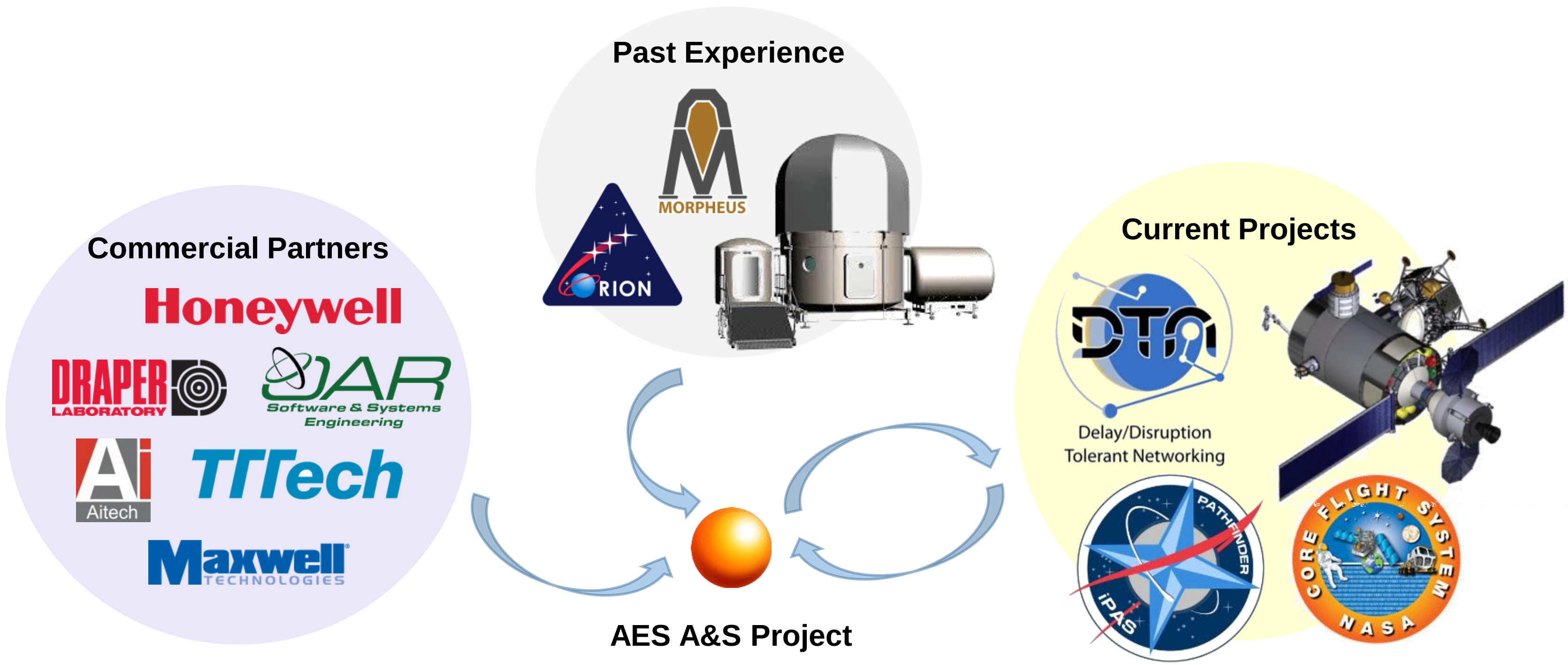
Overview of TTE/CFS Integration

CCSDS SOIS SUBNET/APP WG Meeting
SwRI, San Antonio, TX
8 – 12 May 2017

Andrew Loveless (NASA/JSC)
andrew.loveless@nasa.gov

- The **Avionics and Software (A&S) Project** is developing a mission-agnostic architecture applicable to spacecraft or habitats.
  - Chartered by NASA's Advanced Exploration Systems (AES) Program.
  - Includes participation by most NASA centers and several commercial partners.
  - Mature promising architectures for use in other NASA projects.
  - **Approach:** Minimize development time/cost by utilizing COTS technologies.



IPAS: Integrated Testbed

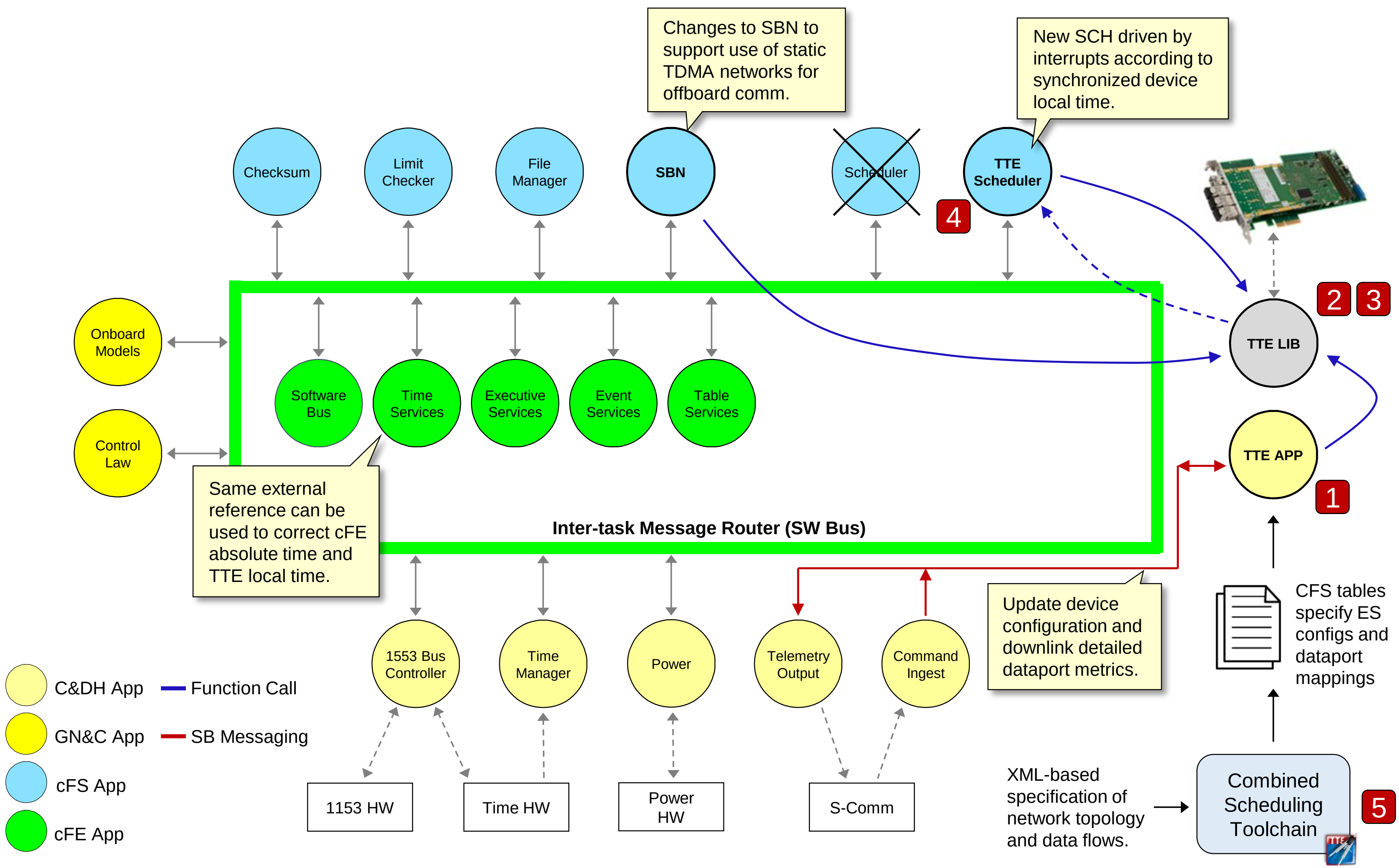
IPAS testbed located at NASA/JSC in B29 (as of May 2017)

Current Work

- Working to include the ability to use Time-Triggered Ethernet (TTE) communication in the NASA Core Flight System (cFS).
- Major Design Objectives:
  - Provides a common set of APIs/Libraries enabling any cFS app to use TTE.
  - Support range of traffic types (e.g. TT, RC, BE), protocol layers (e.g. UDP, IP), and port types (e.g. COM, SAP).
  - Support range of commercial OSs (e.g. Linux, VxWorks 6.X) and targets (e.g. Freescale MPC8548E, Xilinx Zynq 7020).
- Core Components:
  - Application   Device/port configuration, IRQ handling, commanding, telemetry
  - Libraries   APIs for TX/RX, DMA (ESDMA and CPU DMA), IRQ configuration.
  - Drivers   Precompiled drivers to support common target platforms.
  - Scheduler   cFS FSW scheduler driven according to TTE periodic time base.
  - Tooling   Generate cFS port and ES config tables from TTE build products.

Intend to make open source and publically available.

CFS Integration



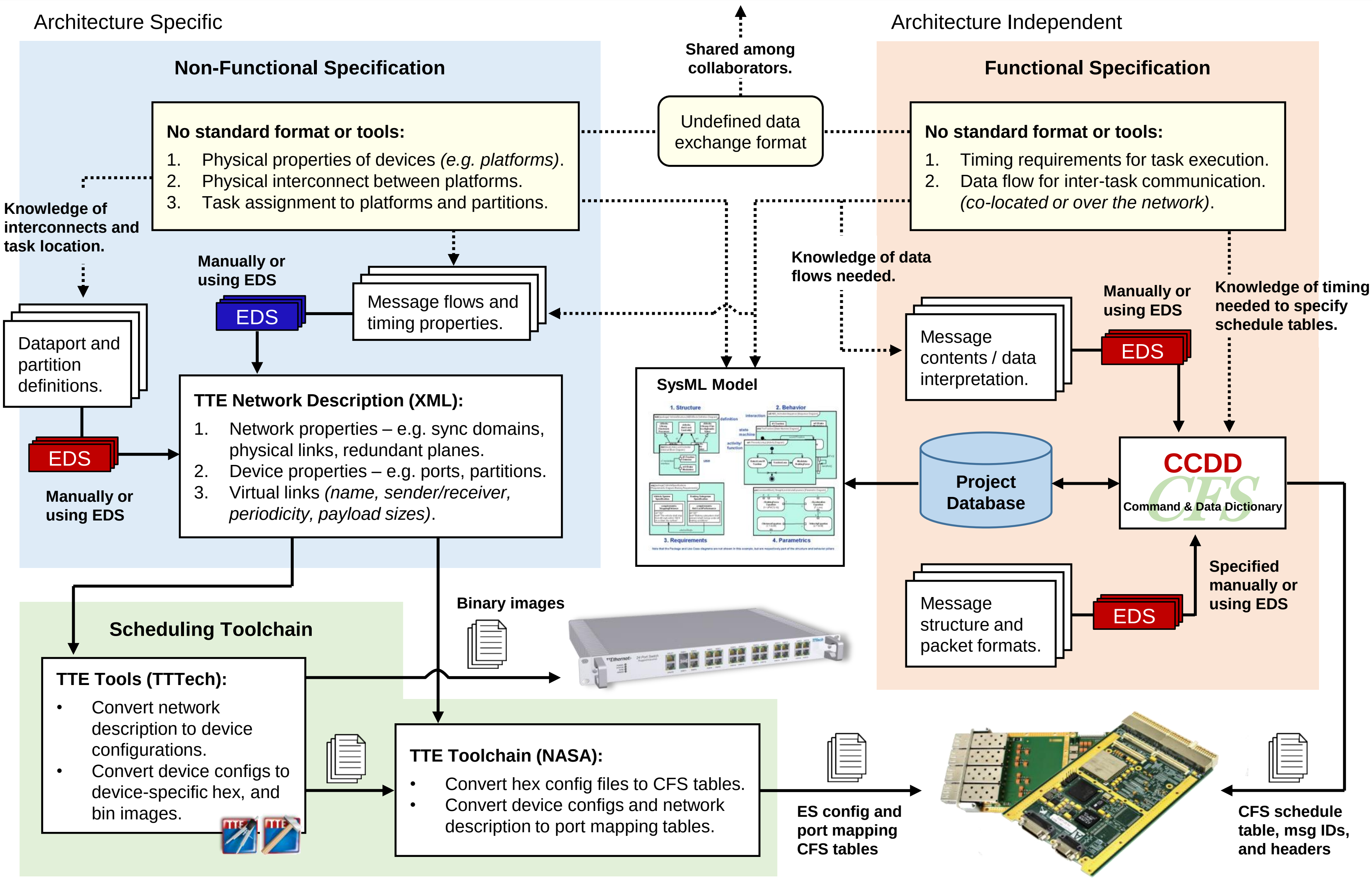
CFS Integration

- Software Bus Network (SBN) application used to extend cFE SB publish/subscribe messaging across partitions/network.
- Revisiting SOIS Packet Service discussion on standardizing API.
  - The more I think about it, the less sense it makes to me.
  - More focus at NASA/JSC on using SBN for cFS apps talking offboard. Apps don't directly access the network interfaces.
    - Different SBN flavors provide internal portability for cFS apps to different subnetworks.
  - Behavior of SBN must depend on subnet used (e.g. for fault management).
    - If using a common API, the FSW cannot make assumptions about subnet (is that feasible?)
- Changes to SBN needed to support scheduled networks with predetermined message flows (similar to ARINC 653).
  - Elimination of announcements/heartbeating for connections between peers.
  - Mapping of message identifiers to dataports instead of specific hosts.
    - Scheduling determines where messages are delivered, SBN is not aware of scheduling.
  - Remove distribution of message subscription tables between peers.
    - Scheduling toolchain + project database (e.g. CCDD) can generate all configuration tables.
- Difficult (Impossible?) to abstract over all subnets from one SBN.

Interoperability Needs

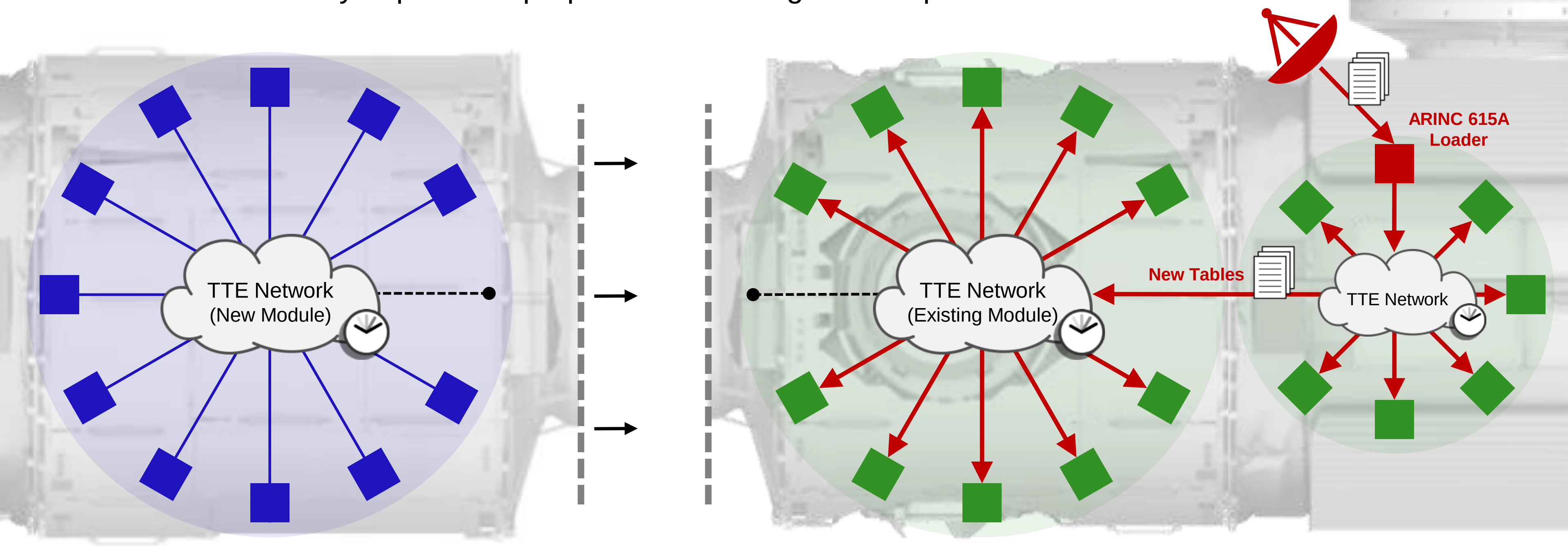
- The development of a Deep Space Habitat would likely be distributed over many parties – commercial/international partners.
- Need for interoperability among tools for the description of physical topologies, traffic flows, network scheduling, message definitions, and software components.
  - E.g. Different FSW architectures must coordinate use of subnet resources.
- Standard data exchange format needed to express relationship between functional/non-functional and software/network pieces.
  - Currently requires manual integration of products produced by separate tools: E.g. SEDS, CDD for SW/network interfaces, vendor tools for network planning.
  - Complex interactions are hard to understand and maintain at the XML level.
  - E.g. Timing requirements for network scheduling and task execution.
- Also must facilitate sharing of information between concurrent component-level and system-level development efforts.
  - E.g. Message packet structure definition and bandwidth allocation.

Toolchain Shortcomings



Incremental Build-Up Approach

- Modules may be developed years apart; therefore have no common scheduling (unlike Orion Crew Module → Service Module approach).
- Goal is to support composability of the network architecture and software.
- Requires additional features in the network scheduling toolchain.
  - The global scheduling approach may not be feasible if there is a requirement for the addition of new scheduled messages, while other TT messages are already defined.
    - *Incremental Scheduling* – determine free transmission gap for each hop, and iteratively modify transmission windows and times in flow as necessary.
    - Still preserves the real-time properties of the existing traffic flows.
  - Also need ability to preserve properties of existing flows to prevent modification.



Timeline

- TTE cFS application targeting (rough) completion by June 2017.
  - Expected public availability around December 2017.
- Christian Fidi working TTE EDS and scheduling to support incremental buildup approach.
- AES A&S integrated test of TTE/cFS in prototype habitat at NASA/JSC in September 2017.