National Aeronautics and
Space Administration

**Jet Propulsion Laboratory**
California Institute of Technology
Pasadena, California

# Distributed Ground Systems for a Large Multi-Instrument Space Mission: Lessons Learned from the Cassini Huygens Program

Barbara Larsen
Leo Cheng

# OUTLINE

- BACKGROUND
  - Paradigm Shift to Distributed, Networked Operations
  - The Cassini Huygens Ground System Design
- DECENTRALIZATION: LESSONS LEARNED FROM THE SHIFT TO MULTIPLE, OVERLAPPING SYSTEMS
  - Logical Architecture for Different Physical Implementations
  - Platform Decisions in a World of Heterogeneous Hardware
  - Differing Institutional Restrictions, Drivers, and Funding
- FUNCTIONAL REALLOCATION: LESSONS LEARNED FROM BLURRED RESPONSIBILITIES BETWEEN SCIENTISTS AND ENGINEERS
  - Electronic, Up-to-date, Widely Accessible Plan
  - Reallocation of Functionality to Tools According to New Charters
  - Synchronization in Change Management
  - Support for Non-Specialist Users
  - Arbitration of Contention for Development Resources
  - Late Interface Changes due to Asynchronous Development

# OVERVIEW OF THE CASSINI GROUND SYSTEM

Drivers:

- Spatial Decentralization
  - Telecommunications / networking
  - Distributed Computing
- Functional Reallocation
  - Dispersion of responsibilities
    - Sequence Construction
    - Monitoring health and safety
    - Spacecraft pointing
    - Flight rule checking
  - Few specialist tool operators

**ground system diagram—**

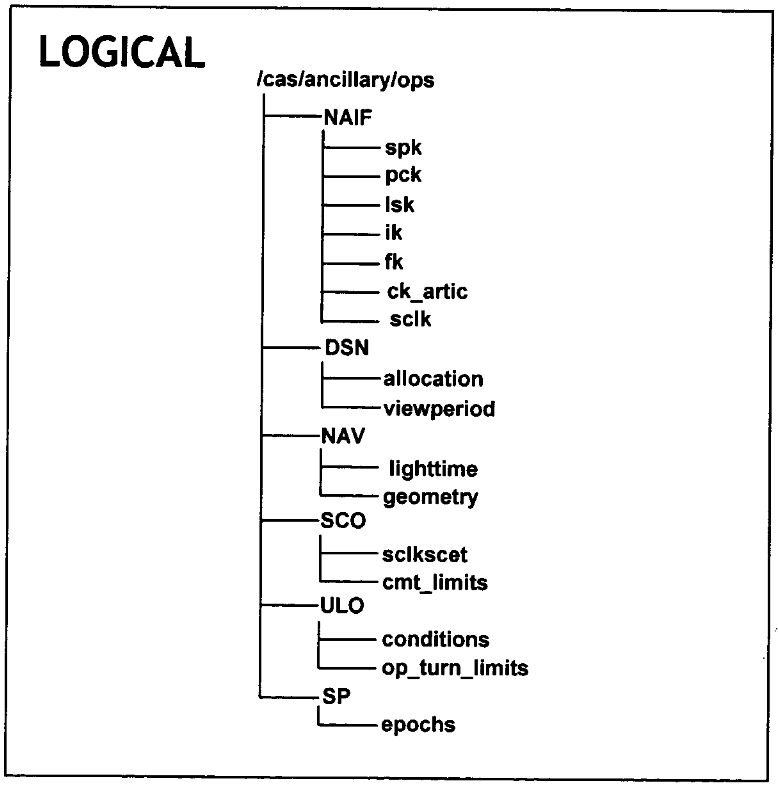
**to be supplied**

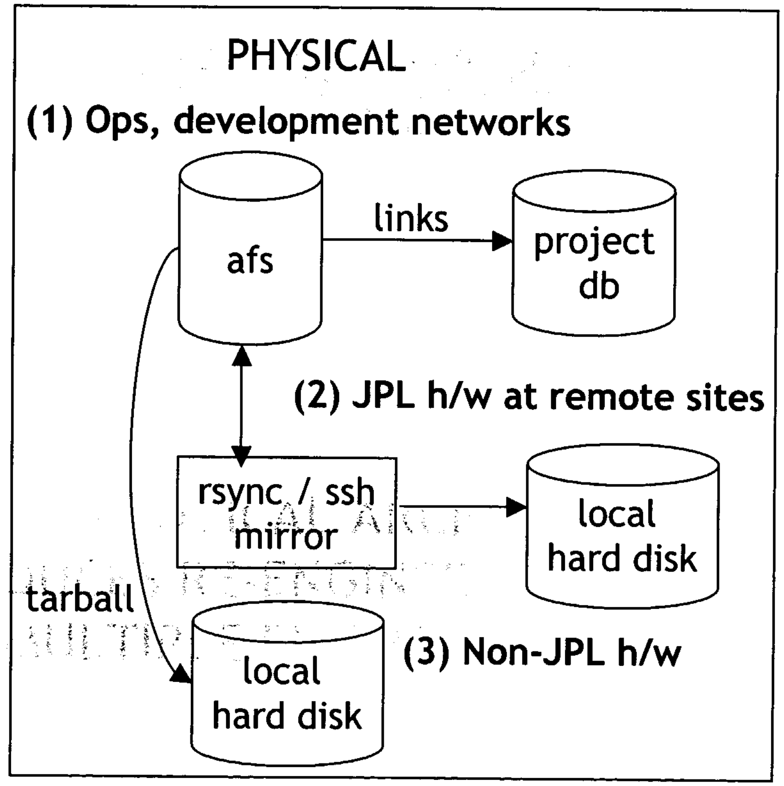
National Aeronautics and
Space Administration

**Jet Propulsion Laboratory**
California Institute of Technology
Pasadena, California

# LESSON 1: A LOGICAL ARCHITECTURE REDUCES RE-ENGINEERING FOR MULTIPLE ENVIRONMENTS

**Example: Structure the ancillary data files needed by uplink software for accessibility by all users**



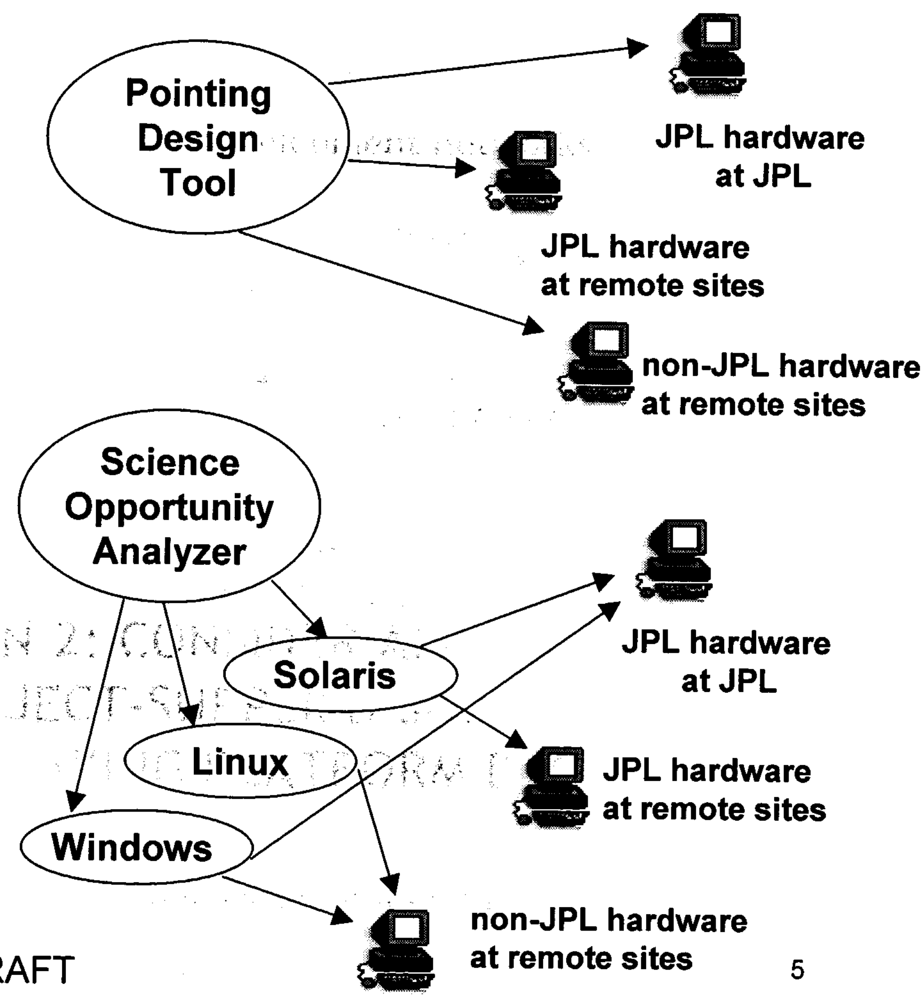
National Aeronautics and Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

# LESSON 2: CONSIDER ALL USERS OF PROJECT-SUPPLIED SOFTWARE WHEN MAKING PLATFORM DECISIONS

- What's Expensive?
  - NASA / JPL view
  - Contracted Scientist View
  - University Co-Investigator View
- What's Preferred?
  - Engineer vs. scientist
  - Europe vs. U.S.
  - University vs. corporate
- How Much Flexibility Is Possible?
  - Cost of Development
  - Maintenance and Testing

**Examples of software deployment to multiple environments:**



# LESSON 3: SYSTEMS AT DIFFERENT INSTITUTIONS WILL EVOLVE AT DIFFERENT RATES

- Aim for Design Robust With Respect To
  - Operating System
  - Security
  - New Hardware

- Communicate Infrastructure Intentions
  - O/S Upgrades and Patches
  - Firewall Changes
  - Security Shut-downs

National Aeronautics and
Space Administration

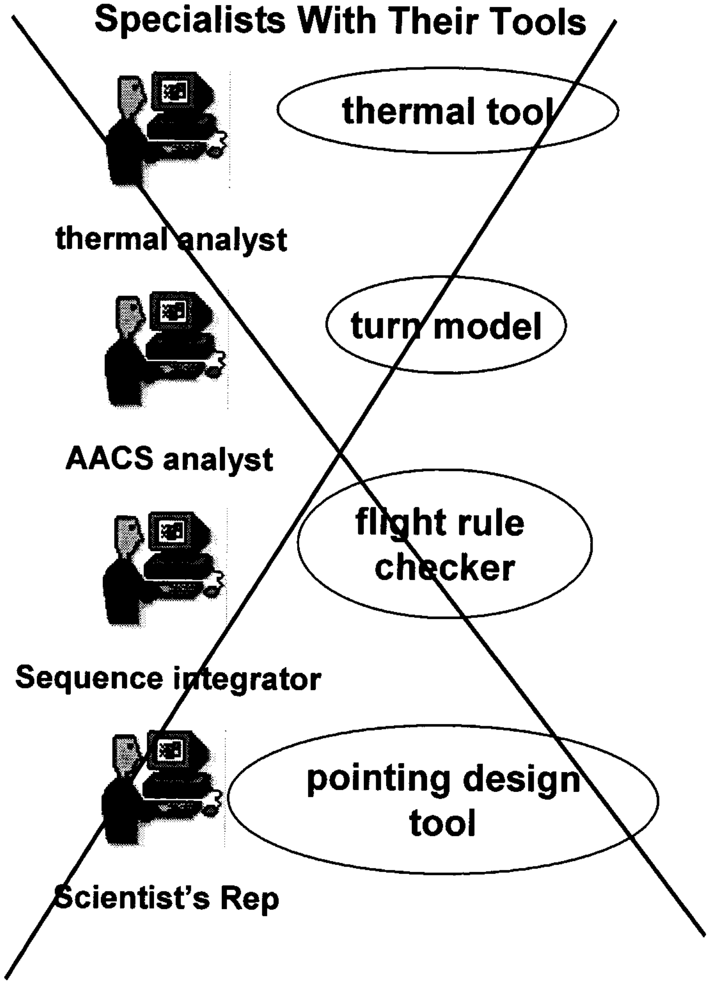
**Jet Propulsion Laboratory**
California Institute of Technology
Pasadena, California

# LESSON 4: CONNECTIVITY ALONE IS NOT ENOUGH

to be supplied

# LESSON 5: SHIFTING RESPONSIBILITIES REQUIRE REALLOCATION OF FUNCTIONALITY TO TOOL AT A HIGH LEVEL

**Old Paradigm:**
**Specialists With Their Tools**

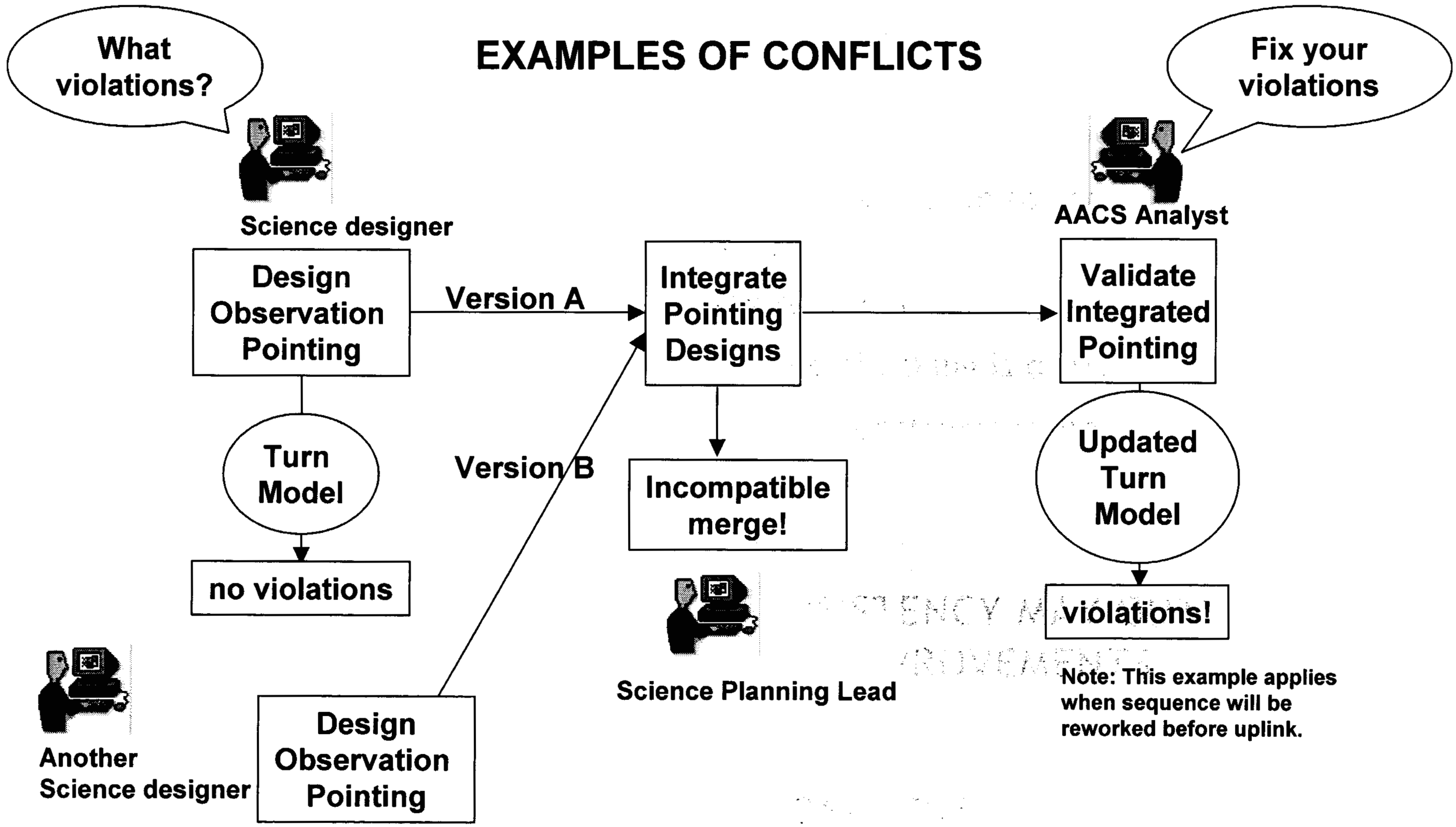



Cassini Examples of Changes in Functional Allocation:

- Turn modeling matters to scientists
- Everyone has to monitor selected thermal heating
- OpNav requires pointing design
- Flight rule checking is distributed
- Each team constructs syntactically correct sequence
- Real-time commands produced from multiple locations

# LESSON 6: CONSISTENCY MAY OUTWEIGH SWIFT IMPROVEMENTS

## EXAMPLES OF CONFLICTS




Note: This example applies when sequence will be reworked before uplink.

# LESSON 6: CONSISTENCY MAY OUTWEIGH SWIFT IMPROVEMENTS

## RESOLUTION

- Uplink software version is linked to process by sequence number
- Multiple sequences are in simultaneous development
- Multiple versions of software are, therefore, in use simultaneously
- Software deliveries must be carefully coordinated with sequence development
- Map from sequence number to software version is table-driven
- User selects version by specifying sequence number to program wrapper, which reads table for correct executable

**example of mapping to be supplied**

# LESSON 7: USERS ARE MORE DEMANDING THAN SPECIALIST OPERATORS

- Escalating Ease of Use Requirements
  - Installation
  - Invocation
  - Configuration
  - "Friendly" user interface
  - Problem reporting and notification
- Added ops / development roles
  - Users Guides rather than operator manuals
  - Trainers and training materials
  - Tech support

Example: Evolution of Configuration of Uplink Tools

- Stage 1:
  - Requisite files obtained by user and recorded in configuration file
- Stage 2:
  - Model configuration file provided by science planners
- Stage 3:
  - Data files in model configuration structured for software availability
- Stage 4:
  - Software assists in finding configuration file and loading referenced data files

# LESSON 8: EXPECT CONTENTION IN TOOLS THAT MEET BOTH SCIENCE AND ENGINEERING NEEDS

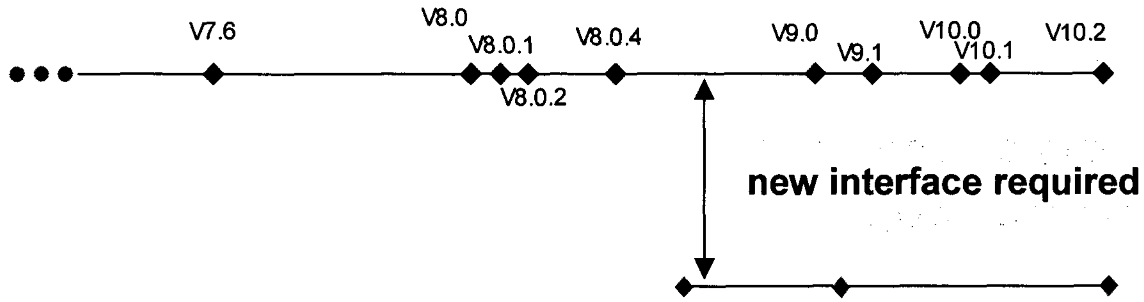
- Requirements vary according to perspective.
  - Fidelity vs. performance (e.g. turn model)
  - Ease of use vs. flexibility (science user vs. engineering operator)

- Allocation of development resources may require arbitration
  - Enhancements for one side against added capabilities for the other
  - Harder to evaluate schedule urgency

# LESSON 9: WITH ASYNCHRONOUS DEVELOPMENT, LATE INTERFACE CHANGES ARE TO BE EXPECTED

**Cassini Example:**

**Mission Sequence System / Pointing Design Tool Delivery Timeline**




**Instrument Ground System Delivery Timeline**

**Impact Mitigation Techniques Used by Cassini**

**•User definable output formats**

**•Flexible interface definitions (e.g. xml)**

**•Glueware and wrappers**

**(But some recoding also required)**

# CONCLUSION

- The Cassini Huygens ground system design is complicated:
  - The spacecraft itself is complex
  - Operations is distributed among JPL and home institutions of scientists
  - The science community plays a greater role in operations
- Environment is now set of integrated systems, not simple data exchange
  - Ground system design requires flexible architecture
  - Platform must be widely accepted
  - Robustness must increase with loss of isolation
  - Coordination is critical
- Responsibilities shift between scientist and engineer
  - Communication is both harder and more important
  - Allocation of requirements to tools must be reconsidered
  - User community has different skill set than old specialist operators