



EOSDIS Core System (ECS) Science Data Processing System (SDPS) Lessons Learned

Panel Members:

Mike Moore (EOSDIS Project) - Chair

Dawn Lowe (EOSDIS Project) - Moderator

Curt Schroeder (EOSDIS Project)

Steve Fox (Raytheon)

Agenda

è Background - Dawn Lowe

- System Overview - Mike Moore
- Program/Project Management Lessons Learned - Curt Schroeder
- System Development Lessons Learned - Mike Moore
- More System Development Lessons Learned - Steve Fox
- Questions and Answers

EOSDIS Science System Purpose

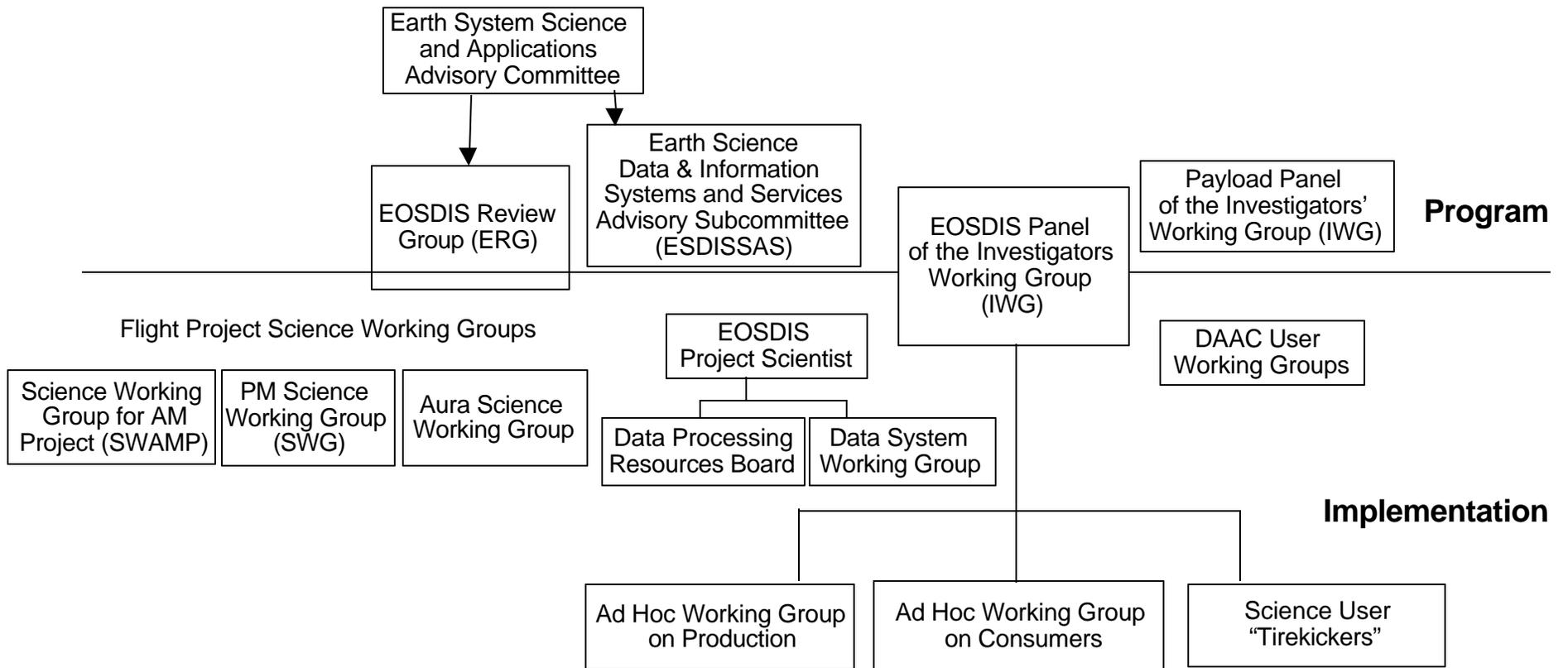
Global Climate Change Research requires:

- Integration of observations from spacecraft, aircraft, and surface observations
- Long term, consistently processed data to support interdisciplinary studies of the Earth as a system
- Access and distribution of diverse types of data to a wide range of scientists
- An integrated view of distributed information
- Management of very large data holdings, with precise metadata

Science Guidance

National Research Council (NRC) Committees
 (e.g., Board on Sustainable Development (BSD),
 Committee on Global Change Research (CGRC),
 Committee on Geophysical and Environmental Data
 (GCED))

Agency



Program Reviews and Science Reviews with Redirection

	1992				1993				1994				1995				1996				1997				1998				1999				2000							
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4				
Program Reviews	▽ Red/Blue Team Rev				▽ EOS Restructuring				▽ EOS Rebaselining				▽ EOS Reshaping				▽ EOS Reshape Implementation Study				▽ EOS Biennial Review				▽ Little's Committee															
					▽ EOS Rescoping				▽ Independent Architecture Studies				▽ EOS Reshape				▽ NASA Zero-Base Review				▽ Internal GSFC Review																			
									▽ IAR				▽ IAR				▽ IAR				▽ IAR				▽ IAR				▽ IAR				IAR ▽							
External Science Reviews	▽ GAO Audit(Technology&Architecture)				▽ NRC EOSDIS Review				▽ ECS Progress Review				▽ NRC Review(LaJolla)-Federation				▽ NRC Refinement of Federation recommendations				▽ EOSDIS Review Group(ERG)				▽ EOSDIS Review Group(ERG)				▽ EOSDIS Review Group(ERG)				▽ ECS Performance Review							

ther External Reviews

94		1995				1996				1997				1998				1999				2000	
3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4		
<div style="display: flex; justify-content: space-between; padding: 10px;"> <div style="width: 15%;"> <p>7 EOSDIS Cost Review(Data Panel)</p> </div> <div style="width: 85%;"> <p style="margin-left: 40px;">▼ EOSDIS Cost Review(IWG)</p> <p style="margin-left: 80px;">▼ EOSDIS Cost Review(Payload Panel)</p> </div> </div>																							

ESDIS Project Reviews

	1993				1994				1995				1996				1997				1998				1999				2000																			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4																
EOSDIS Core System (ECS) Project Reviews	SRR ▼				SDR ▼				PDR ▼				Rel A CDR ▼																																			
									Rel B IDR ▼				Rel B CDR ▼ Data Server Delta CDR ▼				ERG Demo ▼																															
																					Rel 4B RRR ▼																											
																									Rel 5A CSR ▼																							
																					Rel 5B IRR ▼								Rel 5B CSR ▼																			
																													Rel 6A IRR ▼																			
																	L7 MOR ▼																															
																									L7 FOR ▼								Terra ORR ▼															
																													L7 ORR ▼								Aqua MOR ▼											

ESDIS Project/Program/Science/Other External Reviews

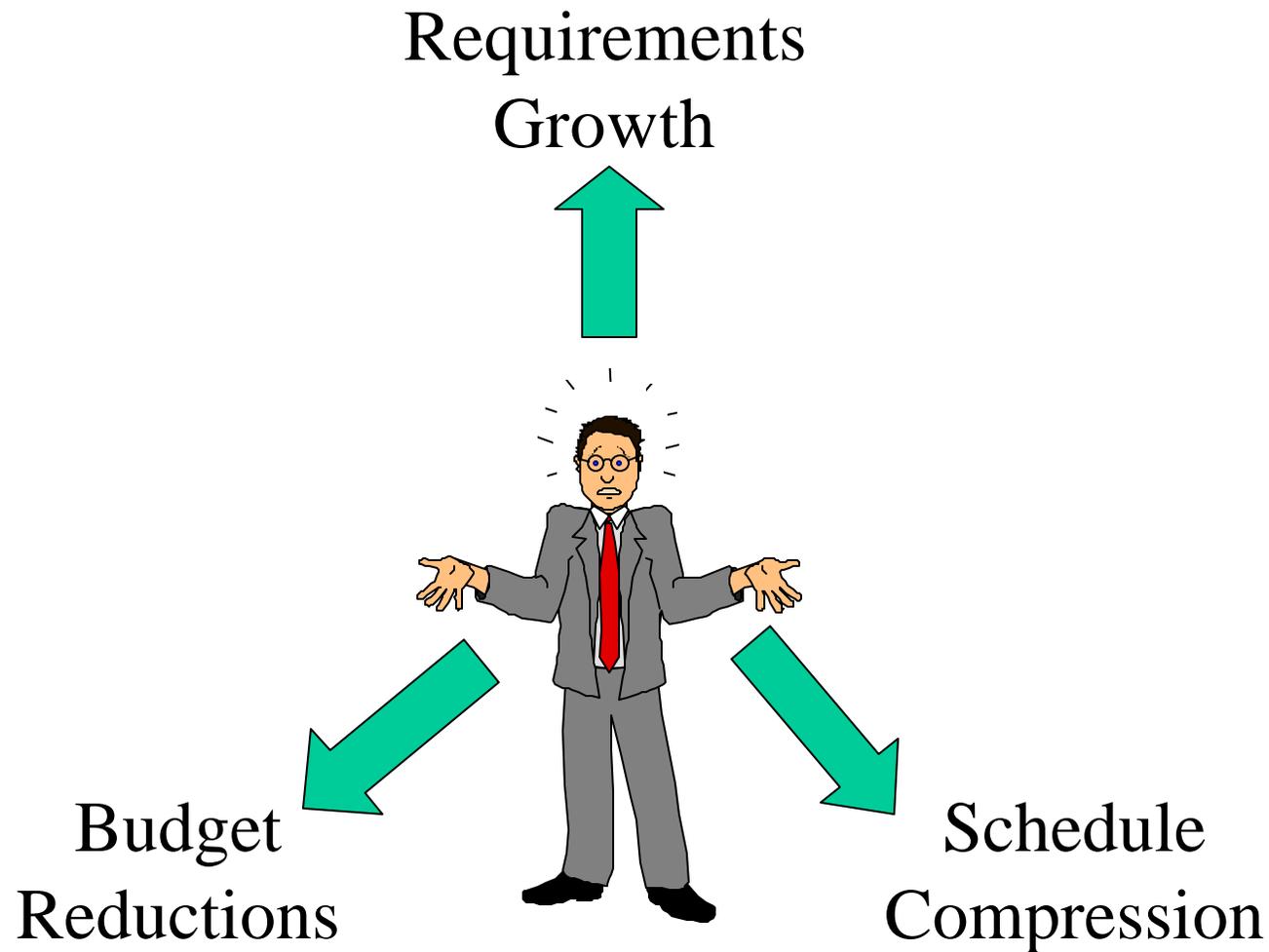
	1992				1993				1994				1995				1996				1997				1998				1999				2000						
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4			
	▽ GAO Audit						▽ ECS SRR		▽ ECS SDR				▽ ECS PDR				▽ ECS Rel B CDR							▽ ERG Demo															▽ IAR
				▽ GAO Audit			▽ EOS Rescoping				▽ GAO Audit		Rel A CDR ▽								▽ Internal GSFC Review									▽ AM1 ORR									
	▽ GAO Audit																													▽ Rel 4B RRR									
	▽ GAO Audit						▽ IG Rapid Action Report/EDOS						▽ ECS RB IDR																	▽ Rel 5A CSR									
				▽ EOS Restructuring					▽ GAO Audit								▽ IAR				▽ EOSDIS Review Group(ERG)									Rel 5B IRR ▽				▽ Rel 5B CSR					
	▽ Red/Blue Team Rev						▽ IAR										▽ EOS Reshape Implem. Study																	▽ Rel 6A IRR					
	▽ NRC EOSDIS Review								▽ Independent Architect. Studies															▽ EOSDIS Review Group(ERG)										▽ Rel 6B IRR					
				▽ EOSDIS Cost Review (Data Panel)									▽ EOS Reshape				▽ NRC Refinement of Federation recomm.													▽ EOSDIS Review Group(ERG)									
							▽ ECS Progress Review																											▽ (ERG)					
							▽ NRC Review(LaJolla)-Federation																																
									▽ EOS Rebaselining																														
									▽ Independent Architecture Studies																														
									▽ IG Audit/ECS Award Fee																														
													▽ IAR																										
																	▽ IG Audit/Subcontract Mgmt																						

Remember Cohn's Law.

The more time you spend in reporting on what you are doing, the less time you have to do anything.

Stability is achieved when you spend all your time doing nothing but reporting on the nothing that you are doing.

The SDPS has experienced the classic Death March scenario.



ECS Science System Today

- Supporting Landsat-7 since April '99, and Terra since January '00
- Managing extraordinary rates and volumes of scientific data:
 - Landsat-7 producing 150 Gbytes of data per day
 - Terra spacecraft produces 194 Gbytes/day:
 - Almost twice as much data as HST (600 MB/day) acquires in an entire year
 - As much data as UARS (345 MB/day) takes in in 1 1/2 years
 - As much as TRMM takes in 200 days
 - When spacecraft data are processed to higher level products, Terra results in more than 1 Tbytes being added to the archive per day
 - In less than a year, the Terra instruments have doubled NASA's Earth Science data holdings.

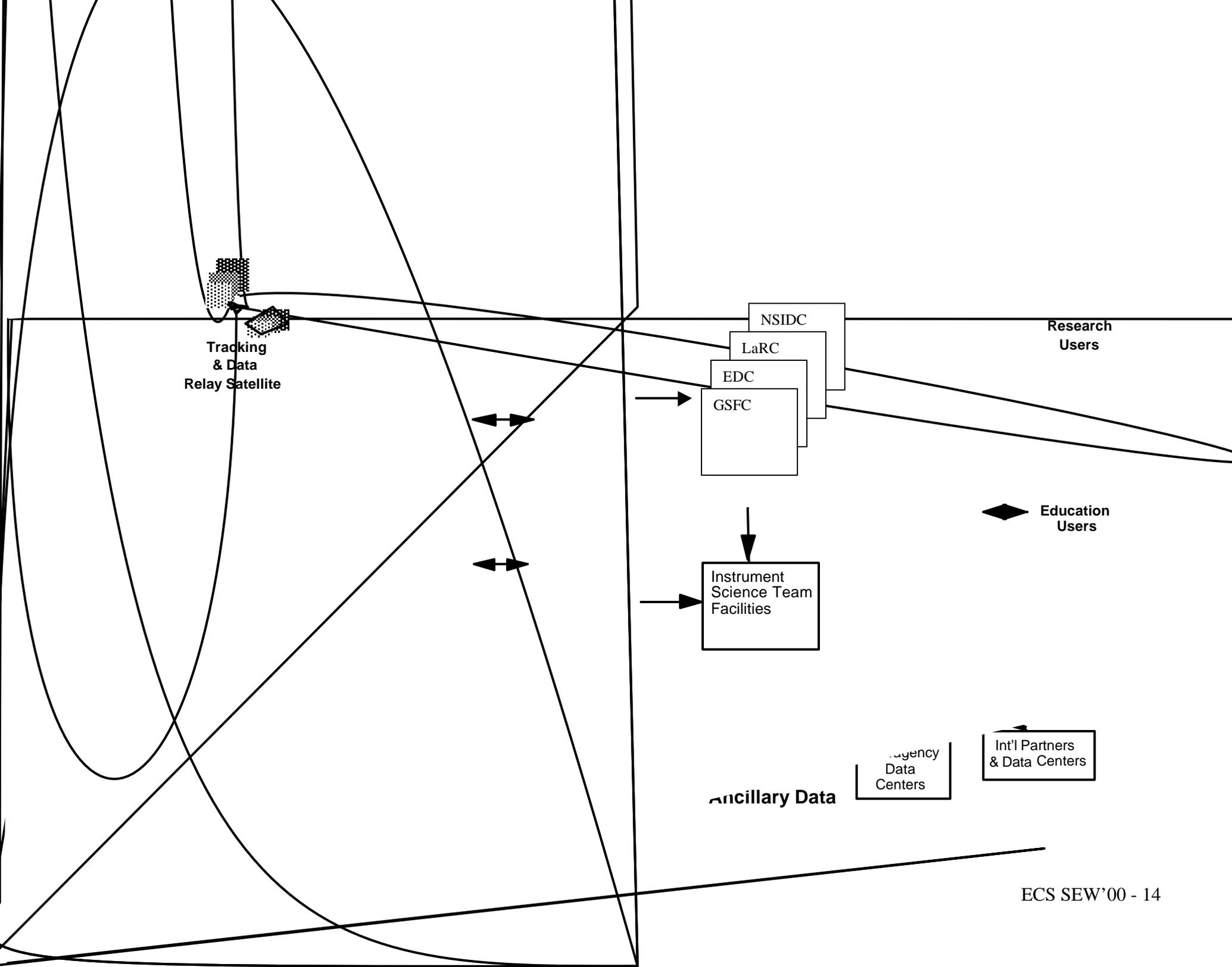
Agenda

- Background - Dawn Lowe
- è System Overview - Mike Moore
- Program/Project Management Lessons Learned - Curt Schroeder
- System Development Lessons Learned - Mike Moore
- More System Development Lessons Learned - Steve Fox
- Questions and Answers

System Business Context

The Earth Observing System (EOS) Data Information System (EOSDIS) Science Data Processing Segment (SDPS) will process, archive and distribute earth science data from 20 science instruments on 9 major spacecraft:

- Long system operational life: 1999 to ~2012
- Distributed across four Data Centers:
 - Earth Resources Observation System (EROS) Data Center (EDC), Sioux Falls, SD
 - Goddard Space Flight Center (GSFC), Greenbelt, MD
 - Langley Research Center (LaRC), Hampton, VA
 - National Snow and Ice Data Center (NSIDC), Boulder, CO
- Many Stakeholders
 - Data Center operations staff
 - Instrument-specific science teams
 - Inter-disciplinary science users
 - Other agencies and data user groups, including general public
- De-centralized Authority
 - Data Centers maintain control of their local facilities and interface with data users
 - Instrument Teams maintain control over their science software and data products



Tracking
& Data
Relay Satellite

NSIDC

LaRC

EDC

GSFC

Research
Users

Education
Users

Instrument
Science Team
Facilities

Ancillary Data

Agency
Data
Centers

Int'l Partners
& Data Centers

System Scale

- Unprecedented scope for NASA
 - Support many different data product types: ~1,300 different product types
 - Archive large amounts of data: adding ~ 1 TB per day (today), ~ 3 PB by 12/03, ~9 PB by 2011
 - Manage large geo-spatial databases: adding ~ 140 MB per day, ~ 293 GB by 12/03
 - Distribute large amounts of data in several formats: ~ 2 TB per day
- Complex system context
 - Execute complex science algorithms provided by science community
 - Interface with ~ 35 external systems
- Large system with heavy dependence on COTS:
 - ~ 1.1 M lines of C++ and Java custom code
 - ~ 50 COTS products integrated with custom code
 - System deployed at Data Centers on 12 to 25 enterprise class Unix servers from multiple vendors

Building software is a learning and re-learning experience!

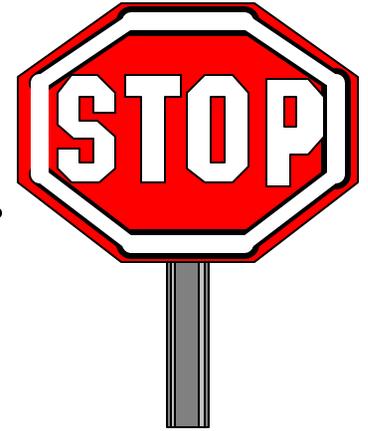


Copyright © 2000 United Feature Syndicate, Inc.
Redistribution in whole or in part prohibited

Agenda

- Background - Dawn Lowe
- System Overview - Mike Moore
- è Program/Project Management Lessons Learned - Curt Schroeder
- System Development Lessons Learned - Mike Moore
- More System Development Lessons Learned - Steve Fox
- Questions and Answers

If You False Start, Restart Sooner, Not Later



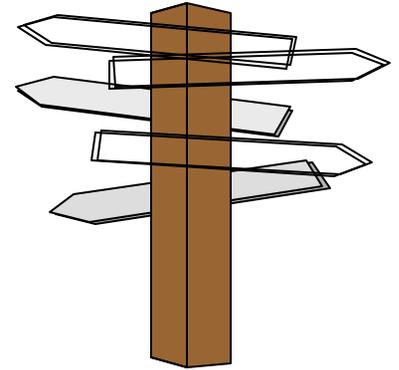
- Science community was not fully supportive of the EOSDIS approach to ECS
 - Multiple versus single implementers
 - Iterative (with feedback) versus waterfall development model
- Instead of concentrating on the design, time was wasted arguing the merits of the approach

If You False Start, Restart Sooner, Not Later (Cont'd)



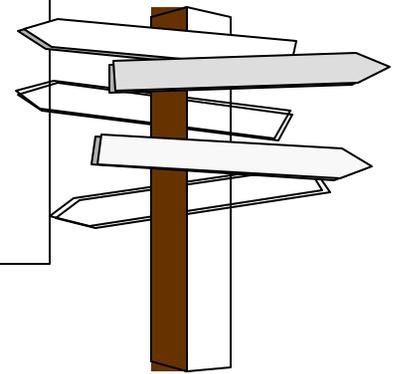
- Eventually restructured contract
 - Reallocated development responsibilities where functions could be isolated to an interface
 - Higher level science product generation by instrument teams
 - End user data access
 - Revisited requirements with user involvement

Requirements and More Requirements



- Many ‘powerful’ stakeholders resulted in:
 - Conflicting agendas, requirements, and priorities
 - No mechanism to reach closure
 - The development team caught in the middle

Requirements and More Requirements (Cont'd)



- To gain some control:
 - Involved users to gain user ownership
 - Sponsored workshops and supported ad-hoc working groups to reach community consensus
 - Instituted process for users to prioritize requirements
 - Involved users in test program
- Don't over-achieve with user involvement in Project Management

Another External Review Never Hurt

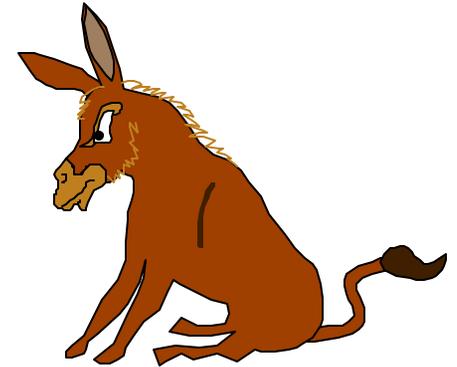


- Except for:
 - Diminishing value added
 - Potential for more conflicting direction
 - Prolonged period of redirections
 - External review teams are not stakeholders
 - Short term commitment
 - Nothing to lose
 - Distracted best technical people from designing the system

ESDIS External Reviews

	1992				1993				1994				1995				1996				1997				1998				1999				2000			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
<div style="display: flex; flex-direction: column; align-items: flex-start;"> <div style="margin-bottom: 10px;">GAO Audit ▼</div> <div style="margin-bottom: 10px;">GAO Audit ▼</div> <div style="margin-bottom: 10px;">GAO Audit ▼</div> <div style="margin-bottom: 10px;">EOS Restructuring ▼</div> <div style="margin-bottom: 10px;">Red/Blue Team Rev ▼</div> <div style="margin-bottom: 10px;">NRC EOSDIS Review ▼</div> <div style="margin-bottom: 10px;">EOSDIS Cost Review (Data Panel) ▼</div> <div style="margin-bottom: 10px;">ECS Progress Review ▼</div> <div style="margin-bottom: 10px;">NRC Review(LaJolla)-Federation ▼</div> <div style="margin-bottom: 10px;">EOS Rebaselining ▼</div> <div style="margin-bottom: 10px;">Independent Architecture Studies ▼</div> <div style="margin-bottom: 10px;">IG Audit/ECS Award Fee ▼</div> <div style="margin-bottom: 10px;">IAR ▼</div> </div>	<div style="display: flex; flex-direction: column; align-items: flex-start;"> <div style="margin-bottom: 10px;">EOS Rescoping ▼</div> <div style="margin-bottom: 10px;">IG Rapid Action Report/EDOS ▼</div> <div style="margin-bottom: 10px;">IAR ▼</div> <div style="margin-bottom: 10px;">Independent Architect. Studies ▼</div> <div style="margin-bottom: 10px;">EOS Reshape ▼</div> <div style="margin-bottom: 10px;">EOSDIS Cost Review(IWG) ▼</div> <div style="margin-bottom: 10px;">EOSDIS Cost Review(Payload Panel) ▼</div> <div style="margin-bottom: 10px;">NASA Zero-Base Rev. ▼</div> <div style="margin-bottom: 10px;">IG Review of DAACs ▼</div> <div style="margin-bottom: 10px;">IG Audit/Subcontract Mgmt ▼</div> </div>	<div style="display: flex; flex-direction: column; align-items: flex-start;"> <div style="margin-bottom: 10px;">GAO Audit ▼</div> <div style="margin-bottom: 10px;">EOS Rescoping ▼</div> <div style="margin-bottom: 10px;">IG Rapid Action Report/EDOS ▼</div> <div style="margin-bottom: 10px;">GAO Audit ▼</div> <div style="margin-bottom: 10px;">Independent Architect. Studies ▼</div> <div style="margin-bottom: 10px;">EOS Reshape ▼</div> <div style="margin-bottom: 10px;">EOSDIS Cost Review(IWG) ▼</div> <div style="margin-bottom: 10px;">EOSDIS Cost Review(Payload Panel) ▼</div> <div style="margin-bottom: 10px;">NASA Zero-Base Rev. ▼</div> <div style="margin-bottom: 10px;">IG Review of DAACs ▼</div> <div style="margin-bottom: 10px;">IG Audit/Subcontract Mgmt ▼</div> </div>	<div style="display: flex; flex-direction: column; align-items: flex-start;"> <div style="margin-bottom: 10px;">IAR ▼</div> <div style="margin-bottom: 10px;">EOS Reshape Implem. Study ▼</div> <div style="margin-bottom: 10px;">NRC Refinement of Federation recomm. ▼</div> <div style="margin-bottom: 10px;">Data Server Delta CDR ▼</div> <div style="margin-bottom: 10px;">IG Review of DAACs ▼</div> <div style="margin-bottom: 10px;">IG Audit/Subcontract Mgmt ▼</div> </div>	<div style="display: flex; flex-direction: column; align-items: flex-start;"> <div style="margin-bottom: 10px;">ERG Demo ▼</div> <div style="margin-bottom: 10px;">Internal GSFC Review ▼</div> <div style="margin-bottom: 10px;">Lucent Tech Review ▼</div> <div style="margin-bottom: 10px;">EOSDIS Review Group(ERG) ▼</div> <div style="margin-bottom: 10px;">EOS Biennial Review ▼</div> <div style="margin-bottom: 10px;">IG-Dissemination of MTPE Info. ▼</div> <div style="margin-bottom: 10px;">IG-Federation ▼</div> <div style="margin-bottom: 10px;">Architecture Review ▼</div> <div style="margin-bottom: 10px;">IAR ▼</div> <div style="margin-bottom: 10px;">ECS Performance Review ▼</div> </div>	<div style="display: flex; flex-direction: column; align-items: flex-start;"> <div style="margin-bottom: 10px;">Terra ORR ▼</div> </div>																															

Don't Fool Yourself



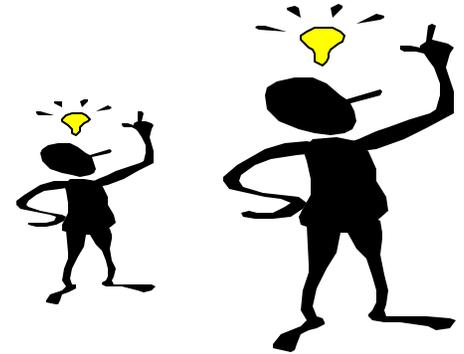
- Wishful thinking will not gain lost schedule time
 - If you are consistently missing milestones, you are very unlikely to recoup schedule time by working more efficiently in the future.
- Fix the problem and accept the losses.
 - Spun-off work where possible
 - Re-establish doable requirements and schedules

What attrition?



- High tech attrition rates (30%) are real
 - Money talks and people walk
 - You have little control
 - Rumors affect people's career decisions
- Factor this reality into your schedules and plans
 - Created pro-career environment (e.g. in-house training)
 - Supported mission-based incentive plan (\$)

Lessons RE-Learned



- Watch what your developer does; not what they say they did
- If you don't measure where you are going, you will not get there
- Don't let event/task progress reporting go longer than 2 weeks

Agenda

- Background - Dawn Lowe
- System Overview - Mike Moore
- Program/Project Management Lessons Learned - Curt Schroeder
- System Development Lessons Learned - Mike Moore
- More System Development Lessons Learned - Steve Fox
- Questions and Answers

Going over big waterfalls is risky!



- Require interim releases
 - we have interim releases every 6 to 9 months
 - focus on most critical and riskiest functionality first
- Be able to define and re-define your releases when it's appropriate
 - we use Science System Release Plans
 - describe functionality, and test and delivery approach
 - created ~3 months before detailed release definition
 - updated as required (see next lesson)

Don't be a Death March drill sergeant!



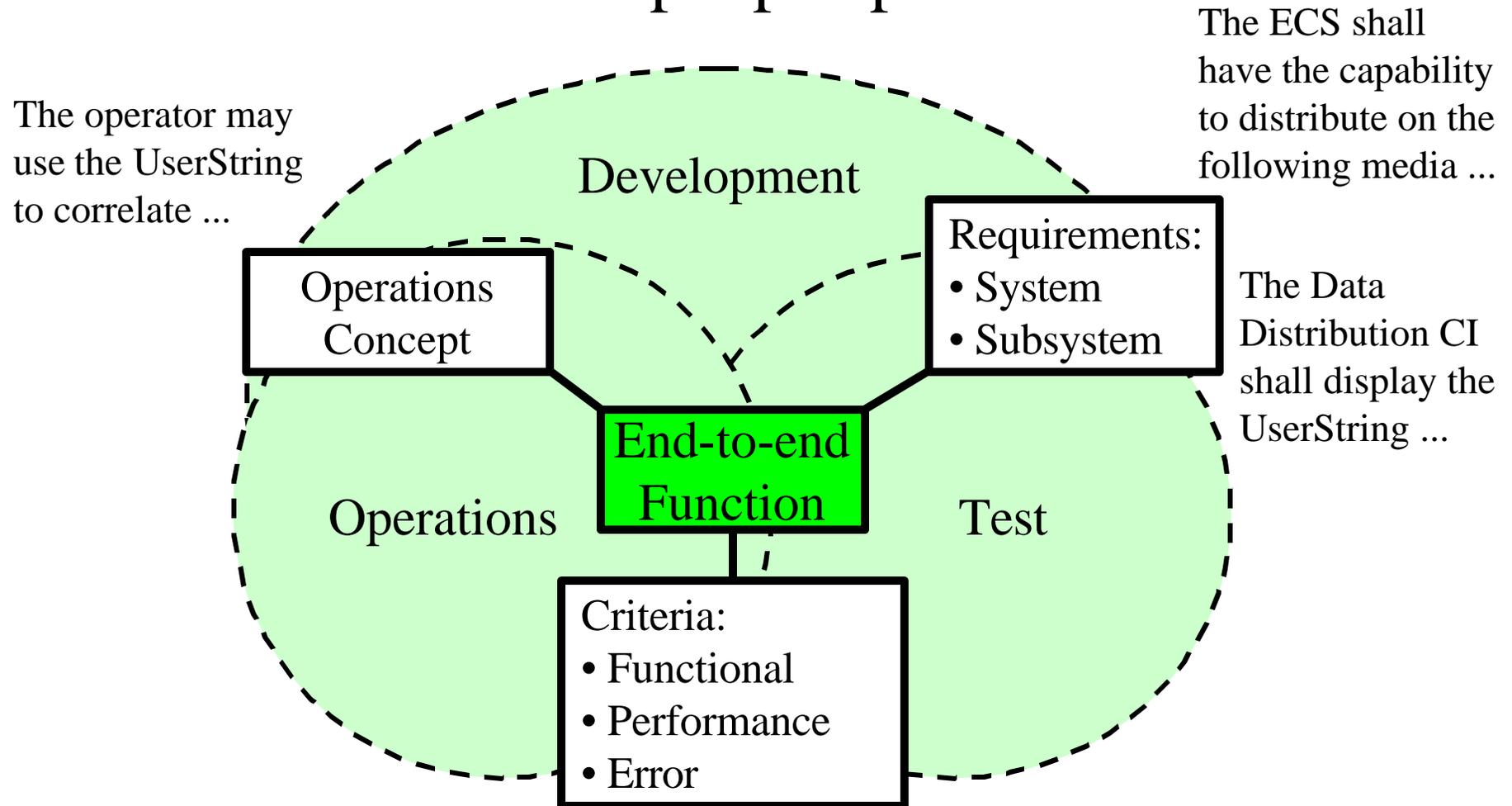
- Be willing to prioritize your requirements; be able to reprioritize when necessary
 - our Priority Board allocates functionality across deliveries (I.e., releases and patches) as needed
- Be able to drop from the bottom when you must
 - Priority Board can move functions to a later delivery
 - Priority Board can drop lower priority sub-functions
 - changes are documented in updates to release plan and "tickets" (see next lesson)
- Tools like the SEI's Architecture Tradeoff Analysis Method can help

Be careful what you ask for and when!



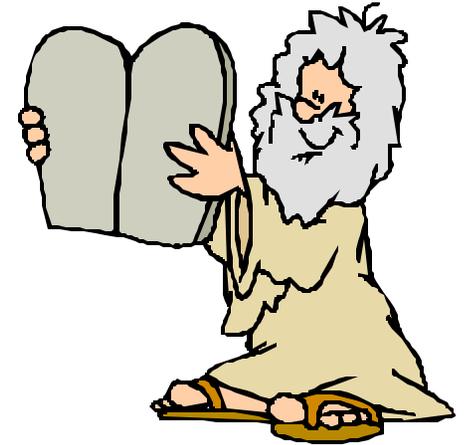
- Defer decisions until needed:
 - we use Tickets to define end-to-end system functions
 - for release plan, include only top-level requirements and constraints
 - during release definition, add detailed operations concept and requirements for associated subsystems
 - prior to release design, add detailed criteria for assessing successful implementation
 - tickets are approved prior to release detailed design
 - tickets are updated to reflect agreed upon changes

A ticket defines an end-to-end function from multiple perspectives.



Demonstrate that ... 4. the UserString appears in the Distribution Notification.
Show that the command line interface can support 25 concurrent requests with ...
Re-submit a request using the same UserString, verify that ...

Every system needs a soul!



- Establish a group with sufficient expertise to oversee all levels of system definition
 - we have established an Architect's Office
 - has detailed domain and implementation expertise needed to make difficult trades
 - leads ticket elaboration
 - guides design, code and test walkthroughs
 - arbitrates resolution of implementation issues

Get and keep their heads in the game!



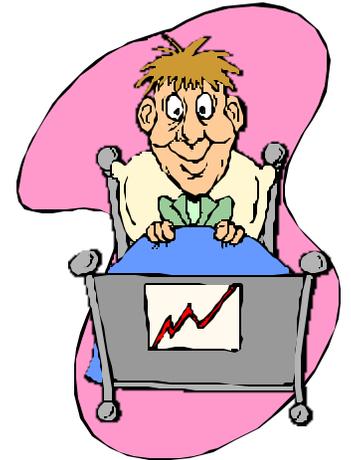
- Establish a system context for all efforts
 - tickets provide end-to-end system context
- Make the context easily accessible by all team members
 - our web database links ticket elements, tests, test results and problem reports
- Encourage knowledge sharing between groups
 - team members build on each other's results
 - criteria -> integration tests -> acceptance tests
 - ops concept + acceptance tests -> ops procedures
 - peer reviews involve all team members

They won't know it until
they see it, so let them
see it!



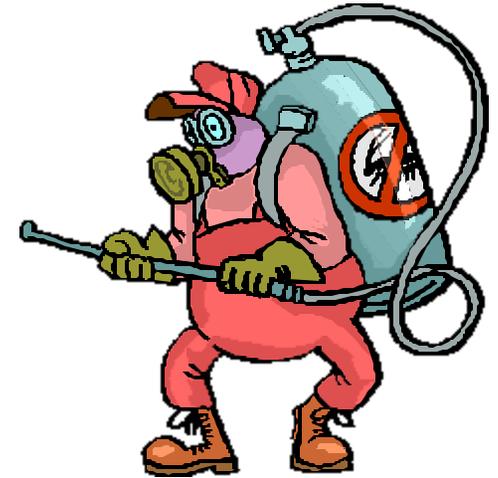
- We make extensive use of operational (end-to-end) prototyping to allow science users and operators to get early looks at key functionality

Check the system's vital signs often!



- Do daily builds
 - we now build all releases of our entire system every night
- Do regression testing frequently
 - we execute an evolving suite of automated regression tests for each internal turnover and each delivery
- Demonstrate new functionality before formal testing
 - a separate integration team with development expertise demonstrates new functionality before formal testing
- Perform on-going performance testing
 - new releases and major patches undergo full performance testing before delivery

Use Zero Defect milestones!



- Establish multiple defect control gates to drive problems out as soon as possible
 - we use many defect control gates for each delivery
 - Code and test peer reviews and walkthroughs
 - Code Merge Reviews and Integration Tests
 - Test Turnovers and Test Readiness Reviews
 - Consent To Ship and Pre-Ship Reviews
 - our control gates are not true zero defect gates, but quality criteria are associated with each gate

Make it possible for others to contribute!



- Establish ways to leverage the community's expertise
 - our Operations Support Software initiative allows others to add non-core components to the baseline system
 - defines delivery and documentation requirements
 - supports configuration management and delivery
 - addresses cross-element problem management
 - many operations-oriented tools are being added to the baseline through this initiative

Agenda

- Background - Dawn Lowe
- System Overview - Mike Moore
- Program/Project Management Lessons Learned - Curt Schroeder
- System Development Lessons Learned - Mike Moore
- è More System Development Lessons Learned - Steve Fox
- Questions and Answers

A Few COTS is Good... but Lots of COTS can be a Nightmare



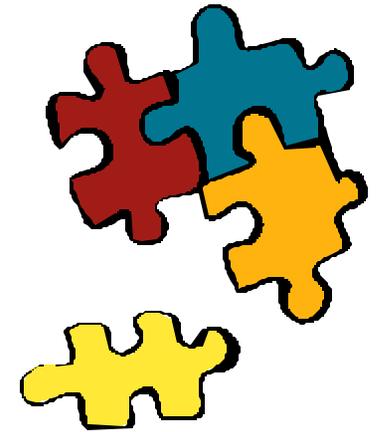
- We integrated over 50 COTS products
 - significant integration and maintenance effort
 - analyze 400 candidate vendor patches per year
 - make 40 COTS upgrades per year
- COTS procurement lessons
 - always fly before you buy
 - never pay the vendor to modify a COTS product for you
 - never allow developers to modify a COTS product
 - adapt your requirements to COTS capabilities
 - you lose leverage if you buy everything up front
 - don't push the envelop of a COTS product's capabilities
 - stay away from bleeding edge products

A Few COTS is Good... but Lots of COTS can be a Nightmare (Cont'd)



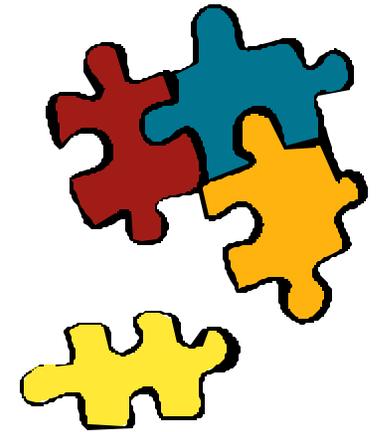
- COTS upgrade lessons
 - wrapping COTS to insulate yourself from their changes reduces risk
 - hardware and software system baselines become obsolete and require upgrading; get over it
 - put together a COTS upgrade plan
 - understand dependencies between COTS
 - make sure you understand the vendor's upgrade schedule and vice versa
 - invest the time to interact with vendor's senior management

Shortcuts in Software CM Will Lead to Chaos When the Rubber Hits the Road



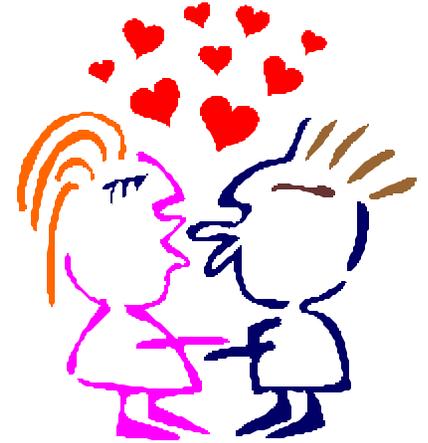
- It's easy to underestimate what's required on a large system with multiple heterogeneous sites
 - We manage 1.1 million lines of custom software (~12,000 components)
 - Support concurrent updates and daily builds against 3 baselines
 - Maintain system build scripts (8,000 lines of code)
 - 8 system deliveries per year
 - 180 patch deliveries per year
- Ensure you have sufficient computing resources and staff expertise for the software CM function

Shortcuts in Software CM Will Lead to Chaos When the Rubber Hits the Road (Cont'd)



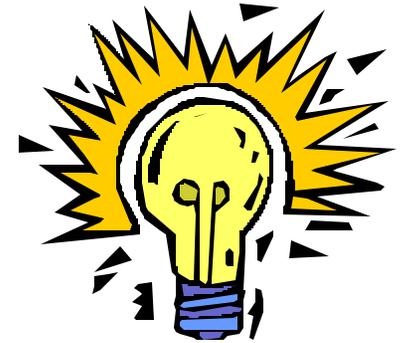
- Whatever you do... don't break the build
 - We've defined a detailed merge process that requires pre-integration of new elements before code may be merged to the baseline
 - Merge justifications are required with all merges to each baseline
 - Peer pressure becomes a strong incentive to carefully check changes before merging

By All Means, KISS if
You Want to be Happy



•

Food for Thought



- Albert, Ceci and Ed Morris, *COTS: Lessons We Continue to Learn*, SEI SE 2000 Conference, Washington D. C., September, 2000.
- Bass, Len, Paul Clements and Rick Kazman, *Software Architecture in Practice*, Addison Wesley Longman, Inc., Reading Massachusetts, 1998.
- Boehm, Barry, *... And Very Few Lead Bullets, Either*, SEI SE 2000 Conference Keynote Talk, Washington D. C., September, 2000.
- Highsmith, James, *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*, Dorset House Publishing Co., New York, New York, 2000.
- McCarthy, Jim, *Dynamics of Software Development*, Microsoft Press, Redmond, Washington, 1995.
- Raymond, Eric, *The Cathedral & The Bazaar*, O'Reilly and Associates, Inc. Sebastopol, CA, 1999.
- Semelsberger, Robert, et al, *Elephant Bungee Jumping: A Software Project Manager's Survival Guide*, National Reconnaissance Office, 1999.
- Yourdan, Edward, *Death March: Managing "Mission Impossible" Projects*, Prentice-Hall, Inc., Upper Saddle River, New Jersey, 1997.

Agenda

- Background - Dawn Lowe
 - System Overview - Mike Moore
 - Program/Project Management Lessons Learned - Curt Schroeder
 - System Development Lessons Learned - Mike Moore
 - More System Development Lessons Learned - Steve Fox
- è Questions and Answers