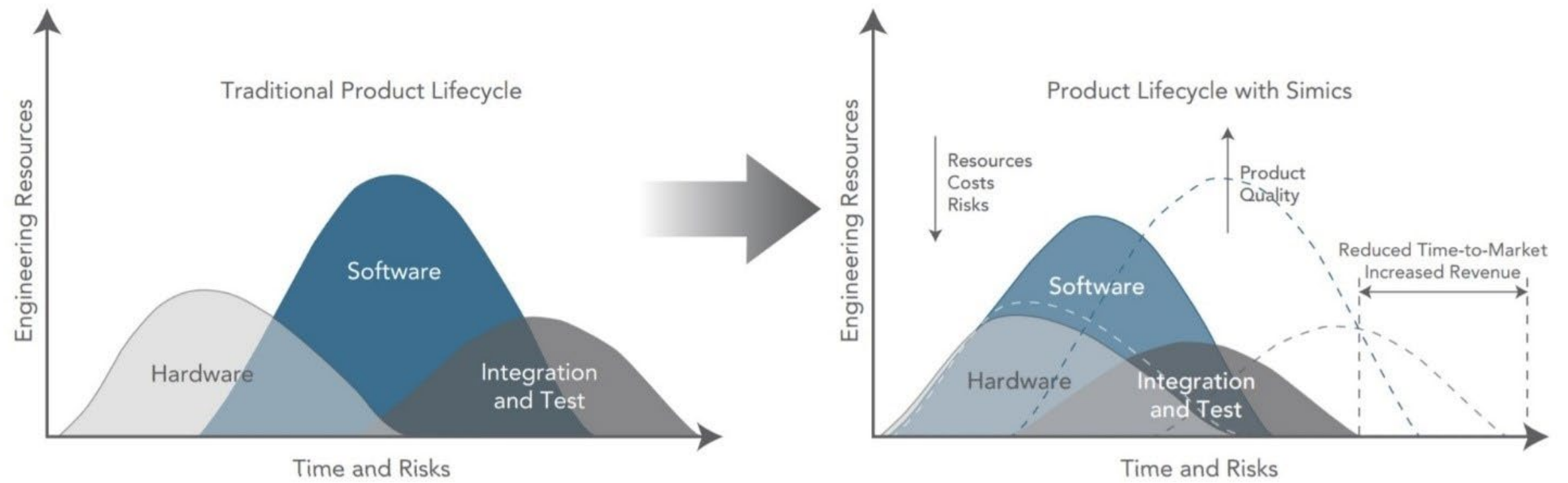
## Researchers

Integrate and demonstrate new technologies

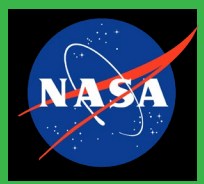
New uses and users reported regularly!

# What's the driver for project utilization?

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



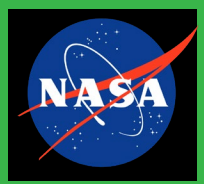
[Shifting Left—Building Systems & Software Before Hardware Lands \(intel.com\)](https://www.intel.com)



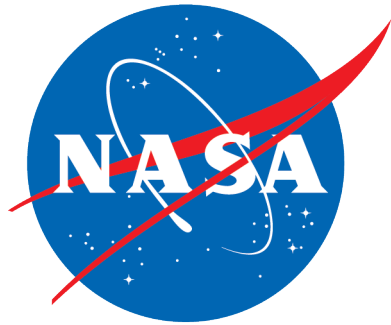
# Path Forward



- 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



Jet Propulsion Laboratory  
California Institute of Technology



Antarctica



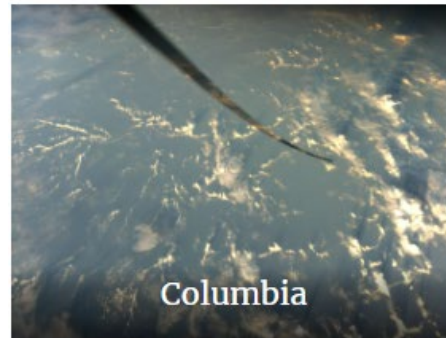
Mongolia



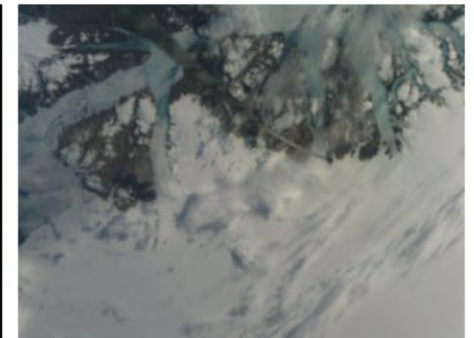
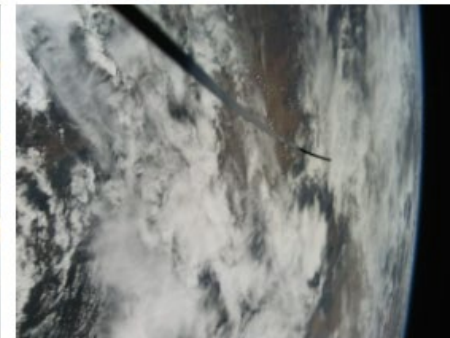
Mongolia



Clouds Over Antarctica



Columbia



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