



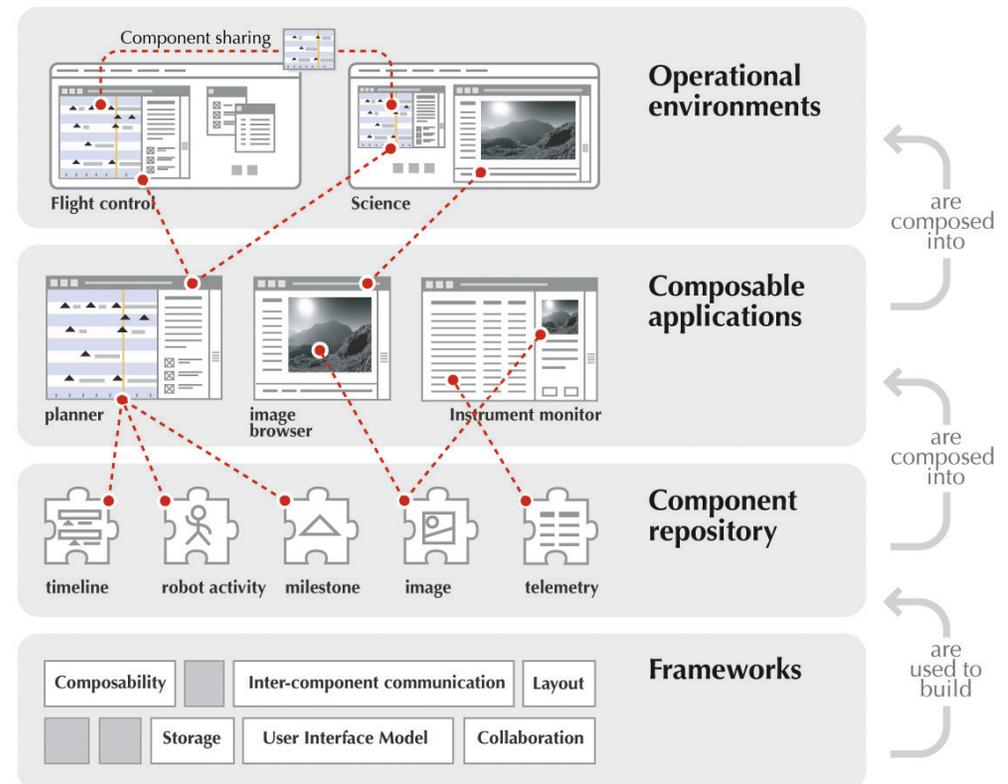
Mission Control Technologies: A New Way of Designing and Evolving Mission Systems

Jay Trimble, Harry Saddler, Joan Walton | Spaceops
June, 2006



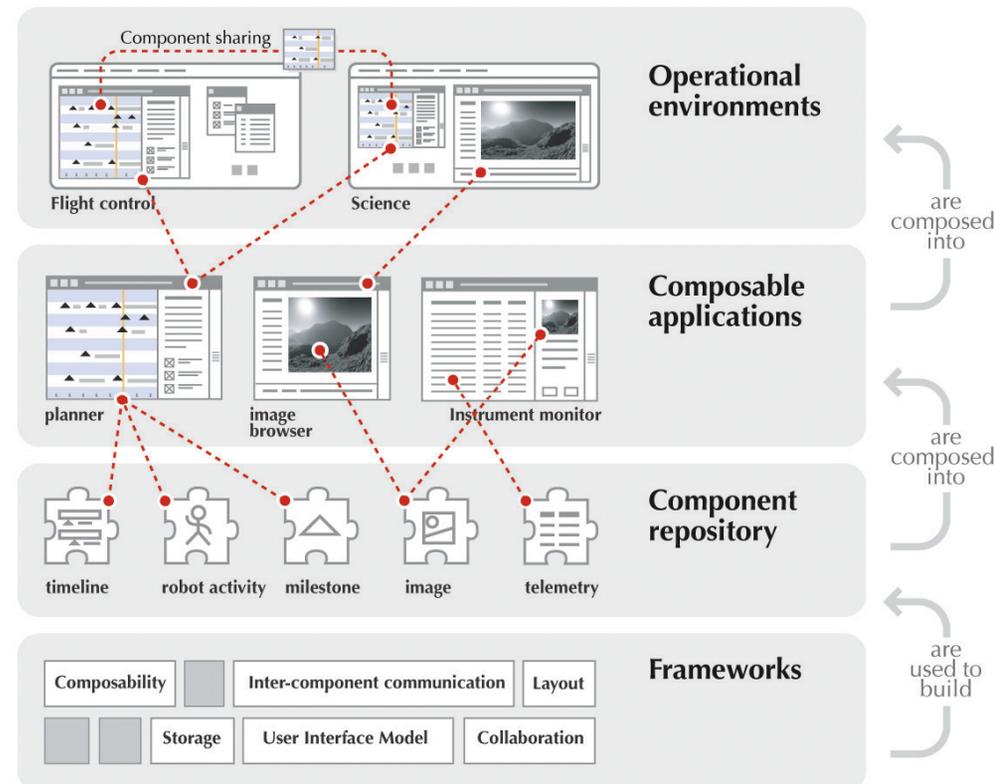
Vision developed in response to observed ops issues

- Fast and flexible adaptation to evolving mission process requirements through composable objects and applications
- Flexible mission processes unconstrained by application boundaries
- Consistent user environments, role-based adaptable displays w/consistent information in multiple representations



Vision developed in response to observed ops issues

- Low cost multi-mission adaptations
- Components developed across multiple organizations working together as a coherent whole
- Collaboration unconstrained by machine boundaries

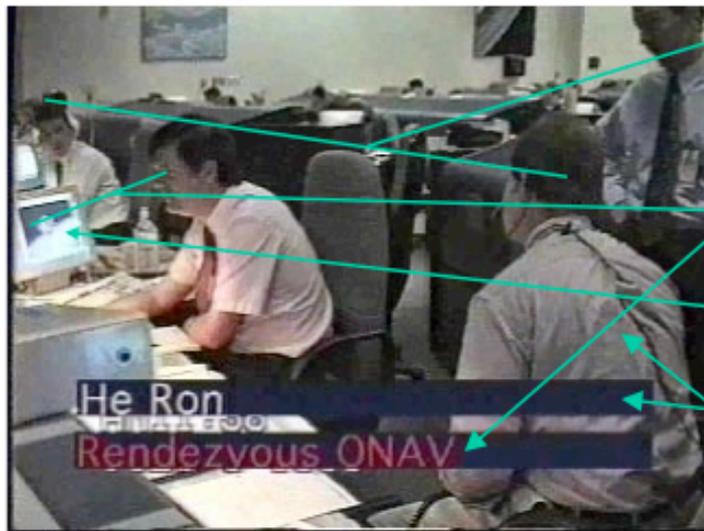


User centered methods applied to human spaceflight

- Design for mission and user needs based on observational methods: “the proof is in the observations.”



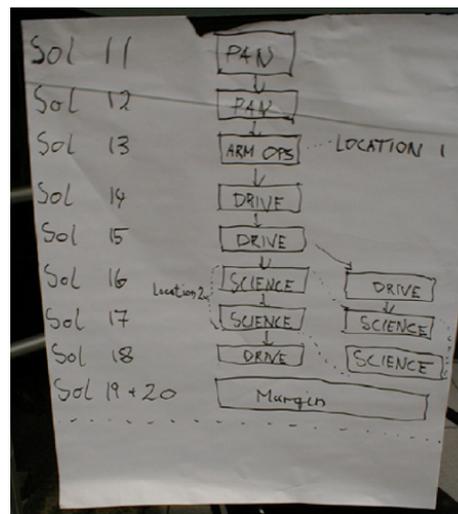
- A total system perspective:



- F2F conversation
- Voice loops
- Computer interfaces
- Model inferences
- Outlouds
- Non-verbal gestures

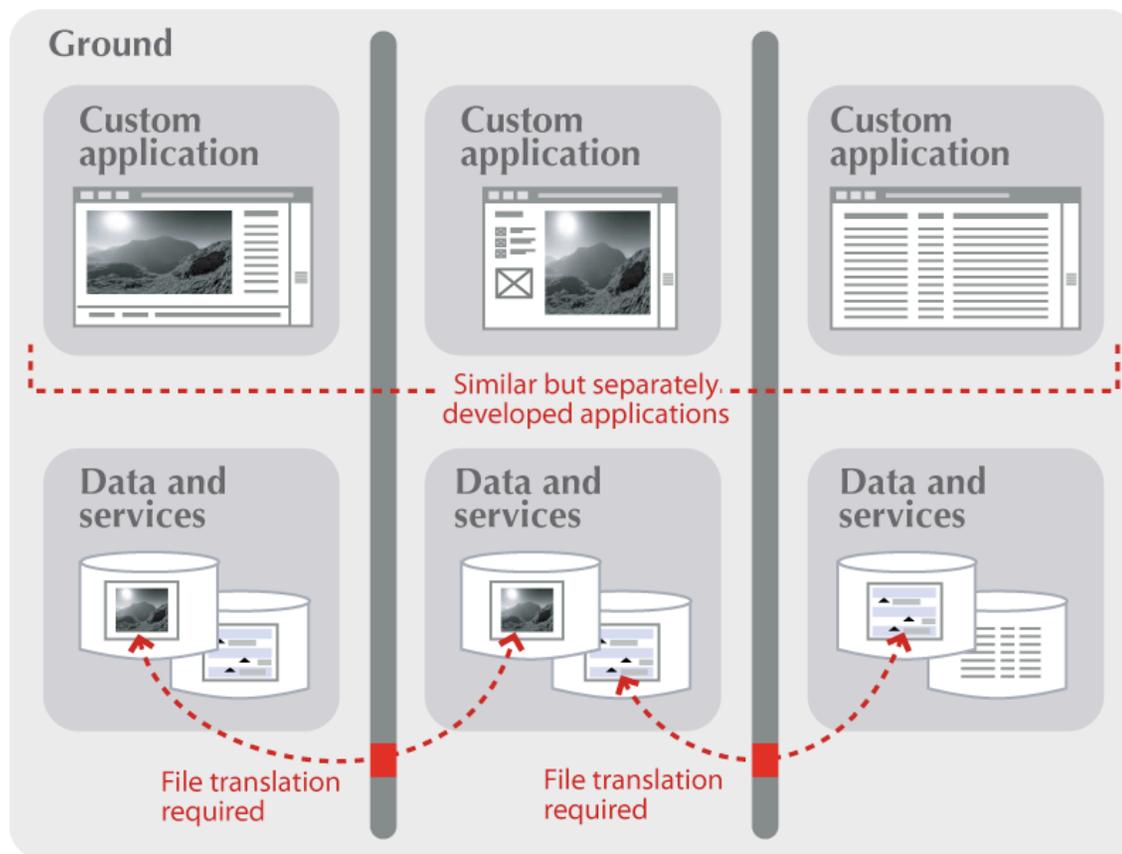
Human centered methods applied to robotic space flight

- Collaborative modeling
- Participatory design
- Evaluation with users



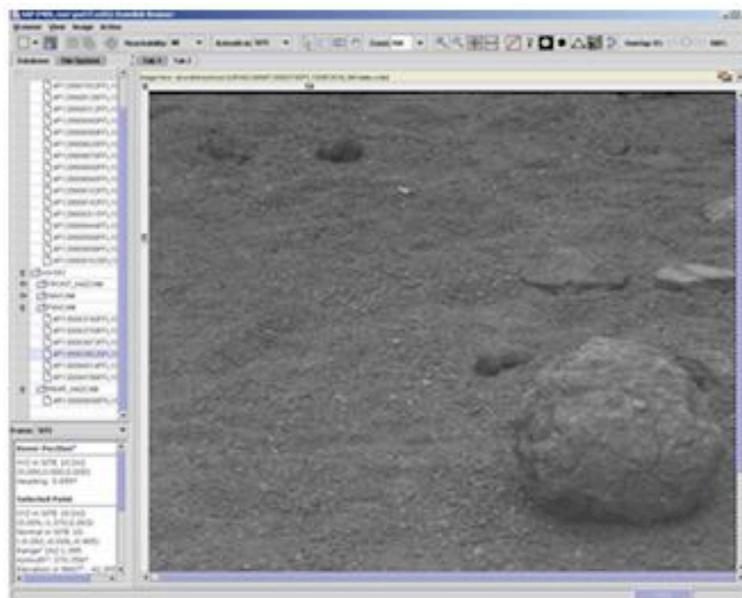
The problem of applications

- Custom-built apps are monoliths with separate data models, preventing dynamic interoperation and incurring code & data duplication



The problem of applications

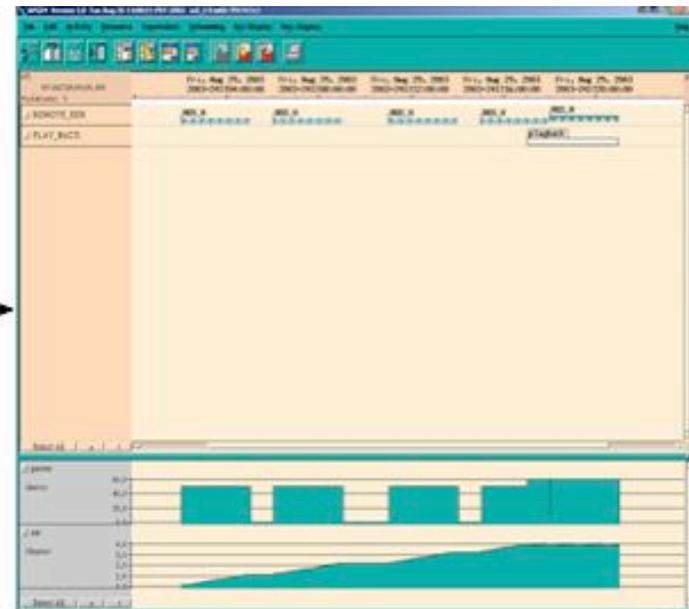
- Separate, inconsistently-designed apps diminish task focus, usability and productivity



Activity creation in one application



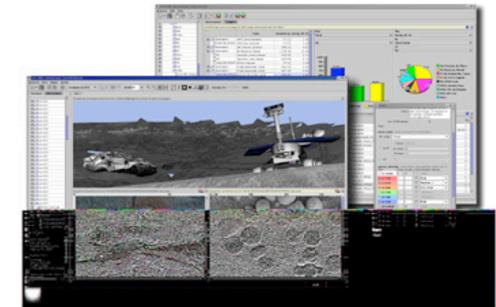
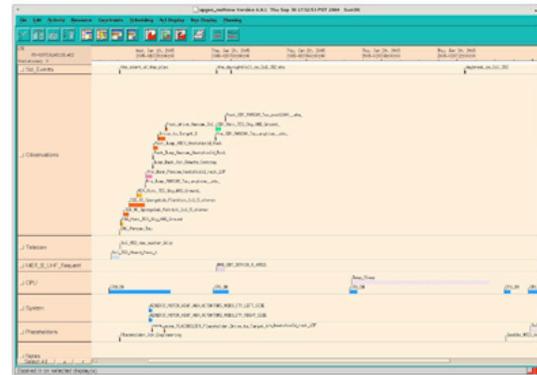
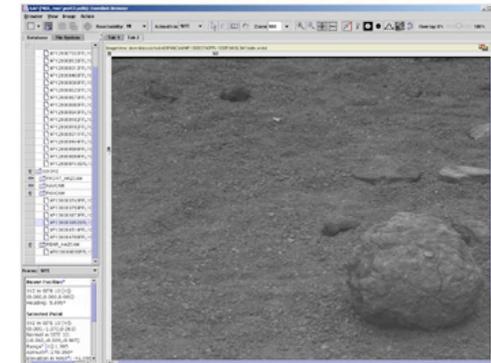
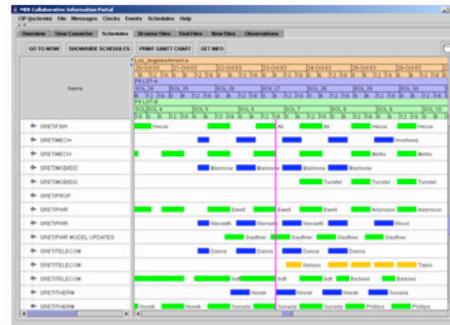
File
Xfer



Activity and resource planning in a second application

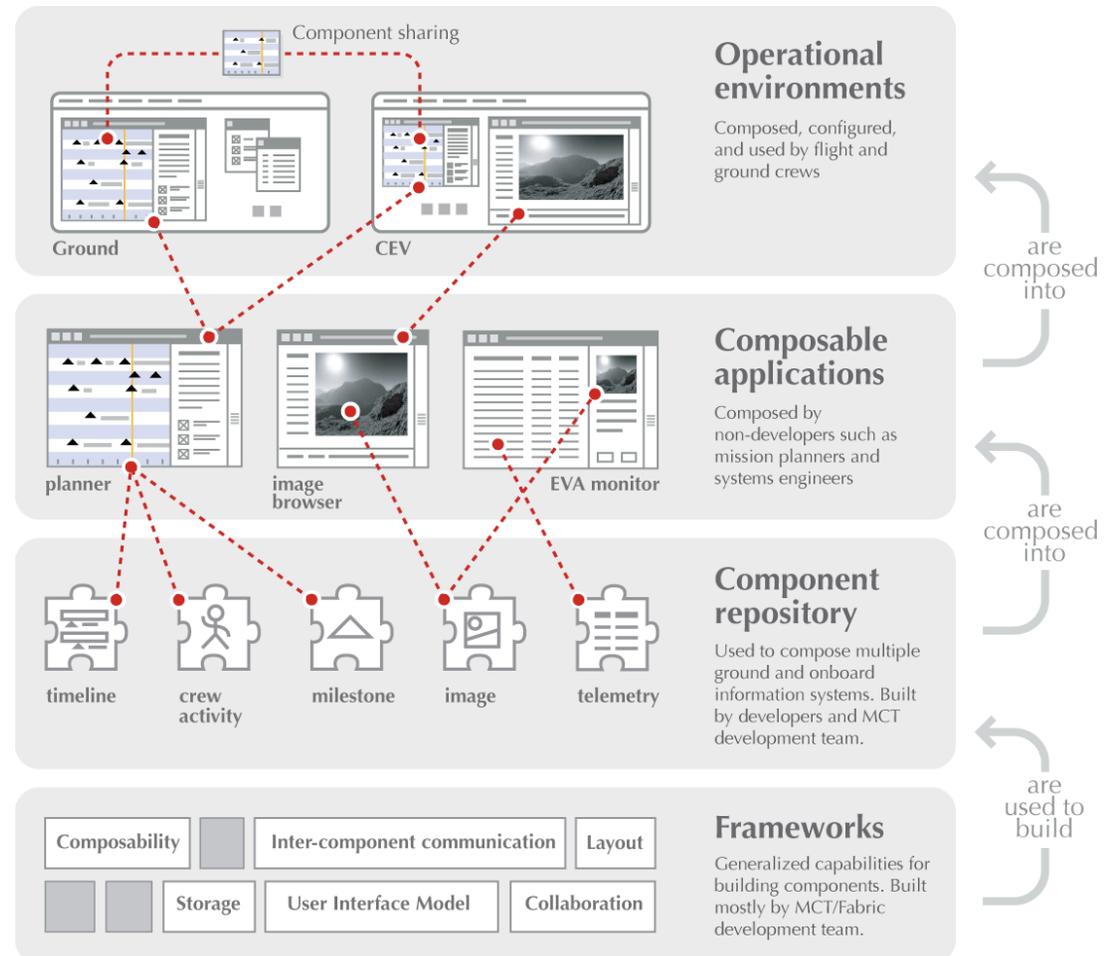
The problem of applications

- Duplication of functionality
- Lack of integration
- Inflexible
- Inconsistent environments = increased training, increased error potential
- Task constraints, stovepiped sequential processes, glueware
- Monolithic apps lead to bloat, inability to deliver feature sets for specific needs without “baggage”



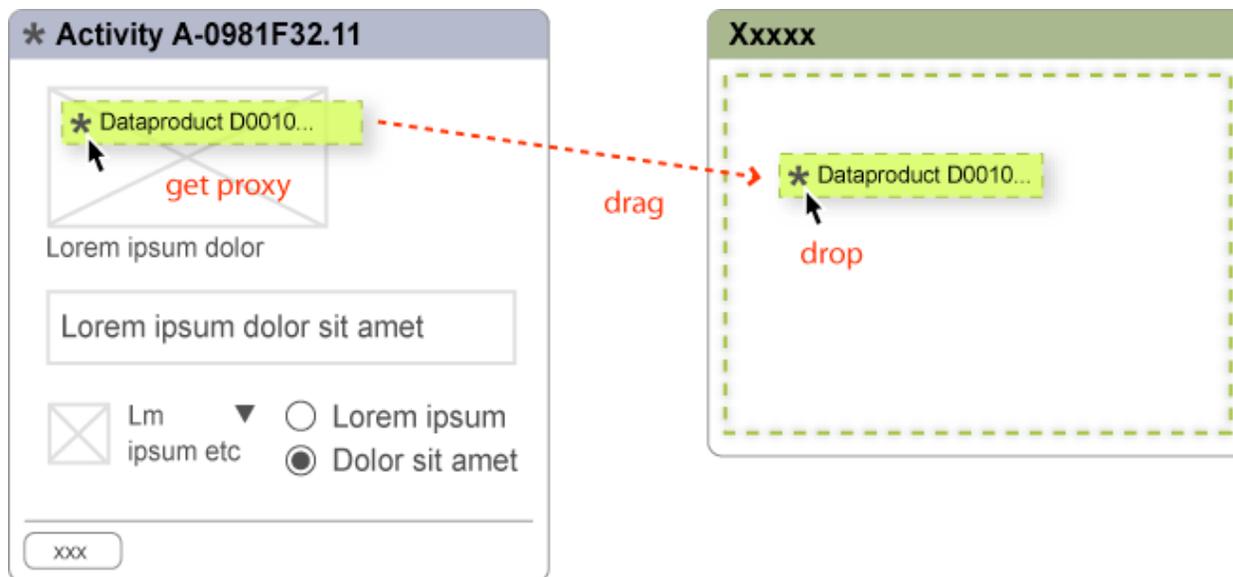
Proposed solutions

- Re-factor software
- Don't think of software as installed applications
- Instead, deliver functionality as collections of services and (user) objects
- Objects and services must be accessible from the users machine, but not necessarily installed directly on it



Benefits

- Composition
 - Flexibility to adapt software to lessons learned, reduce software-induced operational workarounds
 - Enable iterative design, optimization of tasks, specific feature sets for missions and roles



Benefits

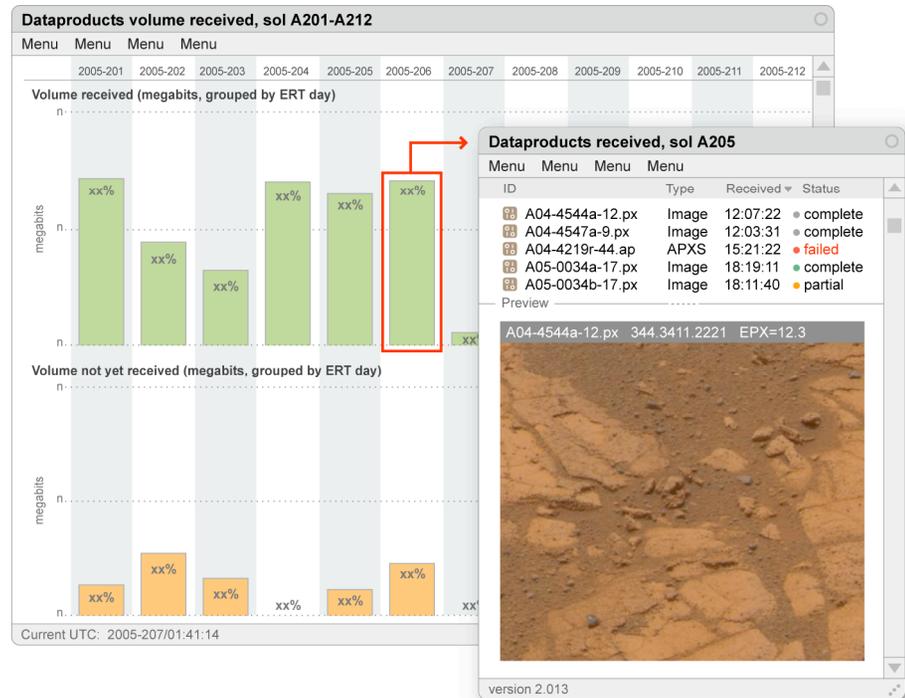
- Interoperability via information models
 - Consistent interfaces and interactions = reduced error potential
 - Components remain autonomous and independent
 - Connections are loosely coupled
 - Relationships and rules are captured in information models
 - “Glueware” is reduced
- Distribution
 - Distributed operations capability for location-independent computing supports future mission ops needs for collaborative distributed mission operations

Mission Control Technologies are...

- *Frameworks* that support development of fine-grained component/services software
 - Think of software not as installed applications on an individual machine, but rather as collections of components and services available on any machine
- *Information model(s)* that supports component/service interoperability
- *Architecture* that supports distributed deployment: messaging, service invocation, collaboration, security, replication, persistence, user management, network integration
- MCT components are compatible with related NASA efforts, such as Ensemble

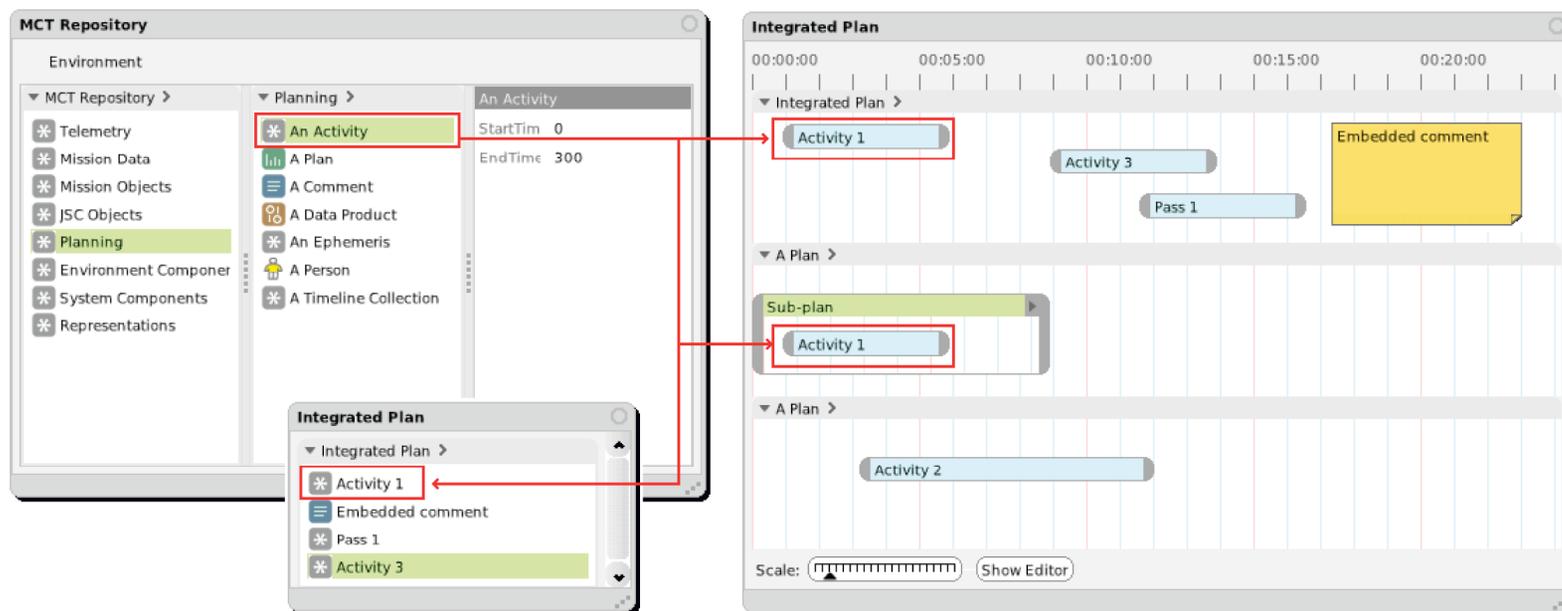
MCT architecture

- Designed for fine-grained components
- Roles
- Multiple representations and views
- “Live” objects
- Compositions
- Information model



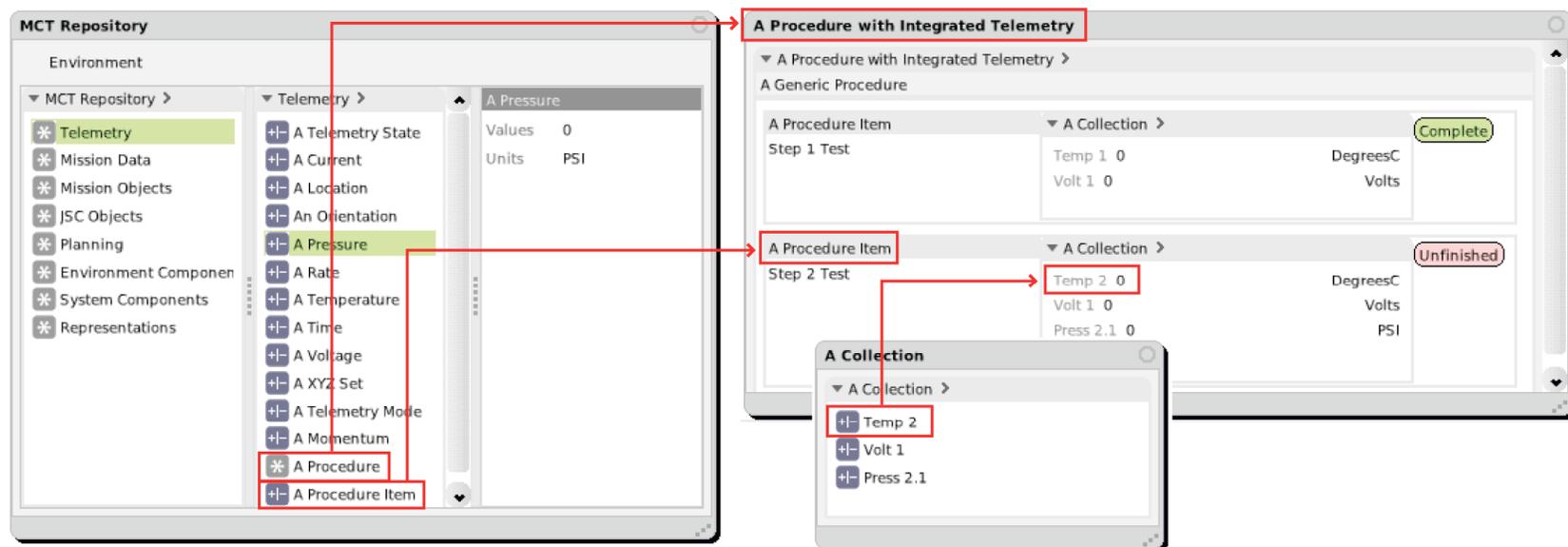
Prototype

- Broad functionality; validate the component model across multiple domains
- Composed in real-time from fine-grained “live” components from a repository
- Plans, activities, plan within plan, embedded objects, such as comment
- Multiple representations of the same component, displayed in different ways



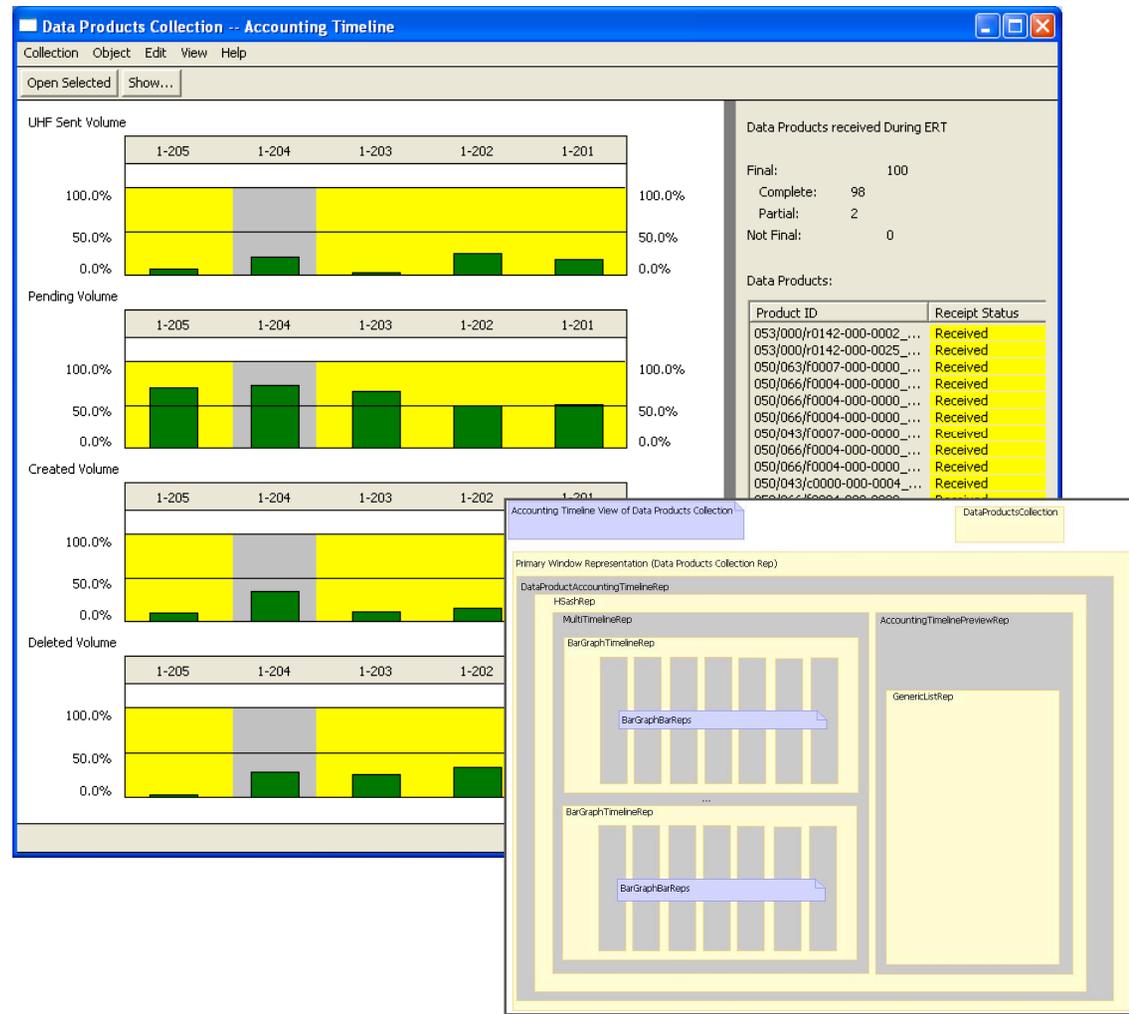
Prototype

- Further component model validation across domains
- Integration of telemetry and procedures
- Telemetry points integrated into procedure steps to form composite display
- Procedural representation and “traditional” representation of the same info



Pilot application

- Build the infrastructure on top of Eclipse
- Develop and test specific functionality, connected to a real data source
- Build on existing collaboration with JPL on data accountability work



First deployable component set being built in 2006

- Telemetry and monitoring components in multiple composable views
- Customer is Johnson Space Center Gemini console positions
- Test reconfiguration for Marshall Space Flight Center w/same components

