Traditional Project Management Meets Agile: Can’t We All Get Along

Dr. Harry Koehnemann
Director of Technology
Rocket Gang
harry@rocketgang.com

Mark Coats
Chief Software Engineer
General Dynamics C4S
Mark.Coats@gdc4s.com
MAY 2011

Traditional Project Management Meets Agile: Can’t We All Get Along

Rocket Gang, 14362 N. Frank Lloyd Wright Blvd, Scottsdale, AZ, 85260

Presented at the 23rd Systems and Software Technology Conference (SSTC), 16-19 May 2011, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License

Approved for public release; distribution unlimited

Same as Report (SAR)

14

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18
Agenda

• Traditional planning and it’s challenges
  √ Focus on team organization, planning, requirements, and architecture
• Making traditional planning agile
• Transitioning from traditional to agile
Traditional Development

Schedule

Earned Value Management

BDUF

Change

Development

PDR

Value

No value until end

Everyone responsible for everything; no one responsible for anything
Agile Development

Traditional

Change

Agile

Product Backlog

Release Backlog

Envisioning

Value

Value

Value

Value

Cross-Functional Teams

Sprint Backlog A

Product Owner

Sprint Backlog B

Scrum Master

* Barely Sufficient Design (Architecture Runway)

Copyright © 2011 Rocket Gang
Agile Value Points

• Defines success as delivering business value, vs. meeting schedule
• Continuous, frequent collaboration points between dev and business
• Encourages frequent feedback and learning
  √ Allows projects to fail fast
  √ Quickly adapts to changes
• Accommodates uncertainty, risk, change
• Plans work and organizes teams around system behavior
• Details requirements, design, schedule at right time
Transitioning to Agile

• Addresses team organization, planning, requirements, architecture
  • Continually define barely-sufficient requirements and architecture runway for feature teams

• Envision/Speculate (vs. Initiate/Plan)
  √ Brainstorm on product vision and promising implementation approach

• Explore (vs. Execute)
  √ Use iterations to explore solution and adapt to new discoveries
  √ Vs. preserve original baselines at all costs

Envision/Speculate: • Vision • Business Goals • Significant features • Architecture models • Constraints • Risks

“Sprint 0” • Architecture infrastructure • Development infrastructure (test automation, etc.) • Risk mitigation plans

Release/Milestone/Wave
Transitioning to Agile Team Organization

- Organize around behavior, not structure or domain
- Teams self-manage
  - Commit to features, decompose work, assign tasks
  - Desire to keep co-located!!
- Strive to push decisions out to feature teams
  - Specialty teams responsible for broader system issues and decisions
- Coupled features may require high collaboration between teams
Transition to Agile Planning

- Plans (WBS) define behavior
  - Releases plan features (20-100 day effort)
  - Decompose into stories (2-10 day)
- Items become more detailed closer to scheduling
- Use milestones/waves as synchronization points
  - Each iteration we learn more about team’s ability to meet schedule
Transitioning to Agile Requirements

- Envisioning defines constraints, significant features
- Exploring plans and details requirements at right time
- Need additional story types to support large, complex systems

Support Non-customer Facing Stories
- Constraints (security, performance)
- Infrastructure building
- Prototyping for risk mitigation
- Inter-team commitments
- Technical tasks
Transitioning to Agile Architecture

- Complex systems also develop their infrastructure
  - Responsible for significant functionality below the waterline
- Prioritize stories that extend and validate infrastructure
  - Includes rich set of automated tests
  - Similar in spirit to the “Sprint 0” agile practice (“Milestone 0”)
Caution on Organizational Change

- Great process cannot make up for having the wrong people
- Need capable people with self-discipline
  - Self-motivated, passionate about solving problems
  - Multi-skilled, adaptive, continuous improvers
  - Proactive, sense of ownership
  - Communicate, collaborate
- Coaching/modeling is key to organizational change

“Most organizations build their bureaucratic rules to manage the few wrong people on the bus, which drives away the right people on the bus, which in turn increases the need for more bureaucratic rules, and so on” -- Jim Collins (2001)
Recommendations

• Blend strengths
  - Agile: continuous feedback, learning, adaptive planning, behavior focus
  - Traditional: up-front assumptions

• Consider agility principles
  - Knowledge and running system are the goals, not documents and schedules
  - Focus on feedback and learning – problem, solution, development process
  - Use continuous, inspect-adapt cycles
  - Plan, track, execute small work
  - Organize teams around features
  - Lead teams, let teams manage themselves

• Repeatable wants to schedule, reliable wants to deliver value
  - Focus more on execution, less on management/control
Conclusions

• Fill the gap between project plans and developer tasks
  º Ensure plans are based on reality
  º Automate progress from development work to project-level plans

• Agile changes how we:
  º Plan Behavior-focused vs. structure/document-focused
  º Measure Business value vs. percent complete
  º Manage Self-managed teams vs. Project Manager
  º Track Feature-centric vs. plan-centric
  º Build & Test Continuously vs. at end
  º Define success Make business smile (value 😊) vs. execute the plan
Thank You

Dr. Harry Koehnemann
Director of Technology
Rocket Gang
harry@rocketgang.com

Mark Coats
Chief Software Engineer
General Dynamics C4S
Mark.Coats@gdc4s.com