



# Next Generation Agile for Space Applications

**Brendan O'Connor**

---

---

Presented to NASA GSFC IS&T Colloquium

Nov. 6, 2013

# Brendan O'Connor

[brendan.oconnor@emergentspace.com](mailto:brendan.oconnor@emergentspace.com)

Chief Systems Architect  
Emergent Space Technologies



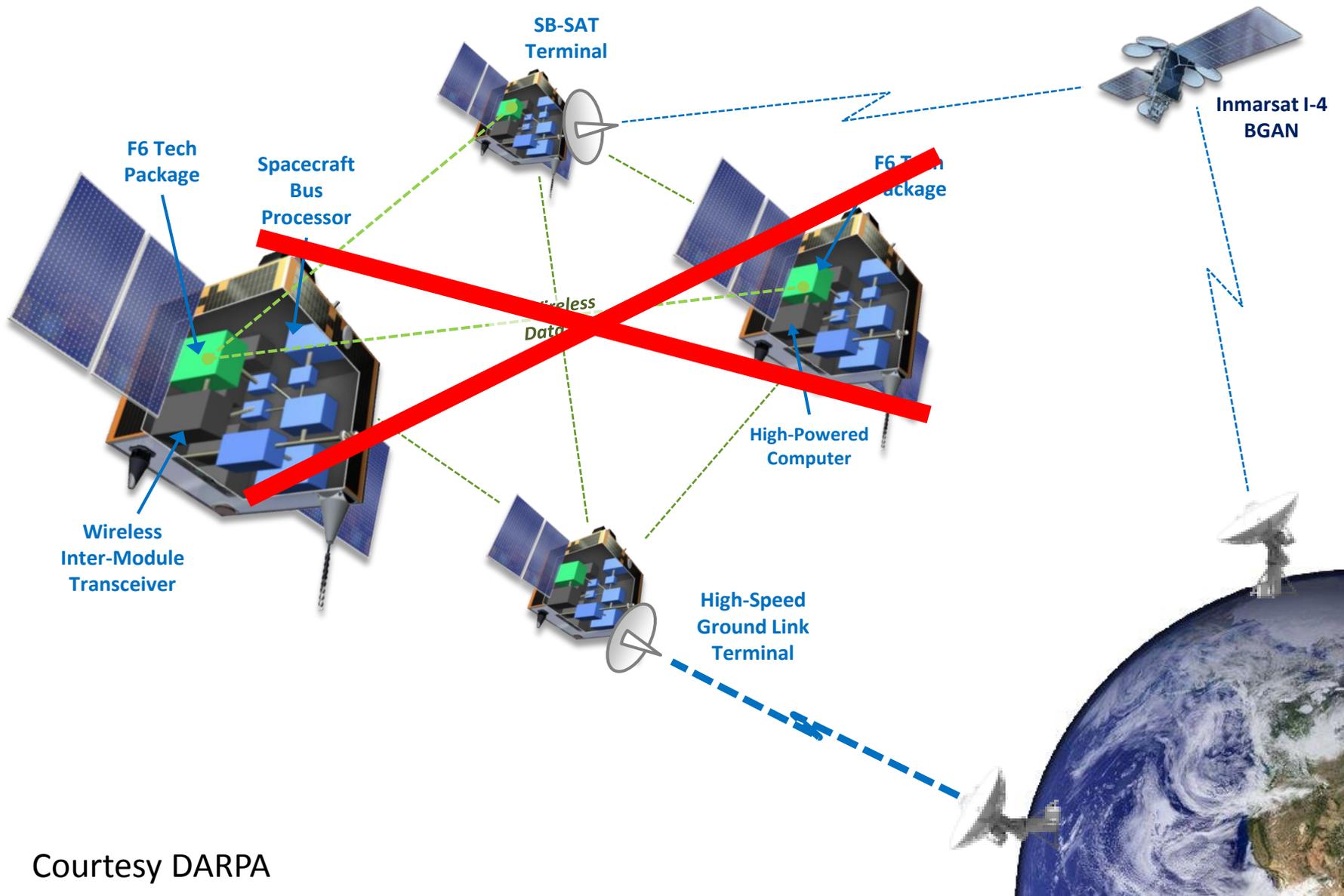
- MS Aerospace Engineering-UTexas
- Software Developer
- Software Architect
- 7 years working for Boeing Space
- 8 years experience as CTO of a Publicly-traded Casino Software Company
- 15+ years experience in Space System Software
- Longtime striver for better software engineering
- Led Emergent's initial efforts to become CMMI L3 Certified
- System Architect for Emergent's System F6 Software for DARPA



# Irate Casino Customers Can be Highly Unpredictable



# Notional System F6 On-Orbit Demo



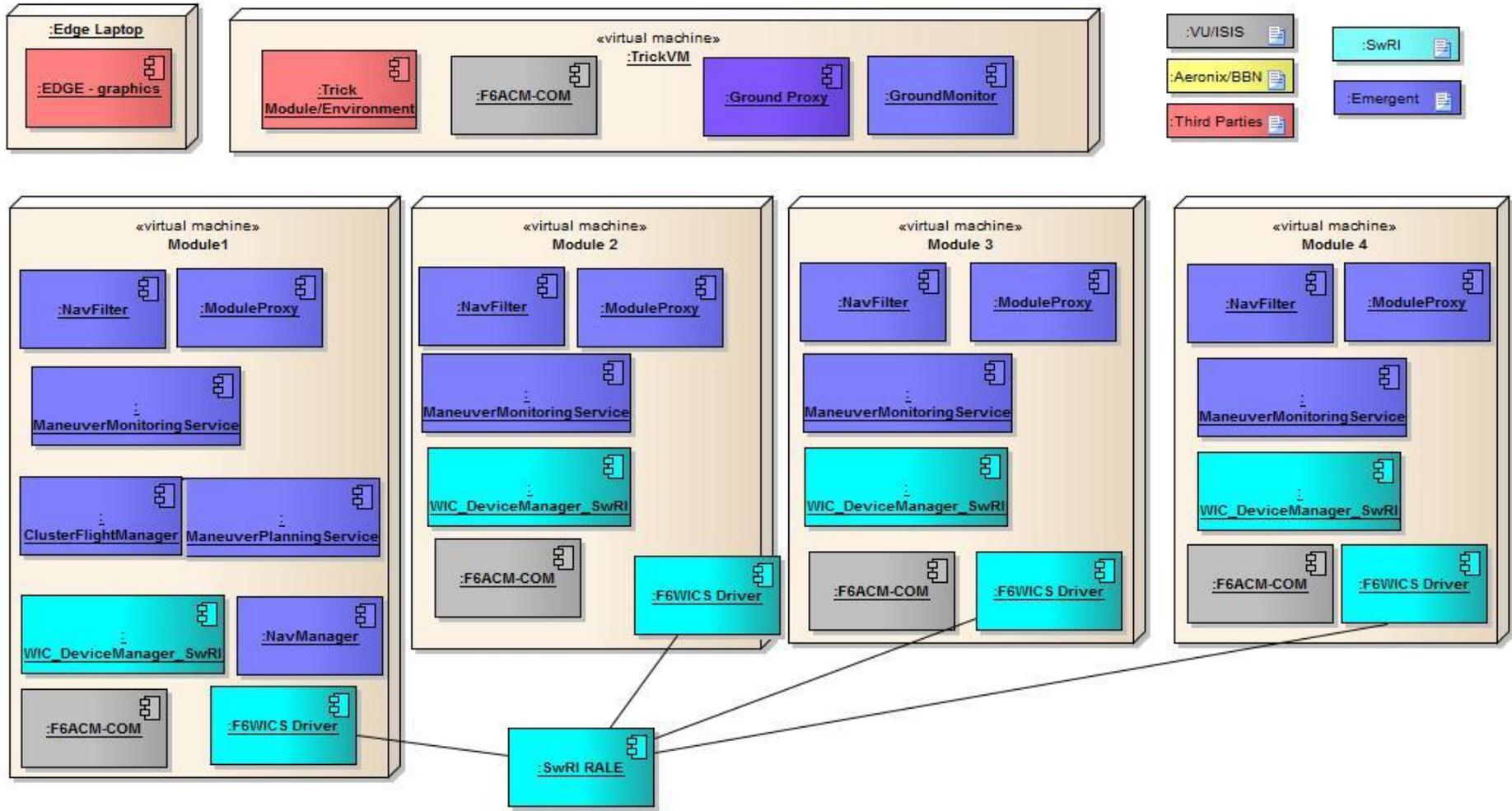
Courtesy DARPA

# The F6 Cluster Flight Software

- Emergent survived a four-team shoot-out to get the Development Contract
- The CFA is a Distributed Multi-spacecraft application providing:
  - Guidance
  - Navigation
  - High-level Control
- Model-based development
- Matlab/C/C++
- Service-based architecture
- Message oriented Middleware
- Multi-year prime contract with DARPA (through NASA-ARC)
- Now targeted for the cFE

# F6 Software Deployment

deployment PI-9 Deployment-SwRI



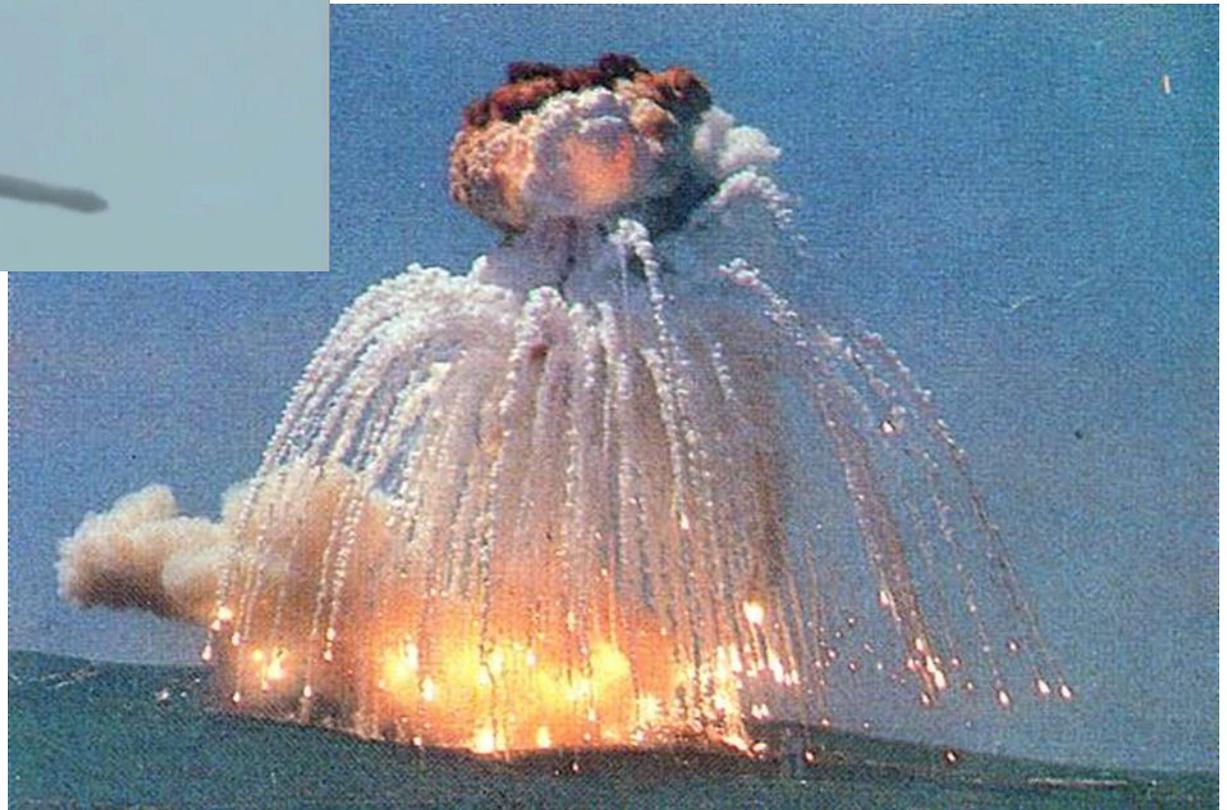
# Have you heard the news?



# Agile is the hot new trend!!

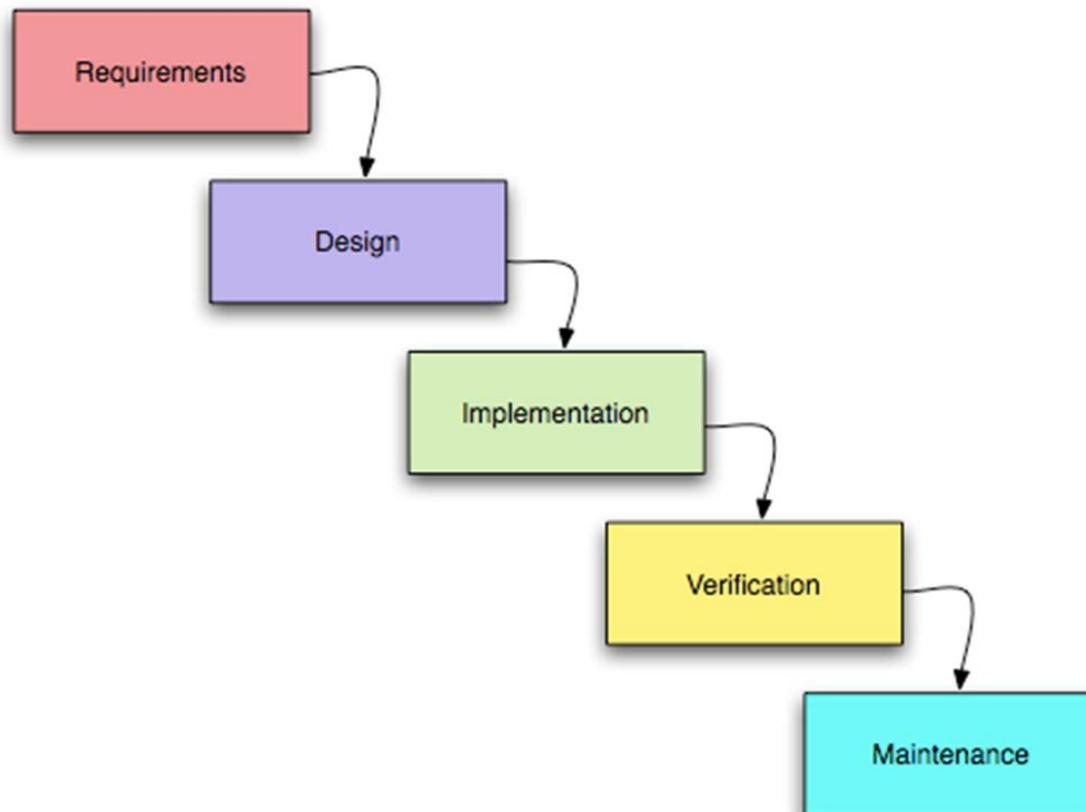
# Latest Software Development Fad?

## What could possibly go wrong?

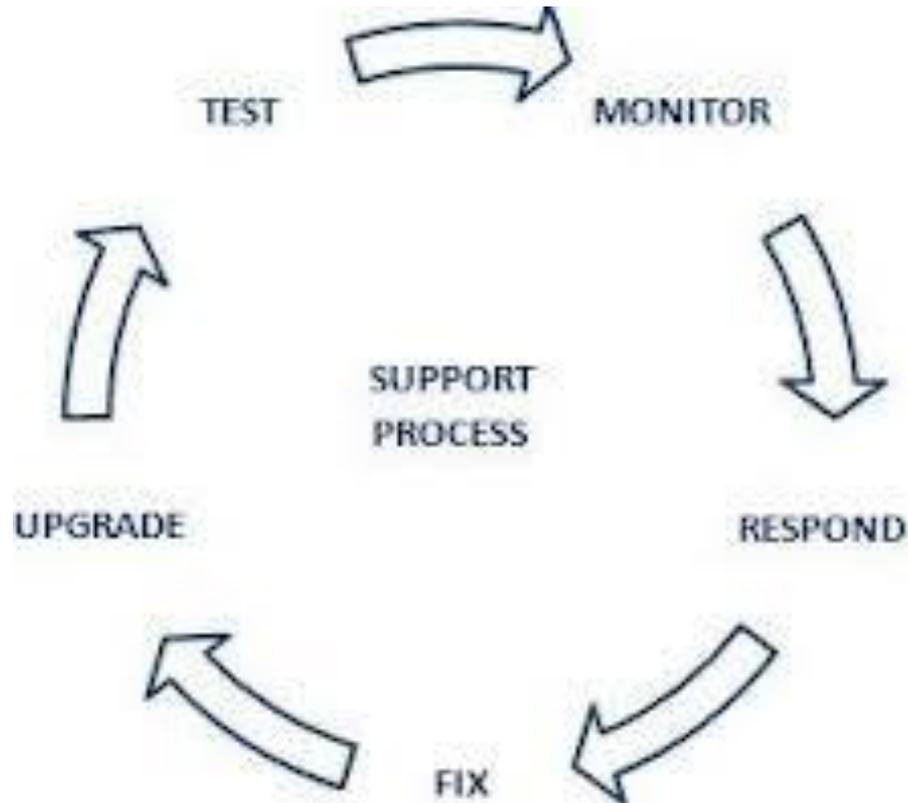


# Why Even Consider Agile for Aerospace Applications?

Lots of good software was written Waterfall.



# Maintenance Lifecycle



# Software Projects Don't Have a Good Track Record

- 25% of all software projects fail outright (No useful software developed).
  - Version One website
- 5 to 15% of all IT projects are abandoned as hopelessly inadequate.
  - IEEE Spectrum 9/2005

## Why do Software projects fail so often?

Among the most common factors:

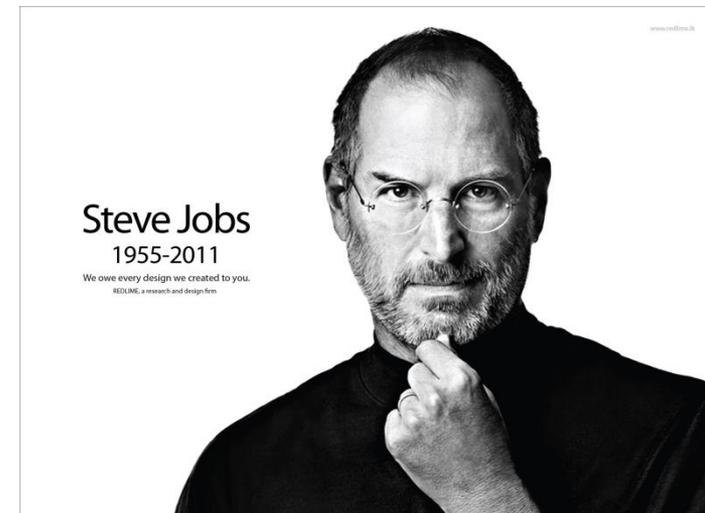
- Unrealistic or unarticulated project goals
- Inaccurate estimates of needed resources
- Badly defined system requirements
- Poor reporting of the project's status
- Unmanaged risks
- Poor communication among customers, developers, and users
- Use of immature technology
- Inability to handle the project's complexity
- Sloppy development practices
- Poor project management
- Stakeholder politics
- Commercial pressures



# Software Can be Better

*“You’re looking at it wrong. You’re looking at it as a hardware person in a fragmented world. You’re looking at it as a hardware manufacturer that doesn’t really know much about software, who doesn’t think about an integrated product, but assumes the software will somehow take care of itself...And you assume that the software will somehow just come alive on this product that you’re dreaming of, but it won’t.”*

-Steve Jobs, Oct. 2010



# Personally Instructive Experience

---

- On a team that developed a multi-satellite mission planning tool
- Many satellites
- Many ground stations
- 35 Developers/testers
- Multi-year development with an iteration every year or so
- State of the art GUI
- State-of-the-art Planning algorithm
- Essentially replaced by a spreadsheet written by one of the operators within 6 months of going operational

# Why did it miss the mark?

- There were essentially 5 use cases:
  - Input requests from users for activities that used resources and their constraints.
  - About Once a Day, Generate a plan for the ground passes for the constellations maximizing the number of satisfied activity requests. Take the time to search for the optimal solution.
  - Occasionally Replan the existing, distributed plan
  - Distribute this plan to the TT&C
  - Monitor the satisfied Activities
  
- The Problem was:
  - We treated all Use Cases as equals and they weren't
  - When replans happened there was frequently a crisis
    - No one cared about an optimal solution, they wanted a fast solution

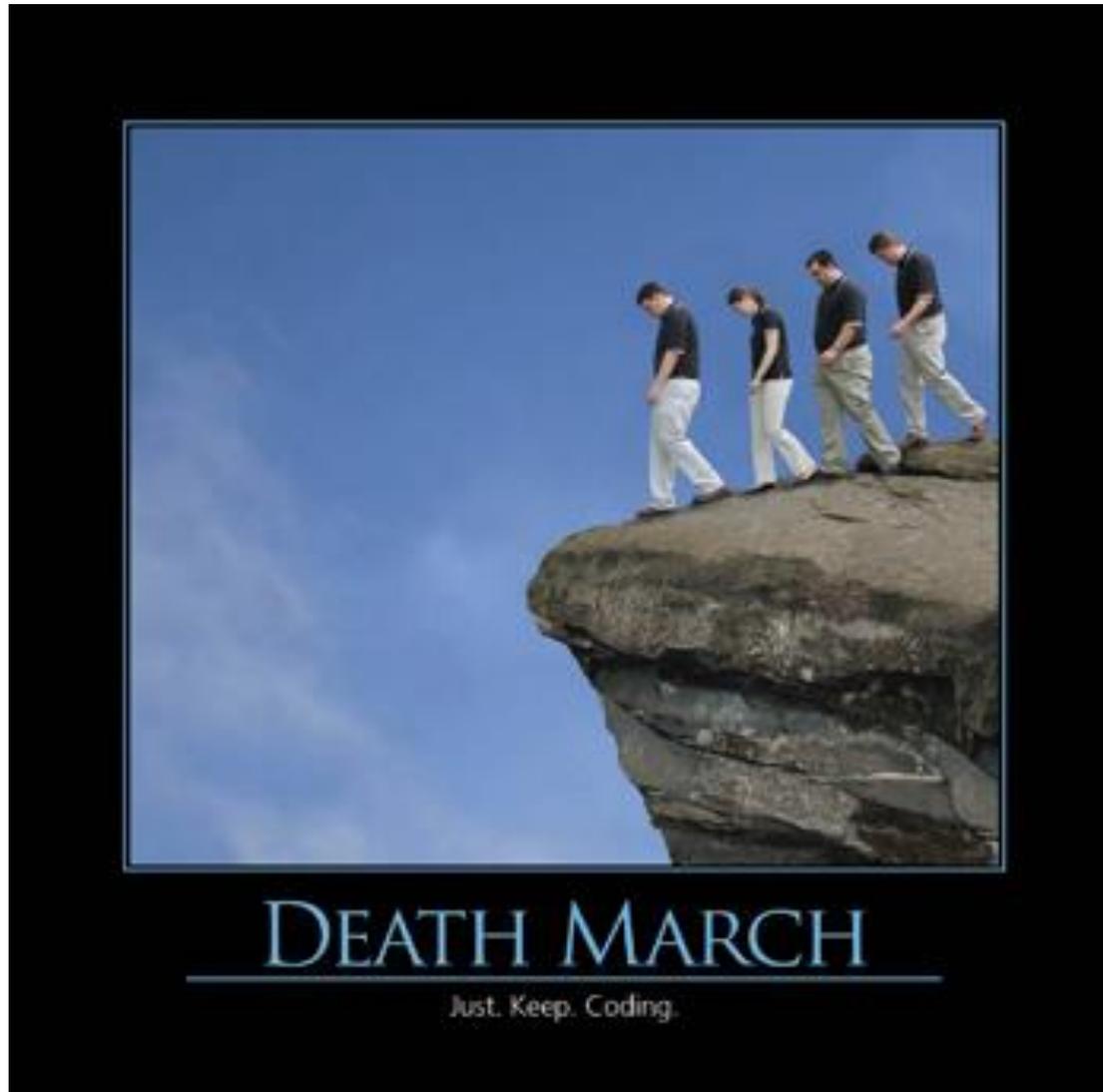
# Management Recognizes There's a Problem, They're Just Not Sure What it Is



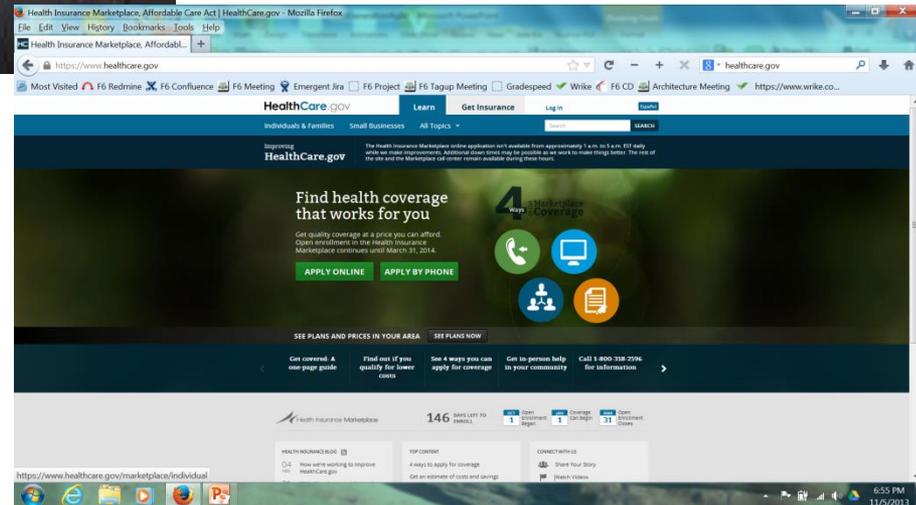
## ■ Actual comments I've heard from Executives

- “The engineers produced the estimates so they should commit to covering the differences.”
- “They should be happy just to have jobs.”
- “You’ve been a software engineer for 15 years, why can’t you accurately predict how long it will take to fix this?”
- “Software causes the bulk of the issues we have in the field so why should we provide incentives to the programmers?”
- “Why does everything take so long?”
- “Can’t you just cut the test time?”
- “When I get here in the morning there are no cars and when I leave in the evening there are no cars in the parking lot. Where are the engineers?”

# For Software Developers, too many projects ended up like this...



# We've all experienced the pain of faulty software



# How do we fix these problems?



- The first step is recognizing you have a problem

## Manifesto for Agile Software Development

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

# What is Agile Software Development?

- A group of [software development methods](#) based on [iterative and incremental development](#).
    - Where requirements and solutions evolve through collaboration between self-organizing, [cross-functional teams](#).
    - It promotes adaptive planning, evolutionary development and delivery, a [time-boxed](#) iterative approach, and encourages rapid and flexible response to change.
    - It is a conceptual framework that promotes foreseen interactions throughout the development cycle.
- Wikipedia

# Popular Agile Methods

---

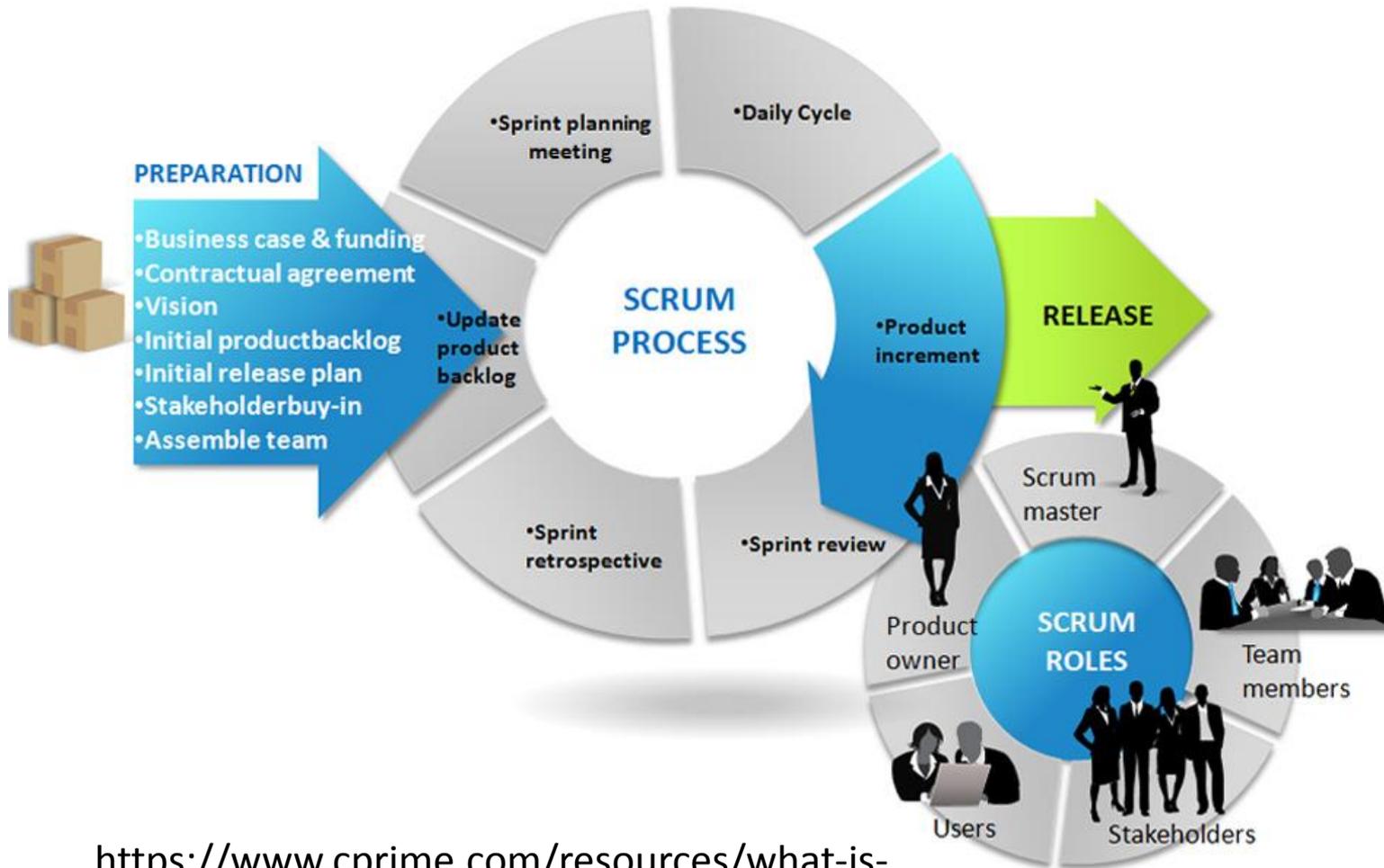
- Extreme Programming
- Kanban
- Lean Software Development
- Scrum

# Emergent Implemented Scrum Process

- A product owner creates a prioritized wish list called a product backlog.
- During sprint planning, the team pulls a small chunk from the top of that wish list, a sprint backlog, and decides how to implement those pieces.
- The team has a certain amount of time — a sprint (usually two to four weeks) — to complete its work, but it meets each day to assess its progress (daily Scrum).
- Along the way, the ScrumMaster keeps the team focused on its goal.
- At the end of the sprint, the work should be potentially shippable: ready to hand to a customer, put on a store shelf, or show to a stakeholder.
- The sprint ends with a sprint review and retrospective.
- As the next sprint begins, the team chooses another chunk of the product backlog and begins working again.

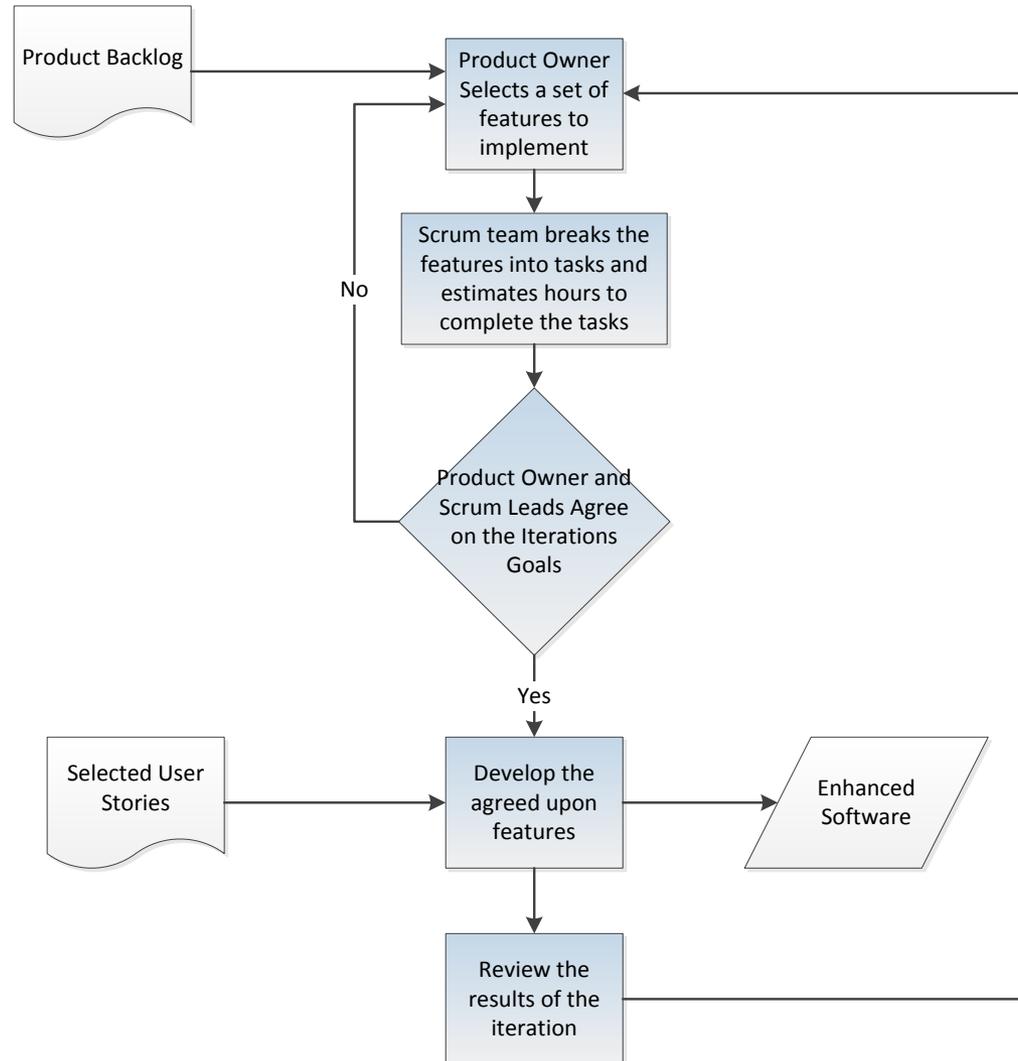
# The Scrum Cycle

## SCRUM PROCESS

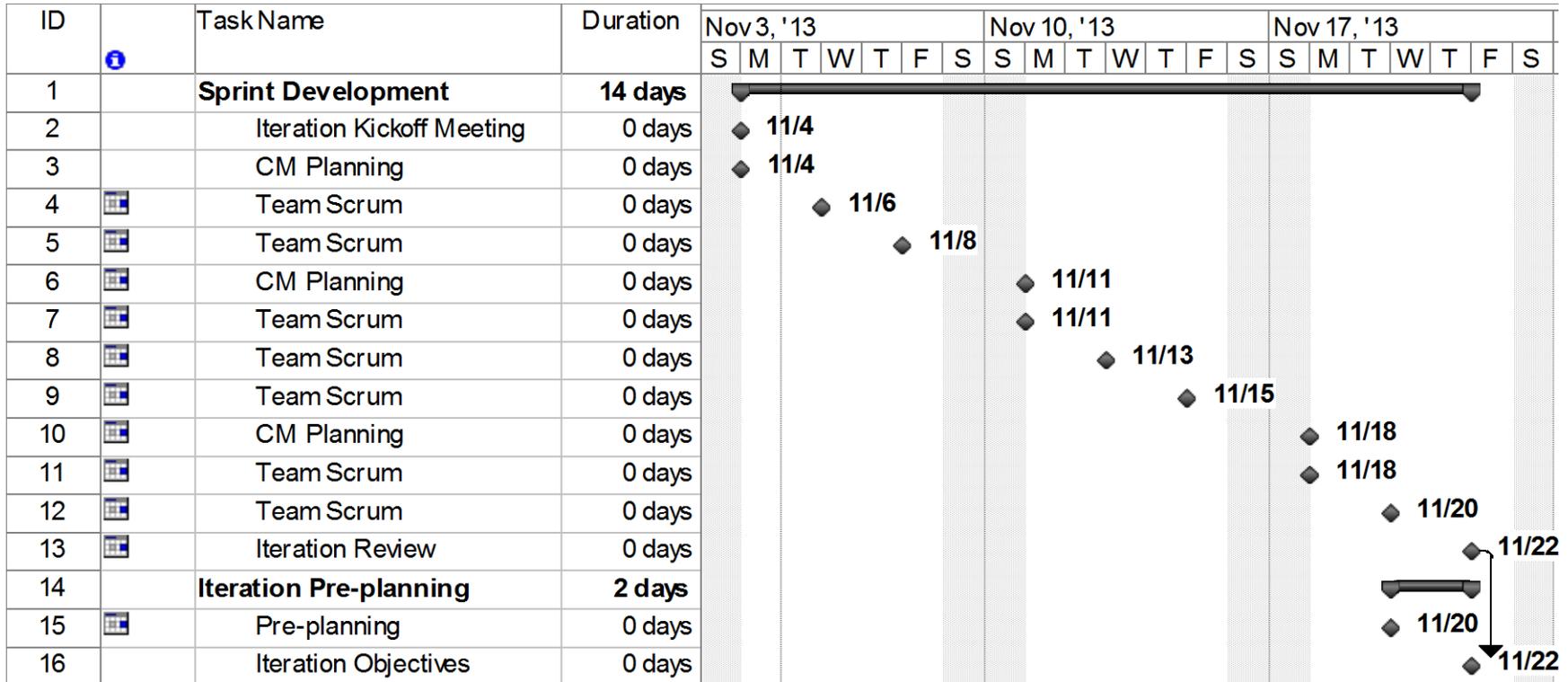


<https://www.cprime.com/resources/what-is-agile-what-is-scrum/>

# Sprint Cycle Detail



# Typical Iteration Schedule



# Schedulable Items

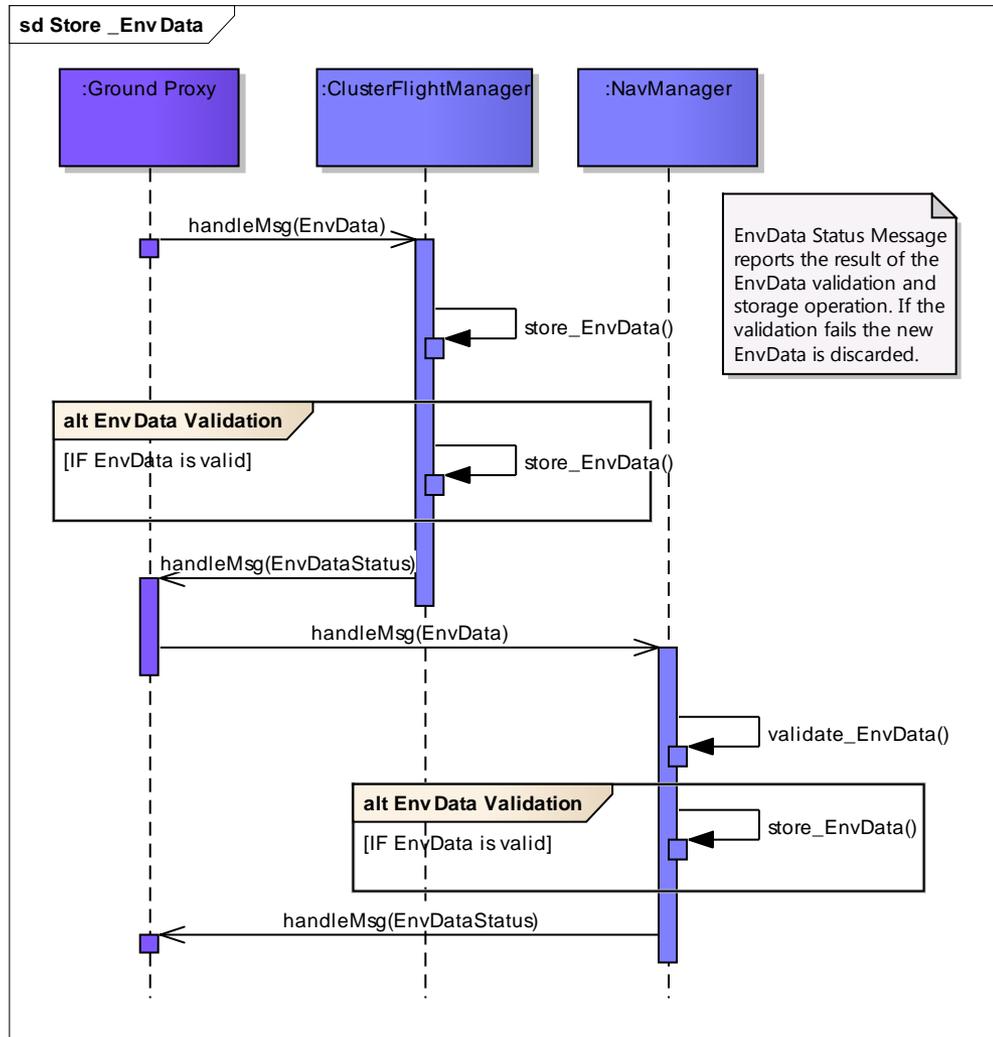
- Action Items
  - Tasks
- Defects
  - Tasks
- Epics
  - User Stories
    - Tasks

# User Stories – the development process drivers



- User Stories are a Requirements Description Technique
- A User Story is a short, simple description of a feature told from the user's perspective
  - Helps to maintain the focus on the customer
- A good user story takes the form:
  - As a [role] I want [some goal] so that [some reason]  
    “As a propulsion system engineer I want to monitor tank temperatures so that I can manually trigger heater cycles”
  - This is a problem for automated software so we adapted them: An X message is received by Y component resulting in Z
- Includes the success criteria
  - May evolve over time
- Writing user stories is a learned technique
- Estimating tasks for user stories is also a learned technique

# Adapting User Stories



# The Product Backlog

## ToDo List

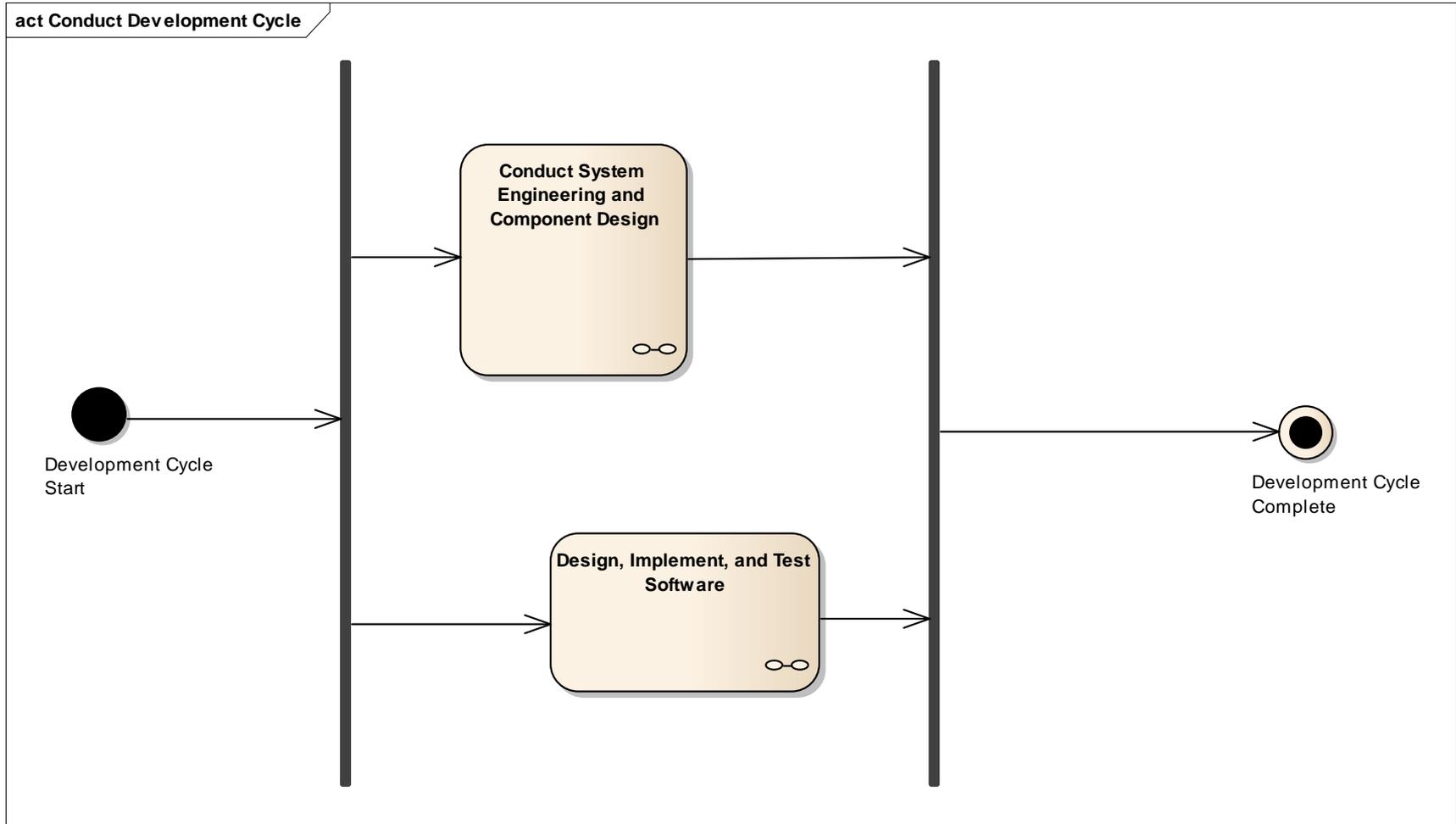
ID	Story	Estimation	Priority
7	As an unauthorized User I want to create a new account	3	1
1	As an unauthorized User I want to login	1	2
10	As an authorized User I want to logout	1	3
9	Create script to purge database	1	4
2	As an authorized User I want to see the list of items so that I can select one	2	5
4	As an authorized User I want to add a new item so that it appears in the list	5	6
3	As an authorized User I want to delete the selected item	2	7
5	As an authorized User I want to edit the selected item	5	8
6	As an authorized User I want to set a reminder for a selected item so that I am reminded when item is due	8	9
8	As an administrator I want to see the list of accounts on login	2	10
<b>Total</b>		<b>30</b>	

# Driving the Scrum Process

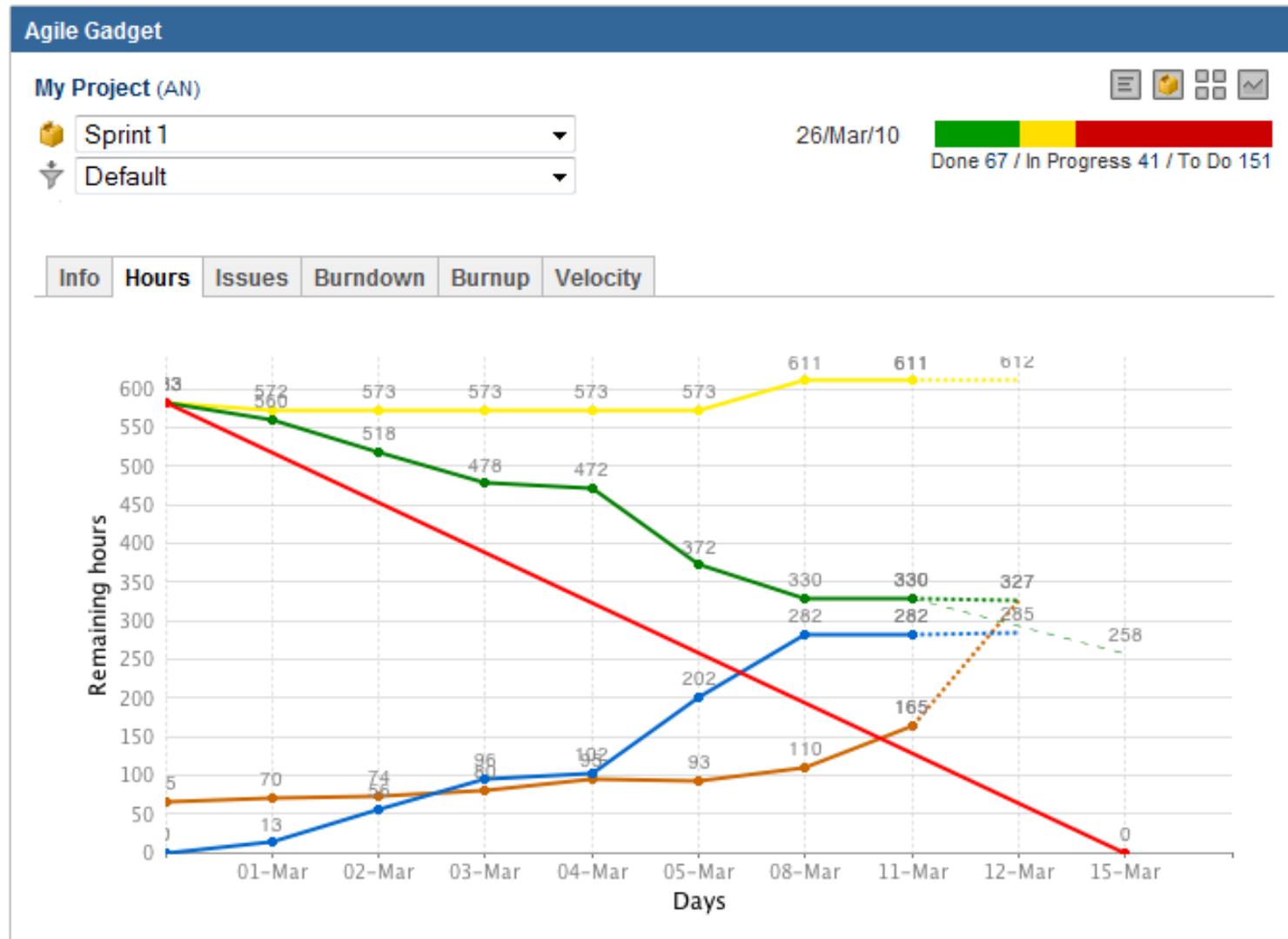
Story	To Do	In Process	To Verify	Done
<p>As a user, I... 8 points</p>	<p>Code the... 9</p> <p>Code the... 2</p> <p>Test the... 8</p> <p>Test the... 4</p>	<p>Code the... DC 4</p> <p>Test the... SC 8</p>	<p>Test the... SC 6</p>	<p>Code the... D</p> <p>Test the... SC 8</p> <p>Test the... SC</p> <p>Test the... SC</p> <p>Test the... SC 6</p>
<p>As a user, I... 5 points</p>	<p>Code the... 8</p> <p>Code the... 4</p> <p>Test the... 8</p> <p>Code the... 6</p>	<p>Code the... DC 8</p>		<p>Test the... SC</p> <p>Test the... SC</p> <p>Test the... SC 6</p>



# Parallel Development and SE Cycles



# Tools: The Burndown Chart



# Reasons Emergent Implemented Scrum



- Focuses on Providing Value
- Respects the Engineers
- Frequent iterations allow the development organization to become a well-oiled machine
- Involves the customer
- Encourages everyone to run the software
- Strong emphasis on testing
- Produces results

# Agile at Emergent



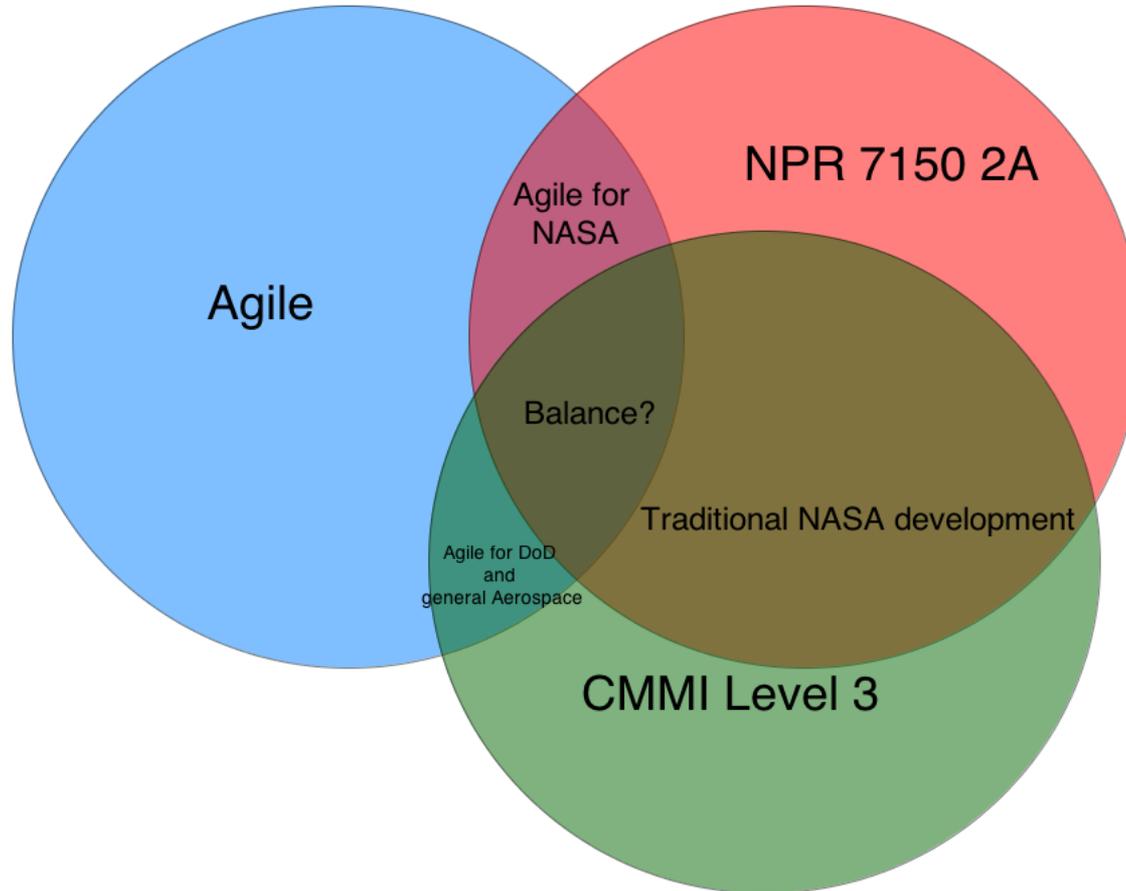
- Use small teams, co-located, if possible
- Modified for CMMI Level 3 by adding steps
- Modified for Aerospace by adding work products
  - Full bidirectional traceability
- Producing Functioning software!
- Parallel Systems Engineering Scrum cycle
- Use web-based tools
- Intermediate length iterations (3 weeks)
  - Provides a balance between rapid feedback and churn
- Scrums are not held daily
- Added the Scrum o' Scrums
- Schedule about 70% of the engineers time

# Impacts to the Four Basic Software Quality Metrics



- Intrinsic Product Quality Metrics
  - Mean time to failure
    - Run the software all the time. Its runnable after each iteration
    - Run it in a operationally valid environment from the beginning
      - Develop the test (sim) first
    - Maintain bidirectional traceability
  - Defect Density
    - Prioritized peer review
    - Employ static analysis
    - Unit test coverage
- Customer Satisfaction Metrics
  - Customer Problems
    - Understand the customer's environment
    - Run the software like the customer would
  - Customer Satisfaction
    - Customers participate in iteration reviews
    - Actively engage the customer
    - Welcome the customer as part of the team

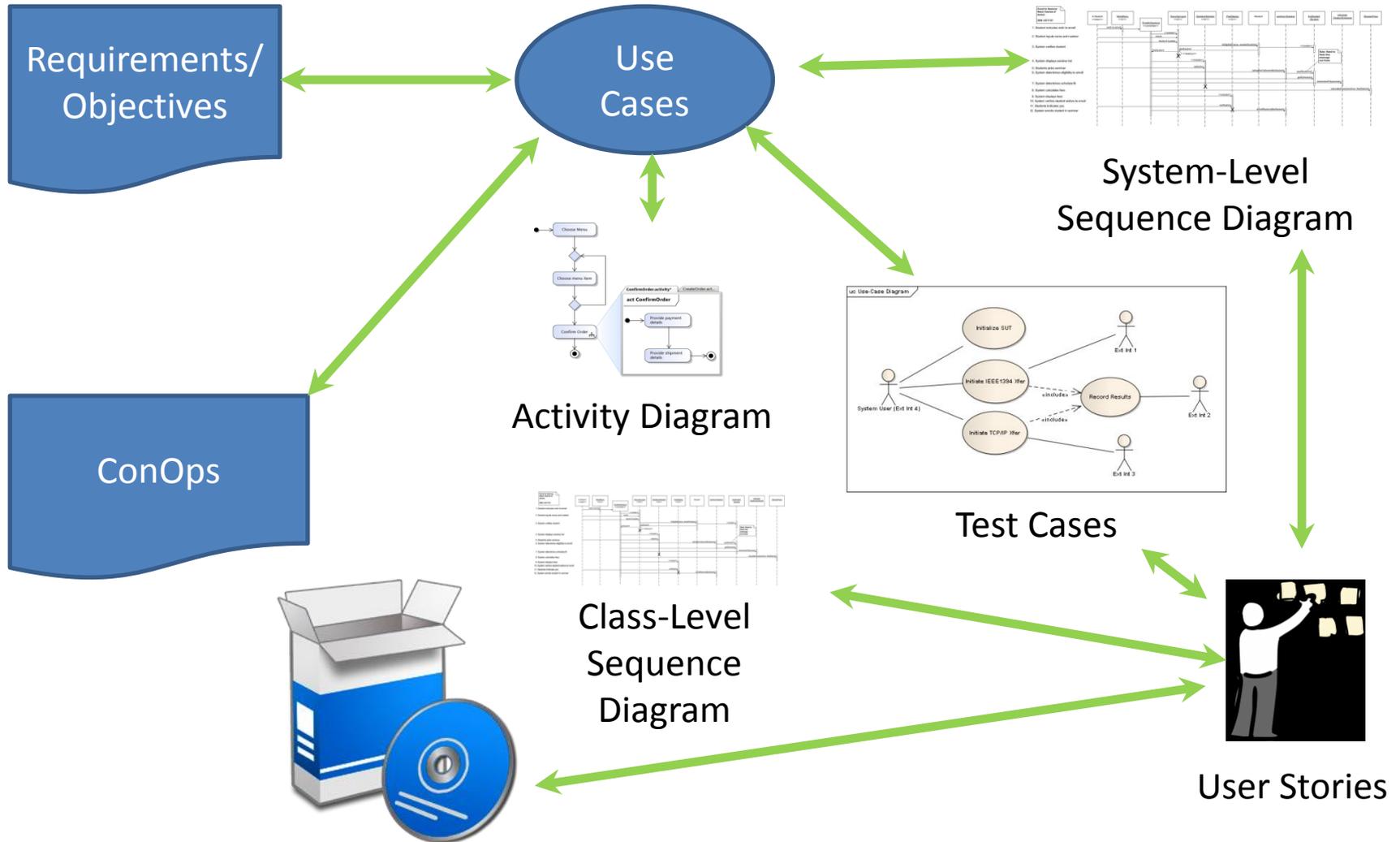
# Where does Agile Fit in?



# Challenges to Agile in Aerospace

- User Stories Don't Fit the Usual Pattern
  - Adapt pieces of Sequences
- Ensuring Independent Verification
  - Agile Practices don't preclude this, but they don't encourage it either
  - Maintain independent testers
- Maintaining and Managing Traceability
  - This is not a part of Scrum or any other agile method
  - Start early and maintain traceability in each iteration
- Achieving Verification and Validation
  - Build your test suite as you go along
  - Evolve your test success criteria as the product evolves
- Keeping the Product Backlog Fed
  - Maintain parallel Systems Engineering Process

# Maintaining Traceability



# Should You Implement an Agile Process In Your Project?

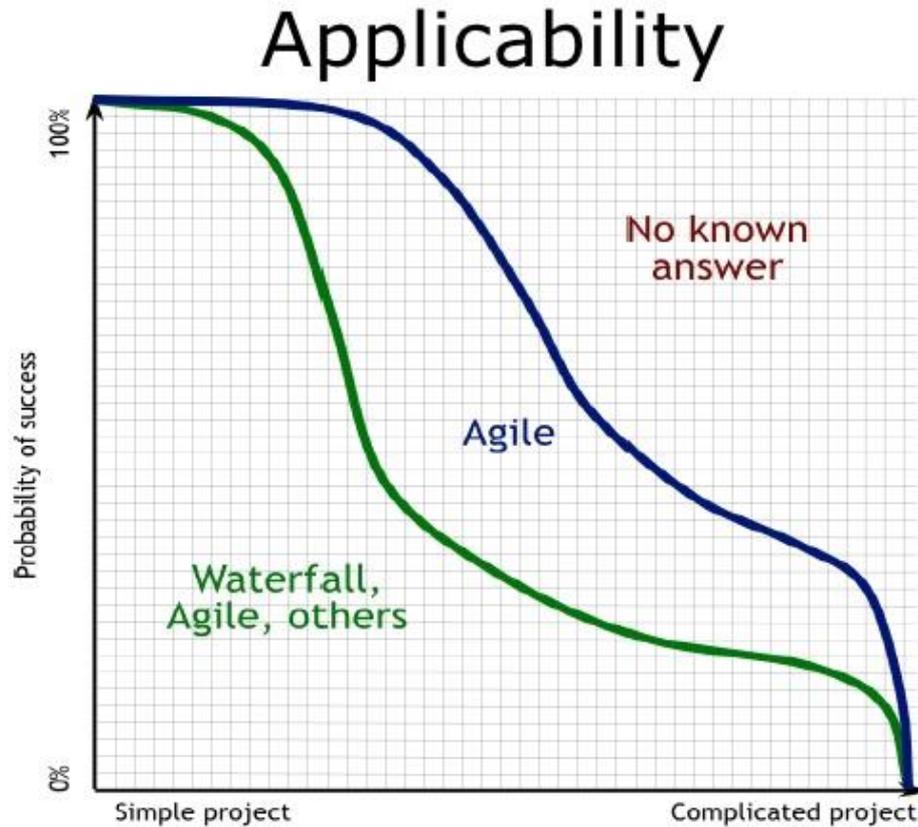
Are flight software developers special?



# Odds of Successful Completion By Project Size

Project Size	Number of People	Length of Program	Odds of Success
Small scale	<10	3-9 months	High
Medium-sized	20-30	<2 years	Slight
Large scale	100-300	3-5 years	Bleak
Mind boggling	1000-2000	5-10 years	Doomed

# Where is agile applicable?



Applicability of Agile (adapted from Ken Schwaber)

<http://duncanpierce.org/node/163>

# Benefits of Agile

- Development costs reduced by up to 70%
- Quality more than 3 times better than industry average
- Customer satisfaction 4.9 on a 5 point scale
- 70% developer satisfaction with process

From ITEA, Project AGILE, <http://www.agile-itea.org>, 2006.

- Provides enhanced insight into how the software is made
  - Moves some of the high risk activities forward (integrations, scenario testing)
  - Allows for continuous improvement
  - Engages all the stakeholders
  - Improves your team's estimation skills
  - The Teams seem more focused
  - The product seems more targeted to what the customer really wanted
- Brendan

Thank you!

# Brendan O'Connor

[brendan.oconnor@emergentspace.com](mailto:brendan.oconnor@emergentspace.com)

Chief Systems Architect  
Emergent Space Technologies  
M. (512) 791-5902

# Where does iterative development fit in?

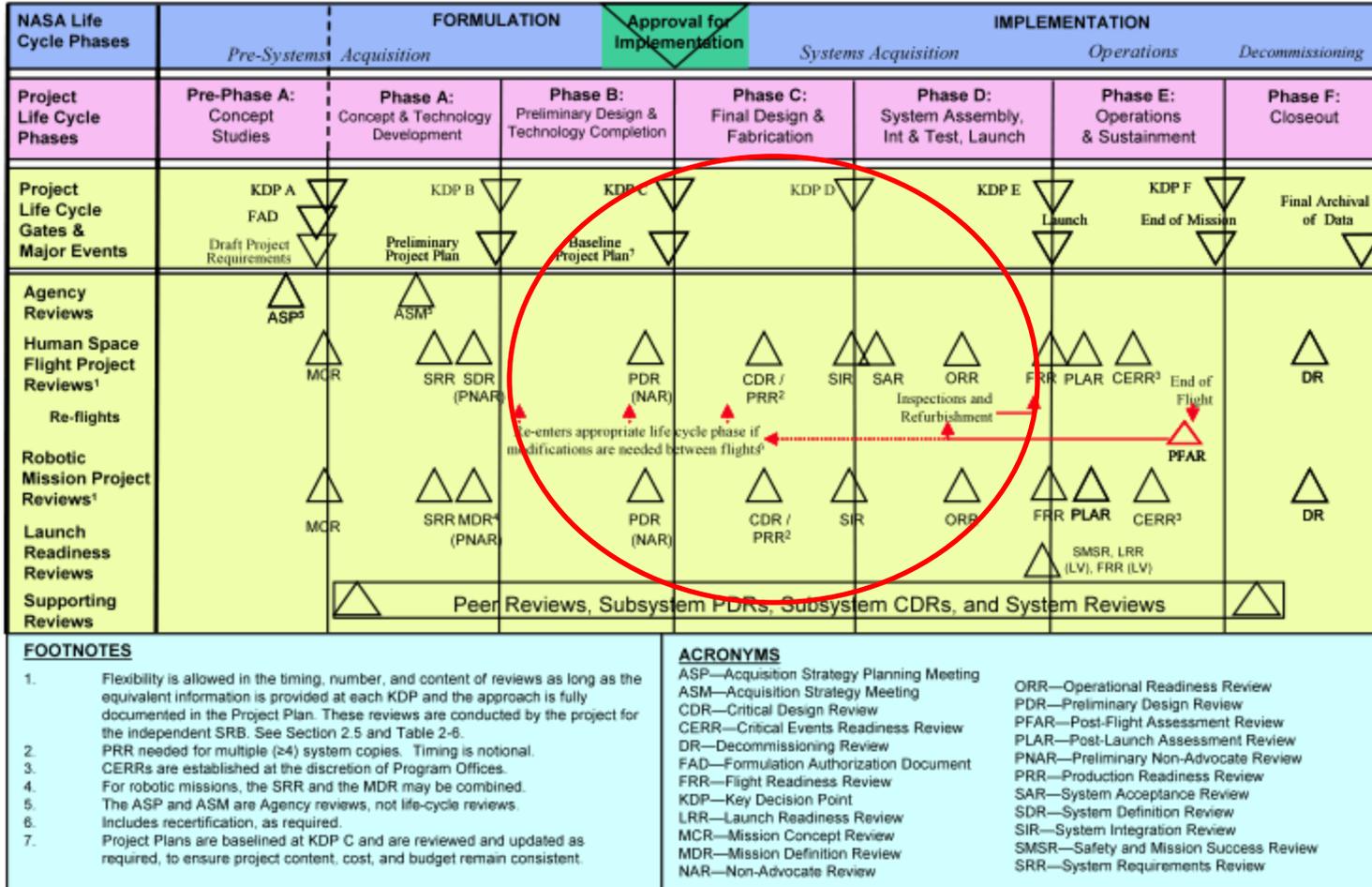


Figure 5-2 – The NASA Project Life Cycle