Independent Test Capability

"Simsmithing"

Converting Words Into Simulations

ITC Team Members

Justin Morris
Steven Seeger
Brandon Bailey
Scott Zemerick
Mark Pitts
Jeff Joltes
Justin McCarty
Dan Nawrocki
Gary Carvell
Peter Medley

Contact Us: ivv-itc@lists.nasa.gov
Internal Website: http://itc.ivv.nasa.gov
External Website: http://www.nasa.gov/centers/ivv/JSTAR/ITC.html

Agenda

- Introduction
  - What is “Simsmithing”?
- “Simsmithing” Process
  - 4 Core Steps
  - Example – JWST Spacecraft
- IV&V Benefits and Results

Introduction & Background

- What is "Simsmithing"?
  - "*The process of creating software-only simulators from mission artifacts.*"
    - Artifacts include requirements, Interface Control Documents (ICDs), Hardware Manuals, etc.

- Goals of IV&V ITC "Simsmithing"
  - Build complete software-only system simulator
  - Reduce costs associated with hardware duplication
  - Build reusable simulation components that can be leveraged across missions
  - Provide complete simulation environment for Flight Software (FSW) verification
  - Make the FSW believe it is running in its normal environment

Introduction & Background

- Benefits
  - IV&V ITC Simulators Fill Gaps
    - Hardware systems are too expensive to duplicate
    - IV&V typically doesn't receive simulators
  - Find more issues
    - During simulator development
    - During IV&V use
  - Increase system knowledge for IV&V engineers and analysts

"Simsmithing" Process

Step 1 - Words

Step 2 - Digest

Step 3 - Build

Step 4 - Certification

Step 1: Words

Acquire All Relevant Artifacts

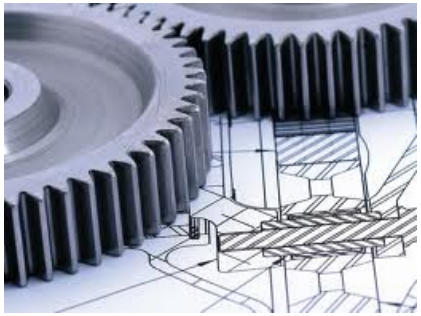
- System/Subsystem Requirements
- Interface Control Documents (ICDs)
- Hardware Manuals
  - Example: JWST SSR manual described how the SSR operates – this was the source of requirements for the simulator
- Test scripts, procedures, and results
- Any document that describes the functionality of the hardware

Step 2: Digest

- Artifacts become simulator requirements and specifications
- Identify components needed to build a fully simulated system
  - What system does the FSW run on? Are there any existing simulators?
  - What peripherals does the FSW interact with?
  - Do external system simulators already exist? Can they be reused?
  - Are there any unused components?

Step 3: Build

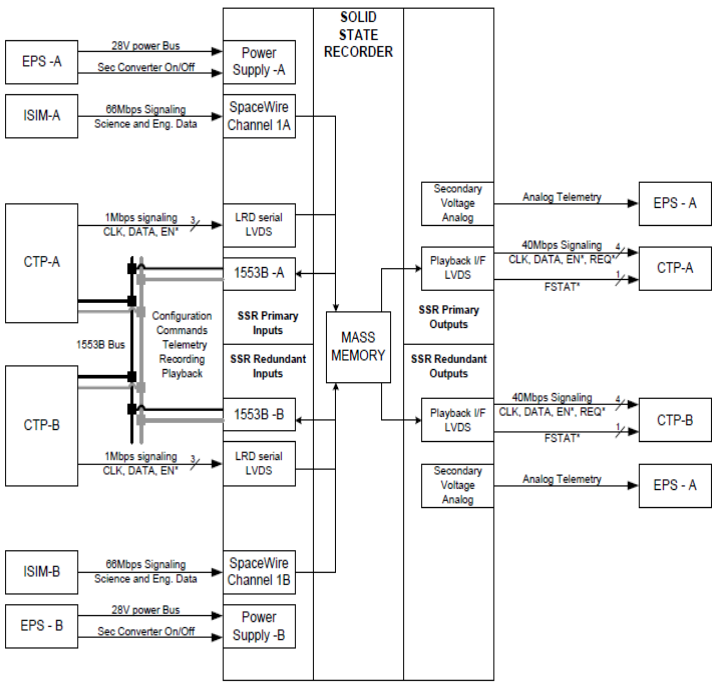
- "Divide and Conquer"
  - Each developer takes ownership of a component
  - Component boundaries typically match system boundaries
    - 1553 Bus, SpaceWire, cPCI, etc
  - Integration of system is team effort

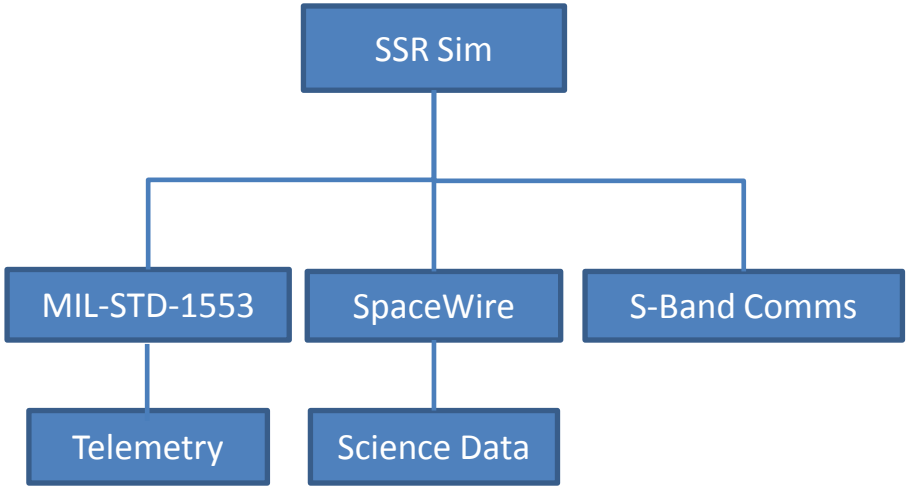


Step 3: Build

- Fidelity
  - How much is needed?
    - Dependent on customer requirements
    - Too much and the simulator executes too slowly to be useful or takes too long to develop
    - Too little and the simulator is ineffective
    - Simulator does not have to simulate every hardware task
    - Must balance testing goals
    - Example: GPM simulator supports two levels of fidelity:
      - Low, primarily for C&DH
      - High, primarily for GN&C

SSR Simulator Example





Task 1: SSRSim responds to FSW MIL-STD-1553 commands and returns responses

Task 2: SSRSim provides telemetry with error injection for IV&V

Task 3: SSRSim provides playback and record functionality

Step 3: Build

- Encountering FSW Bugs
  - Sometimes FSW makes incorrect use of hardware. In order to continue work and make use of the simulation:
    - Patch the binary
      - Causes maintenance issues
      - No longer testing "as-is" software
    - Modify the hardware model
      - Causes maintenance issues
      - Scripts that deal with hardware loading must now be made specific to software version
    - Augment hardware model with an additional layer
      - No maintenance issues
      - Can be added to specific software version script
      - Does not always apply

Step 4: Certification

"Gold Rule"

Execute all original test procedures and obtain the same results as the physical system

Only exception: settling time may need to be increased if the simulator doesn't execute as fast as hardware

Case Study : JIST

- JWST IV&V and ITC team jointly identified need for simulator for IV&V use
  - JWST Project expressed interest as well

- Feb. 2012: ITC team started work on JWST IV&V Simulation and Test (JIST)

- Aug. 2012: ITC team delivered JIST 1.0
  - 6 month turnaround!

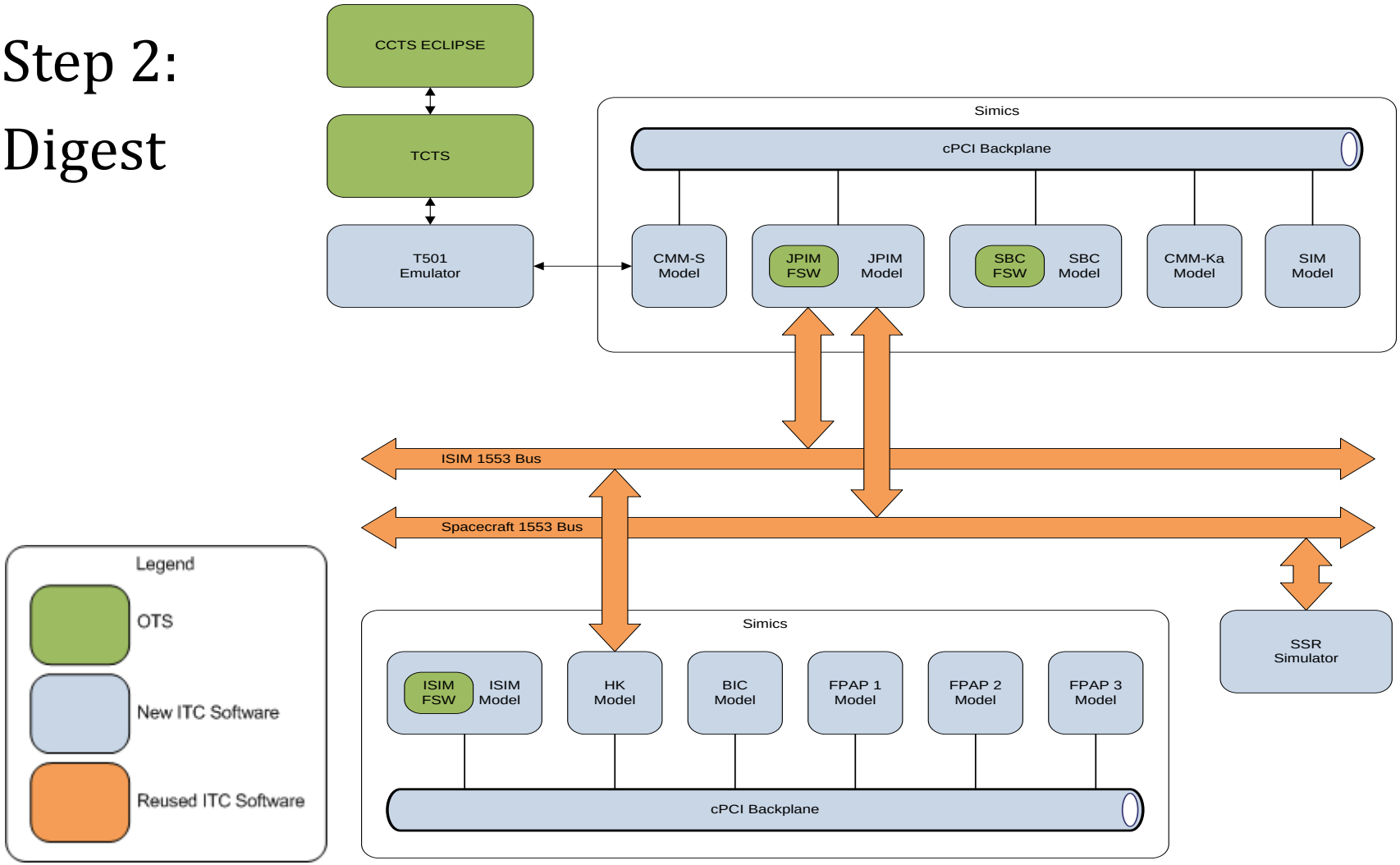
Case Study : JIST

- Step 1: Words
  - Challenging task
    - JWST is almost 10 years old
    - History of design changes
  - Received artifacts:
    - ICDs
    - Hardware Manuals
    - User guides
    - FSW source & binaries
    - EGSE source & binaries
    - Build environments

Case Study : JIST

Step 2:

Digest



Case Study : JIST

- Step 3: Build
  - Divide and Conquer
    - EGSE link, JPIM modeling
    - SSR Simulator
    - SBC modeling
  - Leverage existing simulators where applicable
    - SSR Simulator that was NOT software-only
    - COTS RAD750EN model
  - Utilize ITC simulation architecture
    - Well-tested communications backbone
    - Provides the ability to inject errors for IV&V of FSW
    - Reusable Models

Case Study : JIST

- Step 4: Certification
  - Executed acceptance test procedures from Northrop Grumman
  - **All in-scope tests passed**
    - Only modification was for differences in timing because the simulator executed slower on current development machines
    - Real-time execution for deployed environments

Results & Benefits

- Issues Identified During Simsmithing Process
  - FSW Issues
    - Software incorrectly resets SUMMIT 1553 illegalization registers
    - Software writes 0x1 to the PCI byte swap register; hardware documentation recommends 0xFFFFFFFF instead
    - Software writes invalid bit pattern to FPAP SpaceWire Configuration Register
  - Documentation/Maintenance Issues
    - Ambiguous use of the word "invalid" to describe hardware behavior when particular values are written to a register
    - Software writes a 1 to a bank select register unconditionally in initialization code when the ICD said that it should be writing a 0 and only use 1 if a problem is detected

Results & Benefits

- IV&V Benefits
  - Increased System Understanding
    - "Simsmithing" requires deep understanding of the actual hardware and how it interfaces with the rest of the system as a whole
  - More Issues Found
    - Issues found during "Simsmithing" process
    - Issues found during IV&V's use of simulators
  - Improved IV&V Practices
    - Increased confidence
    - Verify expected software behaviors
    - Expand/Enhance developer test coverage
    - Provide means to test edge-cases and limits of software

Results & Benefits

- Development Project Benefits
  - Development and test activities no longer restricted to limited hardware resources
  - Allows for hardware fault injection without damaging hardware
  - Lab space, storage, and utilities reduced
  - Easily deployed for training
  - Simulate long term operations
  - Increases number of users without additional hardware acquisition

QUESTIONS?