IBM

Agility @ Scale with IBM Rational Software

Jan Van de Poel

03/09/2010

Goal

- What is Agile?
- Does Agile scale?

Anticipate a changing environment

What is Agile?

- Agile is a highly collaborative, evolutionary, quality focused approach to software development.

- How agile is different:
  - Focus on collaboration
  - Focus on quality
  - Focus on working solutions
  - Agilists are generalizing specialists
  - Agile is based on practice, not theory

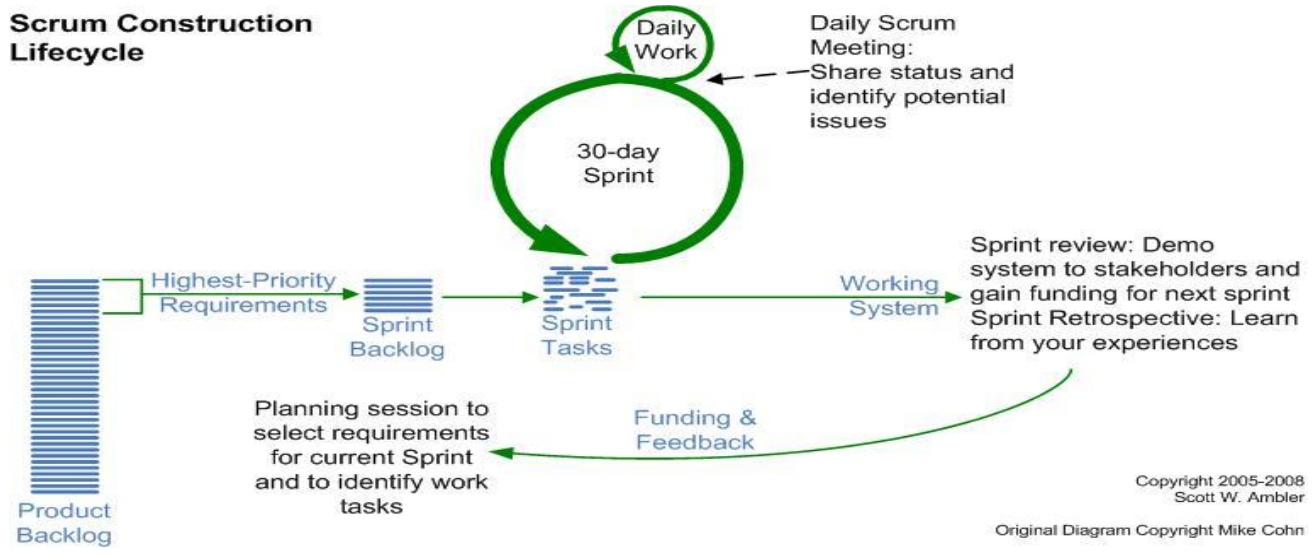
Anticipate a changing environment

Addressing misconceptions about agile

1. Agile teams write documentation
2. Agile teams model
3. Agile requires greater discipline than traditional approaches
4. Agile teams do more planning than traditional teams, but it's just in time (JIT)
5. Agile is more predictable than traditional
6. RUP can be as agile as you want to make it
7. Agile is not a fad, it is being adopted by the majority of organizations
8. Agile can do fixed price, but there's more effective options available to you
9. Agile scales very well

Anticipate a changing environment

The agile construction lifecycle



Anticipate a changing environment

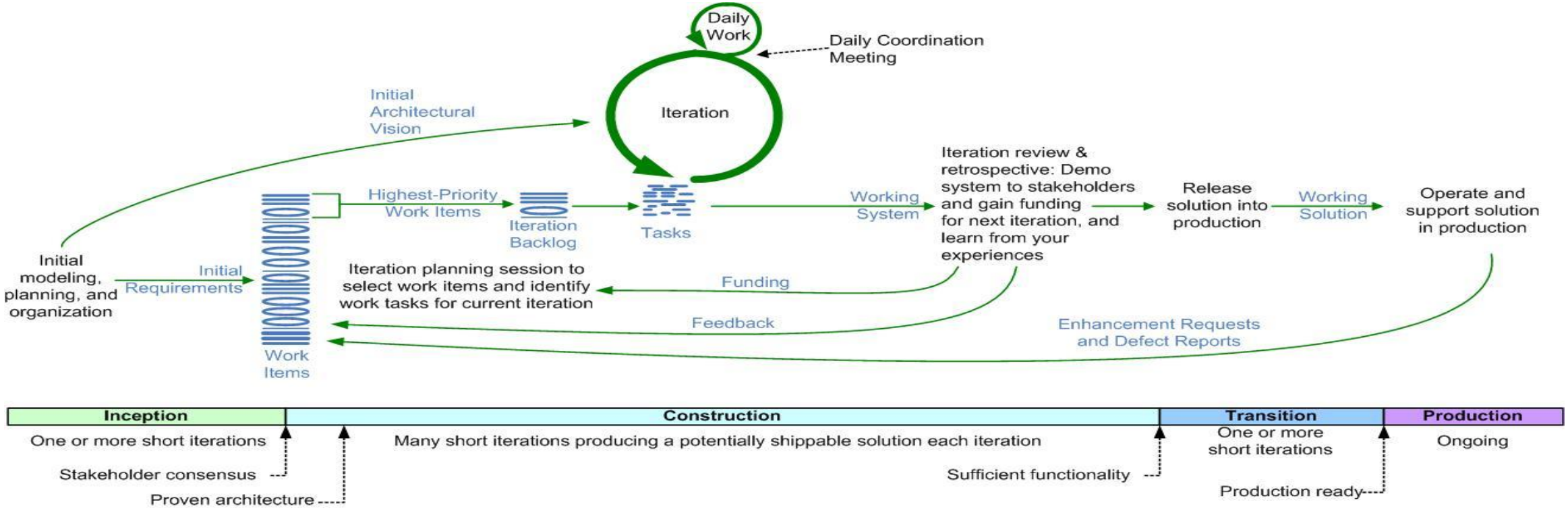
Agile Scaling Model (ASM)

Core Agile Development

- Focus is on construction
- Goal is to develop a high-quality system in an evolutionary, collaborative, and self-organizing manner
- Value-driven lifecycle with regular production of working software
- Small, co-located team developing straightforward software

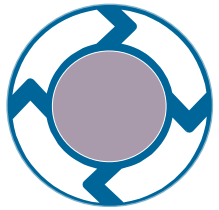
Anticipate a changing environment

The disciplined agile lifecycle: An extension of Scrum



Anticipate a changing environment

Agile Scaling Model (ASM)



Disciplined Agile Delivery

- Extends agile development to address full system lifecycle
- Risk and value-driven lifecycle
- Self organization within an appropriate governance framework
- Small, co-located team delivering a straightforward solution

Core Agile Development

- Focus is on construction
- Goal is to develop a high-quality system in an evolutionary, collaborative, and self-organizing manner
- Value-driven lifecycle with regular production of working software
- Small, co-located team developing straightforward software

Anticipate a changing environment

Agility @ Scale: Agile Scaling Factors

IBM

Team size

Under 10 developers ⟷ 1000's of developers

Compliance requirement

Low risk ⟷ Critical, Audited

Geographical distribution

Co-located ⟷ Global

Domain Complexity

Straight -forward ⟷ Intricate/ Emerging

Enterprise discipline

Project focus ⟷ Enterprise focus

Organization distribution (outsourcing, partnerships)

Collaborative ⟷ Contractual

Organizational complexity

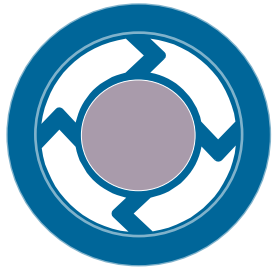
Flexible ⟷ Rigid

Technical complexity

Homogenous ⟷ Heterogeneous, Legacy

Anticipate a changing environment

Agile Scaling Model (ASM)



Agility at Scale

- Disciplined agile delivery and one or more scaling factors applies

Disciplined Agile Delivery

- Extends agile development to address full system lifecycle
- Risk and value-driven lifecycle
- Self organization within an appropriate governance framework
- Small, co-located team delivering a straightforward solution

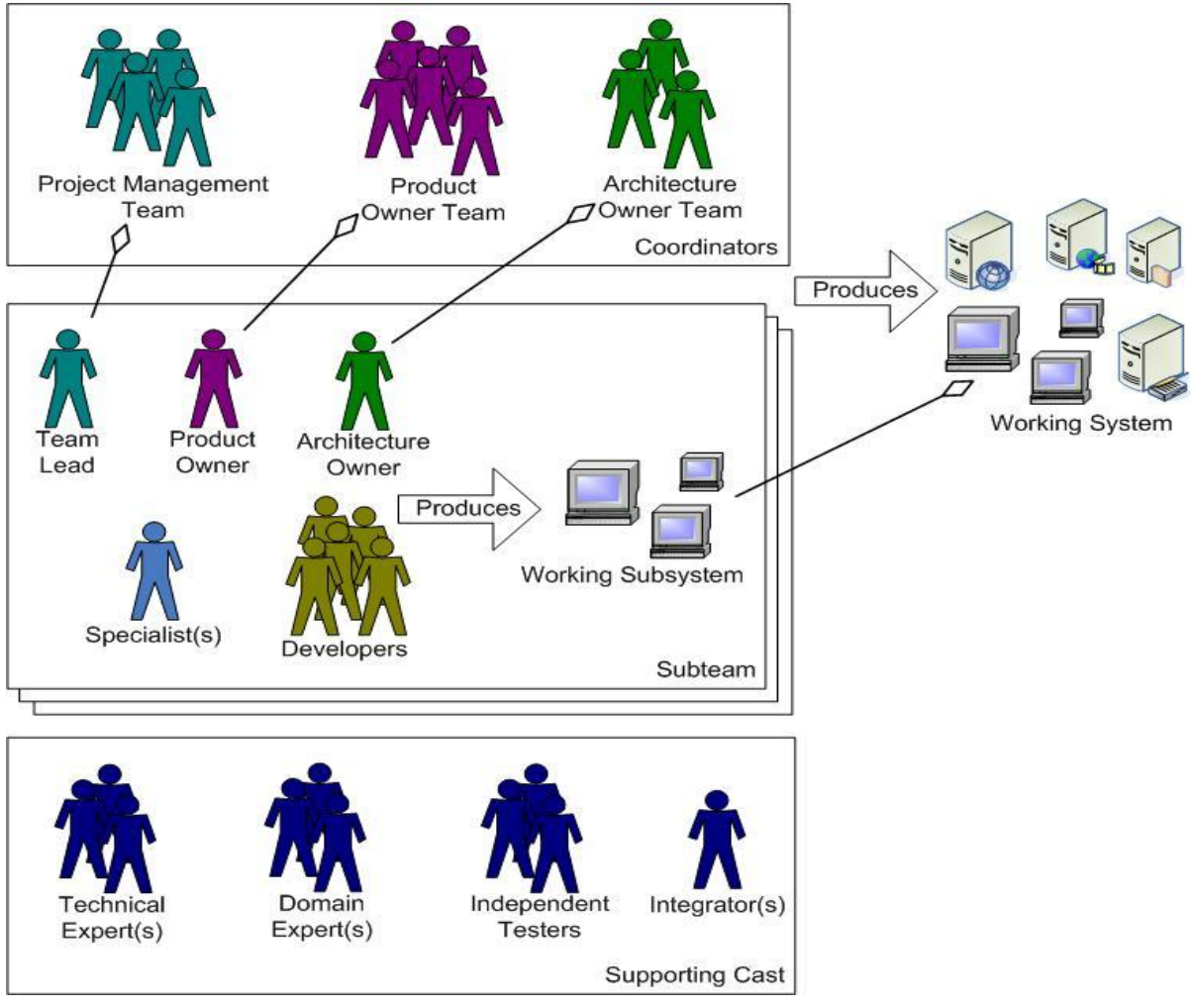
Core Agile Development

- Focus is on construction
- Goal is to develop a high-quality system in an evolutionary, collaborative, and self-organizing manner
- Value-driven lifecycle with regular production of working software
- Small, co-located team developing straightforward software

Anticipate a changing environment

Large agile teams

- Organize the work around your architecture
- Need to coordinate project management, requirements management, and technical issues
- Re-introduce some specialist roles as needed (for example Agile DBAs or UEX experts)
- Provide guidance on infrastructure & development conventions



Anticipate a changing environment

Feature teams vs. component teams

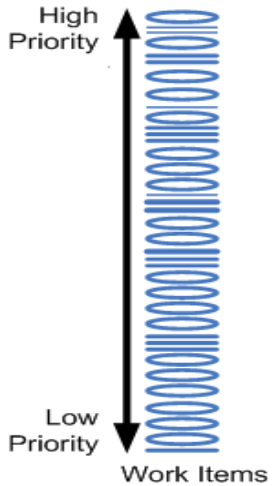
- Component team
  - Responsible for one or more system components (or services, frameworks, …)
  - Does all the work pertaining to that component
- Feature team
  - Responsible for implementing one or more features (or user stories, scenarios, …)
  - Does all the work pertaining to that feature
- Neither approach is perfect, you will likely use both strategies (and combinations thereof), within your organization

Anticipate a changing environment

Single vs. multiple work item lists

Issues:

- Organize work between subteams with minimal redundancy
- Burndowns of subteams need to be rolled-up into the overall project burndown
- Dependencies between work items need to be coordinated
- Subteams should be as independent as possible



Single list:

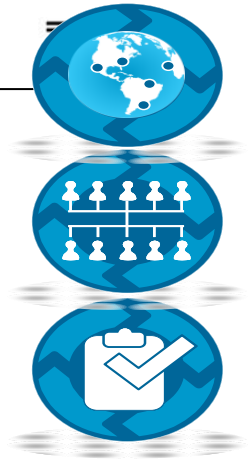
- Progress tracking (i.e. via burndown chart) straightforward
- Work assigned across subteams during iteration planning and evolved during iteration as appropriate

Multiple lists:

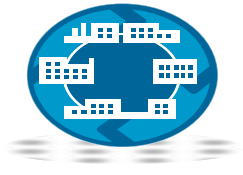
- Work initially assigned across subteams during inception and then evolved throughout the project as the requirements evolve

Anticipate a changing environment

Scaling daily stand up meetings



- Geographic distribution
  - Meeting over phone, video, electronically…
  - Rational Team Concert (RTC) to share information
  - Change meeting times to reflect team distribution – spread the pain
- Team size
  - Kanban strategy is to ask 1 question: What new issues do you foresee?
  - Subteams need to coordinate via coordinators
- Regulatory compliance
  - Take meeting attendance and record action items (if any)
- Organizational distribution
  - Additional coordination between organizations may be required
  - Project dashboard access for external organizations may be required
  - Document decisions/action items pertaining to external organizations

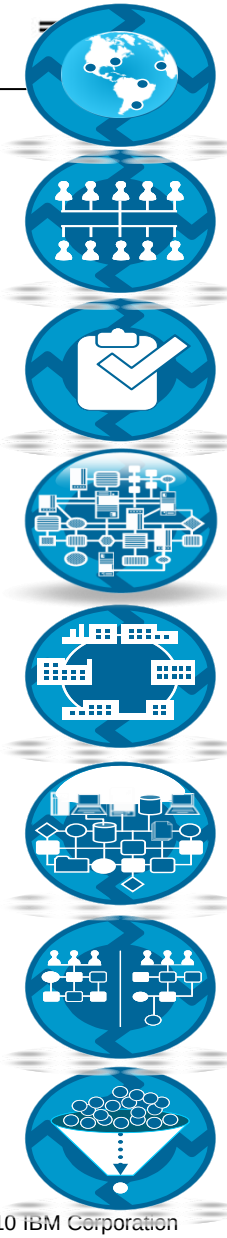


Anticipate a changing environment

Scaling product backlogs

- Disciplined agile delivery
  - Defects treated like requirements and managed on backlog
  - Non-functionality work items, such as training, reviews, can be managed on backlog
- Geographic distribution
  - Manage the backlog electronically
- Team size
  - Subteams may have their own backlogs, but that makes rollups harder
  - Burndowns of subteams need to be rolled up into overall team burndown
- Regulatory compliance
  - May need to manage backlog electronically
- Domain complexity
  - Business analysts look ahead on the product backlog to explore upcoming complexities
- Organizational distribution
  - A given organizational unit may only be allowed to see portions of the backlog
- Technical complexity
  - Team members look a bit ahead on stack to consider upcoming complexities
- Organizational complexity
  - Your team may need to conform to existing change management processes
- Enterprise discipline
  - Electronic backlog management enables automation of burndown charts and other metrics via project dashboard (e.g. in Rational Team Concert), supporting improved governance

Anticipate a changing environment



Why IBM?

- Our integrated tooling based on the Jazz platform enables disciplined agile software development
- Our Measured Capability Improvement Framework (MCIF) service offering helps organizations to successfully improve their IT practices in a sustained manner
- We are one of the largest agile adoption programs in the world
- We understand the enterprise-level issues that you face
- We scale from pilot project consulting to full-scale agile adoption
- Our Accelerated Solutions Delivery (ASD) practice has years of experience delivering agile projects at scale

Anticipate a changing environment

Links

- jazz.net
- ibm.com/**software/rational/agile/**
- ibm.com/**developerworks**/

Anticipate a changing environment

Session Summary

- Agile scales well
- It depends, it depends, it depends ;-)
- You may need to adopt some new practices
- You may need to evolve existing practices

Anticipate a changing environment

Parting thoughts

- Gain some experience
- Have a continuous improvement plan
- Invest in your staff
- The goal is to get better, not to become agile...

Anticipate a changing environment