Software Engineering in the Aerospace Industry

Why it is so hard

CSCI 5828 - Spring 2010
by Jerel Moffatt

Overview

- GAO Study - Large scale project assessment
- Standish Group CHAOS Study
- Difficulties specific to the aerospace industry
- Agile methods vs heavy waterfall methods
- Where do we go from here

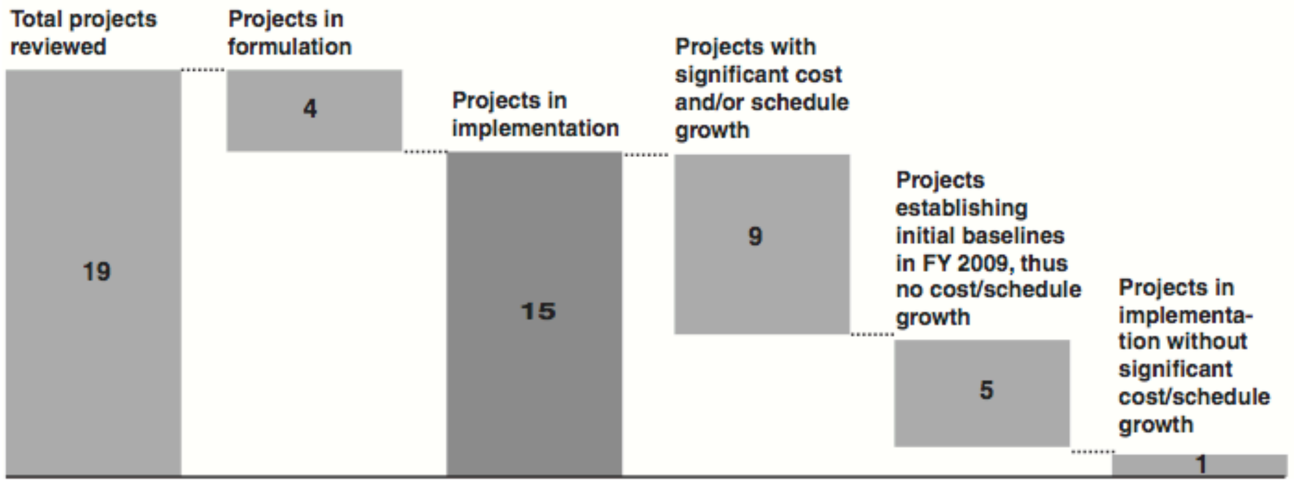
Disclaimer

- Will focus on projects with NASA as customer
- NASA is not the bad guy
    - NASA has done an uncountable number good things
    - Software engineering is ‘just hard’

GAO Study

- In February 2010, the United States Government Accountability Office (GAO) did an assessment on selected large scale NASA projects
  - GAO assessed 19 NASA projects with a combined life-cycle cost of more than $66 billion
- Findings were that most missions were over budget, and behind schedule

Figure 1: Summary of Projects Assessed by Phase of the NASA Project Life Cycle



Source: GAO analysis of NASA project data.

- Of those 19 projects, 4 are still in the formulation phase where cost and schedule baselines have yet to be established, and 5 just entered the implementation phase in fiscal year 2009 and therefore do not have any cost and schedule growth.

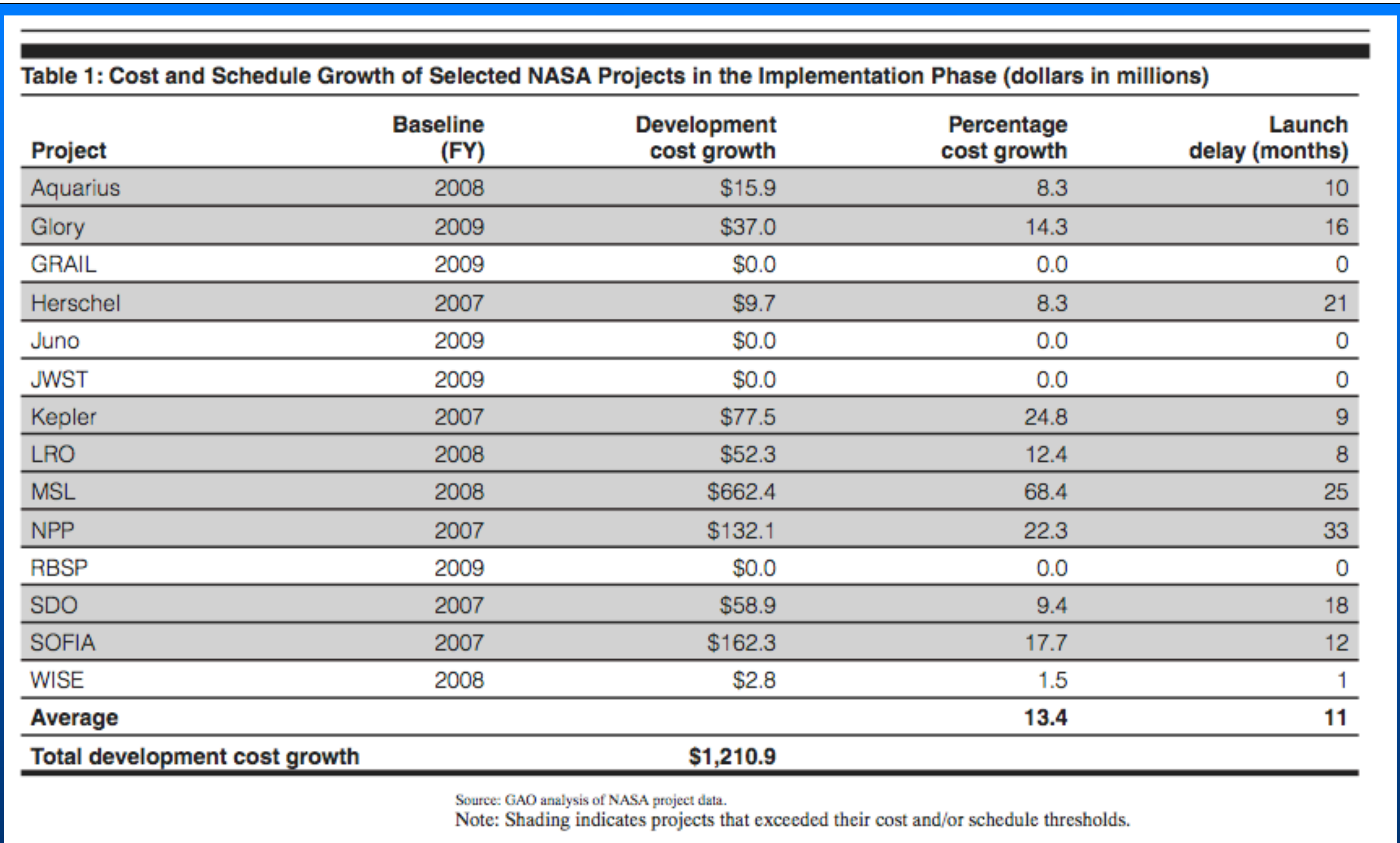


Table 1: Cost and Schedule Growth of Selected NASA Projects in the Implementation Phase (dollars in millions)

Project	Baseline (FY)	Development cost growth	Percentage cost growth	Launch delay (months)
Aquarius	2008	$15.9	8.3	10
Glory	2009	$37.0	14.3	16
GRAIL	2009	$0.0	0.0	0
Herschel	2007	$9.7	8.3	21
Juno	2009	$0.0	0.0	0
JWST	2009	$0.0	0.0	0
Kepler	2007	$77.5	24.8	9
LRO	2008	$52.3	12.4	8
MSL	2008	$662.4	68.4	25
NPP	2007	$132.1	22.3	33
RBSP	2009	$0.0	0.0	0
SDO	2007	$58.9	9.4	18
SOFIA	2007	$162.3	17.7	12
WISE	2008	$2.8	1.5	1
Average			**13.4**	**11**
Total development cost growth		**$1,210.9**		

Source: GAO analysis of NASA project data.
Note: Shading indicates projects that exceeded their cost and/or schedule thresholds.

- 9 of the 10 projects that have been in the implementation phase for several years experienced cost growth ranging from 8 to 68 percent, and launch delays of 8 to 33 months, in the past 3 years.

- These 10 projects had average development cost growth of almost $121.1 million—or 18.7 percent—and schedule growth of 15 months, and a total increase in development cost of over $1.2 billion, with over half of this total—or $706.6 million—occurring in the last year.

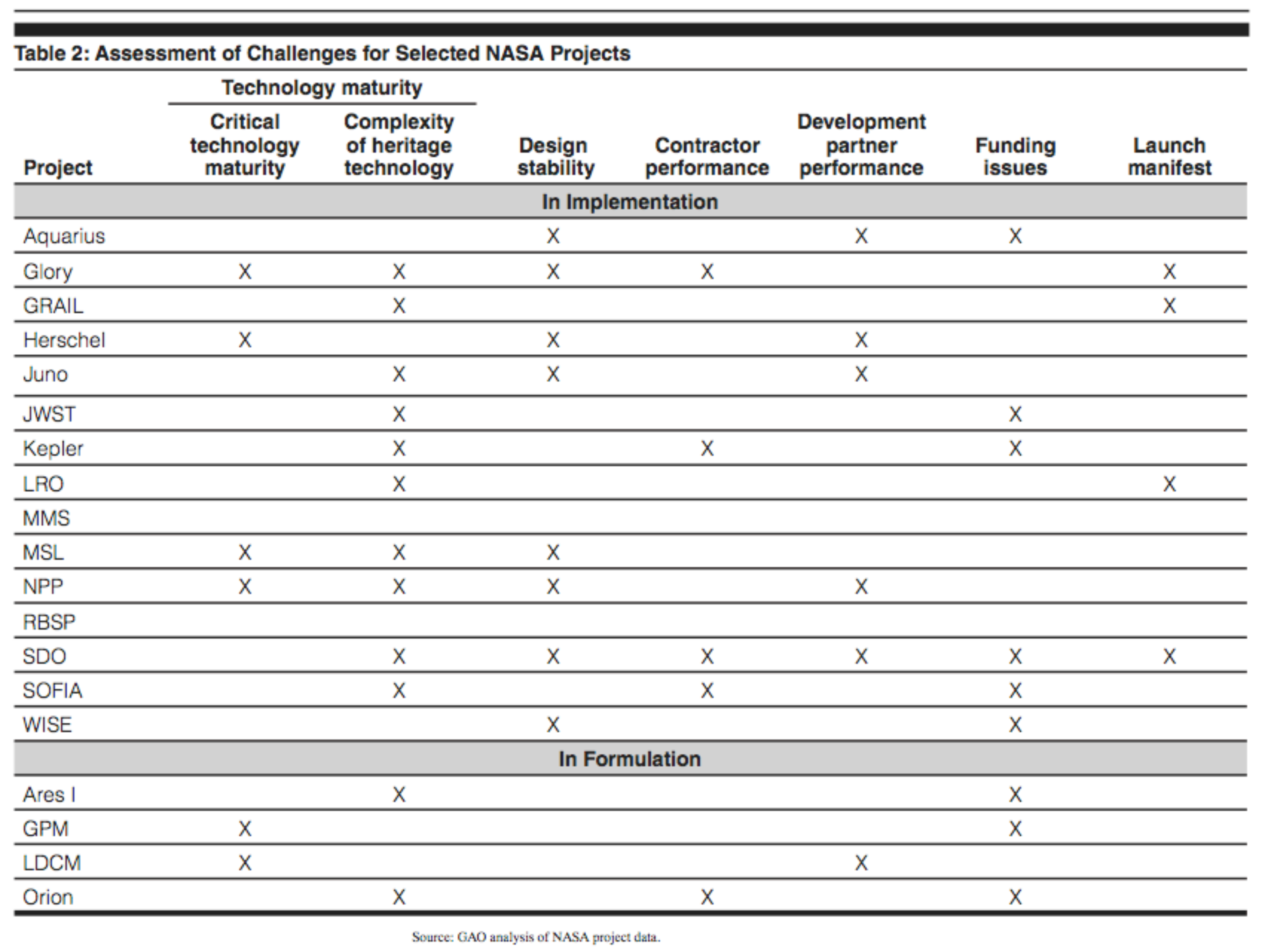
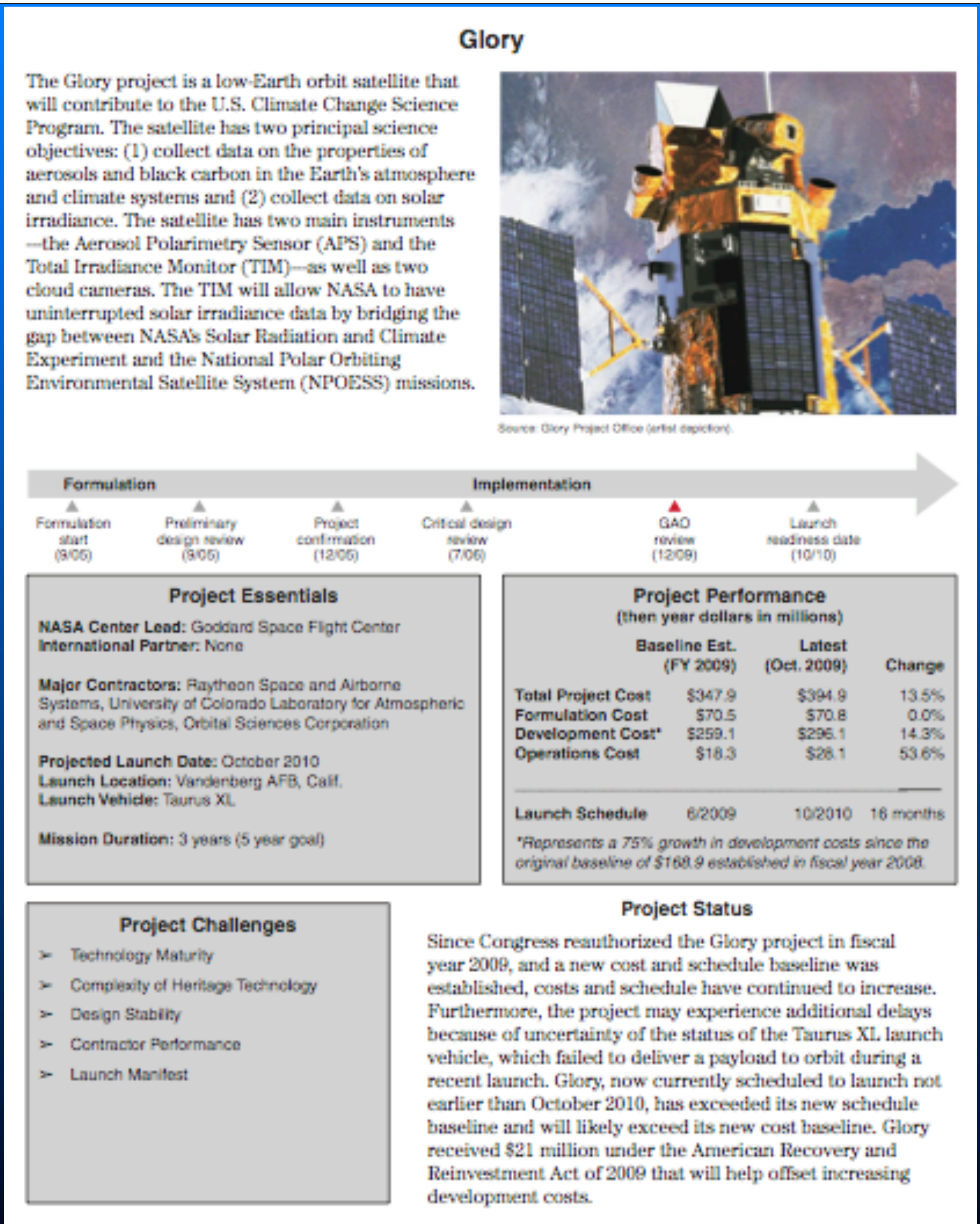


Table 2: Assessment of Challenges for Selected NASA Projects

	Technology maturity						
Project	Critical technology maturity	Complexity of heritage technology	Design stability	Contractor performance	Development partner performance	Funding issues	Launch manifest
In Implementation							
Aquarius			X		X	X	
Glory	X	X	X	X			X
GRAIL		X					X
Herschel	X		X		X		
Juno		X	X		X		
JWST		X				X	
Kepler		X		X		X	
LRO		X					X
MMS							
MSL	X	X	X				
NPP	X	X	X		X		
RBSP							
SDO		X	X	X	X	X	X
SOFIA		X		X		X	
WISE			X			X	
In Formulation							
Ares I		X				X	
GPM	X					X	
LDCM	X				X		
Orion		X		X		X	

Source: GAO analysis of NASA project data.



Glory

The Glory project is a low-Earth orbit satellite that will contribute to the U.S. Climate Change Science Program. The satellite has two principal science objectives: (1) collect data on the properties of aerosols and black carbon in the Earth's atmosphere and climate systems and (2) collect data on solar irradiance. The satellite has two main instruments —the Aerosol Polarimetry Sensor (APS) and the Total Irradiance Monitor (TIM)—as well as two cloud cameras. The TIM will allow NASA to have uninterrupted solar irradiance data by bridging the gap between NASA's Solar Radiation and Climate Experiment and the National Polar Orbiting Environmental Satellite System (NPOESS) missions.

Source: Glory Project Office (artist depiction).

Formulation

Implementation

Formulation start (9/05)

Preliminary design review (9/05)

Project confirmation (12/05)

Critical design review (7/06)

GAO review (12/09)

Launch readiness date (10/10)

Project Essentials

NASA Center Lead: Goddard Space Flight Center
International Partner: None

Major Contractors: Raytheon Space and Airborne Systems, University of Colorado Laboratory for Atmospheric and Space Physics, Orbital Sciences Corporation

Projected Launch Date: October 2010
Launch Location: Vandenberg AFB, Calif.
Launch Vehicle: Taurus XL

Mission Duration: 3 years (5 year goal)

Project Performance
(then year dollars in millions)

	Baseline Est. (FY 2009)	Latest (Oct. 2009)	Change
Total Project Cost	$347.9	$394.9	13.5%
Formulation Cost	$70.5	$70.8	0.0%
Development Cost*	$259.1	$296.1	14.3%
Operations Cost	$18.3	$28.1	53.6%
Launch Schedule	6/2009	10/2010	16 months

*Represents a 75% growth in development costs since the original baseline of $168.9 established in fiscal year 2008.

Project Challenges

- Technology Maturity
- Complexity of Heritage Technology
- Design Stability
- Contractor Performance
- Launch Manifest

Project Status

Since Congress reauthorized the Glory project in fiscal year 2009, and a new cost and schedule baseline was established, costs and schedule have continued to increase. Furthermore, the project may experience additional delays because of uncertainty of the status of the Taurus XL launch vehicle, which failed to deliver a payload to orbit during a recent launch. Glory, now currently scheduled to launch not earlier than October 2010, has exceeded its new schedule baseline and will likely exceed its new cost baseline. Glory received $21 million under the American Recovery and Reinvestment Act of 2009 that will help offset increasing development costs.

- Example - GLORY
- Initial baseline cost at $168.9
- Cost went way over, and was re-baselined at $259.1
- Latest cost Oct. 2009 at $296.1 which is a 14.3% increase since the re-baseline, and a total increase of 75% !

Cobb’s Paradox

- *“We know why projects fail, we know how to prevent their failure -- so why do they still fail?”*
  - - Martin Cobb Treasury Board of Canada Secretariat , Ottawa, Canada
- Standish Group CHAOS Study
- 10 Reasons Projects are late, and/or over budget relating to software development

Success Criteria

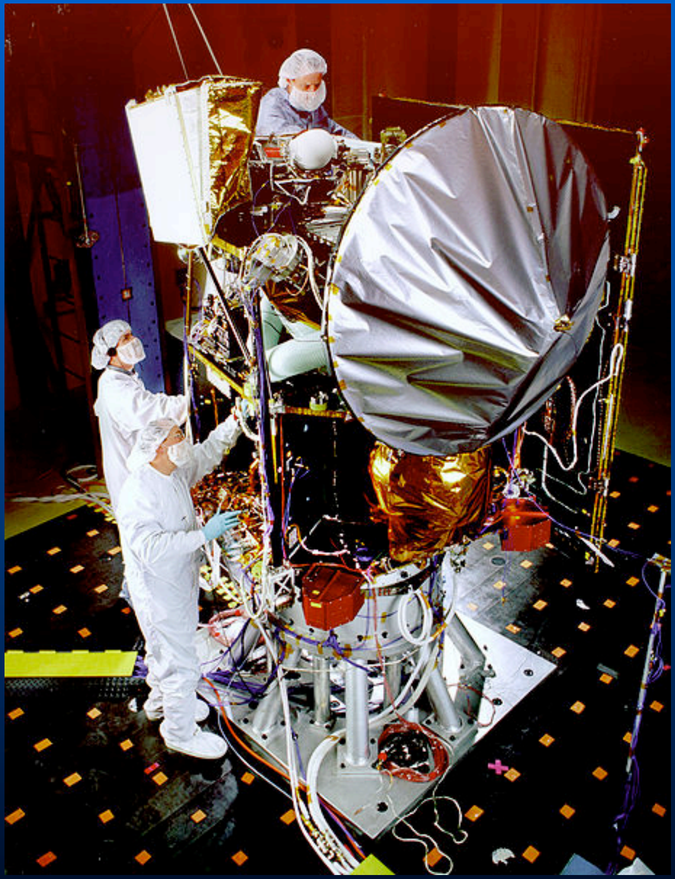
1. User Involvement -19
2. Executive Management Support -16
3. Clear Statement of Requirements -15
4. Proper Planning -11
5. Realistic Expectations -10
6. Smaller Project Milestones -9
7. Competent Staff - 8
8. Ownership - 6
9. Clear Vision & Objectives - 3
10. Hard-Working, Focused Staff - 3

TOTAL 100

User Involvement

- Typically, user involvement consists of:
  - engineers, operators, and scientists who are located in separate facilities, separate states, or even separate countries
- Long distance can hamper communication paths and thus iterative user involvement
  - Often software developers never even meet some of the users
- Politics often plays a role in user involvement, and not for the better
  - Mars Climate Orbiter mission failed due to famous unit conversion error

Mars Climate Orbiter



- Mars Climate Orbiter 1998
- Failed due to unit conversion error
  - Imperial units vs metric
- Caused spacecraft to enter Mars orbit at 57 km instead of the planned 140-150 km orbit
- Drag likely destroyed the vehicle
- Communication errors during testing (between contractor) likely a root cause of this failure

Executive Support & Ownership

- Failure consequences are unacceptable
  - Budgets are high, tax dollars are at stake
  - Projects failing badly have to explain themselves before congress
- Does the project team have a stake?
  - Often there are no direct incentives for success by individual developers
  - Contractors may have bonuses, but largely, the business if funded by government money and there are no 'profits' to share if success is ample
- Who is the boss?
  - Isn't always clear who is in charge sine NASA provides oversight and teams also have dedicated executives

Clear Statement of Requirements

- NASA is big on requirements
  - Insists on formally documented, tracked, and verified requirements
- But requirements are not flexible, or iterative
  - Requirements are viewed by NASA as being solidified up front via a waterfall process definition
  - Does not anticipate and/or allow for iterative requirements
  - Makes it difficult for a contracted organization to implement their own Agile software process beneath

Proper Planning & Milestones

- Schedules are ‘success oriented’
  - But often failure to meet schedule date is unacceptable
  - Launch date may be physically unable to move due to planetary alignments
  - Launch date may be extremely costly to move due to schedules of launch pad activities and launch vehicles
- Does not follow 80/20 rule
  - “20% of a project's features will provide 80% of user benefits”
  - Missions are built on meeting science (or other mission) objectives not on benefits to the user which may be ‘negotiable’. Everything is decided up-front
  - Fixes or updates or improvements post launch are often unacceptable or impossible

Testing Schedule Compromised

- Due to inability for milestones to slip, there are bound to be consequences
- Testing cut short
  - Integration tests merged with system tests
  - System tests merged with acceptance tests
- Impact of adding requirements understated, and impact of removing requirements is under analyzed

Code Bases

- Facilities usually develop software for more than one mission
- Missions have overlap in code, but not necessarily in process definition or rigor
- Changing the code for mission A may also change it for mission B
    - Different missions have different testing, documenting, and verification processes
    - Leads to interesting and sometimes costly configuration management concerns

Conclusion

- So we know that projects are over budget and behind schedule
- And we know why projects are over budget and behind schedule
- This is not a unique problem to aerospace software
  - Most all software projects have similar concerns, perhaps not as strict

What would NASA do?

- There are two extremes to the pendulum
  - More strict bloated process, more meticulous and verbose documentation
  - Streamlined (Agile) processes, only documenting that which provides added benefit
- Over the past decade, NASA has been leaning towards the former solution, not the latter
  - NASA has adopted CMMI to measure their process model
  - In itself, this does not imply a heavier process, but can add to cumbersome oversight when CMMI is attempted to be flowed down to other organizations who might do things differently

Agile Methods

- Agile software development focuses on iterative development
- Uses constant small measurements to evaluate and adapt
- Teams focus on meeting functionality requirements based on user stories and iterative user involvement
- Agile doesn't mean less process, it means more focused process

Agile Methods in Aerospace

- Since NASA executes projects in a waterfall process, working Agile methods beneath is difficult
  - NPR 7150.2 - standards NASA levies on all contracted organizations
  - Defines much of the processes, documentations, and plans that NASA requires for all work on a given project
  - Requirements are due and finalized up front, prior to completing the design, much less actual implementation of the design
    - Specific documentation is often required
      - Keeping user stories on note cards may not be acceptable and would have to be done in addition to other redundant efforts of tracking work

Added Risk?

- NASA may be reluctant to adopting more Agile methods due to a perception of added risk
- Peter Norvig quotes NASA administrator Don Goldin as saying:

> We've got to do the better, faster, cheaper. These space missions cost too much. It'd be better to run more missions and some of them would fail but overall we'd still get more done for the same amount of money.

- Coders at Work (by Peter Seibel)

Where are we going from here?

- Added complexity and weight to the development process does not appear to be helping
- As NASA begins to work with more commercial organizations to develop software, cost savings could be found by allowing them to develop the software under their own methods
  - A delicate balance between managing the outcome of the software with managing the development of the software must be achieved
  - Always keeping in mind, there is 'No Silver Bullet' (Fred Brooks) that will suddenly make software engineering in aerospace an 'easy' task