Agile

Jay Trimble
NASA Ames Research Center
at Philips Agile 4 Ever
October 2013
My Background

• Missions
  • NASA Johnson Space Center, Houston
    • Shuttle Mission Control, Payloads
  • Jet Propulsion Lab
    • Robotic - Voyager Neptune
    • Shuttle - Space Radar Lab, Lead Ops Director
• Current
  • Mission Operations & Ground Data System Manager, Resource Prospector Lunar Rover

Internship in Mission Control (A long time ago)
My Background

- Software Technology
- Human Centered Computing for Mars Rovers
- User centered technologies for mission control
Moments

Science Team after Mars Touchdown 2004

Shuttle Columbia
One Story of Agile at NASA

• This is a bottom up story of how a group at NASA applied agile methods to software development for mission control

• This was approved, but not initiated by, management

• Beyond software

• Taking agile beyond software we are applying agile and lean principles to the development of a Lunar Rover Mission Operations System
The Project

- Our group’s task was to build an architecture for mission control user applications, the primary focus being on developing interaction paradigms and technology for user composable software.
The Collaboration

- Design and Development Team at NASA Ames
- The Customer
  - Mission Control Users at NASA Johnson Space Center
- Using Participatory Design, we created an integrated team that included customer representation
Issues and Mandates

• Some customers want a new product, others do not
• The product must have new capability, but must also not be disruptive within the organization
• Functional and visual connection to legacy product
The Journey

- We began with a six month software delivery cycle
- By iteratively fixing issues, we got the delivery cycle down to three weeks
- It took close to two years to complete the transition
Time for Changes

- Fix the problems iteratively, without a broad proclamation of methodology, i.e. “we are going to be agile” or “we are going to be “lean”

- Just fix the problems
Where we started

- Four six-month deliverables
- One User Experience Spec
Issues we faced

• Long delivery cycle

• Difficult to manage feature prioritization and development, integration and testing

• Progress invisible to customer, lack of meaningful ongoing customer interaction to drive design
  • Mismatch in expectations between design/dev team and customer

• Difficult for the development team to know state of progress relative to goals

• Deliveries focus on subsystems rather than meaningful end user functionality

• Two-year final deliverable created a tendency to defer key issues
# The Team

<table>
<thead>
<tr>
<th>Traditional</th>
<th>Agile 1</th>
<th>Agile 2*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developers 5-9</td>
<td>Developers 7</td>
<td>Developers 4</td>
</tr>
<tr>
<td>User Experience Design (2)</td>
<td>User Experience Design (2)</td>
<td>User Experience Design (1)</td>
</tr>
<tr>
<td>QA/Process Engineers (2)</td>
<td>QA/Process Engineers (2)</td>
<td>QA (.5)</td>
</tr>
<tr>
<td>Project Manager (1)</td>
<td>Project Manager (1)</td>
<td>Developers rotate PM role</td>
</tr>
<tr>
<td>Principle Investigator (Part Time)</td>
<td>Principle Investigator (Part Time)</td>
<td>Principle Investigator (Part Time)</td>
</tr>
<tr>
<td>Interns</td>
<td>Interns</td>
<td>Interns</td>
</tr>
</tbody>
</table>

*Reduced Budget
Six Week Cycle

• We took the six month cycle and divided it into smaller pieces

• This was a start, but still left many issues
Agile

- We shortened the cycle to three weeks
- Replaced discrete events, with integrated interactions
- Integrated strategic and tactical into our ranking process
- Each iteration had clear purpose, goals, ranked priorities
- Release, iterations, daily build
- Strategic road map
Agile Cycle

- Deliver to customer every 3 weeks
- Nightly build
- Release every 3 months

Release to Mission Control User Test Community
Release to Mission Control User Test Community
Release to Mission Control User Test Community
Release to Mission Control Ops

Iteration 1
Iteration 2
Iteration 3
Iteration 4

3 Weeks
6 Weeks
9 Weeks
12 Weeks

jtrimble2@gmail.com
jay.p.trimble@nasa.gov
The Three-Week Cycle

Agile Development Iteration

Priorities/JIRA Rankings

Optional Mid-Iteration Hackathon tests big features

Feature Freeze (-7 days)

Code Freeze (-3 days)

Pre-Ship Hackathon

Start 24 hour test (-2 day)

Deliver to customer

3 Weeks Iteration n

Coding

UE & Tech Spec dates driven by coding dependencies

Issue Tracking Updates/Priorities/Rankings

Nightly Build/Internal testing as features roll out

Daily iteration n

Build to Customer

Test

User Feedback

Feature mods/additions, bug fixes

Customer installs iteration n-1

Customer acceptance test

Customer verification of closed JIRA issues

Customer triages issues it discovered

Optionally, hot patch

Iteration n+1

Monday, September 30, 13
The Release Cycle

Agile Release Into Operations

- **Iteration 1**
  - Release to Mission Control User test Community
  - Customer Feature Verification
  - Coding/UE Specs

- **Iteration 2**
  - Release to Mission Control User test Community
  - Customer Feature Verification
  - Issue Tracking Updates/Priorities/Rankings

- **Iteration 3**
  - Release to Mission Control User test Community
  - Customer Feature Verification
  - Build/Internal testing as features roll out

- **Iteration 4 Bugs/Usability/More Testing**
  - Release to Customer for Mission Control Certification

- **Release**
  - 3 Weeks
  - 6 Weeks
  - 9 Weeks
  - 12 Weeks

Monday, September 30, 13
Strategic Road Map

Event
- Certification
- Slim Adoption
- Flight Adoption
- Legacy Retirement Opportunity

Design
- Custom Views & Canvas Owned Properties
- Static & Dynamic Graphics
- Event Authoring
- Mat Lab Interface (JK/CL)
- Export/Import features
- Enhanced event monitoring
- Command GUI

Implement
- MCT/Chill Monitor Gap Analysis
- EFT-1 DB R5 lit4 start
- Initial Graphics
- Plot/Table tweaks
- Complete PLATO adapter (11/11)
- Initial incorporation of new plot package
- JPL Data Flow
- Support multiple monitors
- Navigation to non-MCT items
- ARC IRIS 1

User Feedback issues and bugs for cert and ops - prioritize as they come in
Complete custom views (1)
Plot/table ready for mission adoption (2)
EFT-1 DB R6 lit1
Import/Export
Event/Limit Architecture Choices/Scope

MatLab Interface
Enhanced Graphics
RESTful interface
Event/Limit Authoring
Initial ODRC Interface
Server-based data buffer with plot interface for historical plot data

User Feedback Issues >= 25%
Lessons Learned

• The measure of progress is working code

• Work on highest priorities first, avoid the temptation to do the easier things first

• Demonstrations, not presentations

• Customer interaction over extensive documentation

• Progress always visible, nightly build available

• Ship each iteration on time, only working features ship
  • Do not delay shipment for features - if a feature is not ready it goes into the next iteration

• Fit the process to your team context and culture, there is no one right way
We are applying agile and lean methods to the design of a Lunar Rover Mission operations and ground data system.
Agile/Lean for Lunar Rover

- Principle
  - Measure of progress is working code
  - Agile - download nightly build
  - Space Mission - demonstrate operational capability through simulation
Agile/Lean for Mission Ops

- Principle
  - Customer interaction over documentation
  - Agile - Participatory design (one method)
  - Space Mission - mission operations design session using PD methods, low fidelity simulation