Agile: From Software to Mission System

Jay Trimble, Mark Shirley, Sarah Hobart
NASA Ames Research Center

SMC-IT 2017
MOS & GDS

- Mission Operations System (MOS)
  - People, Team(s), Products, Processes
- Ground Data System (GDS)
  - Software, hardware, facilities
- Mission System = MOS + GDS
Waterfall to Agile for Software

• We began the journey to agile design and development with software
In the Beginning (Software)

- Delivery every 6 months
- The 6 month delivery cycle created too much time for customer expectations to diverge from what we were building
  - Customers needed to see the product more frequently
- Progress difficult to measure
- Long and formal design specs
  - Too much time talking, not enough time doing
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”
Agile Sprint

- Agile Tailored for our team
- Deliver to customer every 3 weeks
- Nightly build
- Release every 3 months
- Emphasis on constant interaction and use

3 Weeks Iteration n

- 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 days)
- Deliver to customer

Coding

- UE & Tech Spec dates driven by coding dependencies
- JIRA 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
Key attributes of our tailored agile process for software

• The measure of progress is working code

• Rank issues, always focus on the highest priorities

• Demonstrations, not presentations

• Customer interaction over extensive requirements meetings (some meetings still required)

• Visible progress - nightly or continuous builds

• Ship on time, features that are not ready go into the next sprint or release

• Verification using both QA and customer use in context
Agile for MOS

- NASA “standard” process definition is waterfall
  - It’s proven, it works
- Why change a proven process?
  - Potential gains in effectiveness, efficiency
  - Team engagement, culture of doing
  - Cost reductions
Tailored Agile

- For MOS
  - Simulation for Design
  - Assessment of capability through demonstration
  - Early and frequent builds and tests
  - Risk reduction through targeted experiments
  - Maturation of tools and processes through frequent use

- For Software
  - The measure of progress is working code
  - Rank issues, always focus on the highest priorities
  - Demonstrations, not presentations
  - Customer interaction over extensive requirements meetings (some meetings still required)
  - Visible progress - nightly or continuous builds
  - Ship on time, features that are not ready go into the next sprint or release
  - Verification using both QA and customer use in context
The Mission

• Lunar Rover

• Search for volatiles at a polar region

• Launch 2021?

For now, using agile methods

• New design

• Where it increases effectiveness
Resource Prospector Ops

- Lunar surface operations, round trip light time + comm latency = 6 - 30 seconds
- Short duration surface mission 5 - 7 days
- Fully distributed operations, including core teams, everyone operates from home institution
- Lighting, power constraints
- Operation in shadowed regions, no light for a billion years, rover is solar powered
- Continuous comm requirement
- Class D Mission
Agile Applied on RP

- 2015 Distributed Operations Test (DOT)

- Traditional design approach
  - Write a document(s)
  - Present, discuss, review
  - Train the team and test the design in simulations, late in the mission development flow

- Agile approach
  - Write a simple document
  - Conduct a series of simulations, for design, not training (though the team was trained in processes and procedures), early in the flow
  - Test and iteratively improve the concept of operations by trying it
Distributed Operations Test

- Simulation for Design
- Assessment of capability through demonstration
- Maturation of tools and processes through use
- Early and frequent builds and tests

California
Florida
Texas
DOT

- Test/Validate
  - Distributed decision making
  - Distributed command and data flow
  - Integrated situational awareness tools/integration into flight and ground system processes
  - Integrated ground/flight system test procedures
  - Team composition and roles
  - Waypoint driving

- What we did
  - Multiple simulations culminating in the Distributed Operations Test (DOT)
    - Three day DOT
    - Iterative refinement to procedures and processes
    - Mission operations teams using prototype tools
    - Distributed teams
    - Mission Team, science, rover drivers in California, rover hardware, rover systems in Texas, payload in Florida, drill in Southern California
What was Agile about DOT?

- Simulation for Design
- Assessment of capability through demonstration
- Early and frequent builds and tests
- Risk reduction through targeted experiments
- Maturation of tools and processes through frequent use

- How we used the information
- GDS architecture improvements for robustness and reliability
- Updates to team composition and roles
- Updated requirements
- Update software designs
Agile Example: Procedures

- “Say it then sim it”
- We wrote procedures, conducted a brief review then tried them
- Develop procedures by trying them out
- User paper simulations to fill in what’s not there yet
Procedures Sim/Walk Through

- Procedure Sim
- Google Hangouts to connect NASA centers
- Mix of paper and prototype software
An Agile MOS requires an Agile GDS

- DevOps
  - Medium to high fidelity simulations require parts of the GDS
  - Updating GDS across multiple locations
  - Prototype system for near-continuous integration, testing and deployment
    - Kickstart, Ansible, Docker
  - Rapid deployments, containers
Challenges

- Are low fidelity simulations beneficial?
- High fidelity simulations may be complex and labor intensive to set up
  - Focus on flight forward work, don’t put extensive effort into simulation work that is not flight forward
- Difficult to sim frequently
Conclusions

• Ideal - fly early and often

• If we can’t fly as often as we want
  
  • Focus on doing

  • Say it, then sim it

• After an issue is articulated in a meeting, instead of ongoing meetings, say it, then sim it
Conclusion

• Team culture can be shifted from a culture of meetings and documents to a culture of doing