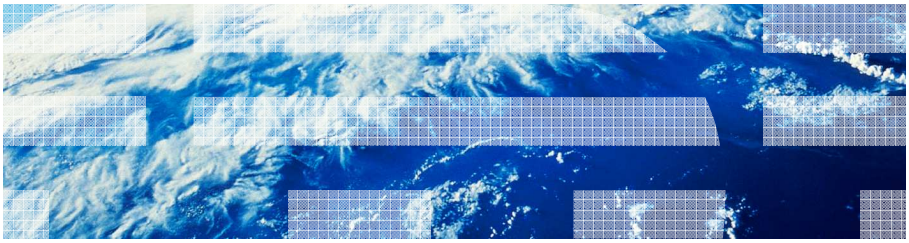
Agile Scaling Model: Be as Agile as You Need to Be

Scott W. Ambler
Chief Methodologist for Agile and Lean, IBM Rational
www.ibm.com/developerworks/blogs/page/ambler
twitter.com/scottwambler

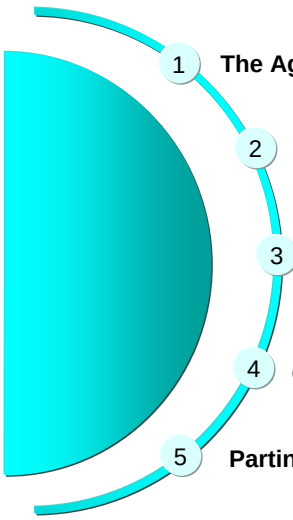


IBM agility@scale™

© 2011 IBM Corporation

This presentation, and others concerning enterprise agile, can be presented to your organization. Please contact your IBM representative to arrange.

Agenda



1. The Agile Scaling Model
2. Example: Scaling Daily Coordination Meetings
3. Scaling some common practices
4. Other scaling strategies
5. Parting thoughts

IBM agility@scale™

© 2011 IBM Corporation

Agile Scaling Model (ASM)

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

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

Agility at Scale

- Disciplined agile delivery and one or more scaling factors applies

3

© 2011 IBM Corporation

The Agile Scaling Model (ASM) defines a roadmap to effectively adopt and tailor agile strategies to meet the unique challenges faced by a system delivery team. The first step to scaling agile strategies is to adopt a disciplined agile delivery lifecycle that scales mainstream agile construction strategies to address the full delivery process from project initiation to deployment into production. The second step is to recognize which scaling factors, if any, are applicable to a project team and then tailor your adopted strategies to address the range of complexities the team faces.

The focus of Agile Development is construction. While clearly important, this is not the full picture. Core or mainstream approaches focus on the fundamentals and is characterized by value-driven lifecycles where high-quality potentially shippable software is produced on a regular basis by a highly collaborative, self-organizing team. The focus is on small (<10) co-located teams developing straightforward systems.

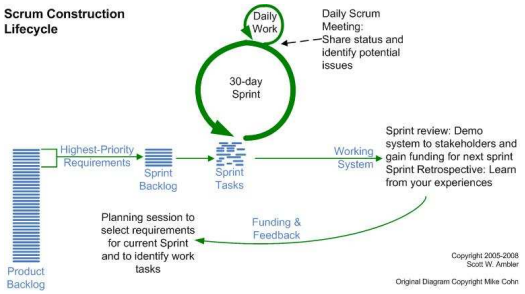
Agile delivery addresses the full delivery lifecycle from project initiation to production. It adds appropriate, lean governance to balance self organization. It adds a risk-driven viewpoint to the value-driven approach in order to increase the chance of project success. The focus is on small (<10) co-located teams developing straightforward systems.

Agility@Scale is disciplined agile development where one or more scaling factors is applicable. It is important to recognize that team size is only one of several issues that agile teams face at scale. The scaling factors that an agile team may face includes team size, physical distribution, organizational distribution, regulatory compliance, cultural or organizational complexity, technical complexity, and enterprise disciplines (such as enterprise architecture, strategic reuse, and portfolio management).

This course focuses on disciplined agile, where the full lifecycle is taken into account, and on Agility@Scale for organizational/enterprise-level development

See http://www.ibm.com/developerworks/blogs/page/ambler?tag=ASM for details.

The Scrum construction lifecycle



4

© 2011 IBM Corporation

See www.ambysoft.com/essays/agileLifecycle.html

This is the Scrum construction lifecycle. There are a lot of good ideas here, but it's not complete.

Scrum practices:

Product Backlog – Prioritized stack of requirements

Value-Driven Lifecycle – Deliver potentially shippable software each sprint

Self Organization – The people who do the work are the ones who plan and estimate it

Release Planning – Develop and then maintain a high-level project plan

Sprint Planning – At the beginning of a sprint, the team plans in detail what they will deliver and how they will do the work

Daily Scrum Meeting – Each day, hold a 15-minute coordination meeting

Sprint Demo – At the end of the sprint demo show what you've built to key stakeholders

Retrospectives – Take the opportunity to identify opportunities for improvement throughout the project.

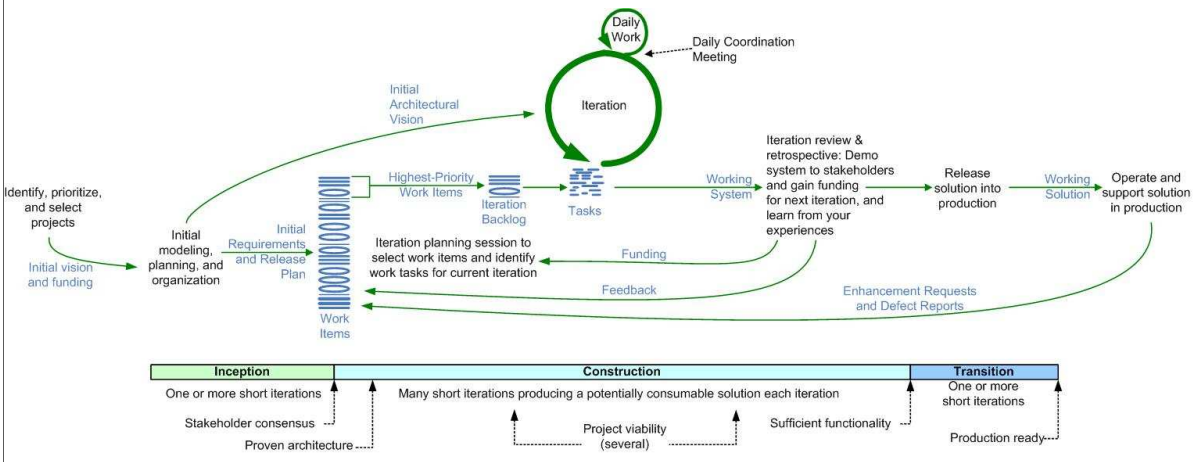
Roles:

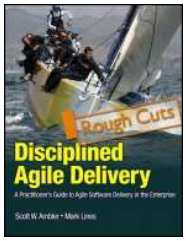
Scrum Master – Team lead

Product Owner – Responsible for prioritizing items in the product backlog, represents the stakeholders

Team Member – Developers, … on the team

The disciplined agile delivery life cycle





The Disciplined Agile Delivery (DAD) process framework is a people-first, learning-oriented hybrid agile approach to IT solution delivery. It has a risk-value lifecycle, is goal-driven, scalable, and is enterprise aware.

5

© 2011 IBM Corporation

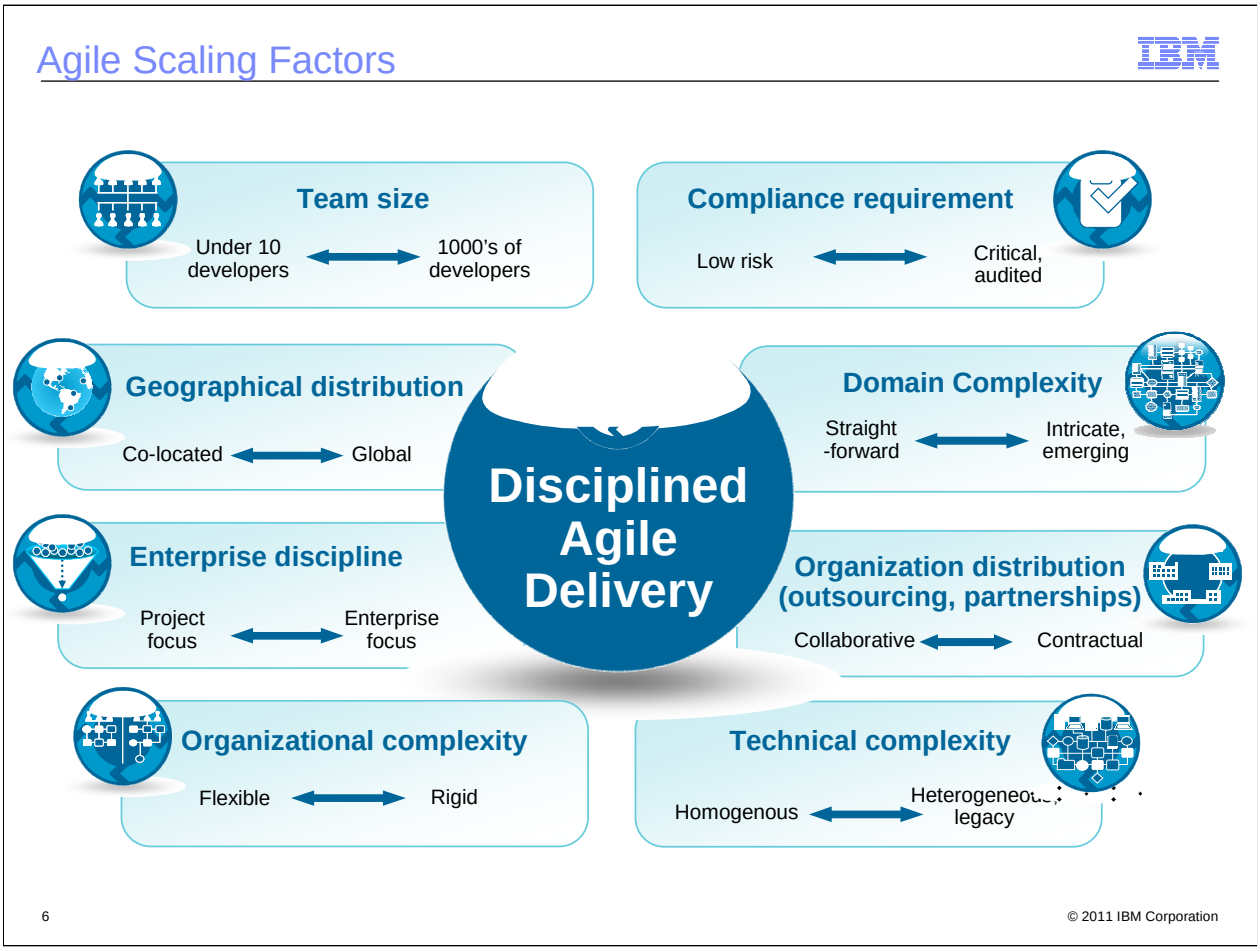
Disciplined agile delivery is an evolutionary (iterative and incremental) approach that regularly produces high quality solutions in a cost-effective and timely manner via a risk and value driven lifecycle. It is performed in a highly collaborative, disciplined, and self-organizing manner within an appropriate governance framework, with active stakeholder participation to ensure that the team understands and addresses the changing needs of its stakeholders. Disciplined agile delivery teams provide repeatable results by adopting just the right amount of ceremony for the situation which they face.

The disciplined agile delivery lifecycle expands upon the Scrum construction lifecycle in three important ways:

It has explicit project phases, recognizing that agile delivery is really iterative in the small and serial in the large.

It includes a full range of practices. This includes initial requirements and architecture envisioning at the beginning of the project to increase the chance of building the right product in the right manner, as well as system release practices.

It includes more robust practices. The lifecycle of this figure explicitly reworks the product backlog in the previous slide into the more accurate concept of a ranked work item list. Not only do agile delivery teams implement functional requirements, they must also fix defects (found through independent testing or by users of existing versions in production), provide feedback on work from other teams, take training courses, and so on.



In the early days, agile development was applied to projects that were small in scope and relatively straightforward. Today, organizations want to apply agile development to a broader set of projects. Agile needs to adapt to increasing complexity. Agility@Scale is about explicitly addressing the complexities that disciplined agile delivery teams face in the real world. The agile scaling factors are:

Geographical distribution. What happens when the team is distributed within a building or across continents?

Team size. Mainstream agile processes work well for small teams (10-15), but but what if the team is fifty people? One hundred people? One thousand people?

Compliance requirement. What if regulatory issues – such as Sarbanes Oxley, ISO 9000, or FDA CFR 21 – are applicable?

Domain complexity. What if the problem domain is intricate (such as bio-chemical process monitoring or air traffic control), or is changing quickly (such as financial derivatives trading or electronic security assurance). More complex domains require greater exploration and experimentation, including but not limited to prototyping, modeling, and simulation.

Organization distribution. Sometimes a project team includes members from different divisions, different partner companies, or from external services firms.

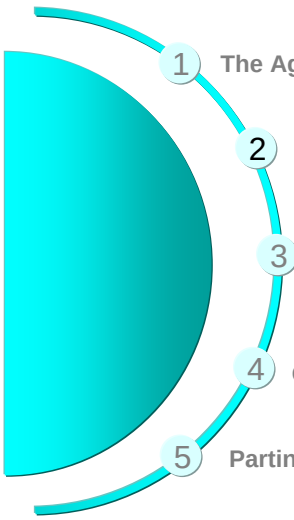
Technical complexity. Working with legacy systems, multiple platforms, or blending disparate technologies can add layers of technical complexity to a solution. Sometimes the nature of the problem is very complex in its own right.

Organizational complexity. The existing organizational structure and culture may reflect traditional values, increasing the complexity of adopting and scaling agile strategies. Different subgroups within the organization may have different visions as to how they should work. Individually, the strategies can be quite effective, but as a whole they simply don't work together effectively.

Enterprise discipline. Organizations want to leverage common infrastructure platforms to lower cost, reduce time to market, and to improve consistency. They need effective enterprise architecture, enterprise business modeling, strategic reuse, and portfolio management disciplines. These disciplines must work in concert with, and better yet enhance, the disciplined agile delivery processes.

Each scaling factor has a range of complexities associated with it. Each team faces a different combination of factors, and therefore needs a process, team structure, and tooling environment tailored to meet their unique situation.

Agenda



1. The Agile Scaling Model
2. **Example: Scaling Daily Coordination Meetings**
3. Scaling some common practices
4. Other scaling strategies
5. Parting thoughts

IBM agility@scale™

© 2011 IBM Corporation

Audience group work: Scaling Daily Coordination Meetings

Let's walk through each scaling factor one at a time:

1. Not at "scale": Agile development/delivery
2. Geographic distribution
3. Team size
4. Regulatory compliance
5. Domain complexity
6. Organizational distribution
7. Technical complexity
8. Organizational complexity
9. Enterprise discipline

Critical Observation:

Practices need to be tailored to reflect the situation you find yourself in.

Context counts.

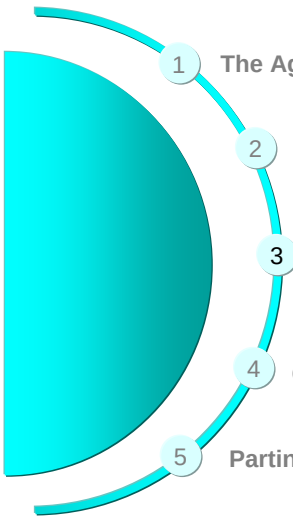
Note: Also known as a "daily stand up" or "Scrum meeting".

© 2011 IBM Corporation

Strategy:

Work through the situations one at a time with the audience and get them to share how they tailor this very simple practice to meet different situations.

Agenda



1. The Agile Scaling Model
2. Example: Scaling Daily Coordination Meetings
3. **Scaling some common practices**
4. Other scaling strategies
5. Parting thoughts

IBM agility@scale™

© 2011 IBM Corporation

Scaling release planning

- Geographic distribution
  - Plan needs to be captured electronically so that everyone has access
- Team size
  - Iteration lengths of the subteams should be a divisor of the larger team
  - Subteams may need their own release plans
  - Overall plan must reflect the plans of the subteams
- Regulatory compliance
  - The plan, and updates to it, may need to be documented
- Organizational distribution
  - Planning across multiple organizations will potentially take longer and require greater detail
- Technical complexity
  - Dependencies on other teams need to be reflected in release plan, particularly non-agile teams which have lower chances of on-time delivery
- Organizational complexity
  - Dependencies on non-agile teams may require changes in the strategies of all teams involved
- Enterprise discipline
  - Release plan must reflect portfolio-level and product-level plans

© 2011 IBM Corporation

A “team of teams” will need:

- An overall release plan
- Release plans for each subteam (usually)
- An overall team rhythm (i.e. 6 weeks)
- Subteams rhythms are a divisor of the team rhythm (i.e. 1, 2, 3, or 6 weeks)

Initial requirements envisioning (http://www.agilemodeling.com/essays/initialRequirementsModeling.htm) and architecture envisioning (http://www.agilemodeling.com/essays/initialArchitectureModeling.htm) are crucial parallel activities, the more complex the situation the more modeling that you’ll need to do (but that doesn’t imply detailed specifications up front)

Dependencies on external teams are critical, particularly non-agile teams which have lower chances of on-time delivery

Scaling iteration planning

- Geographic distribution
  - Need to capture the plan electronically
- Team size
  - Each sub-team is responsible for its own iteration planning
  - Team Leads need to be aware of major dependencies with other subteams, particularly when the other subteams have different iteration lengths/schedules
  - Teams needs to be aware of dependencies between work items lists
  - More likely the teams will need to meet prior to iteration planning to review user stories, disaggregate, estimate discuss dependencies, duplications across stories,…
- Regulatory compliance
  - Iteration plans may need to be documented
- Domain complexity
  - Teams need to engage in look-ahead planning, not just plan for upcoming iteration
- Technical complexity
  - Teams need to be aware of technical dependencies between subsystems
  - Teams need to engage in look-ahead planning, not just plan for upcoming iteration

© 2011 IBM Corporation

Scaling self organization

- Disciplined agile delivery
  - Disciplined agile developers are self organizing within an appropriate governance framework
- Regulatory compliance
  - Plans, estimates, and so on may need to be recorded
- Organizational complexity
  - May need to educate management team on collaborative leadership and facilitation
  - May need to provide education to help others shift from command-control to self-organizing
- Enterprise discipline
  - Application architecture decisions are constrained by enterprise infrastructure and futures directions
  - Application architecture enhanced by leveraging existing infrastructure and reusable assets
  - Release plan must reflect portfolio and project plans
  - Agile teams should follow enterprise-level guidelines (e.g. coding standards, data standards, UI standards, …)
  - Adopt a Lean Development Governance strategy (see paper by Ambler and Kroll)

© 2011 IBM Corporation

Lean development governance whitepaper, and other governance writings, available at http://www.ambysoft.com/onlineWritings.html#Governance

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
- 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 RTC), supporting improved governance

© 2011 IBM Corporation

Large or distributed teams are organized into smaller subteams

Component subteams have their own work item list and product owner

There will be dependencies between work items which need to be coordinated

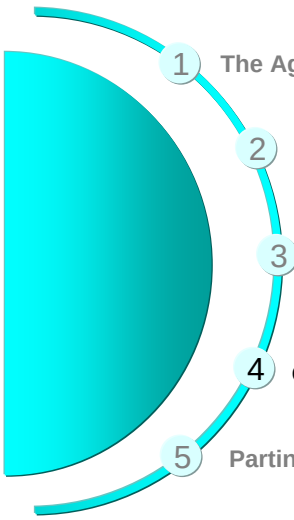
You want to ensure that the work is organized appropriately between subteams with minimal redundancy

Burndowns of subteams need to be rolled-up into the overall project burndown

You need tooling support

If they have separate product backlogs, they don't have to have a common understanding of a story point. Each teams breaks down, estimates work, determines velocity and selects from backlog based on their own velocity. If they have a common backlog, they need to have a normalized/baselined understanding of story points. If they have separate product backlogs and use different meanings of story points, you end up not being able to "sum" the work completed.

Agenda



1. The Agile Scaling Model
2. Example: Scaling Daily Coordination Meetings
3. Scaling some common practices
4. **Other scaling strategies**
5. Parting thoughts

IBM agility@scale™

© 2011 IBM Corporation

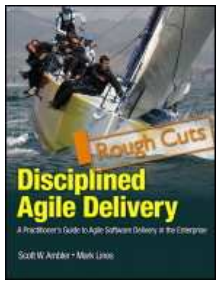
Defer commitment. It's not necessary to start software development by defining a complete specification. You can support the business effectively through flexible architectures that are change tolerant, and by scheduling irreversible decisions at the last possible moment. Frequently, deferring commitment requires the ability to closely couple end-to-end business scenarios to capabilities developed in multiple applications by multiple projects.

Deliver quickly. It is possible to deliver high-quality systems quickly. By limiting the work of a team to its capacity, you can establish a reliable and repeatable flow of work. An effective governance strategy doesn't demand teams do more than they are capable of, but instead asks them to self-organize and determine what they can accomplish. At an organizational level, it's important to enable programs to deliver business value at a pace defined by the fastest-moving projects, rather than at the speed of the slowest project.

Respect people. The Poppendiecks also observe that sustainable advantage is gained from engaged, thinking people. The implication is that you need a human resources strategy that focuses on enabling IT teams—not on controlling them.

Optimize the whole. If you want to govern your development efforts effectively, you must look at the bigger picture. You need to understand the high-level business processes that individual projects support—processes that often cross multiple systems. You need to manage programs of interrelated systems so you can deliver a complete product to your stakeholders. Measurements should address how well you're delivering business value, because that is the raison d'être of your IT department.

Adopt “other” agile strategies



- Requirements envisioning
- Architecture envisioning
- Developing an initial project vision
- Enterprise awareness
- Agile database techniques
- Independent parallel testing
- Lean development governance practices
  - Light-weight milestones
  - Automated monitoring
  - Align stakeholder and IT policies
  - Align team structure, architecture, and requirements
- More robust roles
  - Architecture owner, Domain experts, Technical experts, …

16

© 2011 IBM Corporation

Some references:

Agile data, www.agiledata.org

Independent parallel testing, http://www.ambysoft.com/essays/agileTesting.html

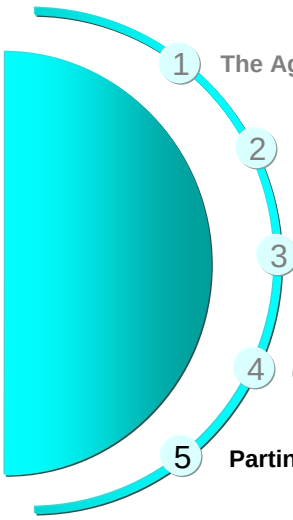
Lean development governance, http://www.ambysoft.com/onlineWritings.html#Governance

For roles, see the forthcoming book Disciplined Agile Delivery (IBM Press, Q1, 2012)

For more on agile enterprise disciplines, see www.enterpriseunifiedprocess.com

The Disciplined Agile Delivery book should be out Q1 2012. Review chapters are available online at Safari as rough cuts.

Agenda



1. The Agile Scaling Model
2. Example: Scaling Daily Coordination Meetings
3. Scaling some common practices
4. Other scaling strategies
5. **Parting thoughts**

IBM agility@scale™

© 2011 IBM Corporation

Does agile scale? Yes!

- Scaling:
  - The majority of agile teams are geographically distributed in some manner
  - Organizations have reported successful agile programs of 500+ people
  - One third of agile teams are in regulatory situations
  - 75% of organizations doing agile are doing so on medium complexity or greater projects
  - 17% of organizations are successfully applying agile in outsourcing situations
  - 78% percent of teams are working with legacy systems
  - 32% of organizations report successful interaction between enterprise architects and agile teams
  - 11% of organizations report that their governance strategy works well with agile teams (yikes)

- Source: DDJ November 2009 State of the IT Union Survey, www.ambysoft.com/surveys/

© 2011 IBM Corporation

Too many people still believe that agile doesn't scale, but fact is it seems to scale as well or better than traditional. There is evidence that organizations are succeeding at applying agile at every single scaling factor described in this presentation.

Address the “5Ps” of IT

1. People
2. Principles/Philosophies
3. Practices/Patterns
4. Products
5. Process

19

© 2011 IBM Corporation

To achieve systemic improvement within an IT organization, you need to address five primary issues:

1. People. Solution delivery is a “team sport”, individuals and the way that they work together are the primary determinants of success on most projects.
2. Principles. You need a consistent and coherent philosophical foundation which reflects your values as an organization to guide your decisions.
3. Practices. Practices are contextual, describing how people work together to achieve a goal.
4. Products. The tools and technologies which people use to perform their work.
5. Process. The “glue” which pulls everything together.

See https://www.ibm.com/developerworks/mydeveloperworks/blogs/ambler/entry/5pofit?lang=en_us

Some agile whitepapers on IBM.com

- The Agile Scaling Model (ASM): Adapting Agile Methods for Complex Environments
  - ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/raw14204usen/RAW14204USEN.PDF
- Scaling Agile: An Executive Guide
  - ftp://public.dhe.ibm.com/common/ssi/sa/wh/n/raw14211usen/RAW14211USEN.PDF
- Improving Software Economics: Top 10 Principles of Achieving Agility at Scale
  - ftp://public.dhe.ibm.com/common/ssi/ecm/en/raw14148usen/RAW14148USEN.PDF
- Enable the Agile Enterprise Through Incremental Adoption of Practices
  - http://public.dhe.ibm.com/common/ssi/ecm/en/raw14077usen/RAW14077USEN.PDF
- Disciplined Agile Delivery: An Introduction
  - http://public.dhe.ibm.com/common/ssi/ecm/en/raw14261usen/RAW14261USEN.PDF

© 2011 IBM Corporation

scott_ambler [at] ca.ibm.com

twitter.com/scottwambler

www.ibm.com/developerworks/mydeveloperworks/blogs/ambler/

www.ibm.com/rational/agile/

www.jazz.net

© Copyright IBM Corporation 2011. All rights reserved.

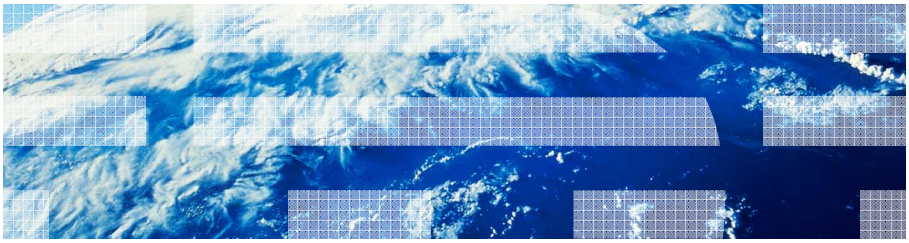
The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

IBM, the IBM logo, the on-demand business logo, Rational, the Rational logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

© 2011 IBM Corporation

Scott Ambler is available to work with your organization regarding a range of enterprise issues, including scaling agile strategies, governing agile projects, enterprise adoption of agile/lean, and analytics for IT. Contact your IBM representative to arrange time with Scott.

Backup Slides



IBM agility@scale™

© 2011 IBM Corporation

Scaling Daily Stand Up Meetings

- Disciplined agile delivery
  - Call them "coordination meetings"
- Geographic distribution
  - Meeting occurs 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, perhaps in a "scrum of scrums"
- 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
- Enterprise discipline
  - Enterprise professionals should be invited to meetings, better yet the teams

© 2011 IBM Corporation

•Domain complexity increases the need for regular coordination

•Technical complexity increases the need for regular coordination

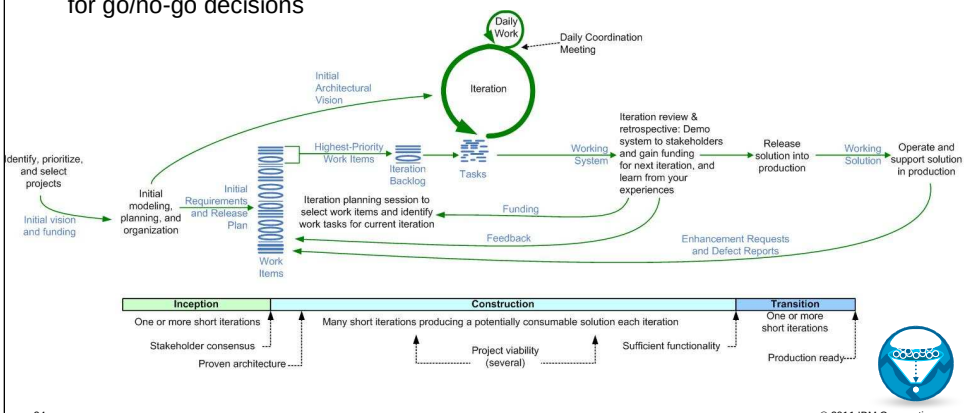
•Organizational complexity MAY make it difficult for you to hold these meetings (it may not be allowed)

•Enterprise architecture, business modelers, … may decide to attend the meetings

For more information about RTC, visit www.jazz.net

Scaling Value Lifecycle to Risk-Value Lifecycle

- Provides the extended team with explicit milestones centered on balancing risk mitigation and value creation
- Observations:
  - Key stakeholders frequently do not have time to carefully review and discuss the results of every iteration
  - Iteration demos are still a good idea for getting feedback, but not effective for go/no-go decisions



© 2011 IBM Corporation

Extends the value driven lifecycle of Scrum to address basic governance and risk-mitigation issues. This strategy was first introduced by Unified Process in the early 1990s.

Scaling iteration demos

- Geographic distribution
  - Demos occurs physically where possible, but transmitted electronically
  - Change demo times to reflect team distribution – spread the pain
  - Greater need to schedule the demos ahead of time
- Team size
  - Prioritize which subteam(s) should demo to the entire team
  - Individual subteams should do their own demos to their smaller community
- Regulatory compliance
  - Take meeting attendance and record action items (if any)
- Domain complexity
  - Invite a wider range of stakeholders required, not just insiders
- Organizational distribution
  - Invite stakeholders from each organization unit
  - Separate demos may be required by some organization units specific to them
- Technical complexity
  - Some of the work may not be visible through user interface, so may need to do a technical walkthrough instead

© 2011 IBM Corporation

Organizational complexity MAY make it difficult to hold regular demos

Enterprise discipline – You MAY choose to discuss how what you're demoing maps back to your enterprise business model/vision

Stakeholders - don't forget groups other than insiders.

Need to have planned ahead of time to identify the which aspects external stakeholders are interested in.

Relationships with stakeholders established during inception.

Unlikely a customer will be able to meet every two weeks. Don't forget business partners who can help influence purchases later.

Need to arrange attendance ahead of time.

Make sure you have good representation across stakeholder types.

Scaling reflections/retrospectives

- Disciplined agile delivery
  - Track your progress with IBM Rational SelfCheck
- Geographic distribution
  - Hold retrospectives electronically to include everyone
- Team size
  - Subteams should hold their own retrospectives
  - Subteams should share ideas with each other
- Regulatory compliance
  - May need to record any changes to your process
- Organizational distribution
  - May not be allowed to share improvements between organizations
- Organizational complexity
  - Existing process improvement strategies may focus on reviews (post mortems) at the end of the project
  - Process improvement may be "owned" by a specific process group
- Enterprise discipline
  - Consider an agile center of competency (CoC) to share ideas

26

© 2011 IBM Corporation

Review the effectiveness of the development process and environment

Retrospectives: www.retrospectives.com

Ask

- What went well?
- What could be changed to deliver better results?
- What have we learned?
- What still puzzles us?

Capture lessons learned

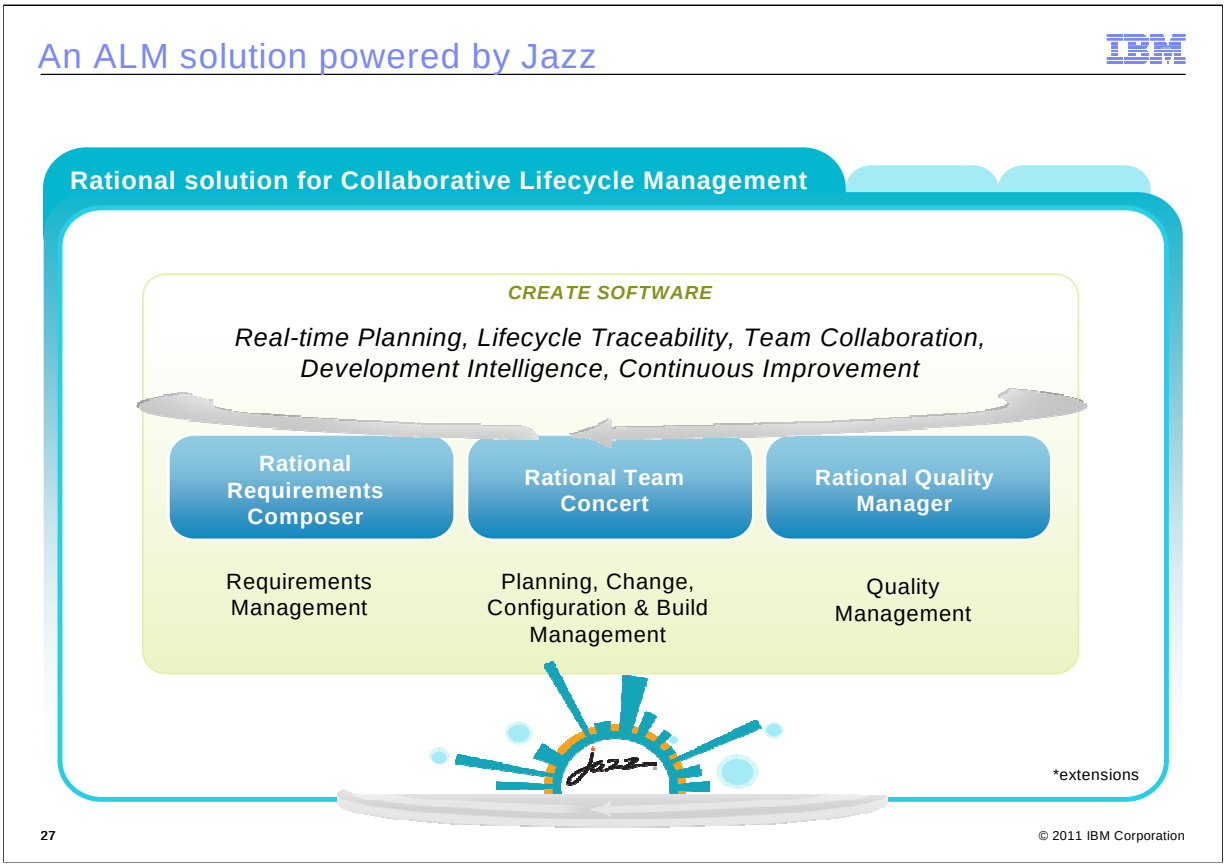
- Target the top 3 issues; do not take on too much

Next iteration

- Take 1 – 2 concrete actions, for example: "We need to stop doing X", "We need to get better at Y"
- Invest the appropriate amount of time versus value to be gained

Take retrospectives to the next level

- Consider tracking your results
- Consider using IBM® Rational® Self Check



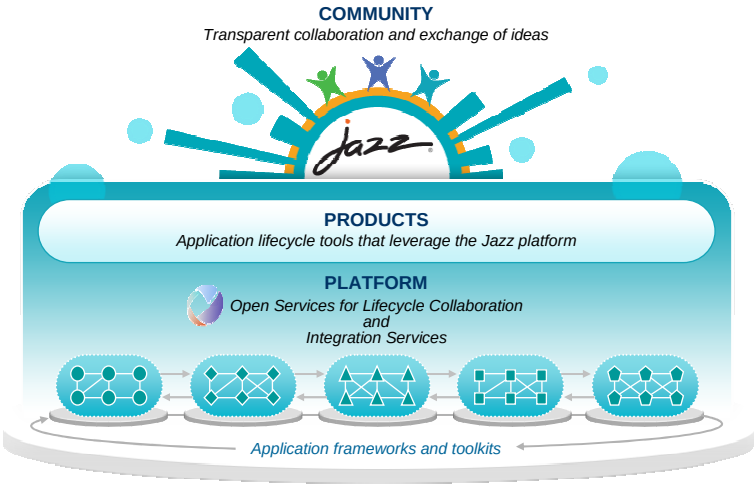
For organizations seeking a turnkey ALM solution for project teams – featuring the latest technology based on the Jazz platform – we offer the Rational solution for Collaborative Lifecycle Management. The Rational CLM solution provides the highest level of ALM interoperability in one easy-to-install and easy-to-use solution that can be optimized for Agile or traditional teams. The intention with CLM is to make it easy for customers to implement an ALM solution that meets the vast majority of their needs. Organizations requiring additional capabilities can extend the solution with other Rational tools.

For detailed descriptions of RTC's capabilities, go to http://www-01.ibm.com/software/rational/products/rtc/

For detailed descriptions of RQM's capabilities go to http://www-01.ibm.com/software/awdtools/rqm/

For detailed descriptions of RRC's capabilities go to http://www-01.ibm.com/software/awdtools/rrc/

Jazz provides open collaboration across the software and systems lifecycle

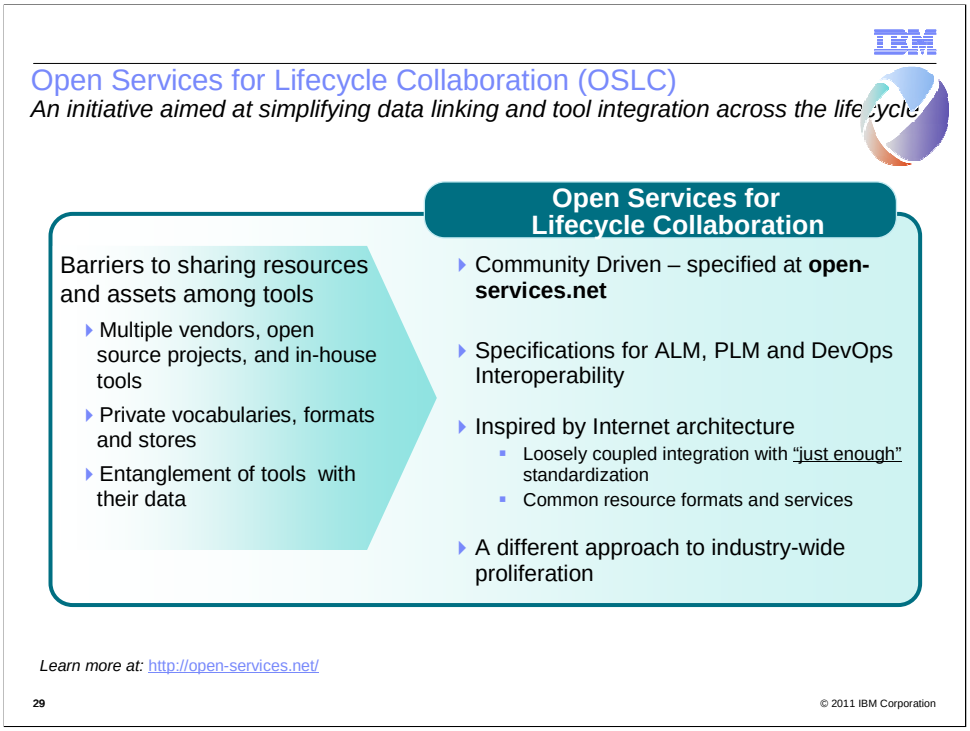


Learn more at: https://jazz.net/about/

28

© 2011 IBM Corporation

Application frameworks and toolkits: Frameworks and toolkits to aid tool writers in the development of tools. Rational Team Concert, Rational Quality Manager, and Rational Requirements Composer were all built leveraging application frameworks.



Slide Source: Andy Gurd

Open Services for Lifecycle Collaboration (OSLC) is an industry initiative, initially proposed by IBM in June 2008, aimed at simplifying collaboration across the software delivery lifecycle. The OSLC community specifies resource formats and RESTful APIs for lifecycle collaboration.

Emphasis has shifted from "foundation integration" services to "loosely coupled domain" integration services

Priority has shifted from a "complete" representation of a particular domain tool interface to identifying what's required for specific cross-domain integrations

Barriers to sharing resources and assets across the software lifecycle

- Multiple vendors, open source projects, and in-house tools
- Private vocabularies, formats and stores

For each domain of interest:

Collect our own thinking

- Support for extensibility
- Proper granularization
- Consistency of approach across ALM

Involve others

- Who are the players?
- What are the integration scenarios?

Form work groups

- openly explore the scenarios

The answer to this question is summarized in the five ALM principles, or Imperatives, you see here. These are the criteria that, in our work with hundreds of customers on thousands of ALM projects, must be satisfied in order to provide and effective and productive ALM environment.

Presentation Description

The Agile Scaling Model (ASM) provides the context and advice for effectively tailoring agile strategies. It describes how to extend the agile construction life cycle into a full-fledged disciplined agile delivery life cycle from project initiation to production. It then describes how to tailor agile practices to address eight scaling factors: team size, geographical distribution, domain complexity, organizational distribution, regulatory compliance, organizational complexity, technical complexity, and enterprise disciplines (such as enterprise architecture, reuse, and governance).

Process/Mechanics

- I'll summarize the three categories of the ASM and how the build on each other. 1. Core agile methods which focus on construction 2. Disciplined agile delivery methods which focus on the full delivery lifecycle 3. Agility at scale
- I'll then walk through the eight scaling factors, explaining each one and summarizing appropriate industry statistics.
- I'll walk through several common agile practices and show how each change when being applied at scale — Same practice, different implementation based on the context you find yourself in.
- I'll summarize several not-so-common practices applicable at scale
- Questions will be taken throughout as I prefer to have discussions.

Learning outcomes

- Discover how to address the full delivery lifecycle
- Learn that there is more to scaling than large or distributed teams
- Learn how to balance self organization with lean governance
- Learn how to take a risk-value driven approach to disciplined agile delivery

© 2011 IBM Corporation