IBM Software Group

Agile Software Development and Security

Scott W. Ambler
Chief Methodologist/Agile&SOA
scott_ambler@ca.ibm.com

Rational. software

Agenda

- Introduction
- Agile process maturity
- Agile and security
- Parting thoughts

Agenda

- Introduction
  - Determining if a team is agile
  - Misconceptions about agile
  - Industry statistics
  - When agile is not applicable
- Agile process maturity
- Adopting and security
- Parting thoughts

Criteria to determine if a team is agile

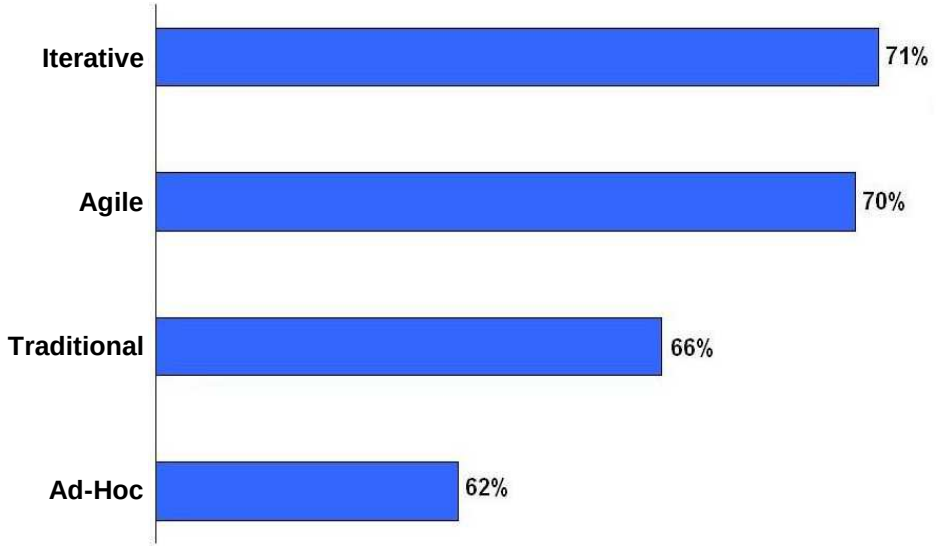
Disciplined agile teams:

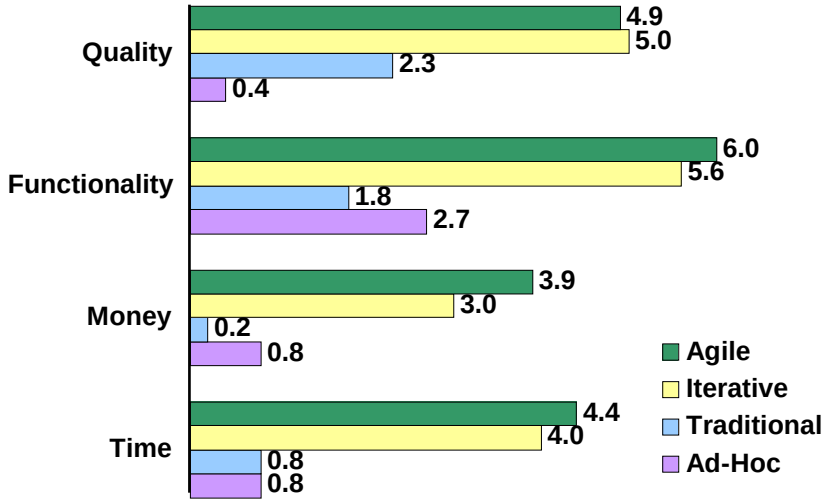
1. Produce working software on a regular basis.
2. Do continuous regression testing, and better yet take a Test-Driven Development (TDD) approach.
3. Work closely with their stakeholders, ideally on a daily basis.
4. Are self-organizing, and disciplined teams work within an appropriate governance framework.
5. Regularly reflect, and measure, on how they work together and then act to improve on their findings in a timely manner.

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. Agile scales very well
7. RUP can be as agile as you want to make it
8. Agile is not a fad, it is being adopted by the majority of organizations
9. Agile can do fixed price, but it's still poor practice to do so

Project success rates





Bottom Line: *Agile teams produce higher quality work, are quicker to deliver, are more likely to deliver the right functionality, and more likely to provide greater ROI than traditional teams*

Source: Dr Dobb's 2008 Project Success Survey

When agile is not applicable

- The culture of the organization is the primary determinant.
- Potential cultural pitfalls:
  - Waterfall culture
  - Low-trust environment
  - Unwillingness to change
- Very expensive to redeploy the system
- Significant dependencies on new hardware development
- They’re doing a good job with non-agile approaches

Agenda

- Introduction
- Agile Process Maturity
  - Towards a maturity model
  - Disciplined agile development
  - Agility at scale
  - Industry statistics
- Agile and security
- Parting Thoughts

Agile Process Maturity Model

1 Core Agile Software Development

Focus is on construction

Goal is to develop a high-quality system in an evolutionary, collaborative manner

Value-driven lifecycle with regular production of working software

2 Disciplined Agile Software Development

Extends agile development to address full system lifecycle

Risk and value-driven lifecycle

Lean development governance

3 Agility at Scale

- Addresses one or more scaling factors, including:
  - Team size
  - Geographical distribution
  - Organizational distribution
  - Regulatory compliance
  - Environmental complexity

Agile Values: A Foundation for Agile Development

1

We value

- Individuals Interactions
- Working Software
- Customer Collaboration
- Respond to Change

over

- Processes and Tools
- Comprehensive Documentation
- Contract Negotiation
- Following a Plan

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

Source: www.agilemanifesto.org

What is disciplined agile?

Disciplined agile software development is an evolutionary (iterative and incremental) approach to development which regularly produces high quality software in a cost effective and timely manner via a risk and value driven lifecycle.

It is performed in a highly collaborative and self-organizing manner, with active stakeholder participation to ensure that the team understands and addresses the changing needs of its stakeholders.

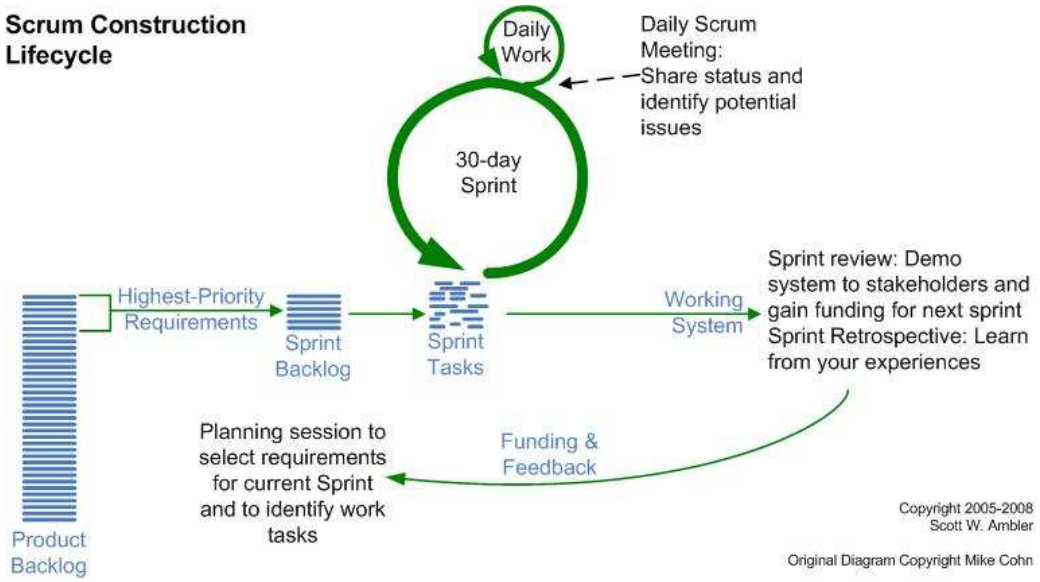
Disciplined agile development teams provide repeatable results by adopting just the right amount of ceremony for the situation which they face.

2

Core Principles

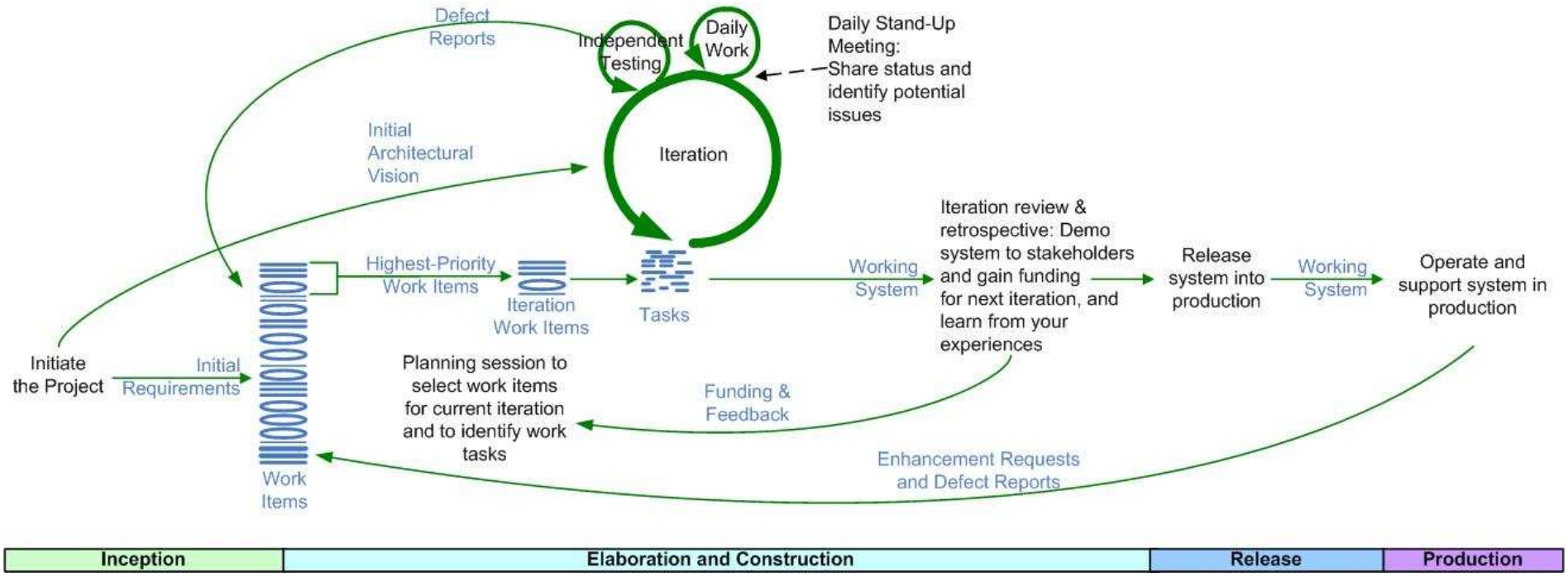
- "Fits just right" process
- Continuous testing and validation
- Consistent team collaboration
- Rapid response to change
- Ongoing customer involvement
- Frequent delivery of working software

The Agile Construction Lifecycle



Source: www.ambysoft.com/essays/agileLifecycle.html

The Full Agile Development Lifecycle



Source: www.ambysoft.com/essays/agileLifecycle.html

Manifesto for Software Craftsmanship

2

Not only

- Individuals Interactions
- Working Software
- Customer Collaboration
- Respond to Change

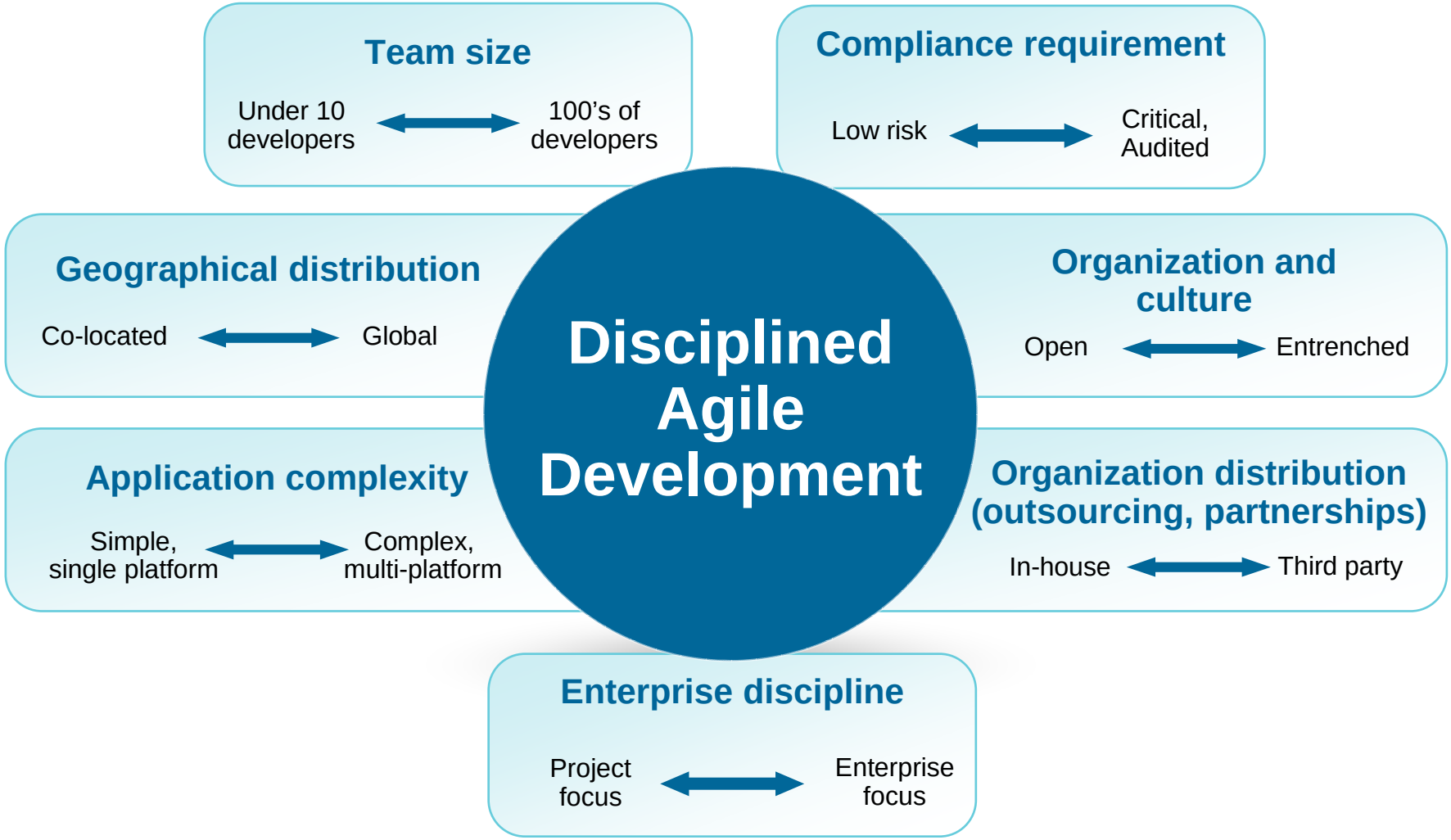
but also

- A Community of Professionals
- Well-Crafted Software
- Productive Partnerships
- Steadily Adding Value

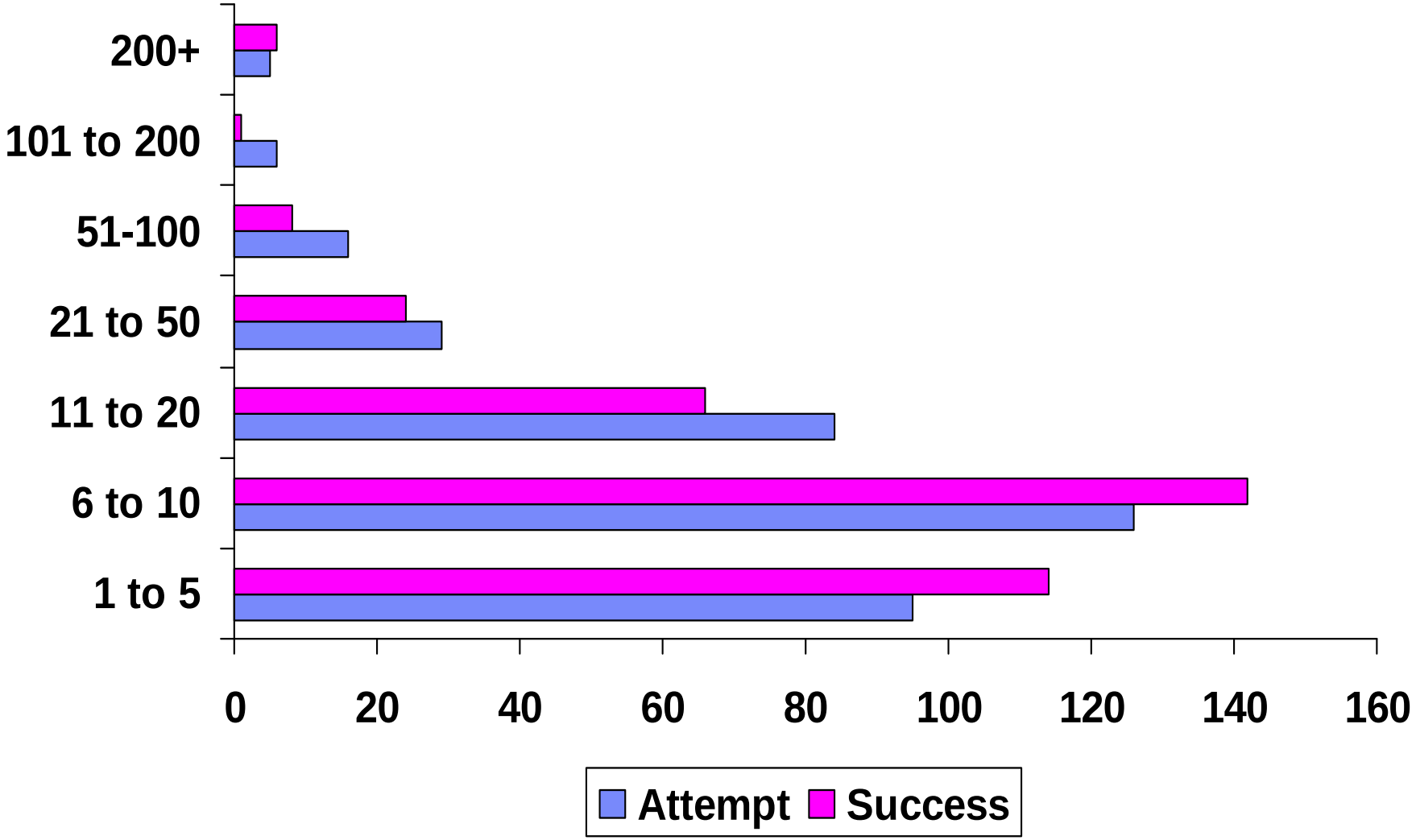
That is, in pursuit of the items on the left we have found the items on the right to be indispensable.

Source: manifesto.softwarecraftsmanship.org

What is Agility at Scale?

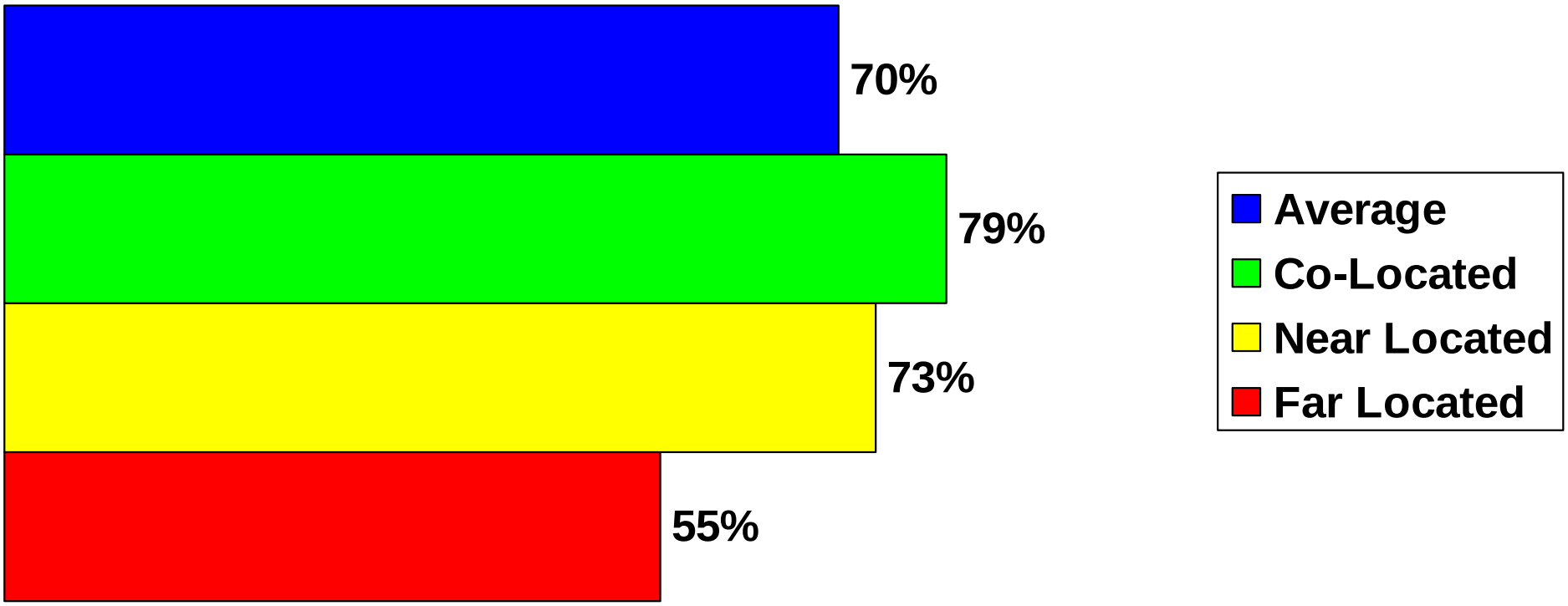


Largest team size: Attempted vs. Successful



Source: Dr Dobb's 2008 Agile Adoption Survey

Agile project success rates: the effect of distribution

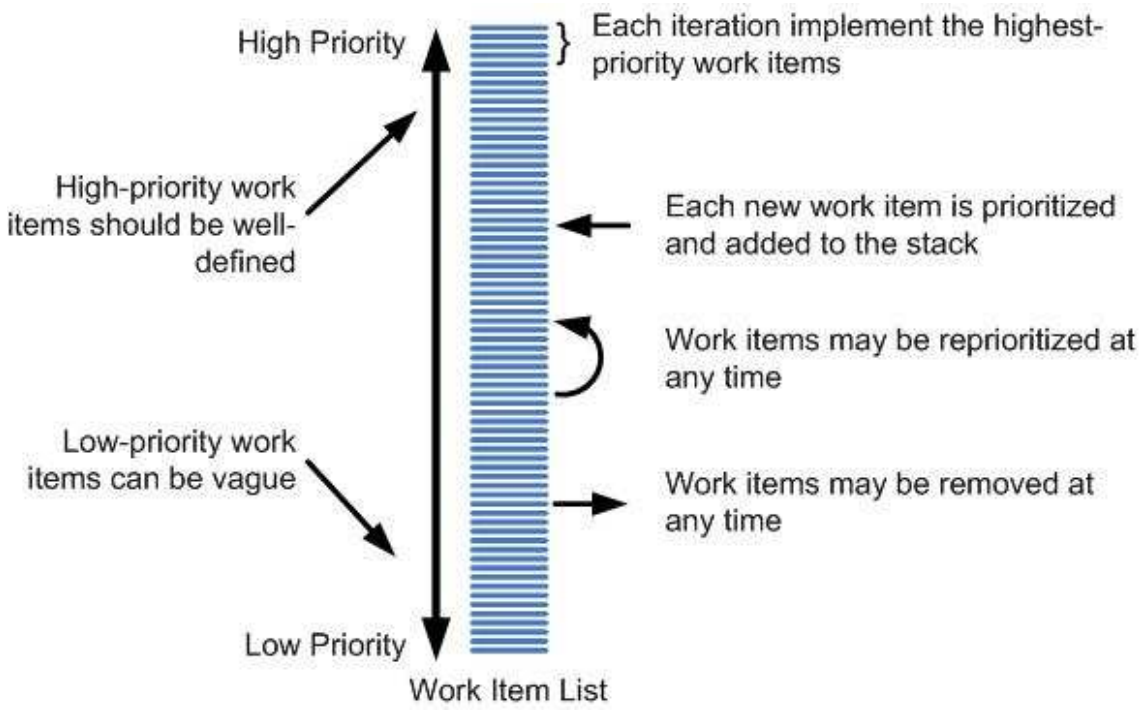


Source: Dr Dobb's 2008 Project Success Survey

Agenda

- Introduction
- Agile Process Maturity
- Agile and security
  - Recognize limitations of “level 1” strategies
  - Product owners
  - Initial modeling
  - Parallel independent testing
  - Invest in your staff
  - Tooling
- Parting Thoughts

Recognize That the Agile Requirements Strategy Doesn't Work Well for "ilities"

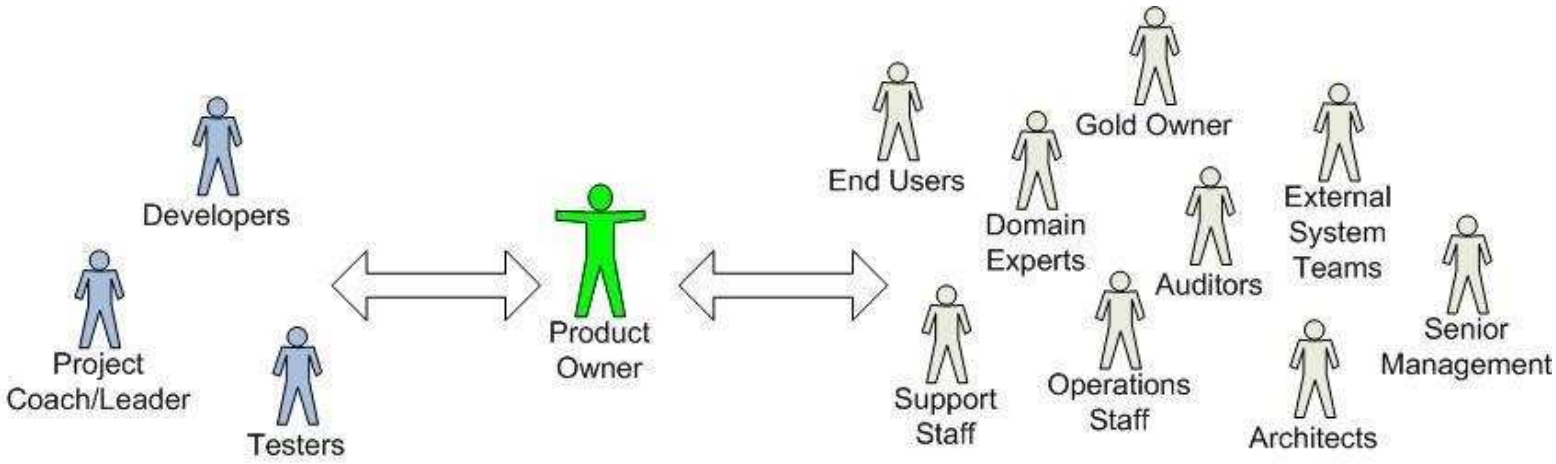


- Requirements are prioritized by stakeholders
- Requirements are estimated by the development team
- Requirements will evolve throughout the project
- Stakeholders see working software each iteration
- Stakeholders can change the level of funding as appropriate
- Stakeholders determine when "enough is enough"

Source: www.agilemodeling.com/essays/prioritizedRequirements.htm

Product Owners Must Represent a Wide Range of Stakeholders – Including Security Stakeholders

- On-site customer is nice, so put them to work
  - Stakeholders can be active participants in modeling
- Product owner is really a communication conduit between the team and stakeholders
  - Must have agile business analysis skills
  - PO gets the team access to the relevant stakeholders just in time
  - Negotiate, negotiate, negotiate



Source: www.agilemodeling.com/essays/agileRequirementsBestPractices.htm

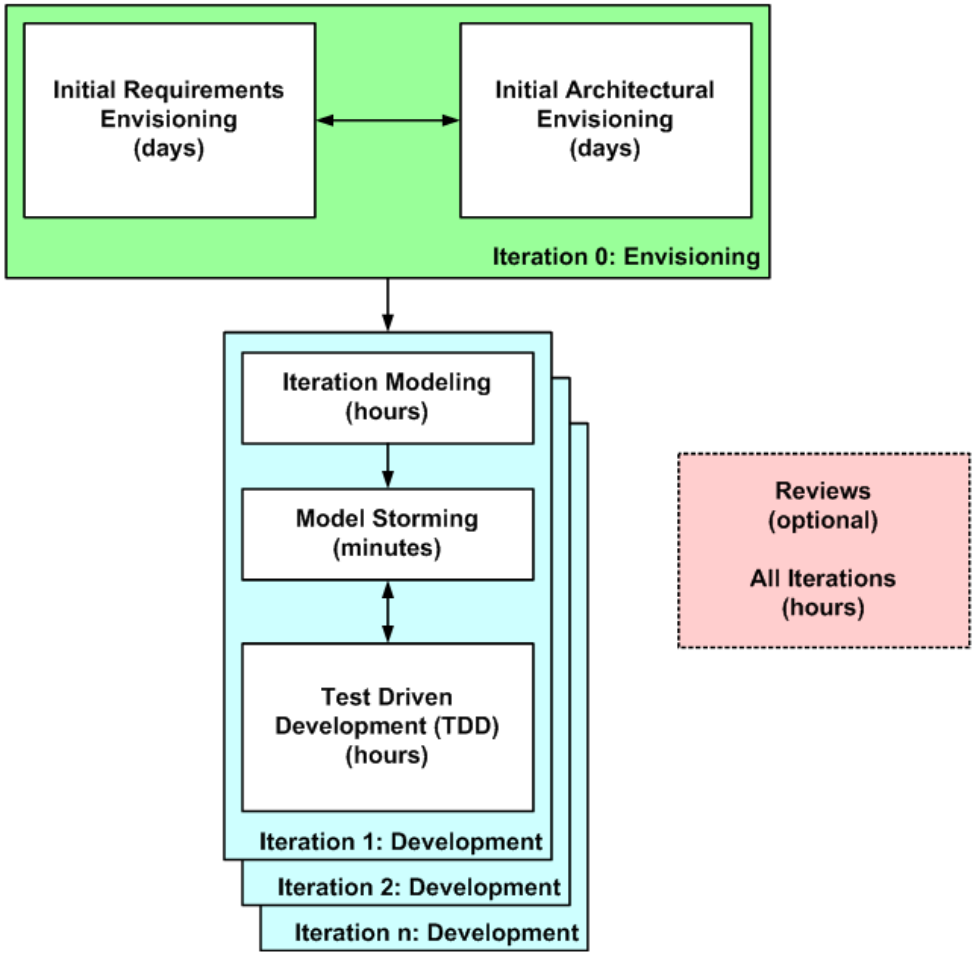
Do Some Up Front Modeling

Do some initial requirements envisioning to identify the high-level requirements

Do some initial architecture envisioning to identify a viable architecture strategy

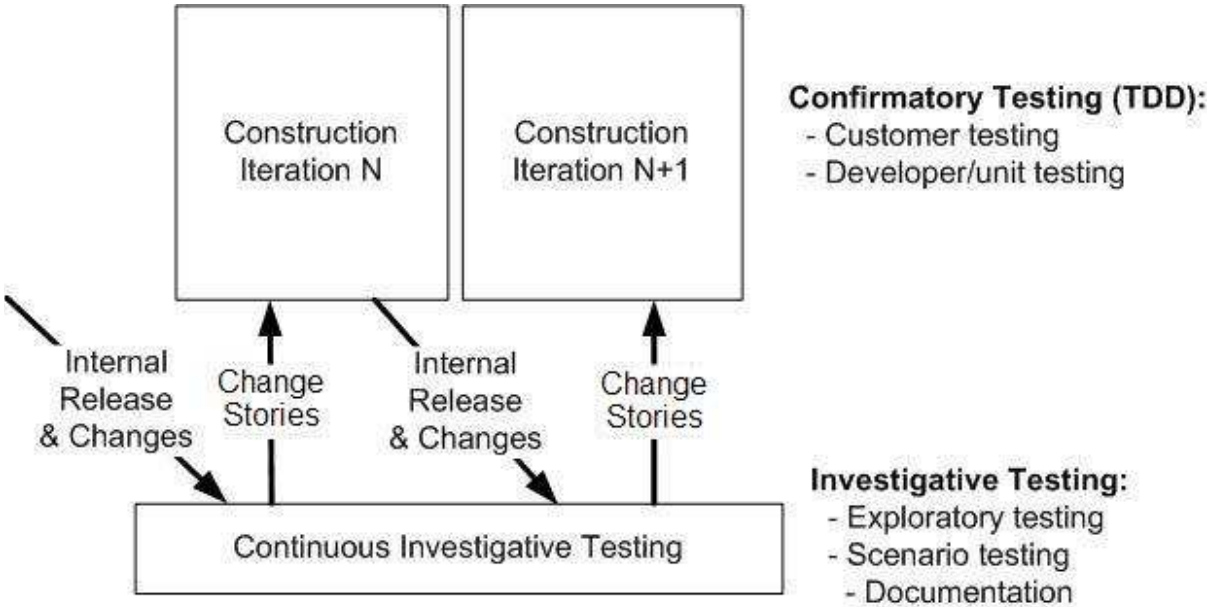
DO NOT write detailed specifications at the beginning of the project

The goal is to reduce risk by thinking critical issues through but not take on the burden (and risk) of detailed specifications



Source: www.agilemodeling.com/essays/amdd.htm

Have an Independent Test Team Looking for "ilities"



Effective agile teams push their working builds to an independent test team on a regular basis for investigative testing

Change stories must be prioritized and put back on the team's work stack

Defects == Requirements

Scales TDD: TDD is a form of confirmatory testing. TDD is a great start, but it's not the full testing picture

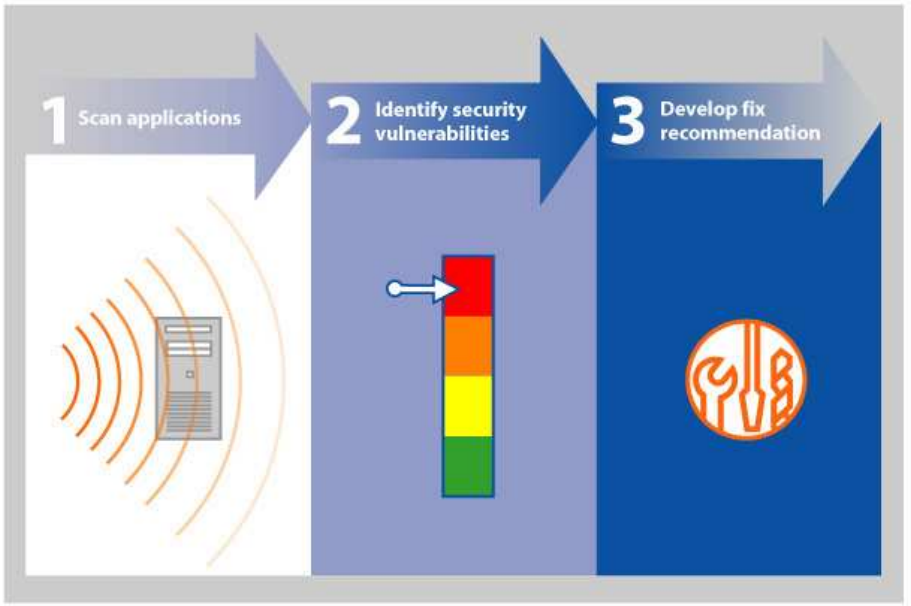
Critical strategy for addressing non-functional requirements

Invest in Your Staff

- Developers should understand the fundamentals of security
  - General: Developers should understand the fundamentals of a wide range of topics, security being one of many
- One-to-two day “intro to security” training
- Mentoring
- Non-solo development
  - Developers can easily pick up security skills from others who already have them

Tooling: IBM Rational AppScan
Testing web applications for security problems

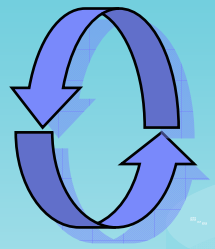
- AppScan is an automated Web application security solution that accurately pinpoints critical vulnerabilities and provides advice on how to fix them.
- Helps you to evaluate, understand and resolve security issues and greatly reduce the business risks related to Web vulnerabilities.
- Facilitates PCI compliance
  - Greatly reducing the need for manual testing, freeing up resources
  - Making in-house web penetration testing automated and affordable
  - Protecting company web infrastructure and brand



Source: www.ibm.com/software/awdtools/appscan/

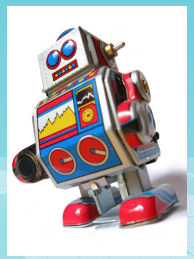
Tooling: IBM Rational Software Analyzer Automating Code Inspection

Continuous



Continuous, comprehensive, and collaborative quality management throughout the lifecycle reduces costs and improves credibility

Automated



Automating workflows across business processes by streamlining and eliminating redundancies to improve return on investment

Visibility

Moment-by-moment grasp of software quality and project metrics for immediate corrective action

Source: www.ibm.com/software/awdtools/swanalyzer/

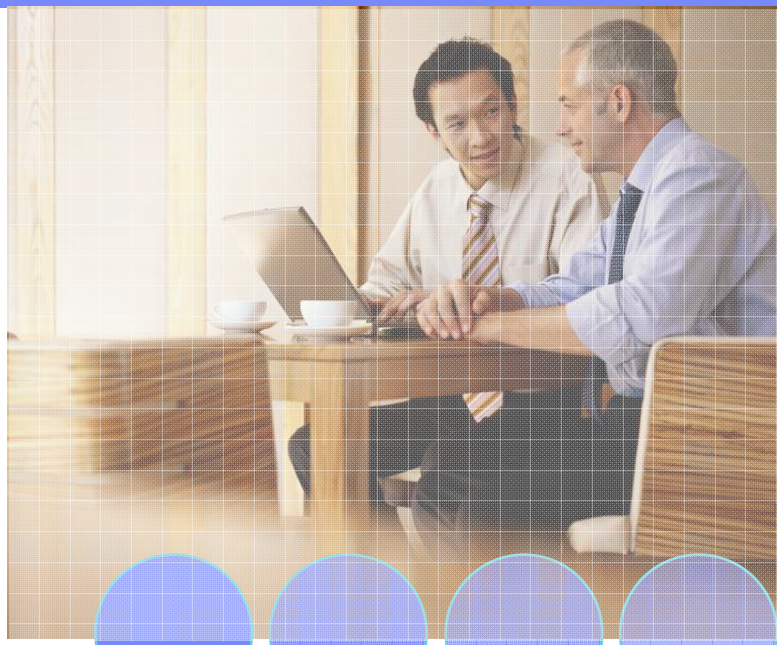
Agenda

- Introduction
- Agile Process Maturity
- Agile and Security
- Parting Thoughts
  - Why IBM?
  - Critical Resources

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

Critical resources



- www.ibm.com/rational/agile/
- www.ibm.com/developerworks/
- www.ibm.com/developerworks/blogs/page/ambler
- www.jazz.net