Agile Systems Engineering Approach to Software Project Development

A look at how we combined agile & traditional systems engineering methods

Chris Ritter – Director of Software Development, SPEC Innovations
Daniel Hettema – Systems Engineer, SPEC Innovations
Steven H. Dam, Ph.D. – President & Founder, SPEC Innovations
About Us
What SPEC Innovations Does

• Systems Engineering
  • Architecture Development
  • Proposal Development
  • Full Lifecycle Support
  • Systems Training
• Contract Software Development
• Commercial Product: Innoslate
About The Software Team

Software Developers: 2 to 7
Systems Engineers: 1 to 3
Project Leads: 1

# of computer screens: 14, minimum

Max Team Size: 10
Our Culture

• Management sets top-level requirements
• No separate Q & A team
• All software developers had worked on prior projects together
• The team is highly collaborative with everyone providing suggestions throughout the process
• Team input in all aspects of design
• Team members worked in the same location, at the same office
What is Agile?
Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

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

That is, while there is value in the items on the right, we value the items on the left more.
Agile Systems Engineering

• Focus on continuous delivery of a product
• Accept and implement constant changes in requirements
  • Backlog requirements that cannot be performed in the current cycle
  • Accept requirements that can be easily added in the current cycle
• Daily collaboration within the development team, marketing team and support team
• Trust the input from all members of the team
Innoslate Development

The Cliff Note Version
“Develop a modern systems engineering tool, that has a function flow diagram, simulator, requirements management component with the end goal to support of the entire lifecycle.”

- Steve Dam
  Primary Stakeholder & Boss
Year One, Prototype Phase

- Two separate teams
  - Simulation Team
    - 1 Software Engineer
    - ½ Systems Engineer
  - Modeling Team
    - 3 Software Engineers
    - ½ Systems Engineers

- No formal process
- Used to generate the requirements for the development phase
Year One, Early 2011 Screenshot
Year One, Late 2011 Screenshot
Year One, Systems Engineer Role

• Aided in prototype requirements
  • Developed use cases
  • Mock ups
  • Modeled processes to specify functionality
• Aided as stakeholder with constant feedback
  • 15-minute Meetings daily
  • Researched ideas from the SE community
• Spearheaded development of the Lifecycle Modeling Language to support the tool
Year Two, Pre-Release Phase

• One cohesive team
  • 5 Software Engineers
  • 1 Software Lead
  • 3 Systems Engineers
  • 1 Project Leader
• Quality Assurance
  • Developed in-depth test cases
  • Added quality as a requirement at every phase
  • Developed releasable builds bi-weekly
Year Two, Screenshot
Year Two, Systems Engineers Role

- Aided in level two requirements
- Live daily collaboration with software team
- Weekly meetings to discuss status
- Spearheaded standardization of the Lifecycle Modeling Language to support this tool and open it to the public and other tools
- Used working prototype to define
  - New requirements
  - Issues and risks
Year Two, The Process

**Bi-Weekly “Sprint” Software Production Cycle**

- **Monday – Week 1**: Plan & Evaluate Priorities
- **Tuesday – Week 1**: Analysis & Design
- **Wednesday – Week 1 to Wednesday – Week 2**: Coding (New Features/Bug Fixes) & Unit/Developmental Testing (Local Host)
- **Friday Late PM – Week 2**: Deploy to Cloud Site
- **Friday PM – Week 2**: Final OT & Document Updates
- **Friday AM – Week 2**: Final fixes (if necessary) and review
- **Thursday – Week 2**: Operational Test (Cloud Test Site)

User Feedback (anytime from “Features Tracker” webpage)
Baseline Functional Requirements
Internal Lessons Learned
New Version Available to Users
Current, Released Phase

- One cohesive team
  - 3 Software Engineers
  - 1 Software Lead
  - 1 Systems Engineer
  - 1 Project Leader (Stakeholder)
- Two Formal Release Processes
Current, Overall Organization

- Daily collaboration with the entire team
- Weekly
  - Meeting with software lead and management
  - Meeting with project stakeholder
- Bi-Weekly with all stakeholders
- Monthly releases to customers
- Yearly major releases to customers
- Bi-Yearly design review and planning session
Current, Minor Release Process

**Monthly “Sprint” Software Production Cycle**

- **Monday – Week 1**: Plan & Evaluate Priorities
- **Tuesday – Week 1**: Analysis & Design
- **Week 1 to Week 2**: Coding (New Features/Bug Fixes) & Unit/Developmental Testing (Local Host)

**New Version Available to Users**

- **Final Day**: Deploy to Cloud Site
- **End of Week 4**: Final Test & Document Updates
- **Week 4**: Final fixes (if necessary) and review
- **Week 3**: Operational Test (Cloud Test Site)

**User Feedback**
- (anytime from “Features Tracker” webpage)
Current, Major Release Process

1. **START**
2. **Loop** once a month
3. Continue
4. Plan and evaluate priorities (Week 1)
5. Analysis and Design (Week 1)
6. New Features, Unit Testing (Week 1-4)
7. Internal Testing (Week 4)
8. Demo to internal stakeholders (Final day)
9. Capabilities Complete
10. Operational Testing and Fixes (4 Weeks)
11. Final testing and fixes (1 Week)
12. Deploy to cloud site
13. **END**
What We Learned & Conclusion

Hint: Agile Doesn’t Always Work
Lessons Learned

Works With Agile
- Software Bug fixes
- Decoupled Major Features
- Minor features
- Top-level documentation (2-5 pages)
- Low risk requirements

Doesn’t Work With Agile
- Major Backend changes
- Complex new systems
- Marketing
- In-depth Documentation (10+ pages)
- High risk top level requirements
Conclusion

• Keep teams small and in the same location
• Allow members of the team to provide design and feature input at all stages
• Encourage constant customer feedback
  • Commercial: Via email, feature tracker
  • Services: Via email, presentations, interactive demonstrations
• Allow low risk requirements creep
Questions?
Back Up
Other Projects

- Databasing tool
- Assessment Tool for Disaster Management
- Big data analytics and visualization

3D Comparison Diagram
Customer Feedback

- Quick turn arounds
- New features added monthly
- Allows stakeholders to better understand and derive better requirements
- High customer satisfaction